

**UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA**

Alisson Granemann Abreu, Fernando Burigo Teixeira

**ALPC - APLICAÇÃO PARA LEVANTAMENTO DE
PARTICIPAÇÃO CIENTÍFICA**

Florianópolis

2017

Alisson Granemann Abreu, Fernando Burigo Teixeira

**ALPC - APLICAÇÃO PARA LEVANTAMENTO DE
PARTICIPAÇÃO CIENTÍFICA**

Trabalho de Conclusão de Curso submetido ao curso de graduação em Ciência da Computação para a obtenção do Grau de Bacharel.

Orientadora: Carina Friedrich Dornelles

Florianópolis

2017

Catálogo na fonte elaborada pela biblioteca da
Universidade Federal de Santa Catarina

A ficha catalográfica é confeccionada pela Biblioteca Central.

Tamanho: 7cm x 12 cm

Fonte: Times New Roman 9,5

Maiores informações em:

<http://www.bu.ufsc.br/design/Catalogacao.html>

Alisson Granemann Abreu, Fernando Burigo Teixeira

**ALPC - APLICAÇÃO PARA LEVANTAMENTO DE
PARTICIPAÇÃO CIENTÍFICA**

Este Trabalho de Conclusão de Curso foi julgado aprovado para a obtenção do Título de “Bacharel”, e aprovado em sua forma final pelo curso de graduação em Ciência da Computação.

Florianópolis, 1 de junho 2017.

Renato Cislaghi
Coordenador

Banca Examinadora:

Carina Friedrich Dorneles
Orientadora

Ronaldo dos Santos Mello

José Luís Almada Güntzel

RESUMO

A Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (Capes), é uma fundação do Ministério da Educação (MEC) e desempenha papel fundamental na expansão e consolidação da pós-graduação em todos os estados da Federação Brasileira. Dentre as atividades da Capes, tem-se a avaliação da pós-graduação stricto sensu (mestrado e doutorado) e a divulgação da produção científica. Essas duas atividades são feitas através da realização de relatórios com indicativos da produção científica, com a cooperação docente-docente, docente-aluno, a participação discente em projetos do orientador, prêmios recebidos, etc. Todo esse levantamento de dados é feito de forma manual, onde os gestores de programas de pós-graduação manualmente analisam os currículos de seus discentes, coletando as informações necessárias para preencher os relatórios. O presente trabalho tem como intenção facilitar esse processo manual, oferecendo uma ferramenta Web, na qual toda a coleta de dados é feita de forma automatizada. A colaboração, produção de pesquisadores, alunos e pares, não será mais feita de forma exaustiva, levando horas analisando todos os currículos. Basta acessar de forma simples e prática o site, importar os currículos e todos os dados serão disponibilizados de forma organizada para visualização e consulta.

Palavras-chave: Produção Científica; Colaboração Científica; Dados Curriculum Vitae Lattes; Mapeamento da Produtividade

ABSTRACT

The Higher Education Personnel Improvement Coordination (Capes), is a foundation of the Ministry of Education (MEC) and plays a key role on expansion and consolidation of graduate studies in all states of the Brazilian Federation. Among the activities of the Capes, there are *stricto sensu*, graduate courses evaluation (master's and doctorate's degree) and the dissemination of scientific production. These two activities are done by conducting reports with indicative of the scientific production, with professor-professor and professor-student cooperation, the student participation in the professor projects, received awards, etc. All this data collection is done by hand, i.e. heads of graduate programs manually analyze the resumes of their students, collecting the information needed to complete the reports. This work intends to facilitate this manual process, offering a Web application in which all data collection is be done in an automated way. The collaboration, production of researchers, students and peers no more need to be done in a exhaustive manner, taking hours analyzing all resumes. It is only necessary to access the site, import the resumes and all data will be available in an organized way to view and query.

Keywords: Scientific production; Scientific collaboration; Data Curriculum Vitae Lattes; Productivity mapping

LISTA DE FIGURAS

Figura 1	Recorte do Data Type Definition do Lattes	24
Figura 2	Visão geral da aplicação	34
Figura 3	Modelo lógico do controle de usuário e colaboração docente- discente	36
Figura 4	Modelo lógico da graduação e colaboração docente-docente	37
Figura 5	Importar currículo	38
Figura 6	Tela de importação	38
Figura 7	Visão Geral do Processamento	40
Figura 8	Fluxo realizado para procurar um currículo	42
Figura 9	Lógica para salvar uma formação	43
Figura 10	Fluxo básico de busca para colaboração docente-docente	44
Figura 11	Tela de relatórios do programa	49
Figura 12	Tela de relatórios do docente	49
Figura 13	Tela de relatórios do discente	50

LISTA DE TABELAS

Tabela 1	Perfil q-gram de x e y	26
Tabela 2	Comparação entre trabalhos.....	32
Tabela 3	Pessoa 1 e 2 indexadas	35

LISTA DE ABREVIATURAS E SIGLAS

ALPC	Aplicação para Levantamento de Participação Científica	18
CNPq	Conselho Nacional de Desenvolvimento Científico e Tecnológico	23
W3C	World Wide Web Consortium	23
XML	Extensible Markup Language	23
SGML	Standard Generalized Markup Language	24
ISO	International Organization for Standardization	24
IDE	Integrated Development Environment	51
JAXB	Java Architecture for XML Binding	51
EJB	Enterprise JavaBeans	52
HTTP	Hypertext Transfer Protocol	52
REST	Representational State Transfer	52
SGBD	Sistema de Gerenciamento de Banco de Dados	55

SUMÁRIO

1 INTRODUÇÃO	17
1.1 OBJETIVOS	19
1.1.1 Objetivo geral	19
1.1.2 Objetivos específicos	19
1.2 METODOLOGIA	19
1.3 ORGANIZAÇÃO DO TRABALHO	20
2 FUNDAMENTAÇÃO TEÓRICA	23
2.1 CURRÍCULO LATTES	23
2.2 Q-GRAM	25
2.3 ALGORITMOS DE SIMILARIDADE	25
2.3.1 Distância Q-Gram	25
2.3.2 Jaro-Winkler	26
2.3.3 Levenshtein	27
3 TRABALHOS RELACIONADOS	29
3.1 SAPOS/UFF	29
3.2 SAPOS/UFPR	29
3.3 SCRIPTLATTES	30
3.4 COBALTO	31
3.5 ANÁLISE COMPARATIVA	31
4 ALPC	33
4.1 VISÃO GERAL	33
4.2 PESQUISA DE TEXTO COMPLETA	34
4.3 ESQUEMA DO BANCO DE DADOS	35
4.4 IMPORTAÇÃO	36
4.5 PROCESSAMENTO	39
4.5.1 Buscar Currículo	39
4.5.2 Extração dos dados	41
4.5.2.1 Formação	43
4.5.2.2 Orientação	44
4.5.2.3 Produção	45
4.5.2.4 Atuação profissional	47
4.5.2.5 Linha de pesquisa	47
4.5.2.6 Projeto de pesquisa	48
4.5.2.7 Graduação	48
4.6 RELATÓRIOS	48
4.7 FERRAMENTAS, BIBLIOTECAS E UTILITÁRIOS	51
4.7.1 Eclipse	51

4.7.2	JAXB	51
4.7.3	Spring Framework	52
4.7.4	Apache Lucene	52
4.7.5	Node.js	52
4.7.6	Gulp	53
4.7.7	ECMAScript 6	53
4.7.8	Babel	53
4.7.9	React	54
4.7.10	REST	54
4.7.11	Git e Github	54
4.7.12	Maven	55
4.7.13	PostgreSQL	55
5	CONCLUSÕES E TRABALHOS FUTUROS	57
	REFERÊNCIAS	59
	APÊNDICE A – Artigo	71
	APÊNDICE B – Código	73

1 INTRODUÇÃO

Anualmente, a Capes - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior¹ - realiza avaliações (CAPES, 2017) dos PPGs - Programas de Pós-Graduação, com a participação da comunidade acadêmica-científica através de consultores destinados a essa finalidade. A avaliação é um meio de assegurar e manter a qualidade dos cursos de Mestrado e Doutorado no país. A avaliação é baseada, essencialmente, nos dados inseridos na Plataforma Sucupira², uma ferramenta desenvolvida pela Capes e UFRN, com o objetivo de coletar informações, realizar análises e avaliações e ser a base de referência do Sistema Nacional de Pós-Graduação (SNPG). A partir dos dados inseridos, a plataforma gera um relatório, que é utilizado como base para a avaliação. Um dos pontos considerados nesta avaliação é a análise da colaboração científica entre docente-docente e/ou docente-discente, ou seja, como os docentes colaboram entre si e como os estudantes estão envolvidos nestes trabalhos. Além dessa colaboração, a participação discente em projetos do orientador, prêmios recebidos, etc, também são itens importantes.

Todas estas informações são mantidas, individualmente, por docentes e discentes em seus currículos cadastrados na Plataforma Lattes³. A plataforma Lattes é uma sistema web que disponibiliza uma ferramenta para montagem e armazenamento de currículos. Seu alcance se estende à gestão e operacionalização do fomento do CNPq, outras agências de fomento federais e estaduais, das fundações estaduais de apoio à ciência e tecnologia, das instituições de ensino superior e dos institutos de pesquisa. Toda produção discente/docente é incluída em seus devidos currículos, com todas as informações relevantes.

Para alimentar certos campos de dados da plataforma Sucupira e gerar estatísticas prévias para planejamentos estratégicos de PPGs, seus gestores devem analisar cada currículo dos integrantes do Programa, avaliando todas as produções, para então relacionar docente/-discente a cada trabalho. Considerando que cada programa possui um número considerável de pesquisadores e alunos, o tempo gasto para a conclusão desta tarefa é alto, além de ser passível de erro.

Uma solução para isso é o projeto e construção de alguma plataforma computacional que realize o cruzamento dos dados dos currículos

¹capes.gov.br

²<https://sucupira.capes.gov.br>

³lattes.cnpq.br

dos participantes de um Programa de Pós-Graduação e gera relatórios estatísticos baseados no agrupamento de informações relevantes para a avaliação da CAPES, como número de produções bibliográficas e técnicas, participações em projetos, etc, levando em consideração a colaboração entre pares.

O grande desafio no desenvolvimento de tal plataforma é a heterogeneidade dos dados presentes nos currículos Lattes. Apesar dos currículos estarem em formato XML, que prevê interoperabilidade, os dados inseridos podem ter valores distintos para a mesma informação. Por exemplo, uma colaboração está contida em uma orientação, de mestrado a doutorado, e é a partir das suas informações que a relação é estabelecida. Sendo assim, um orientador possuirá as informações da sua orientação em seu currículo e o orientando também, no entanto de forma distinta. O orientador adiciona ao currículo uma orientação em andamento ou concluída, e o orientando em sua formação. Dados como nome do orientador/orientando e título do trabalho são comuns em ambos os currículos, portanto é possível encontrar a colaboração quando há a correspondência de um trabalho entre o discente e docente, aplicando as condições necessárias. É possível tratar este problema através da aplicação de funções de similaridade a textos (DORNELES; GALANTE, 2008), como em nomes de autores, instituição ou título do trabalho. Estas funções de similaridade recebem dois textos como valor de entrada e retornam um valor entre 0 e 1 (ou 0 à 100%) equivalente a taxa de similaridade entre eles, ou seja, textos exatamente iguais recebem o valor 1 e caso contrário, o valor de similaridade difere entre as funções, baseado na proximidade entre os textos.

Neste trabalho é descrito o ALPC (Aplicação para Levantamento de Participação Científica), uma plataforma que tem como principal finalidade vincular toda e qualquer colaboração científica, entre docente e discente, através de seus respectivos currículos Lattes, analisando e processando suas informações. A interação com o usuário é feita por uma aplicação Web disponibilizada em um servidor único, onde cada usuário é identificado com um papel dentro do sistema (coordenador, docente e discente) correspondendo às suas funções. Para o estabelecimento da relação em uma colaboração, o ALPC tem o seguinte processo: o primeiro é indexar as entidades (uma pessoa, uma produção, uma orientação) através de um recurso chamado *full-text search*, que utiliza o conteúdo dos campos da entidade e quebra-os em termos utilizando um *tokenizer*, transformando-os em seu próprio índice. A partir disso, sempre que uma busca é feita, apenas as entidades com a maior probabilidade de correspondência são retornadas e comparadas

através das funções de similaridade, a fim de garantir o correto relacionamento das colaborações e evitar duplicar qualquer dado, após o correto relacionamento, as colaborações são salvas para então ficarem disponíveis para a visualização pelo usuário.

1.1 OBJETIVOS

1.1.1 Objetivo geral

O ALPC tem como principal finalidade vincular toda e qualquer colaboração científica, entre docente-discente e docente-docente, através de seus respectivos currículos Lattes. Deve-se fazer uma análise do currículo em busca dessas colaborações que são uteis aos relatórios da Capes e como relaciona-las, salvar esses relacionamentos e disponibilizá-los por meio de uma aplicação Web.

1.1.2 Objetivos específicos

- Fazer um estudo sobre a estrutura disponibilizada pelo Lattes (formato XML) do currículo de pesquisadores e alunos, a fim de reunir as informações necessárias para a busca desejada, e então armazená-los em uma base.
- A partir da base, com os dados processados, aplicar algoritmos de similaridade (previamente escolhidos), para assim tornar possível a busca dos vínculos, baseado em certos parâmetros, como nome(s) do(s) professor(es)/aluno(s), por exemplo.
- Agrupar as informações geradas com as relações, separadas por áreas, como informações relacionadas a docente, discente e o programa todo e disponibilizá-las ao usuário.
- As informações da busca (entrada e saída), devem ser coletadas e mostradas através de interface amigável, em uma aplicação web, desenvolvida neste trabalho.

1.2 METODOLOGIA

A aplicação ALPC foi desenvolvida sem o uso de qualquer tipo de metodologia específica de Engenharia de Software. Porém, algumas etapas foram planejadas com a intenção de organizar o processo de

desenvolvimento.

Etapa 1: o primeiro passo foi definir o projeto. O que exatamente deveria ser feito, como deveria ser implementado e o que deveria ser retornado ao usuário. Foram identificados quais seriam os dados necessários no relatório entregue para a avaliação da CAPES e quais poderiam ser extraídos do currículo Lattes de todos os docentes e discentes participantes do Programa de Pós-Graduação. Tornou-se necessário o estudo de toda a estrutura do currículo Lattes a fim de mapear estes dados.

Etapa 2: revisão de literatura, buscando artigos e trabalhos relacionados às funções de similaridade textuais. Foi feita uma avaliação da melhor aplicação das funções dentro do contexto deste trabalho. Além disso, foi feito um estudo e escolha das ferramentas, *frameworks* e linguagens utilizados no desenvolvimento do sistema, baseando-se no conhecimento dos integrantes, facilidade da aprendizagem/implementação e eficiência performática.

Etapa 3: a etapa 3 focou no desenvolvimento de código utilizando os dados recolhidos nas duas primeiras etapas. Não foi seguido nenhum processo de desenvolvimento de código específico, mas foram adotadas boas práticas de codificação, sincronização da dupla e realização de reuniões semanais com a orientadora, garantindo o desenvolvimento das funcionalidades do sistema dentro dos requisitos.

Etapa 4: redigir o relatório do trabalho, documentando todas as partes do processo do desenvolvimento do sistema, paralelamente à etapa 3.

Etapa 5: finalização do desenvolvimento do trabalho, aplicação de testes e estudo sobre os métodos utilizados, para avaliar a eficácia do sistema proposto.

1.3 ORGANIZAÇÃO DO TRABALHO

Para uma melhor compreensão de como foi desenvolvido o sistema ALPC, este documento está agrupado em capítulos. No segundo capítulo a fundamentação teórica é descrita, enfatizando alguns conceitos pertinentes ao desenvolvimento do trabalho. No terceiro capítulo são mostrados trabalhos relacionados, finalizando com um comparativo entre os sistemas. O ALPC é apresentado no quarto capítulo, mostrando seus módulos, como foram desenvolvidos, suas funções dentro do sistema e seus resultados, além de todas as ferramentas utilizadas neste trabalho. No quinto capítulo são destacadas funcionalidades que

são passíveis de serem adicionadas ao trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo aborda os conceitos básicos para a compreensão deste trabalho. É apresentada a estrutura do currículo Lattes e seus agrupamentos, importantes para que a sua análise seja completa. Também são apresentados os algoritmos utilizados no sistema para a similaridade entre termos: Q-gram, Jaro-Winkler e Levenshtein.

2.1 CURRÍCULO LATTES

A Plataforma Lattes¹ criada e mantida pelo CNPq, é um ambiente que integra bases de dados de currículos, de grupos de pesquisa e de instituições em um único sistema de informações. O Currículo Lattes se tornou um padrão nacional no registro dos estudantes e pesquisadores do país, sendo adotado pela maioria das universidades e institutos de pesquisa do país.

A Comunidade CONSCIENTIAS/LMPL² definiu o DTD (*Data Type Definition*)(CONSCIENTIAS/LMPL, 2014) do Currículo Lattes, ou seja, a estrutura em que é montada o currículo (como mostra a Figura 1), evoluindo posteriormente até se tornar uma gramática *XML Schema*, uma recomendação da W3C(World Wide Web Consortium)³, que descreve formalmente os elementos da linguagem XML(Extensible Markup Language). Com esse modelo, as universidades brasileiras podem extrair informações do currículo Lattes e/ou gerar informações para o mesmo a partir dos seus sistemas corporativos.

Extensible Markup Language (XML)(W3C, 2015) é um formato de texto simples, muito flexível derivado de SGML (ISO 8879). Originalmente concebido para enfrentar os desafios da publicação eletrônica em grande escala, XML também está desempenhando um papel cada vez mais importante na troca de uma ampla variedade de dados na Web e em outros lugares.

A estrutura do currículo Lattes é dividida em cinco grandes elementos: Dados gerais, Dados complementares, Produção bibliográfica, Produção técnica e Outras produções. Cada um destes grupos pode apresentar um ou mais elementos-filhos que são descritos a seguir. Vários destes elementos citados são utilizados no processamento e mani-

¹<http://lattes.cnpq.br/>

²<http://lmpl.cnpq.br/lmpl/>

³<https://www.w3.org/>

```

<!ELEMENT DADOS-GERAIS (RESUMO-CV?,OUTRAS-INFORMACOES-RELEVANTES? , ENDereco?,
FORMACAO-ACADEMICA-TITULACAO?, ATUACOES-PROFISSIONAIS?, AREAS-DE-ATUACAO?, IDIOMAS?,
PREMIOS-TITULOS?)>
<!ATTLIST DADOS-GERAIS
  NOME-COMPLETO CDATA #REQUIRED
  NOME-EM-CITACOES-BIBLIOGRAFICAS CDATA #REQUIRED
  NACIONALIDADE CDATA #REQUIRED
  CPF CDATA #IMPLIED
  NUMERO-DO-PASSAPORTE CDATA #IMPLIED
  PAIS-DE-NASCIMENTO CDATA #IMPLIED
  UF-NASCIMENTO CDATA #IMPLIED
  CIDADE-NASCIMENTO CDATA #IMPLIED
  FORMATO-DATA-DE-NASCIMENTO NMTOKEN #FIXED "DDMMAAAA"
  DATA-NASCIMENTO CDATA #IMPLIED
  SEXO (MASCULINO | FEMININO) #REQUIRED
  NUMERO-IDENTIDADE CDATA #IMPLIED
  ORGAO-EMISSOR CDATA #IMPLIED
  UF-ORGAO-EMISSOR CDATA #IMPLIED
  FORMATO-DATA-DE-EMISSAO NMTOKEN #FIXED "DDMMAAAA"
  DATA-DE-EMISSAO CDATA #IMPLIED
  NOME-DO-PAI CDATA #IMPLIED
  NOME-DA-MAE CDATA #IMPLIED
  PERMISSAO-DE-DIVULGACAO (SIM | NAO) #REQUIRED
  NOME-DO-ARQUIVO-DE-FOTO CDATA #IMPLIED
  TEXTO-RESUMO-CV-RH CDATA #IMPLIED
  OUTRAS-INFORMACOES-RELEVANTES CDATA #IMPLIED
  DATA-FALECIMENTO CDATA #IMPLIED
  SIGLA-PAIS-NACIONALIDADE CDATA #IMPLIED
  PAIS-DE-NACIONALIDADE CDATA #IMPLIED
  RACA-OU-COR CDATA #IMPLIED
>

```

Figura 1 – Recorte do Data Type Definition do Lattes

pulação das informações deste trabalho.

Dentro de **Dados gerais** encontram-se as informações pessoais como endereço, idiomas, formação acadêmica, atuações profissionais, áreas de atuação e prêmios e títulos. Já em **Dados complementares** são encontradas participações em eventos e bancas, orientações em andamento e informações adicionais. O grupo **Produção bibliográfica** possui trabalhos em eventos, artigos aceitos e publicados, trabalhos em eventos, livros, capítulos, textos em jornais ou revistas e demais tipos de produção bibliográfica. Em **Produção técnica**, situam-se os trabalhos técnicos, software, patente, desenho industrial, marca, topografia de circuito integrado, produto tecnológico, processos ou técnicas, trabalho técnico, cultivar registrada ou protegida e demais tipos de produção técnica. Por último, em **Outras produções**, encontra-se orientações concluídas, produções artísticas/culturais e demais trabalhos.

2.2 Q-GRAM

Um *Q-Gram* ou *N-Gram* (UKKONEN, 1992) é uma sequência de caracteres de tamanho n : um 2-gram (ou bigram) pode ser uma sequência de letras como "b", "bc" ou "cd" e assim por diante. Um 3-gram (ou trigram) pode ser uma sequência de três caracteres como "abc" ou "def". As palavras não necessariamente são "legíveis", podendo ser fonemas, sílabas ou letras concatenadas.

Dada uma *string* A ao longo de um alfabeto finito Σ , $|A|$ é o comprimento de A , A_i refere-se ao caractere i de A , e $A_{i,j}$ é a *substring* de A que começa com o caractere i e termina com o j . A substring de comprimento $q > 0$ de A é um q -gram de A .

Por exemplo, o nome Frodo tem as seguintes *q-grams* com $q = 2$: Fr ro od do. O nome Freud dividida nestes *q-grams* resulta em: Fr re eu ud.

2.3 ALGORITMOS DE SIMILARIDADE

Nesta seção são apresentados os algoritmos de similaridade utilizados neste trabalho, para a comparação de textos. São explicadas suas definições e exemplos são demonstrados.

2.3.1 Distância Q-Gram

O algoritmo de distância *Q-Gram* (NAVARRO; SUTINEN; TARHIO, 2000; RASMUSSEN; STOYE; MYERS, 2006; JURAFSKY; MARTIN, 2014) é uma função de similaridade utilizando *q-grams*, explicados na Seção 2.2. Quanto menor a distância por duas *strings*, mais similares elas são. A medida é dada através da contagem de ocorrências de distintos *q-grams* de tamanho n nas duas *strings* a serem comparadas.

Seja uma *string* $s = a_1 \cdot \dots \cdot a_k$ e q um *q-gram* de tamanho n de um alfabeto Σ^q . Se $a_i a_{i+1} \cdot \dots \cdot a_{i+n-1} = q$ para algum i , então s tem uma ocorrência de q . É chamado o perfil *q-gram* de s , um vetor contendo a quantidade de ocorrências para todo $q \in \Sigma^q$, dado por $G(s) = G(s)[q]$.

A distância entre duas *strings* é definida como o módulo da di-

ferença entre seus perfis q -gram, dado por

$$D_q(x, y) = \sum_{q \in \Sigma^q} |G(x)[q] - G(y)[q]|$$

A distância relativa normalizada, ou taxa de erro, é definida como a distância dividida pelo comprimento da maior *string* envolvida.

Exemplo: Dadas as *strings* $x = abcde$ e $y = acdbc$, assumindo tamanho 2 para os q -grams, temos $\Sigma^q = ab, bc, cd, de, ac, db$. Então o perfil q -gram de x e y é definido por:

Tabela 1 – Perfil q -gram de x e y

	ab	bc	cd	de	ac	db
x	1	1	1	1	0	0
y	0	1	1	0	1	1

Assim, a distância é $D(x, y) = 4$.

2.3.2 Jaro-Winkler

O Jaro-Winkler (COHEN; RAVIKUMAR; FIENBERG, 2003) é uma medida de similaridade entre duas strings. Assim como a distância Q-Gram, quanto menor a distância, mais similares as strings são. A métrica de distância Jaro-Winkler é mais adequada para strings curtas, como nomes de pessoas ou outros nomes próprios. A pontuação é normalizada tal que 0 equivale a nenhuma semelhança e 1 é uma correspondência exata. Este algoritmo é uma variante da métrica de Jaro, explicada a seguir.

Dadas as strings $s = a_1 \dots a_K$ e $t = b_1 \dots b_L$, define-se um caractere a_i em s para ser comum com t onde há um $b_j = a_i$ em t tal que $i - H \leq j \leq i + H$, sendo $H = \frac{\min(|s|, |t|)}{2}$. Assumindo $s' = a'_1 \dots a'_K$, sendo os caracteres que s têm em comum com t (na mesma ordem que aparecem em s) e que $t' = b'_1 \dots b'_L$, sendo análogo, aplica-se uma transposição em s' e t' em uma posição i onde $a'_i \neq b'_i$. Assumindo $T_{s', t'}$ como a metade do número de transposições para s' e t' , a métrica de Jaro é dada por

$$Jaro(s, t) = \frac{1}{3} * \left(\frac{|s'|}{|s|} + \frac{|t'|}{|t|} + \frac{|s'| - T_{s', t'}}{|s'|} \right)$$

e sua variante, Jaro-Winkler, também usa o comprimento P do maior prefixo comum entre s e t. Sendo $P' = \max(P, 4)$, define-se

$$Jaro - Winkler(s, t) = Jaro(s, t) + \frac{P'}{10} * (1 - Jaro(s, t))$$

Exemplo:

Dadas as strings s = MARTHA e t = MARHTA são encontrados os valores:

- $m = 6$
- $|s'| = 6$
- $|t'| = 6$
- $T_{s',t'} = \frac{2}{2} = 1$

Assim, o valor de Jaro é

$$Jaro(s, t) = \frac{1}{3} * \left(\frac{6}{6} + \frac{6}{6} + \frac{6-1}{6} \right) = 0.944$$

Para encontrar o valor de Jaro-Winkler, é necessário P', que é igual à 3 (já que o maior prefixo comum é MAR). Então

$$Jaro - Winkler(s, t) = 0.944 + \frac{3}{10} * (1 - 0.944) = 0.961$$

2.3.3 Levenshtein

Levenshtein (LEVENSHTEIN, 1966) é uma das funções de similaridade pioneiras, sendo uma das mais simples. A distância entre duas strings é dada pelo número mínimo de operações necessárias para transformar uma string em outra. Estão inclusas nas operações a inserção, deleção ou substituição de um caractere.

Exemplo:

A distância entre "barato" e "sapatos" é três, dado que são necessárias duas substituições e uma deleção.

- barato -> sarato (substituição)
- sarato -> sapato (substituição)
- sapato -> sapatos (inserção)

3 TRABALHOS RELACIONADOS

Este capítulo trata de trabalhos que possuem propostas semelhantes ao ALPC, descrevendo brevemente cada um e, posteriormente, fazendo uma análise comparativa entre todos. Os trabalhos relacionados foram selecionados baseados na semelhança entre as funcionalidades propostas pelo ALPC.

3.1 SAPOS/UFF

O SAPOS¹ (Sistema de Apoio à Pós) (FERREIRA; AMARO, 2013) da Universidade Federal Fluminense vem com a função de apoio ao programa de pós-graduação da universidade, como gerenciamento de matrículas, orientações, bolsas, disciplinas, credenciamentos, entre outros.

Seu desenvolvimento foi iniciado por professores da instituição, posteriormente incluindo alunos da graduação de Ciência da Computação no time de desenvolvimento. É um software livre e de código aberto. O framework utilizado é o Ruby on Rails, com a linguagem de programação Ruby.

As principais funcionalidades do SAPOS são os cadastros e gerenciamento de alunos, professores, bolsas e etapas. Com relação aos alunos, são cadastrados seus dados pessoais e sua relação com uma ou mais matrículas. Com relação aos professores, além de seus dados pessoais, são vinculadas suas orientações. No segmento de bolsas, são feitos os cadastros e monitoramento das bolsas concedidas, relacionadas aos alunos. A área de etapas contempla as etapas que os alunos são obrigados a completar, durante o curso de Pós-Graduação, para receber sua titulação.

3.2 SAPOS/UFPR

O SAPOS² (Sistema de Apoio à Pós)(FABRO; ALMEIDA; SLUZARSKI, 2012) da Universidade Federal do Paraná é uma aplicação web para gestão de sistemas de pós-graduação (Mestrado e Doutorado). É um software livre com código aberto, portanto seu uso é gratuito. Ini-

¹<https://sapos.ic.uff.br/homologacao/credits/show>

²<http://web.inf.ufpr.br/didonet/sapos-sistema-de-apoio-a-pos-graduacao>

ciou a partir da necessidade da gestão do Programa de Pós Graduação em Informática da Universidade Federal do Paraná. Seu desenvolvimento envolve alunos da Ciência da Computação da universidade e seu número varia de acordo com o número de bolsas disponíveis. O sistema foi desenvolvido com a linguagem Java e utiliza o framework Spring.

Assim como o SAPos da Universidade Federal Fluminense, suas funcionalidades são voltadas à gestão do programa de pós-graduação, como cadastro pessoal de alunos e professores e gestão de bolsas. Porém o SAPos da Universidade Federal do Paraná possui algumas outras funcionalidades, como o gerenciamento de disciplinas e turmas, controle de defesas e relatórios.

3.3 SCRIPTLATTES

Outra ferramenta relacionada é o ScriptLattes³ (MENA-CHALCO; JUNIOR, 2009). É um software livre, de código aberto, com a finalidade de auxiliar à Secretaria de Pós-graduação do IME-USP. Ele foi projetado para funcionar através de terminal (console) no sistema operacional Linux, entretanto, pode ser compilado e configurado para ser utilizado sob outros sistemas operacionais, como MS Windows ou MacOS. Suas primeiras versões foram programadas usando Perl e posteriormente refatorado para a linguagem Python.

Foi desenvolvido com o intuito de extrair currículos no site da Plataforma Lattes, para a geração de relatórios e gráficos de interesse. Entre eles estão as produções, artigos, orientações, prêmios, entre outros. Sua principal função é auxiliar grupos de professores ou departamentos no processo de coleta de dados para avaliação do corpo docente.

É importante destacar que o ScriptLattes baixa automaticamente os currículos Lattes (em formato HTML) de um grupo de pessoas de interesse e compila as listas de produções, tratando apropriadamente as produções duplicadas e similares. Ele gera os relatórios com listas de produções e orientações separadas por tipo, e, adicionalmente, a ferramenta permite a criação automática de grafos de coautoria entre os membros do grupo e um mapa de geolocalização dos membros e alunos com orientação concluída.

³<http://scriptlattes.sourceforge.net/>

3.4 COBALTO

COBALTO ⁴ é um sistema desenvolvido pela UFPel (Universidade Federal de Pelotas) que pretende integrar os diversos sistemas de Tecnologia da Informação (TI) da UFPel. É um software livre e em plataforma Web.

Por seu intermédio é possível o registro e o acesso de informações referentes ao ensino, à pesquisa, à extensão e ao gerenciamento administrativo, atendendo tanto a graduação quanto a pós-graduação. Dentre os usuários que possuem acesso autenticado ao sistema estão: candidatas a vagas na Universidade, alunos, professores e funcionários administrativos. Para cada tipo de usuário, é liberado o acesso a uma área do sistema.

Ele contempla todo o ciclo de vida de um estudante na Instituição, desde o preenchimento online da ficha de inscrição para os selecionados nos diferentes tipos de ingresso até sua formatura e diplomação. No que se refere a recursos aos docentes, existem funcionalidades para cadastro de suas atividades, como as avaliações, publicação das notas, frequência e diário de classe. O professor pode ainda consultar as suas atividades, acompanhar o desempenho dos alunos, bem como emitir atas de provas e exames, entre outros.

3.5 ANÁLISE COMPARATIVA

Esta seção apresenta um comparativo simples entre os trabalhos relacionados e o ALPC. Foram comparadas as principais funcionalidades entre os trabalhos, demonstrando a capacidade deste trabalho em relação aos outros. A Tabela 2 mostra a comparação, baseada entre as possíveis áreas dentro do sistema. Valores 'Sim' representam que a aplicação possui a área e 'Não', caso contrário.

Na Tabela 2, podemos perceber que os SAPos da UFPR e UFF e o Cobalto são voltados principalmente para gerência da parte acadêmica de docentes e discentes, com as áreas de disciplinas e/ou bolsas. Já o ALPC e o scripLattes estão voltados para a geração de relatórios, a partir de certos dados enviados à eles. É importante ressaltar que, apesar de o ALPC não estar voltado à gerência da parte acadêmica, ele é organizado por áreas e cada uma contém uma lista ordenada com informações pertinentes para cada item. Isso não acontece com o

⁴<https://cobalto.ufpel.edu.br/>

Tabela 2 – Comparação entre trabalhos

	ALPC	Sapos/ UFF	Sapos/ UFPR	scriptLattes	COBALTO
Docente	Sim	Sim	Sim	Não	Sim
Discente	Sim	Sim	Sim	Não	Sim
Orientações	Sim	Sim	Sim	Sim	Não
Produções	Sim	Não	Não	Sim	Não
Atuações	Sim	Não	Não	Não	Não
Disciplinas	Não	Não	Sim	Não	Sim
Bolsas	Não	Não	Sim	Não	Não
Relatórios	Sim	Não	Não	Sim	Não
Gráficos	Não	Não	Não	Sim	Não

scriptLattes, que está totalmente voltado ao levantamento de colaborações, geração de relatórios com informações gerais e gráficos. O foco do ALPC é principalmente coletar os dados necessários para o relatório de avaliação do Programa de Pós-Graduação das instituições, mas ele vem como um intermediário entre as funcionalidades de gerência (SAPos's e Cobalto) e agrupamento de informações (scriptLattes).

4 ALPC

A ferramenta ALPC foi criada seguindo um processo contínuo de desenvolvimento. Neste capítulo é dada uma visão geral do funcionamento da aplicação, mostrando desde como é feita a persistência, a seleção dos dados, até os módulos e suas funcionalidades. Em seguida são apresentados os fluxos de importação e processamento do dados extraídos do currículo Lattes, bem como os relatórios que podem ser exibidos. Por último, são abordadas as principais ferramentas usadas para alcançar os objetivos do trabalho.

4.1 VISÃO GERAL

O ALPC possui duas principais fases, sendo a primeira delas a importação dos currículos da plataforma Lattes (no formato XML) para dentro do sistema. Nesse momento é feita a extração de alguns dados apenas para identificar o currículo e este é serializado e persistido no banco de dados. A segunda etapa é o processamento desses currículos, onde o currículo é carregado do banco de dados e convertido para objetos da linguagem de programação Java. Neste estágio, o currículo convertido passa por uma série de buscas e processos, a fim de evitar a duplicidade e alguma inconsistência de qualquer docente, discente ou colaboração, para que futuramente se possa extrair os relatórios sem erros de contabilização. Para realizar o processo de busca, foi montado um segundo banco de dados além da relacional, onde os dados são indexados por palavras/termos.

Dentro do sistema, cada currículo é tratado como uma pessoa para o sistema, e assim será durante todo o processo. Todas as buscas e persistência são feitas limitando-se ao programa de pós-graduação no qual o processamento está ocorrendo, ou seja, cada programa possui seus próprios dados. É importante ressaltar que se um currículo for processado para um programa e depois para outro, duas pessoas serão criadas no sistema, cada uma vinculada à seu devido programa. Assim, uma pessoa representa um currículo em um programa. Caso essa pessoa possua uma ou mais formações dentro do programa, o papel de discente lhe é conferido. Se possuir orientações no programa, lhe é dado o papel de docente. Para a avaliação sobre a orientação/formação pertencer ou não ao programa, o nome do curso é comparado com o nome do programa de pós-graduação através de uma função de similaridade.

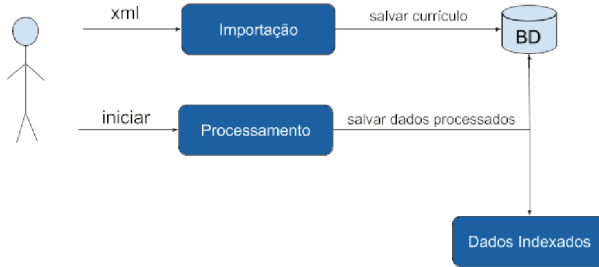


Figura 2 Visão geral da aplicação

A Figura 2 mostra uma visão geral do fluxo das funcionalidades do ALPC. A funcionalidade de importação é detalhada na Seção 4.4, o processamento na Seção 4.5, o banco de dados de indexado por palavras na Seção 4.2 e o banco de dados relacional na Seção 4.3.

4.2 PESQUISA DE TEXTO COMPLETA

Para evitar a duplicidade de informações e relacionar as devidas colaborações, o uso de funções de similaridade seria empregado entre todas as colaborações. Entretanto, se para cada colaboração extraída dos currículos fosse aplicada estas funções comparando-a com todas as outras colaborações adquiridas até o momento, o tempo de execução aumentaria significativamente conforme a quantidade de informações fosse crescendo. Assim, tornou-se necessário um filtro em cima do grande volume de dados nos quais os resultados fossem retornados em tempo aceitável, para que então as funções de similaridade fossem devidamente aplicadas. Esse processo é explicado com mais detalhes na Seção 4.5.

Pesquisa de texto completa ou *full-text search* (BERNARD et al., 2017) é uma técnica para buscar um registro utilizando o próprio conteúdo de um ou mais campos da entidade. Uma entidade pode ser um docente, um discente, uma produção ou uma orientação. A pesquisa de texto completa é dividida em duas partes: indexação e busca. Na indexação, o valor do campo é quebrado em termos, como no q-gram (Seção 2.2), para que se tornem o índice do registro. Na busca, aplica-se a mesma quebra em termos ao texto de entrada e utilizando esses termos, a consulta retorna diretamente o registro, evitando uma busca sequencial em todos os registros da base.

Por exemplo, dado o registro de duas pessoas, a pessoa 1 possui o nome João da Silva e a pessoa 2 João Souza, com o indexador separando as palavras em termos a partir dos campos, ficamos com o seguinte cenário:

Tabela 3 – Pessoa 1 e 2 indexadas

Termo	Registro
João	Pessoa 1 e 2
da	Pessoa 1
Silva	Pessoa 1
Souza	Pessoa 2

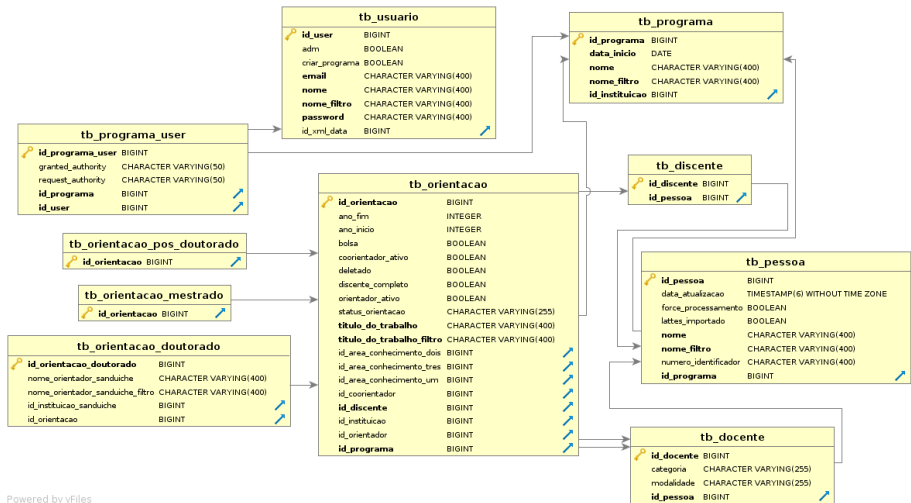
A quebra do texto em termos para cada registro do sistema é dada por uma função única e customizável. Por exemplo, para a indexação de nomes de pessoas, a função aplicada utiliza o nome completo, retira-se acentuação, todas as letras são passadas para minúsculas e os termos são quebrados com tamanho quatro.

4.3 ESQUEMA DO BANCO DE DADOS

Depois de extraídos e manipulados, os dados são persistidos em um banco de dados relacional. O banco de dados foi projetado para facilitar as consultas e a extração dos relatórios. Através de um diagrama do Modelo Lógico a Figura 3 mostra a estrutura que armazena o controle de usuários e as colaborações docente-discente e a Figura 4 a estrutura que armazena a colaboração docente-docente e a graduação.

A tabela *tb_programa* é onde são persistidos os programas de pós-graduação cadastrados pelos usuários, com uma relação n..n com a tabela *tb_usuario*, onde ficam os usuários cadastrados no sistema, através do *tb_programa_user*, já que um usuário pode estar contido em mais de um programa. Uma relação 1..n com a tabela *tb_pessoa*, que constitui a tabela que representa um currículo no sistema e essa pode ganhar um relacionamento de 1..1 com a(s) tabela(s) *tb_docente* e/ou *tb_discente*, quando a pessoa possuir o(s) papel(is) de docente e/ou discente no sistema.

As tabelas *tb_orientacao_mestrado*, *tb_orientacao_doutorado* e *tb_orientacao_pos_doutorado* são responsáveis por armazenar as colaborações de mestrado, doutorado e pós-doutorado, respectivamente, entre docentes e discentes, referenciando a tabela *tb_orientacao*, contendo os dados comuns entre todas as orientações/formações. A *tb_orientacao*



Powered by yfiles

Figura 3 – Modelo lógico do controle de usuário e colaboração docente-discente

possui uma relação de n..1 com as tabelas *tb_docente* e *tb_discente*.

Para as produções, foi criada a tabela *tb_produção*, que tem relação 1..n com a tabela *tb_docente_produção*, que se relaciona com a *tb_docente* com n..1. Aqui é salva a colaboração entre docentes nas produções. Uma tabela *tb_autor* é utilizada para relacionar os n autores a cada produção. Os projetos de pesquisa têm o mesmo comportamento que as produções, através das tabelas *tb_projeto_pesquisa* e *rl_projeto_pesquisa_docente*, com seus devidos autores.

As linhas de pesquisa são armazenadas na *tb_linha_pesquisa*, que por meio da *rl_linha_pesquisa_docente*, faz uma relação n..n com a *tb_docente*. As atuações profissionais possuem relação n...1 com os docentes e são salvas na tabela *tb_atuação_profissional*.

4.4 IMPORTAÇÃO

Esta seção descreve como é realizado o processo de importação de um currículo para o sistema. Como mostrado na Figura 5, ela constitui em uma funcionalidade com poucos passos, onde em um primeiro momento o currículo é convertido do XML para um objeto da linguagem Java, para então ser salvo no banco de dados e futuramente ser

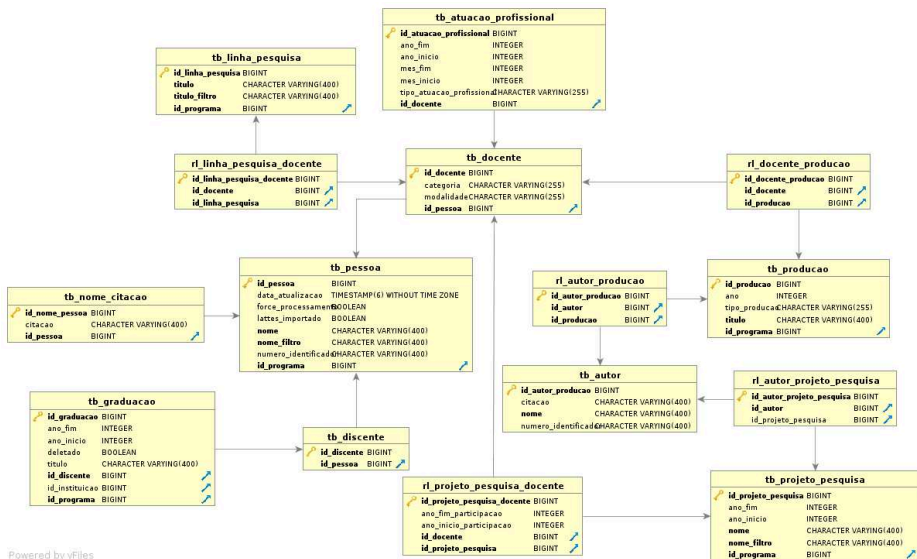


Figura 4 – Modelo lógico da graduação e colaboração docente-docente processado. A Figura 6 mostra a sua tela no sistema:

- Extrair dados

A primeira parte do processo, constitui em extrair os dados do currículo Lattes que estão no formato XML e colocá-los em objetos da linguagem Java, facilitando assim a manipulação desses dados. Para que a conversão aconteça, foi realizado um mapeamento de cada elemento do currículo em formato XML para um objeto correspondente na linguagem Java. Esse conversor pode ser utilizado para a conversão em ambas direções, de XML para objetos da linguagem e vice-versa. O mapeamento foi desenvolvido com o auxílio de uma biblioteca existente, chamada JAXB (Java Architecture for XML Binding), que é explicado na Seção 4.7. A utilização desta biblioteca dispensa o uso de qualquer algoritmo para extração dos dados, necessitando apenas o próprio mapeamento descrito e o carregamento do arquivo XML com os dados em memória.

- Converter dados

Na última etapa do processo de importação de um currículo para o sistema, alguns dados são extraídos do currículo, com o objetivo

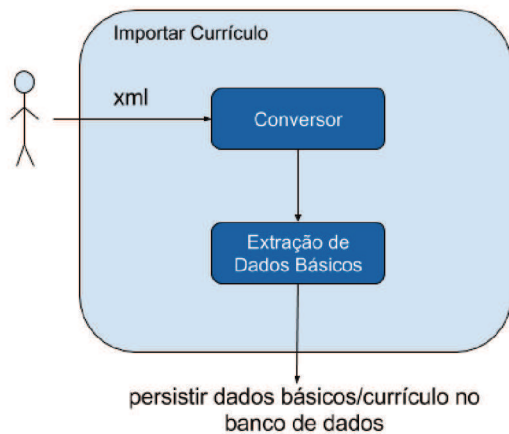


Figura 5 Importar currículo

ALPC PROGRAMA DOENTE DISCENTE IMPORTAR USUÁRIOS Alisson Brarenstein Aloni

Importe aqui os currículos do programa, utilize formato xml ou zip

IMPORTAR CURRÍCULO

Importe seu currículo para utilização em qualquer programa que participe

IMPORTAR SEU CURRÍCULO

Nome

mm/dd/yyyy mm/dd/yyyy

mm/dd/yyyy mm/dd/yyyy

Nome	Data importação	Data atualização		
Alko von Wengenheim	2017-04-26	2017-02-02	↕	🗑
Antonio Augusto Medeiros Freitas	2017-04-26	2017-04-24	↕	🗑
Carina Friedrich Domeles	2017-04-26	2017-03-27	↕	🗑
Carla Monika Westphal	2017-04-26	2017-03-14	↕	🗑
Carlos Becker Westphal	2017-04-26	2017-04-24	↕	🗑

1 2 3 4 5 6 >

EXCLUIR DADOS
PROCESSAR TODOS

Figura 6 Tela de importação

de apresentar ao usuário o currículo importado e identificá-lo, tanto para o usuário quanto ao sistema. Os campos são: número identificador (número que identifica unicamente cada currículo existente na plataforma Lattes), nome completo (nome da pessoa a qual o currículo se refere) e data de atualização (última vez que esse currículo foi atualizado na plataforma Lattes). Após a extração desses dados, o currículo XML é serializado e salvo no banco de dados para futuramente ser processado.

4.5 PROCESSAMENTO

Para otimizar todo o processo de importar currículos e gerar relatórios a partir de suas informações, foi adotada a estratégia de manipular os dados de cada currículo uma única vez (aceitando posteriores atualizações), processá-los e persisti-los em um novo modelo de dados, diminuindo o tempo total a cada nova solicitação de processamento feita pelo usuário, além de facilitar todo o processo interno. Dessa forma, o usuário pode consultar as informações requeridas rapidamente e de forma customizada aplicando filtros. Lembrando que embora o processamento possa ser lento, pois ocorre em segundo plano e não necessita de qualquer envolvimento com o usuário, deseja-se que seja feito em tempo viável e que tenha boa escalabilidade. Portanto, é preciso que o processo leve apenas alguns minutos. Para casos onde ocorra apenas atualização do currículo, a função deve ocorrer de forma mais rápida ainda. O processamento de um currículo é dividido em passos, explicados a seguir. A Figura 7 mostra o seu fluxo.

4.5.1 Buscar Currículo

Como primeira parte do processamento de um currículo, é necessário verificar se uma pessoa equivalente aquele currículo já foi importada anteriormente a fim de evitar duplicidades e/ou esforço desnecessário. Caso encontrado sua data de atualização é avaliada: se a data de atualização do currículo no sistema é mais antiga que a data de atualização do novo currículo, então o novo currículo é processado e seus dados extraídos. Isso indica que a pessoa que está no banco de dados está desatualizada. Para os casos onde encontra-se uma pessoa no sistema com data de atualização igual ou maior ao novo currículo, nada será feito.

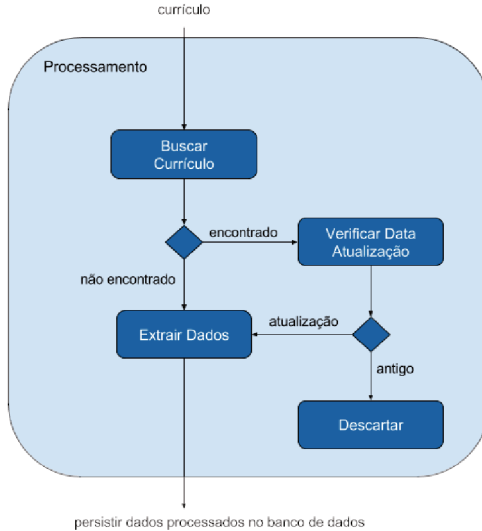


Figura 7 Visão Geral do Processamento

Caso um currículo equivalente não seja encontrado no processo de busca, uma nova pessoa é criada para sua representação. Posteriormente, são atualizados os campos identificadores do currículo: nome completo, número identificador e data de atualização.

O processo de busca do currículo pode ser dividido em três partes. Na Figura 8, é possível visualizar esse fluxo de busca. Os passos são os seguintes:

1. Por número identificador

A primeira busca é simples e feita com correspondência total pelo número identificador do próprio currículo Lattes. É aplicada diretamente no banco de dados relacional, caso haja uma pessoa representando esse currículo. Isso acontece quando um currículo foi processado previamente, com seu número identificador preenchido. Existe também a possibilidade de um orientador preencher o número identificador do orientando dentro de sua orientação ou o discente preencher o número de seu orientador dentro de sua formação. Este passo não é o único pois o número identificador não é obrigatório, então nem sempre está preenchido.

2. Por formação/orientação

Caso não seja encontrada uma pessoa com o número identificador do currículo, é feita uma busca usando as formações e orientações contidas no currículo. Isso ocorre pelo fato de que uma orientação/ formação é uma colaboração, e a mesma colaboração pode ter sido importada por outro currículo. Primeiramente, uma busca é feita utilizando a técnica de pesquisa de texto completa com o título da colaboração e o nome do orientando. Os resultados retornados da busca tem o orientador e/ou coorientador e o discente comparados com a funções de similaridade Jaro-Winkler, juntamente com a comparação do título da colaboração com a distância Q-Gram.

3. Por nome da pessoa

Caso as buscas anteriores falhem, em último caso, uma busca no banco de palavras/termos indexados é feita usando apenas o nome completo da pessoa. Os resultados são comparados com a função de similaridade Jaro-Winkler. Se nada for encontrado, provavelmente este currículo ainda não possui uma pessoa que o represente no sistema ou está incompleto.

4.5.2 Extração dos dados

Um dos principais desafios, é evitar a duplicidade de dados e fazer o correto relacionamento docente-discente e docente-docente através das colaborações. Por exemplo, o núcleo para o relacionamento entre docente-discente (ou orientador-orientando), no currículo, está dentro das orientações para o docente e nas formações para o discente. Estas são as principais informações utilizadas para a busca e ligação entre o docente e discente. Se o currículo do discente for importado anteriormente ao de seu(s) orientador(es), e suas devidas informações persistidas, ao importar, posteriormente, o currículo do(s) docente(s), suas orientações devem ser comparadas às formações do discente, já persistidas, evitando assim a duplicação da(s) colaboração(ões). Caso o currículo do docente seja importado primeiramente, ao importar o currículo do(s) discente(s) que tem como orientador este docente, suas formações devem ser comparadas às orientações, já persistidas, para suas devidas correspondências. Portanto, a extração de dados do currículo deve levar em consideração a falta de ordem em que os currículos foram importados e processados no sistema, mantendo sempre o mesmo resultado.

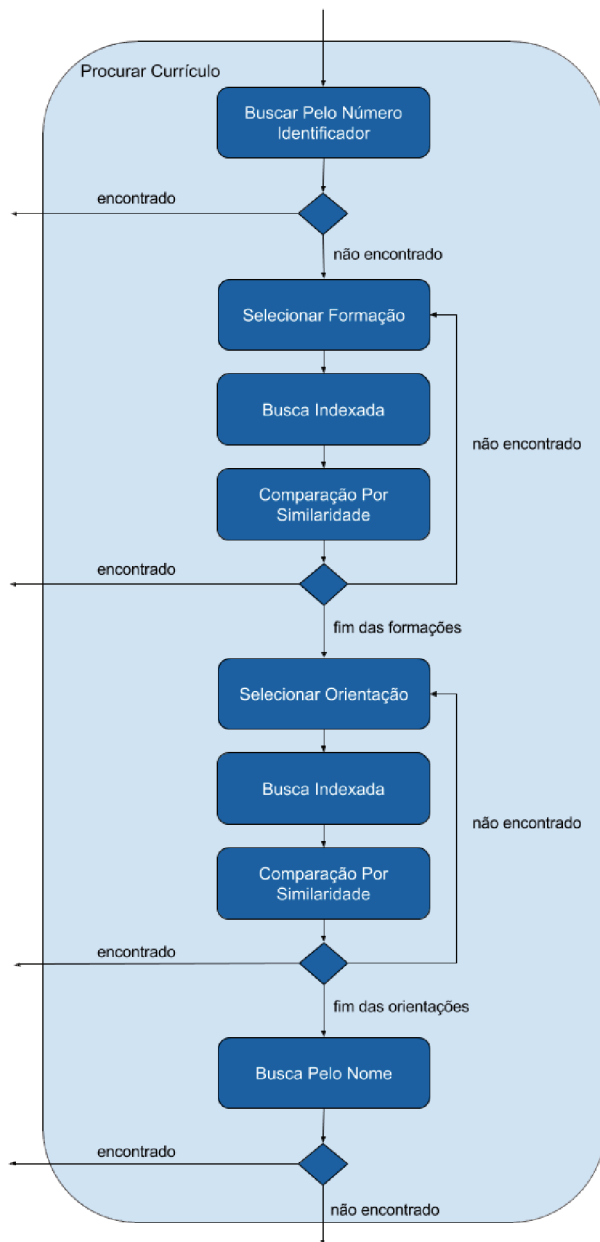


Figura 8 Fluxo realizado para procurar um currículo

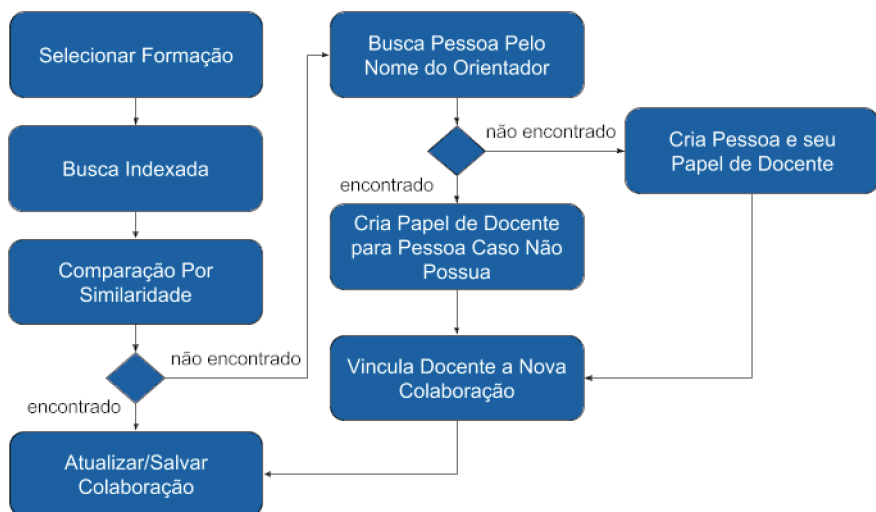


Figura 9 Lógica para salvar uma formação

A seguir é detalhado cada um dos passos executados de extração de dados do currículo usados na elaboração dos relatórios.

4.5.2.1 Formação

O processo de persistência inicia com as formações de mestrado, doutorado e pós-doutorado do currículo, seguindo uma busca similar à busca de formações/orientações explicada na Seção 4.5.1 item 2. Caso a colaboração seja encontrada, esta é atualizada. Se não, uma nova colaboração é criada e, equivalente a busca usando o nome da pessoa na Seção 4.5.1 item 3, o docente orientador da formação é procurado. Caso não seja encontrado, significa que o currículo do orientador da formação ainda não foi importado. Uma pessoa com o papel de docente é criada no programa, utilizando o nome e o número identificador, preenchidos na formação. Isso é feito para que futuramente, na importação do currículo do docente relacionado a esta formação, seja vinculado corretamente. A falta de obrigatoriedade do número identificador é o que traz complexidade ao processo de vincular docente-discente, pois ao importar o currículo deste docente posteriormente, não há uma ma-

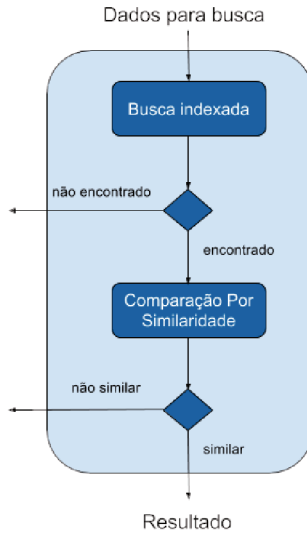


Figura 10 Fluxo básico de busca para colaboração docente-docente

neira absoluta de identificá-lo na base. Esse fluxo pode ser visualizado na Figura 9.

4.5.2.2 Orientação

Após salvas as formações, são persistidas as orientações de mestrado e doutorado (caso o currículo contenha alguma). O processo é similar ao de salvar uma formação, com a diferença que, para as orientações não encontradas, é realizada uma busca adicional e possivelmente persistência do discente orientando. Caso encontrada uma colaboração relativa a esta orientação, que foi salva através de uma formação do discente, a orientação não atualiza nenhum campo dessa colaboração, ou seja, os dados de uma formação sempre terão preferência sobre os dados de uma orientação.

4.5.2.3 Produção

Finalizado o processo das orientações/formações e os papéis atribuídos, caso o currículo possua o papel de docente, são também extraídas as informações sobre suas produções bibliográficas e produções técnicas. É neste processo que será encontrada a colaboração docente-docente. Primeiramente, é feita uma busca no banco de termos indexados usando o título da produção, ano, tipo da produção e nome do autor, nomes em citações e o número identificador, estes três últimos vindos dos dados gerais do currículo a ser importado. A partir dos resultados retornados, o título e os dados referentes ao autor são comparados com funções de similaridade e, caso não haja resultados compatíveis, a nova informação é persistida.

Toda produção possui seus autores e estes são persistidos separadamente e relacionados à produção para que aumente a eficácia da busca, não se atendo apenas aos dados da produção. Na busca no banco de palavras/termos indexados, ao menos um dos campos do autor (nome completo, nome da citação ou número identificador) e todos os campos da produção devem conter ao menos um termo equivalente. O fluxo pode ser visualizado na Figura 10.

Não há distinção na extração dos dados de produção bibliográfica e técnica, pois possuem o mesmo conjunto de informações. São classificadas por tipo de produção, que se refere a especificação dentro das produções. Para as produções bibliográficas, são considerados:

- Artigos publicados
- Artigos aceitos
- Trabalhos em eventos
- Livros
- Capítulos
- Textos em jornais
- Revistas
- Demais produções

Para produções técnicas considera-se:

- Cultivar registrada

- Cultivar protegida
- Desenho industrial
- Marca
- Patente
- Processo técnico
- Produto tecnológico
- Software
- Topografia
- Trabalho técnico
- Apresentação de trabalho
- Carta ou mapa
- Curso de curta duração
- Desenvolvimento de material didático
- Editoração
- Manutenção de obra artística
- Maquete
- Mídia social
- Organização de evento
- Programa de rádio ou TV
- Relatório de pesquisa e outra produção técnica

4.5.2.4 Atuação profissional

As atuações profissionais do docente não são colaborativas, ou seja, não possuem nenhum vínculo com outros docentes ou discentes. Sendo assim, não existe a necessidade de buscas e funções de similaridade para evitar duplicidade. As informações já persistidas (caso existam) sempre são deletadas para dar lugar às novas informações. Para as pessoas que possuem o papel de docente, as suas atuações profissionais realizadas na instituição do programa de pós-graduação são simplesmente salvas e vinculadas à pessoa do currículo sendo importado. Cada atuação tem seu tipo, que são:

- Conselho, comissão ou consultoria
- Direção ou administração
- Ensino
- Extensão universitária
- Participação em projeto
- Pesquisa e desenvolvimento
- Serviço técnico especializado
- Treinamento ministrado
- Outra atividade

4.5.2.5 Linha de pesquisa

As linhas de pesquisa são um subgrupo das pesquisas e desenvolvimento, que estão contidas nas atuações profissionais. Sendo assim, são processados apenas linhas de pesquisa que satisfizerem as condições descritas na Seção 4.5.2.4. Linhas de pesquisa são colaborativas entre docentes e sua busca é dada apenas pelo seu título. Caso não encontrada, a nova linha de pesquisa é persistida. Linhas de pesquisa são distintas dentro de um programa de pós-graduação. O fluxo pode ser visualizado na Figura 10.

4.5.2.6 Projeto de pesquisa

Os projetos de pesquisa, que estão contidos nas participações em projeto das atuações profissionais. Assim, são processados apenas projetos de pesquisa de atuações profissionais que satisfizerem as condições descritas na Seção 4.5.2.4. O processo de busca e persistência de um projeto de pesquisa é semelhante ao processo descrito na Seção 4.5.2.3, diferenciando alguns dos campos utilizados na pesquisa: retira-se o tipo de produção a adiciona um campo do tipo ano, resultando em ano início e ano fim.

4.5.2.7 Graduação

A graduação não foi importada de forma colaborativa, pois ela é utilizado para verificar a procedência dos discentes, ou seja, não é necessário o vínculo docente-discente, apenas o vínculo com o discente sendo importado.

4.6 RELATÓRIOS

Os relatórios são uma parte de extrema relevância da aplicação e são organizados de forma simples e compacta, agrupados por sua amplitude: informações sobre o programa, docentes e discentes.

Na Figura 11 mostra a tela onde está a maioria dos dados referente ao programa, nessa tela existem subgrupos, sendo estes: atividades acadêmicas, linha de pesquisa, participação em projetos, procedência, produção bibliográfica e produção técnica. A procedência é classificada por serem as universidades onde os discentes estiveram anteriormente ao ingresso do programa.

Para os grupos atividades acadêmicas, produção bibliográfica e produção técnica, são listados os seus respectivos tipos, atrelando a cada um sua quantidade, calculada para todo o programa. Há a possibilidade de filtragem por período entre anos. Para os grupos projeto, linha de pesquisa e procedência, os dados estão organizados em forma de lista e retornam resultados relacionados ao programa, e para cada um uma quantidade de docentes em projeto e linha de pesquisa e discentes em procedência. A linha de pesquisa tem como filtro opcional seu título. Já para projetos de pesquisa e procedência, os filtros são dados pelo nome e período por ano.

ALPC

PROGRAMA | DOCENTE | DISCENTE | IMPORTAR | USUÁRIOS | Fomando Bolog

Atividades acadêmicas | Linha de pesquisa | Participação em projetos | Presidência | Produção bibliográfica | Produção técnica

Produção bibliográfica

Ano de início: _____ Ano de fim: _____

Produção	Quantidade
Artigos publicados	534
Artigos aceitos para publicação	11
Trabalho em eventos	2361
Livros publicados ou organizados	89
Capítulos de livros publicados	115
Traduções	0
Partituras	0
Profusão e posafino	1
Textos em jornais ou revistas	20
Outras produções	86

Figura 11 Tela de relatórios do programa

ALPC

PROGRAMA | DOCENTE | DISCENTE | IMPORTAR | USUÁRIOS | Fomando Bolog

Nome do orientador: _____
 Área de conhecimento: _____
 Ano de início: _____ Ano de fim: _____
 < Mostrar mais filtros >

Número de bolsistas PI: S: 0 / IB: 0 / IA: 0 / IC: 0 / ID: 0 / Z: 0
 Número de bolsistas DI: TA: 0 / ID: 0 / IC: 0 / ID: 0 / Z: 0
 Número orientadores de mestrado: 26
 Número orientadores de doutorado: 16

Aldo von Wangenheim

Defina tipo de bolsa

Em andamento		Concluído	
Discentes de mestrado	2	Discentes de mestrado	44
Discentes de doutorado	3	Discentes de doutorado	1
Total de discentes	5	Total de discentes	45

< Mostrar mais estatísticas >

Antonio Augusto Medeiros Fröhlich

Defina tipo de bolsa

Em andamento		Concluído	
Discentes de mestrado	2	Discentes de mestrado	22
Discentes de doutorado	2	Discentes de doutorado	1
Total de discentes	4	Total de discentes	23

Figura 12 Tela de relatórios do docente

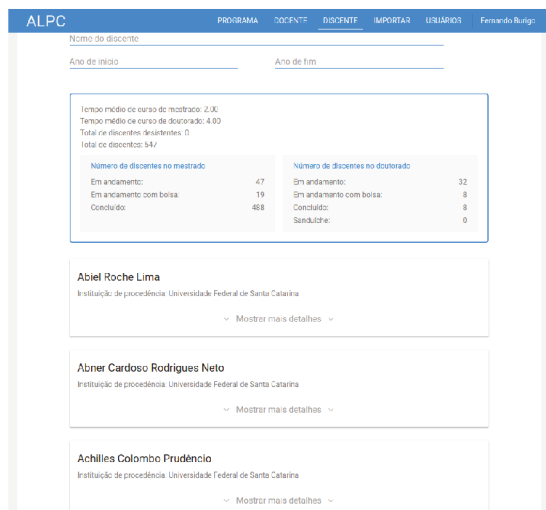


Figura 13 Tela de relatórios do discente

No tela dos docentes mostrado na Figura 12 é exibido um resumo com número de bolsistas e número de orientadores de mestrado e doutorado. Juntamente, há uma listagem de todos os docentes do programa. Cada docente é complementado com as informações sobre as orientações em andamento e concluídas, possibilitando a visão detalhada de cada orientação, além de um resumo com quantidade de discentes de mestrado, quantidade de discentes de doutorado e total. É possível também uma visão detalhada sobre o docente, como formações e o quantidade de produções bibliográficas. O filtro aplicado à listagem está entre nome do orientador, área do conhecimento e período por ano.

A Figura ?? mostra o grupo dos discentes que segue a diagramação dos docentes. O resumo apresenta informações relacionadas ao programa, como tempo médio de curso, total de discentes desistentes, total de discentes e números de discentes em mestrado ou doutorado, tanto em andamento quanto concluído. Dentro da listagem, cada item exhibe o tipo de orientação e seu ano, com possível detalhamento de informações da formação do discente.

Desses três grupos se obtêm vários dados cobrados pelo relatório da Capes, do programa é possível visualizar a quantidade de cada tipo das produções técnicas e atividades acadêmicas, a quantidade de docentes em linhas de pesquisa, projetos de pesquisa, quantos são orientadores de mestrado e/ou doutorado, quantos discentes estão regularmente

matriculados, defenderam suas teses, estão em doutorado sanduíche e quantos são bolsistas. Já dos docentes temos sua formação completa com a quantidade de produções e discentes e todas as suas orientações. Nos discentes temos sua formação, sua instituição de procedência e ano de entrada no curso. Como os relatórios da Capes são anuais, todos os dados possuem filtro por período, assim conseguindo visualizar os dados ano a ano.

4.7 FERRAMENTAS, BIBLIOTECAS E UTILITÁRIOS

Esta seção abordada toda a base necessária para o desenvolvimento da aplicação, com uma breve explicação sobre a utilidade de cada uma. Todas as ferramentas, bibliotecas e utilitários empregados no lado do servidor (Eclipse, JAXB, Spring, REST e Lucene), no cliente (Gulp, ECMAScript 6) e também para o conjunto (REST, GIT).

4.7.1 Eclipse

Eclipse¹ é uma IDE para desenvolvimento de software, que neste trabalho foi utilizado com a linguagem Java. Contém um espaço de trabalho (workspace) e um sistema de plugins extensíveis para personalizar o ambiente. É escrito em sua maioria na linguagem Java e sua principal utilização é para desenvolvimento de aplicativos Java, mas também existem versões para desenvolver aplicações em outras linguagens de programação, como C e C++, ou até mesmo através do uso de plugins.

4.7.2 JAXB

JAXB² é uma biblioteca que permite mapear classes Java para representações XML. JAXB oferece duas características principais: a capacidade de converter objetos Java em XML e o inverso, ou seja, o XML volta a ser objeto Java. Em outras palavras, JAXB permite armazenar e recuperar dados na memória em qualquer formato XML, sem a necessidade de implementar um conjunto específico de rotinas de carregamento e persistência de XML para a estrutura de classes do programa. Dentro do trabalho foi utilizado na conversão do currículo

¹<https://eclipse.org>

²<https://jaxb.java.net>

Lattes para objetos do Java, para a manipulação dos dados de maneira rápida e eficaz.

4.7.3 Spring Framework

O Spring³ é um framework para aplicações e inversão de controle (IoC), sendo o IoC um meio consistente de configuração e gerenciamento de objetos Java usando a reflexão. Existe um responsável por gerenciar os ciclos de vida dos objetos: criar esses objetos, chamando seus métodos de inicialização e configurar ligando-os, que é chamado de contêiner. Embora o framework não impõe qualquer modelo de programação específica, tornou-se popular na comunidade Java como uma alternativa de substituição para o modelo EJB . Um ponto importante é que ele tem suporte a servlet, que vem da ideia de um pequeno servidor (servidorzinho, em inglês), cujo objetivo é receber chamadas, normalmente do protocolo HTTP, processá-las e retorná-las ao cliente. Também fornece extensão e customização para aplicações web e serviços REST. Foi o framework escolhido para o desenvolvimento da aplicação deste trabalho, utilizando o protocolo REST para a comunicação cliente-servidor.

4.7.4 Apache Lucene

Apache Lucene é uma biblioteca para recuperação de informações, para aplicações que requerem pesquisa de texto completo. Apache é um servidor que é distribuído sob uma licença de código aberto. Compreende quatro componentes principais acerca dos textos: indexação, busca, verificação ortográfica, chance de acerto na busca e uso de tokens. Seu uso neste trabalho é para a indexação das informações.

4.7.5 Node.js

Node.js é um ambiente de execução multi-plataforma para o desenvolvimento de aplicações web, do lado do servidor. Fornece uma arquitetura orientada a eventos e uma API de I/O sem bloqueio, projetado para otimizar a vazão e escalabilidade para aplicações web em tempo real. Ele usa motor V8 JavaScript do Google para executar có-

³<http://projects.spring.io/spring-framework/>

digo, e uma grande porcentagem dos módulos básicos são escritos em JavaScript. Utilizado na compilação da aplicação do trabalho.

4.7.6 Gulp

Gulp⁴ é um sistema para construção de aplicações, e sua função é automatizar tarefas comuns no desenvolvimento. É construído sobre o Node.js, e tanto a sua origem quanto seu arquivo, onde são definidas as tarefas, são escritos em JavaScript. É possível escrever tarefas para detectar erros e potenciais problemas no código JavaScript e CSS, analisar seus modelos e etc. O Gulp é apenas o gerenciador, os plugins disponíveis é que são os responsáveis pelas tarefas. Usado neste trabalho para tarefas de gerências de builds e análise do código.

4.7.7 ECMAScript 6

ECMAScript⁵ é a especificação de linguagem de script padronizada pela Ecma International. Implementações bem conhecidas da linguagem, tais como JavaScript, JScript e ActionScript são amplamente utilizados para execução de scripts do lado do cliente na Web. Suporta programação orientada a objetos, além de outras propriedades: as funções são objetos, os objetos têm protótipos, construtores "pré-compilados", etc. Linguagem utilizada para os diversos scripts executados no lado do cliente da aplicação.

4.7.8 Babel

Infelizmente não se pode garantir que os browsers suportarão a versão 6 do ECMAScript, portanto é necessário fazer um "rebaixamento" para que a aplicação funcione garantidamente. Babel⁶ é um compilador JavaScript, especificamente um compilador fonte-a-fonte, muitas vezes chamado de "transpiler" (compilador/tradutor). Isto significa que ele analisa o código, modifica-o e gera o novo código de volta para a outra versão. Como a versão 6 do ECMAScript é aplicada neste trabalho, tornou-se necessário o uso do Babel.

⁴<http://gulpjs.com/>

⁵<http://www.ecmascript.org/>

⁶<https://babeljs.io/>

4.7.9 React

React⁷ é uma biblioteca JavaScript para criação de interfaces. Visa um modelo em que subcomponentes não podem afetar diretamente componentes anexos, codificação eficiente para a atualização quando dados são alterados e uma clara separação entre os componentes de uma aplicação. Gerencia automaticamente todas as atualizações de interface do usuário quando há alterações de dados subjacentes, e conceitualmente atinge o botão de "refresh", atualizando somente os componentes alterados, evitando o carregamento da página inteira. Empregado na aplicação deste trabalho justamente pelo ganho de tempo e diminuição de operações necessárias ao atualizar uma página do sistema.

4.7.10 REST

REST(W3C, 2011) é uma arquitetura de software para Web. Fornece um conjunto coordenado de restrições para o design de componentes em um sistema, com alta performance e velocidade de resposta. Sistemas que utilizam esta arquitetura podem ser chamados de RESTful. Normalmente, sua comunicação é baseada no protocolo HTTP, com os mesmos comandos (GET, POST, PUT, DELETE), que os navegadores da Web usam para recuperar páginas e enviar dados para servidores remotos. Protocolo usado na comunicação servidor-cliente deste trabalho por sua simplicidade e velocidade.

4.7.11 Git e Github

Git⁸ é um sistema de controle de versão distribuído e de gerenciamento de código fonte. Cada diretório de trabalho do Git é um repositório com um histórico completo e habilidade total de acompanhamento das revisões, não dependente de acesso a uma rede ou a um servidor central. GitHub⁹ é um serviço de hospedagem compartilhado para projetos que usam o Git. Possui funcionalidades de uma rede social como feeds, seguidores, wiki e alguns gráficos, com a história de commits, contribuições, etc. Utilizado para o desenvolvimento da

⁷<https://facebook.github.io/react/>

⁸<https://git-scm.com/>

⁹<https://github.com/>

aplicação deste trabalho principalmente pela familiaridade dos desenvolvedores com a ferramenta, além de todas as suas facilidades.

4.7.12 Maven

Maven ¹⁰ é uma ferramenta de gerenciamento de projetos de software. Baseado no conceito de um modelo de objeto do projeto (POM), pode gerenciar a compilação do projeto, elaboração de relatórios e outras características. Um arquivo XML (o POM) é quem descreve o projeto que está sendo compilado, as suas dependências a outros módulos externos e componentes, a ordem de compilação, diretórios e plugins necessários. Ele baixa dinamicamente as bibliotecas Java e os plugins a partir de um ou mais repositórios, como o Maven 2 Central Repository, armazenando-os em uma cache local. Também empregado no desenvolvimento da aplicação pela afinidade dos desenvolvedores com a ferramenta, que em um projeto deste porte se torna fundamental.

4.7.13 PostgreSQL

Para o banco de dados relacional, foi utilizado o SGBD (Sistema de Gerenciamento de Banco de Dados) Postgresql (POSTGRESQL, 2017). O PostgreSQL é um sistema de banco de dados objeto-relacional de código aberto. Possui mais de 15 anos de desenvolvimento ativo e uma arquitetura comprovada que lhe confere uma forte reputação de confiabilidade, integridade de dados e correção. Ele é executado em todos os principais sistemas operacionais, incluindo Linux, UNIX (macOS) e Windows. Inclui a maioria dos tipos de dados SQL:2008, como *INTEGER*, *NUMERIC*, *BOOLEAN*, *CHAR*, *VARCHAR*, *DATE*, *INTERVAL* e *TIMESTAMP*. Também suporta armazenamento de objetos grandes binários, incluindo imagens, sons ou vídeo. Ele tem interfaces de programação nativas para C/C++, Java, .Net, Perl, Python, Ruby, entre outros.

¹⁰<https://maven.apache.org/>

5 CONCLUSÕES E TRABALHOS FUTUROS

O ALPC é uma ferramenta projetada para a melhoria da gerência, a qualidade e a eficiência dos cursos de pós-graduação, extraíndo e organizando as informações do currículo Lattes. Todas as suas funcionalidades foram desenvolvidas com ênfase no público alvo. O principal objetivo do trabalho, é reunir as informações necessárias e relatá-las ao usuário de acordo com os currículos em que se desejava obter estes dados, que não estão explícitos, foi atingido.

Apesar de todo o processo ter foco nas similaridades, esperando então que as informações extraídas entre colaborações não estejam totalmente equivalentes, ainda existe uma margem de erro em casos nos quais as informações não contenham o mínimo de aproximação. Infelizmente, isso depende total e exclusivamente da forma com que os currículos tenham sido preenchidos.

Mesmo com problemas no desenvolvimento, alcançou-se uma boa base para a continuação do trabalho, onde foi desenvolvida uma biblioteca que já realiza a conversão de todo o currículo Lattes, facilitando assim futuramente a extração de outros dados.

Sugestões de trabalhos futuros são:

- A validação dos currículos ao importá-los para dentro do sistema. Verificar se todos os dados necessários estão preenchidos de forma adequada;
- A extração de mais dados do currículo Lattes, como cooperações nacionais e internacionais dos docentes do programa, prêmios, entre outros, trazendo assim uma maior aproximação da total completude dos relatórios exigidos pela Capes;
- A disponibilização de módulos onde um usuário possa editar os resultados obtidos do processamento e assim consertar algum possível erro, originado tando do processamento quanto de um currículo incorreto e não precisar importar um novo currículo para o sistema e ter que fazer um processamento parcial ou completo das informações.

REFERÊNCIAS

- BERNARD, E. et al. *Hibernate Search*. 2017. <https://docs.jboss.org/hibernate/search/5.7/reference/en-US/pdf/hibernate_search_reference.pdf>. Acessado em 10/02/2017.
- CAPES. *Avaliação do Sistema Nacional de Pós-Graduação*. 2017. <<http://capes.gov.br/avaliacao/sobre-a-avaliacao>>. Acessado em 03/03/2017.
- COHEN, W. W.; RAVIKUMAR, P.; FIENBERG, S. E. A comparison of string distance metrics for name-matching tasks. *Proceedings of the workshop on Data Cleaning and Object Consolidation at the International Conference on Knowledge Discovery and Data Mining*, 2003.
- CONSCIENTIAS/LMPL. *DTD (Data Type Definition) do Currículo Lattes*. 2014. <<http://lmpl.cnpq.br/lmpl/Gramaticas/Curriculo/DTD/Documentacao/DTDC>>. Acessado em 20/08/2016.
- DORNELES, C. F.; GALANTE, R. Aplicação de funções de similaridade e detecção de diferenças em grandes volumes de dados distribuídos. *Jornadas de Atualização em Informática*, 2008.
- FABRO, M. D. D.; ALMEIDA, E. C. de; SLUZARSKI, F. Teaching web application development: a case study in a computer science course. *Informatics in Education*, v. 11, p. 29–44, 2012.
- FERREIRA, R. D.; AMARO, T. M. P. *SAPOS - Sistema de apoio à pós-graduação*. Tese (Doutorado) — Universidade Federal Fluminense, 2013.
- JURAFSKY, D.; MARTIN, J. H. *Speech and Language Processing*. [S.l.]: Draft of September 1, 2014.
- LEVENSHTAIN, V. I. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 1966.
- MENA-CHALCO, J. P.; JUNIOR, R. M. C. scriptlattes: An open-source knowledge extraction system from the lattes platform. *Journal of the Brazilian Computer Society*, v. 15, p. 31–39, 2009.

NAVARRO, G.; SUTINEN, E.; TARHIO, J. Indexing text with approximate q-grams. *In Proceedings of the 11th Annual Symposium on Combinatorial Pattern Matching (CPM'2000)*, 2000.

POSTGRESQL. *Postgresql*. 2017.

<<https://www.postgresql.org/docs/current/static/pgtrgm.html>>.

Acessado em 07/01/2017.

RASMUSSEN, K. R.; STOYE, J.; MYERS, E. W. Efficient q-gram filters for finding all epsilon-matches over a given length. *J. Comput. Biol*, 2006.

UKKONEN, E. Approximate string-matching with q-grams and maximal matches. *Theoretical Computer Science*, v. 92, p. 191–211, 1992.

W3C. *REST*. 2011. <<https://www.w3.org/2001/sw/wiki/REST>>.

Acessado em 15/04/2016.

W3C. *Extensible Markup Language (XML)*. 2015.

<<http://www.w3.org/XML/>>. Acessado em 05/08/2016.

APÊNDICE A – Artigo

ALPC - Aplicação para Levantamento de Participação Científica

Alisson Granemann Abreu¹, Fernando Burigo Teixeira¹

¹Departamento de Informática e Estatística
Universidade Federal de Santa Catarina (UFSC)
Florianópolis, SC, Brasil.

alissongranemannabreu@gmail.com, fernandobt8@gmail.com

Abstract. *Each year, Capes - Higher Education Personnel Improvement Coordination - conducts evaluations of the country's graduate courses. This activity is done through the production of reports with scientific production indicatives. With professor-professor and professor-student cooperation. All this data collection is done manually by graduate courses program managers, who analyze the resumes of their students and professors, collecting the information needed to complete the reports. The present work intends to facilitate this manual process, offering a tool on the Web, in which all data collection is done in an automated way.*

Resumo. *Anualmente, a Capes - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - realiza avaliações dos Programas de Pós-Graduação do país. Essa atividade é feita através da realização de relatórios com indicativos da produção científica, com a cooperação docente-docente e docente-aluno. Todo esse levantamento de dados é feito de forma manual pelos gestores de programas de pós-graduação, que analisam os currículos de seus discentes e docentes, coletando as informações necessárias para preencher os relatórios. O presente trabalho tem como intenção facilitar esse processo manual, oferecendo uma ferramenta Web, na qual toda a coleta de dados é feita de forma automatizada.*

1. Introdução

Anualmente, a Capes - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - realiza avaliações dos PPGs - Programas de Pós-Graduação, com a participação da comunidade acadêmica-científica através de consultores destinados a essa finalidade. A avaliação é um meio de assegurar e manter a qualidade dos cursos de Mestrado e Doutorado no país. A avaliação é baseada, essencialmente, nos dados inseridos na Plataforma Sucupira, uma ferramenta desenvolvida pela Capes e UFRN, com o objetivo de coletar informações, realizar análises e avaliações e ser a base de referência do Sistema Nacional de Pós-Graduação (SNPG). A partir dos dados inseridos, a plataforma gera um relatório, que é utilizado como base para a avaliação. Um dos pontos considerados nesta avaliação é a análise da colaboração científica entre docente-docente e/ou docente-discente, ou seja, como os docentes colaboram entre si e como os estudantes estão envolvidos nestes trabalhos. Além dessa colaboração, a participação

discente em projetos do orientador, prêmios recebidos, etc, também são itens importantes.

2. Problematização

Todas as informações são mantidas, individualmente, por docentes e discentes em seus currículos cadastrados na Plataforma Lattes. A plataforma Lattes é uma sistema web que disponibiliza uma ferramenta para montagem e armazenamento de currículos. Toda produção discente/docente é incluída em seus devidos currículos, com todas as informações relevantes. Para alimentar certos campos da plataforma Sucupira e gerar estatísticas prévias para planejamentos estratégicos de PPGs, seus gestores devem analisar cada currículo dos integrantes do Programa, avaliando todas as produções, para então relacionar docente/discente a cada trabalho.

Uma solução para isso é a construção de alguma plataforma computacional que realiza o cruzamento dos dados dos currículos dos participantes de um Programa de Pós-Graduação e gera relatórios baseados no agrupamento de informações relevantes à avaliação da CAPES, como número de produções bibliográficas e técnicas, participações em projetos, levando em consideração a colaboração entre pares.

3. Solução

O ALPC (Aplicação para Levantamento de Participação Científica), uma plataforma que tem como principal finalidade vincular toda e qualquer colaboração científica, entre docente e discente, através de seus respectivos currículos Lattes, analisando e processando suas informações, através da interação com uma aplicação Web.

O grande desafio no desenvolvimento de tal plataforma é a heterogeneidade dos dados presentes nos currículos Lattes. Apesar dos currículos estarem em formato XML, que provê interoperabilidade, os dados inseridos podem ter valores distintos para a mesma informação. Por exemplo, uma colaboração está contida em uma orientação, de mestrado a doutorado, e é a partir das suas informações que a relação é estabelecida. Sendo assim, um orientador possuirá as informações da sua orientação em seu currículo e o orientando também, no entanto de forma distinta. Dados como nome do orientador/orientando e título do trabalho são comum em ambos os currículos, portanto é possível encontrar a colaboração quando há a correspondência de um trabalho entre o discente e docente, aplicando as condições necessárias. É possível tratar este problema através da aplicação de funções de similaridade a textos [DORNELES AND GALANTE 2008], como em nomes de autores, instituição ou título do trabalho. Estas funções de similaridade recebem dois textos como valor de entrada e retornam um valor de 0 à 1 (ou 0 à 100%) equivalente a taxa de similaridade entre eles.

3. ALPC

O ALPC possui duas principais fases, sendo a primeira delas a importação dos currículos da plataforma Lattes (no formato XML) para dentro do sistema. Nesse momento é feita a extração de alguns dados apenas para identificar o currículo e este é serializado e persistido no banco de dados. A segunda etapa é o processamento desses currículos, onde o currículo é carregado do banco de dados e convertido para objetos da linguagem de programação Java. Neste estágio, o currículo convertido passa por uma série de buscas e processos, a fim de evitar a duplicidade e alguma inconsistência de qualquer docente, discente ou colaboração, para que futuramente se possa extrair os relatórios sem erros de contabilização. Para realizar o processo de busca, foi montado um segundo banco de dados além da relacional, onde os dados são indexados por palavras/termos.

Dentro do sistema, cada currículo é tratado como uma pessoa para o sistema, e assim será durante todo o processo. Todas as buscas e persistência são feitas limitando-se ao programa de pós-graduação no qual o processamento está ocorrendo, ou seja, cada programa possui seus próprios dados. É importante ressaltar que se um currículo for processado para um programa e depois para outro, duas pessoas serão criadas no sistema, cada uma vinculada à seu devido programa. Assim, uma pessoa representa um currículo em um programa. Caso essa pessoa possua uma ou mais formações dentro do programa, o papel de discente lhe é conferido. Se possuir orientações no programa, lhe é dado o papel de docente. Para a avaliação sobre a orientação/formação pertencer ou não ao programa, o nome do curso é comparado com o nome do programa de pós-graduação através de uma função de similaridade.

3.1. Importação

Constitui em uma funcionalidade com poucos passos, onde em um primeiro momento o currículo é convertido do XML para um objeto da linguagem, para então ser salvo no banco de dados e futuramente ser processado.

A primeira parte do processo constitui em extrair os dados do currículo Lattes, que estão no formato XML, e colocá-los em objetos da linguagem em que o trabalho está sendo desenvolvido, facilitando a manipulação desses dados. Para que a conversão aconteça, foi indispensável um mapeamento de cada elemento do currículo para objetos da linguagem. Foi manualmente elaborado e pode ser utilizado para a conversão em ambas direções - de XML para objetos da linguagem e vice-versa.

Na última etapa do processo de importação de um currículo para o sistema, alguns dados são extraídos do currículo, com o objetivo de apresentar ao usuário o currículo importado e identificá-lo, tanto para o usuário quanto ao sistema. Os campos são: número identificador (número que identifica unicamente cada currículo existente na

plataforma Lattes), nome completo (nome da pessoa a qual o currículo se refere) e data de atualização (última vez que esse currículo foi atualizado na plataforma Lattes). Após a extração desses dados, o currículo XML é serializado e salvo no banco de dados para futuramente ser processado.

3.2. Processamento

Para otimizar todo o processo de importar currículos e gerar relatórios a partir de suas informações, foi adotada a estratégia de manipular os dados de cada currículo uma única vez (aceitando posteriores atualizações), processá-los e persisti-los em um novo modelo de dados, diminuindo o tempo total a cada interação do usuário para seus devidos fins, além de facilitar todo o processo interno. Dessa forma, o usuário pode consultar as informações requeridas rapidamente e de forma customizada aplicando filtros. Lembrando que embora o processamento possa ser lento, deseja-se que seja feito em tempo viável e que tenha boa escalabilidade. Portanto, é preciso que o processo leve apenas alguns minutos. Para casos onde ocorra apenas atualização do currículo, a função deve ocorrer de forma mais rápida ainda. O processamento de um currículo é dividido em passos, explicados a seguir.

Como primeira parte do processamento de um currículo, é necessário verificar se uma pessoa equivalente aquele currículo já foi importada anteriormente a fim de evitar duplicidades e/ou esforço desnecessário. Caso encontrado sua data de atualização é avaliada: se a data de atualização do currículo no sistema é mais antiga que a data de atualização do novo currículo, então o novo currículo é processado e seus dados extraídos. Isso indica que a pessoa que está no banco de dados está desatualizada. Para os casos onde encontra-se uma pessoa no sistema com data de atualização igual ou maior ao novo currículo, nada será feito.

Caso um currículo equivalente não seja encontrado no processo de busca, uma nova pessoa é criada para sua representação. Posteriormente, são atualizados os campos identificadores do currículo: nome completo, número identificador e data de atualização.

O processo de busca do currículo pode ser dividido em três partes. Os passos são os seguintes:

- Por número identificador

A primeira busca é simples e feita com correspondência total pelo número identificador do próprio currículo Lattes. É aplicada diretamente no banco de dados relacional, caso haja uma pessoa representando esse currículo. Isso acontece quando um currículo foi processado previamente, com seu número identificador preenchido. Existe também a possibilidade de um orientador preencher o número identificador do orientando dentro de sua orientação ou o discente preencher o número de seu orientador dentro de sua formação. Este passo não é o único pois o número identificador não é obrigatório, então nem sempre está preenchido.

- Por formação/orientação

Caso não seja encontrada uma pessoa com o número identificador do currículo, é feita uma busca usando as formações e orientações contidas no currículo. Isso ocorre

uma colaboração relativa a esta orientação, que foi salva através de uma formação do discente, a orientação não atualiza nenhum campo dessa colaboração, ou seja, os dados de uma formação sempre terão preferência sobre os dados de uma orientação.

Finalizado o processo das orientações/formações e os papéis atribuídos, caso o currículo possua o papel de docente, são também extraídas as informações sobre suas produções bibliográficas e produções técnicas. É neste processo que será encontrada a colaboração docente-docente. Primeiramente, é feita uma busca no banco de termos indexados usando o título da produção, ano, tipo da produção e nome do autor, nomes em citações e o número identificador, estes três últimos vindos dos dados gerais do currículo a ser importado. A partir dos resultados retornados, o título e os dados referentes ao autor são comparados com funções de similaridade e, caso não haja resultados compatíveis, a nova informação é persistida.

Toda produção possui seus autores e estes são persistidos separadamente e relacionados à produção para que aumente a eficácia da busca, não se atendo apenas aos dados da produção. Na busca no banco de palavras/termos indexados, ao menos um dos campos do autor (nome completo, nome da citação ou número identificador) e todos os campos da produção devem conter ao menos um termo equivalente.

Não há distinção na extração dos dados de produção bibliográfica e técnica, pois possuem o mesmo conjunto de informações. São classificadas por tipo de produção, que se refere a especificação dentro das produções

As atuações profissionais do docente não são colaborativas, ou seja, não possuem nenhum vínculo com outros docentes ou discentes. Sendo assim, não existe a necessidade de buscas e funções de similaridade para evitar duplicidade. As informações já persistidas (caso existam) sempre são deletadas para dar lugar às novas informações. Para as pessoas que possuem o papel de docente, as suas atuações profissionais realizadas na instituição do programa de pós-graduação são simplesmente salvas e vinculadas à pessoa do currículo sendo importado.

As linhas de pesquisa são um subgrupo das pesquisas e desenvolvimento, que estão contidas nas atuações profissionais. Linhas de pesquisa são colaborativas entre docentes e sua busca é dada apenas pelo seu título. Caso não encontrada, a nova linha de pesquisa é persistida. Linhas de pesquisa são distintas dentro de um programa de pós-graduação.

Os projetos de pesquisa, que estão contidos nas participações em projeto das atuações profissionais. O processo de busca e persistência de um projeto de pesquisa é semelhante ao processo descrito na produção, diferenciando alguns dos campos utilizados na pesquisa: retira-se o tipo de produção e adiciona um campo do tipo ano, resultando em ano início e ano fim.

A graduação não foi importada de forma colaborativa, pois ela é utilizada para verificar a procedência dos discentes, ou seja, não é necessário o vínculo docente-discente, apenas o vínculo com o discente sendo importado.

uma colaboração relativa a esta orientação, que foi salva através de uma formação do discente, a orientação não atualiza nenhum campo dessa colaboração, ou seja, os dados de uma formação sempre terão preferência sobre os dados de uma orientação.

Finalizado o processo das orientações/formações e os papéis atribuídos, caso o currículo possua o papel de docente, são também extraídas as informações sobre suas produções bibliográficas e produções técnicas. É neste processo que será encontrada a colaboração docente-docente. Primeiramente, é feita uma busca no banco de termos indexados usando o título da produção, ano, tipo da produção e nome do autor, nomes em citações e o número identificador, estes três últimos vindos dos dados gerais do currículo a ser importado. A partir dos resultados retornados, o título e os dados referentes ao autor são comparados com funções de similaridade e, caso não haja resultados compatíveis, a nova informação é persistida.

Toda produção possui seus autores e estes são persistidos separadamente e relacionados à produção para que aumente a eficácia da busca, não se atendo apenas aos dados da produção. Na busca no banco de palavras/termos indexados, ao menos um dos campos do autor (nome completo, nome da citação ou número identificador) e todos os campos da produção devem conter ao menos um termo equivalente.

Não há distinção na extração dos dados de produção bibliográfica e técnica, pois possuem o mesmo conjunto de informações. São classificadas por tipo de produção, que se refere a especificação dentro das produções

As atuações profissionais do docente não são colaborativas, ou seja, não possuem nenhum vínculo com outros docentes ou discentes. Sendo assim, não existe a necessidade de buscas e funções de similaridade para evitar duplicidade. As informações já persistidas (caso existam) sempre são deletadas para dar lugar às novas informações. Para as pessoas que possuem o papel de docente, as suas atuações profissionais realizadas na instituição do programa de pós-graduação são simplesmente salvas e vinculadas à pessoa do currículo sendo importado.

As linhas de pesquisa são um subgrupo das pesquisas e desenvolvimento, que estão contidas nas atuações profissionais. Linhas de pesquisa são colaborativas entre docentes e sua busca é dada apenas pelo seu título. Caso não encontrada, a nova linha de pesquisa é persistida. Linhas de pesquisa são distintas dentro de um programa de pós-graduação.

Os projetos de pesquisa, que estão contidos nas participações em projeto das atuações profissionais. O processo de busca e persistência de um projeto de pesquisa é semelhante ao processo descrito na produção, diferenciando alguns dos campos utilizados na pesquisa: retira-se o tipo de produção e adiciona um campo do tipo ano, resultando em ano início e ano fim.

A graduação não foi importada de forma colaborativa, pois ela é utilizado para verificar a procedência dos discentes, ou seja, não é necessário o vínculo docente-discente, apenas o vínculo com o discente sendo importado.

4. Relatórios

Os relatórios são uma parte de extrema relevância da aplicação e são organizados de forma simples e compacta, agrupados por sua amplitude: informações sobre o programa, docentes e discentes.

Dentro do programa, existem subgrupos, sendo estes: atividades acadêmicas, linha de pesquisa, participação em projetos, procedência, produção bibliográfica e produção técnica. A procedência é classificada por serem as universidades onde os discentes estiveram anteriormente ao ingresso do programa. Para os grupos atividades acadêmicas, produção bibliográfica e produção técnica, são listados os seus respectivos tipos, atrelando a cada um sua quantidade, calculada para todo o programa. Há a possibilidade de filtragem por período entre anos. Para os grupos projeto, linha de pesquisa e procedência, os dados estão organizados em forma de lista e retornam resultados relacionados ao programa, e para cada um uma quantidade de docentes em projeto e linha de pesquisa e discentes em procedência. A linha de pesquisa tem como filtro opcional seu título. Já para projetos de pesquisa e procedência, os filtros são dados pelo nome e período por ano.

No grupo dos docentes é exibido um resumo com número de bolsistas e número de orientadores de mestrado e doutorado. Juntamente, há uma listagem de todos os docentes do programa. Cada docente é complementado com as informações sobre as orientações em andamento e concluídas, possibilitando a visão detalhada de cada orientação, além de um resumo com quantidade de discentes de mestrado, quantidade de discentes de doutorado e total. É possível também uma visão detalhada sobre o docente, como formações e o quantidade de produções bibliográficas. O filtro aplicado à listagem está entre nome do orientador, área do conhecimento e período por ano.

O grupo discentes segue a diagramação dos docentes. O resumo apresenta informações relacionadas ao programa, como tempo médio de curso, total de discentes desistentes, total de discentes e números de discentes em mestrado ou doutorado, tanto em andamento quanto concluído. Dentro da listagem, cada item exibe o tipo de orientação e seu ano, com possível detalhamento de informações da formação do discente.

5. Conclusão

O ALPC é uma ferramenta projetada para a melhoria da gerência, a qualidade e a eficiência dos cursos de pós-graduação, extraíndo e organizando as informações do currículo Lattes. Todas as suas funcionalidades foram desenvolvidas com ênfase no público alvo. O principal objetivo do trabalho, é reunir as informações necessárias e relatá-las ao usuário de acordo com os currículos em que se desejava obter estes dados, que não estão explícitos, foi atingido.

Apesar de todo o processo ter foco nas similaridades, esperando então que as informações extraídas entre colaborações não estejam totalmente equivalentes, ainda existe uma margem de erro em casos nos quais as informações não contenham o

mínimo de aproximação. Infelizmente, isso depende total e exclusivamente da forma com que os currículos tenham sido preenchidos.

Mesmo com problemas no desenvolvimento, alcançou-se uma boa base para a continuação do trabalho, onde foi desenvolvida uma biblioteca que já realiza a conversão de todo o currículo Lattes, facilitando assim futuramente a extração de outros dados.

Referências

DORNELES, C. F.; GALANTE, R. Aplicação de funções de similaridade e detecção de diferenças em grandes volumes de dados distribuídos. Jornadas de Atualização em Informática, 2008.

COHEN, W. W.; RAVIKUMAR, P.; FIENBERG, S. E. A comparison of string distance

metrics for name-matching tasks. Proceedings of the workshop on Data Cleaning and Object Consolidation at the International Conference on Knowledge Discovery and Data Mining, 2003.

NAVARRO, G.; SUTINEN, E.; TARHIO, J. Indexing text with approximate q-grams. In

Proceedings of the 11th Annual Symposium on Combinatorial Pattern Matching (CPM'2000), 2000.

APÊNDICE B – Código


```

package ufsc.alpc.converter;

import java.io.File;
import java.io.InputStream;
import java.io.OutputStream;

import javax.xml.bind.JAXBContext;
import javax.xml.bind.JAXBException;
import javax.xml.bind.Marshaller;
import javax.xml.bind.Unmarshaller;

import org.xml.sax.InputSource;

import ufsc.alpc.xml.dto.CurriculoVitaeXmlDto;

public class LattesXmlConverter {

    public static CurriculoVitaeXmlDto unmarshal(String filePath) throws JAXBException {
        JAXBContext jaxbContext =
            ↪ JAXBContext.newInstance(CurriculoVitaeXmlDto.class);
        Unmarshaller jaxbUnmarshaller = jaxbContext.createUnmarshaller();
        return (CurriculoVitaeXmlDto) jaxbUnmarshaller.unmarshal(new
            ↪ InputSource(filePath));
    }

    public static CurriculoVitaeXmlDto unmarshal(File file) throws JAXBException {
        JAXBContext jaxbContext =
            ↪ JAXBContext.newInstance(CurriculoVitaeXmlDto.class);
        Unmarshaller jaxbUnmarshaller = jaxbContext.createUnmarshaller();
        return (CurriculoVitaeXmlDto) jaxbUnmarshaller.unmarshal(file);
    }

    public static CurriculoVitaeXmlDto unmarshal(InputStream inputStream) throws
        ↪ JAXBException {
        JAXBContext jaxbContext =
            ↪ JAXBContext.newInstance(CurriculoVitaeXmlDto.class);
        Unmarshaller jaxbUnmarshaller = jaxbContext.createUnmarshaller();
        return (CurriculoVitaeXmlDto) jaxbUnmarshaller.unmarshal(inputStream);
    }

    public static void marshal(CurriculoVitaeXmlDto curriculo, String filePath) throws
        ↪ JAXBException {
        JAXBContext jaxbContext =
            ↪ JAXBContext.newInstance(CurriculoVitaeXmlDto.class);
        Marshaller createMarshaller = jaxbContext.createMarshaller();
        createMarshaller.marshal(curriculo, new File(filePath));
    }

    public static void marshal(CurriculoVitaeXmlDto curriculo, File file) throws JAXBException
        ↪ {
        JAXBContext jaxbContext =
            ↪ JAXBContext.newInstance(CurriculoVitaeXmlDto.class);
        Marshaller createMarshaller = jaxbContext.createMarshaller();
        createMarshaller.marshal(curriculo, file);
    }

    public static void marshal(CurriculoVitaeXmlDto curriculo, OutputStream outputStream)
        ↪ throws JAXBException {
        JAXBContext jaxbContext =
            ↪ JAXBContext.newInstance(CurriculoVitaeXmlDto.class);
        Marshaller createMarshaller = jaxbContext.createMarshaller();
        createMarshaller.marshal(curriculo, outputStream);
    }
}

package ufsc.alpc.xml.dto.producao.bibliografica;

public interface DadosBasicosProducao {

    String getTitulo();

    String getAno();
}

package ufsc.alpc.xml.dto.producao.bibliografica.trabalhoseventos;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;

```

```

import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class TrabalhosEmEventosXmlDto {

    // TRABALHO-EM-EVENTOS*
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlAttribute(name = "SEQUENCIA-PRODUCAO")
    protected String sequenciaProducao;

    @XmlElement(name = "TRABALHO-EM-EVENTOS")
    private List<TrabalhoEmEventosXmlDto> trabalhosEventos;

}

package ufsc.alpc.xml.dto.producao.bibliografica.trabalhoseeventos;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutorXmlDto;
import ufsc.alpc.xml.dto.common.PalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;
import ufsc.alpc.xml.dto.producao.bibliografica.trabalhoseeventos.dadosbasicos.DadosBasicosTrabalhoXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.trabalhoseeventos.detalhamento.DetalhamentTrabalhoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class TrabalhoEmEventosXmlDto extends PalavraAreaSetorInfoAgrupadorXmlDto implements
↳ Producao {

    // DADOS-BASICOS-DO-TRABALHO?
    // DETALHAMENTO-DO-TRABALHO?
    // AUTORES*
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DO-TRABALHO")
    private DadosBasicosTrabalhoXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DO-TRABALHO")
    private DetalhamentTrabalhoXmlDto detalhamento;

    @XmlElement(name = "AUTORES")
    private List<AutorXmlDto> autores;

}

package ufsc.alpc.xml.dto.producao.bibliografica.trabalhoseeventos.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosProducaoBibliograficaXmlDto;
import ufsc.alpc.xml.dto.domain.NaturezaTrabalho;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosTrabalhoXmlDto extends DadosBasicosProducaoBibliograficaXmlDto {

    // NATUREZA (COMPLETO | RESUMO | RESUMO_EXPANDIDO) #IMPLIED

```

```

// TITULO-DO-TRABALHO CDATA #IMPLIED
// ANO-DO-TRABALHO CDATA #IMPLIED
// PAIS-DO-EVENTO CDATA #IMPLIED
// IDIOMA CDATA #IMPLIED
// MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
↪ MEIO_DIGITAL |
// FILME | HIPERTEXTO
// | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
// HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
// FLAG-RELEVANCIA (SIM | NAO) "NAO"
// DOI CDATA #IMPLIED
// TITULO-DO-TRABALHO-INGLES CDATA #IMPLIED
// FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"

@XmlAttribute(name = "NATUREZA")
protected NaturezaTrabalho idioma;

@XmlAttribute(name = "TITULO-DO-TRABALHO")
protected String titulo;

@XmlAttribute(name = "ANO-DO-TRABALHO")
protected String ano;

@XmlAttribute(name = "PAIS-DO-EVENTO")
protected String pais;

@XmlAttribute(name = "TITULO-DO-TRABALHO-INGLES")
protected String tituloIngles;
}
package ufsc.alpc.xml.dto.producao.bibliografica.trabalhoseventos.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.domain.ClassificacaoEvento;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoTrabalhoXmlDto {

// CLASSIFICACAO-DO-EVENTO (INTERNACIONAL | NACIONAL | REGIONAL |
↪ LOCAL |
// NAO_INFORMADO) "NAO_INFORMADO"
// NOME-DO-EVENTO CDATA #IMPLIED
// CIDADE-DO-EVENTO CDATA #IMPLIED
// ANO-DE-REALIZACAO CDATA #IMPLIED
// TITULO-DOS-ANAIS-OU-PROCEEDINGS CDATA #IMPLIED
// VOLUME CDATA #IMPLIED
// FASCICULO CDATA #IMPLIED
// SERIE CDATA #IMPLIED
// PAGINA-INICIAL CDATA #IMPLIED
// PAGINA-FINAL CDATA #IMPLIED
// ISBN CDATA #IMPLIED
// NOME-DA-EDITORIA CDATA #IMPLIED
// CIDADE-DA-EDITORIA CDATA #IMPLIED
// NOME-DO-EVENTO-INGLES CDATA #IMPLIED

@XmlAttribute(name = "CLASSIFICACAO-DO-EVENTO")
protected ClassificacaoEvento classificacaoEvento;

@XmlAttribute(name = "NOME-DO-EVENTO")
protected String nomeEvento;

@XmlAttribute(name = "CIDADE-DO-EVENTO")
protected String cidadeEvento;

@XmlAttribute(name = "ANO-DE-REALIZACAO")
protected String anoRealizacao;

@XmlAttribute(name = "TITULO-DOS-ANAIS-OU-PROCEEDINGS")
protected String tituloAnaisProceedings;

@XmlAttribute(name = "VOLUME")
protected String volume;

@XmlAttribute(name = "FASCICULO")

```

```

protected String fasciculo;

@XmlAttribute(name = "SERIE")
protected String serie;

@XmlAttribute(name = "PAGINA-INICIAL")
protected String paginaInicial;

@XmlAttribute(name = "PAGINA-FINAL")
protected String paginaFinal;

@XmlAttribute(name = "ISBN")
protected String isbn;

@XmlAttribute(name = "NOME-DA-EDITORA")
protected String nomeEditora;

@XmlAttribute(name = "CIDADE-DA-EDITORA")
protected String cidadeEditora;

@XmlAttribute(name = "NOME-DO-EVENTO-INGLES")
protected String nomeEventoIngles;
}

package ufsc.alpc.xml.dto.producao.bibliografica.artigos.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosProducaoBibliograficaXmlDto;
import ufsc.alpc.xml.dto.domain.NaturezaArtigo;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosArtigoXmlDto extends DadosBasicosProducaoBibliograficaXmlDto {

    // NATUREZA (COMPLETO | RESUMO | NAO_INFORMADO) #IMPLIED
    // TITULO-DO-ARTIGO CDATA #IMPLIED
    // ANO-DO-ARTIGO CDATA #IMPLIED
    // PAIS-DE-PUBLICACAO CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    // ← MEIO_DIGITAL |
    // FILME | HIPERTEXTO
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-DO-ARTIGO-INGLES CDATA #IMPLIED
    // FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaArtigo natureza;

    @XmlAttribute(name = "TITULO-DO-ARTIGO")
    protected String titulo;

    @XmlAttribute(name = "ANO-DO-ARTIGO")
    protected String ano;

    @XmlAttribute(name = "PAIS-DE-PUBLICACAO")
    protected String pais;

    @XmlAttribute(name = "TITULO-DO-ARTIGO-INGLES")
    protected String tituloIngles;
}

package ufsc.alpc.xml.dto.producao.bibliografica.artigos.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

```

```

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoArtigoXmlDto {
    // TITULO-DO-PERIODICO-OU-REVISTA CDATA #IMPLIED
    // ISSN CDATA #IMPLIED
    // VOLUME CDATA #IMPLIED
    // FASCICULO CDATA #IMPLIED
    // SERIE CDATA #IMPLIED
    // PAGINA-INICIAL CDATA #IMPLIED
    // PAGINA-FINAL CDATA #IMPLIED
    // LOCAL-DE-PUBLICACAO CDATA #IMPLIED

    @XmlAttribute(name = "TITULO-DO-PERIODICO-OU-REVISTA")
    protected String tituloPeriodicoRevista;

    @XmlAttribute(name = "ISSN")
    protected String issn;

    @XmlAttribute(name = "VOLUME")
    protected String volume;

    @XmlAttribute(name = "FASCICULO")
    protected String fasciculo;

    @XmlAttribute(name = "SERIE")
    protected String serie;

    @XmlAttribute(name = "PAGINA-INICIAL")
    protected String paginaInicial;

    @XmlAttribute(name = "PAGINA-FINAL")
    protected String paginaFinal;

    @XmlAttribute(name = "LOCAL-DE-PUBLICACAO")
    protected String localPublicacao;
}
package ufsc.alpc.xml.dto.producao.bibliografica.artigos.publicados;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ArtigosPublicadosXmlDto {
    // ARTIGO-PUBLICADO*

    @XmlElement(name = "ARTIGO-PUBLICADO")
    private List<ArtigoPublicadoXmlDto> artigoPublicado;
}
package ufsc.alpc.xml.dto.producao.bibliografica.artigos.publicados;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;
import ufsc.alpc.xml.dto.producao.bibliografica.artigos.dadosbasicos.DadosBasicosArtigoXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.artigos.detalhamento.DetalhamentoArtigoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)

```

```

@XmlAccessorType(XmlAccessType.FIELD)
public class ArtigoPublicadoXmlDto extends
↳ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {

    // DADOS-BASICOS-DO-ARTIGO?
    // DETALHAMENTO-DO-ARTIGO?
    // AUTORES*
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATE #IMPLIED
    // ORDEM-IMPORTANCIA CDATE #IMPLIED

    @XmlAttribute(name = "ORDEM-IMPORTANCIA")
    protected String ordemImportancia;

    @XmlElement(name = "DADOS-BASICOS-DO-ARTIGO")
    private DadosBasicosArtigoXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DO-ARTIGO")
    private DetalhamentoArtigoXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.bibliografica.artigos.aceitos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;
import ufsc.alpc.xml.dto.producao.bibliografica.artigos.dadosbasicos.DadosBasicosArtigoXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.artigos.detalhamento.DetalhamentoArtigoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ArtigoAceitoParaPublicacaoXmlDto extends
↳ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {

    // DADOS-BASICOS-DO-ARTIGO?
    // DETALHAMENTO-DO-ARTIGO?
    // AUTORES*
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATE #IMPLIED

    @XmlElement(name = "DADOS-BASICOS-DO-ARTIGO")
    private DadosBasicosArtigoXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DO-ARTIGO")
    private DetalhamentoArtigoXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.bibliografica.artigos.aceitos;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ArtigosAceitosParaPublicacaoXmlDto {

    // ARTIGO-ACEITO-PARA-PUBLICACAO*

    @XmlElement(name = "ARTIGO-ACEITO-PARA-PUBLICACAO")
    private List<ArtigoAceitoParaPublicacaoXmlDto> artigoAceito;
}

```



```

}
package ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.prefacioposfacio;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;
import
↪ ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.prefacioposfacio.dadosbasicos.DadosBasicosPrefacioP
import
↪ ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.prefacioposfacio.detalhamento.DetalhamentoPrefacioP

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class PrefacioPosfacioXmlDto extends
↪ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {
    // DADOS-BASICOS-DO-PREFACIO-POSFACIO?
    // DETALHAMENTO-DO-PREFACIO-POSFACIO?
    // AUTORES*
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlElement(name = "DADOS-BASICOS-DO-PREFACIO-POSFACIO")
    private DadosBasicosPrefacioPosfacioXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DO-PREFACIO-POSFACIO")
    private DetalhamentoPrefacioPosfacioXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.prefacioposfacio.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosProducaoBibliograficaXmlDto;
import ufsc.alpc.xml.dto.domain.NaturezaPrefacioPosfacio;
import ufsc.alpc.xml.dto.domain.TipoPrefacioPosfacio;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosPrefacioPosfacioXmlDto extends
↪ DadosBasicosProducaoBibliograficaXmlDto {
    // TIPO (PREFACIO | POSFACIO | APRESENTACAO | INTRODUCAO) #IMPLIED
    // NATUREZA (LIVRO | OUTRA | REVISTAS_OU_PERIODICOS | NAO_INFORMADO)
    // "NAO_INFORMADO"
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS-DE-PUBLICACAO CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    ↪ MEIO_DIGITAL |
    // FILME | HIPERTEXTO
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "TIPO")
    protected TipoPrefacioPosfacio tipo;

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaPrefacioPosfacio natureza;
}

```

```

    @XmlAttribute(name = "TITULO")
    protected String titulo;

    @XmlAttribute(name = "ANO")
    protected String ano;

    @XmlAttribute(name = "PAIS-DE-PUBLICACAO")
    protected String paisPublicacao;

    @XmlAttribute(name = "TITULO-INGLES")
    protected String tituloIngles;
}
package ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.prefacioposfacio.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoPrefacioPosfacioXmlDto {

    // NOME-DO-AUTOR-DA-PUBLICACAO CDATA #IMPLIED
    // TITULO-DA-PUBLICACAO CDATA #IMPLIED
    // ISSN-ISBN CDATA #IMPLIED
    // NUMERO-DA-EDICAO-REVISAO CDATA #IMPLIED
    // VOLUME CDATA #IMPLIED
    // SERIE CDATA #IMPLIED
    // FASCICULO CDATA #IMPLIED
    // EDITORA-DO-PREFACIO-POSFACIO CDATA #IMPLIED
    // CIDADE-DA-EDITORIA CDATA #IMPLIED

    @XmlAttribute(name = "NOME-DO-AUTOR-DA-PUBLICACAO")
    protected String nomeAutorPublicacao;

    @XmlAttribute(name = "TITULO-DA-PUBLICACAO")
    protected String tituloPublicacao;

    @XmlAttribute(name = "ISSN-ISBN")
    protected String issnIsbn;

    @XmlAttribute(name = "NUMERO-DA-EDICAO-REVISAO")
    protected String numeroEdicaoRevisao;

    @XmlAttribute(name = "VOLUME")
    protected String volume;

    @XmlAttribute(name = "SERIE")
    protected String serie;

    @XmlAttribute(name = "FASCICULO")
    protected String fasciculo;

    @XmlAttribute(name = "EDITORIA-DO-PREFACIO-POSFACIO")
    protected String editoriaPrefacioPosfacio;

    @XmlAttribute(name = "CIDADE-DA-EDITORIA")
    protected String cidadeEditoria;
}
package ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.partiturasmusical;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;
import
↪ ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.partiturasmusical.dadosbasicos.DadosBasicosPartituraXmlDto;
import

```

```

↪ ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.partituramusical.detalhamento.DetalhamentoPartitura

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class PartituraMusicalXmlDto extends
↪ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {
    // DADOS-BASICOS-DA-PARTITURA?
    // DETALHAMENTO-DA-PARTITURA?
    // AUTORES*
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlElement(name = "DADOS-BASICOS-DA-PARTITURA")
    private DadosBasicosPartituraXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DA-PARTITURA")
    private DetalhamentoPartituraXmlDto detalhamento;
}

package ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.partituramusical.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosProducaoBibliograficaXmlDto;
import ufsc.alpc.xml.dto.domain.NaturezaArranjoMusical;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosPartituraXmlDto extends DadosBasicosProducaoBibliograficaXmlDto {
    // NATUREZA (CANTO | CORAL | ORQUESTRA | OUTRO | NAO_INFORMADO)
    ↪ #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS-DE-PUBLICACAO CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    ↪ MEIO_DIGITAL |
    // FILME | HIPERTEXTO
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaArranjoMusical natureza;

    @XmlAttribute(name = "TITULO")
    protected String titulo;

    @XmlAttribute(name = "ANO")
    protected String ano;

    @XmlAttribute(name = "PAIS-DE-PUBLICACAO")
    protected String paisPublicacao;

    @XmlAttribute(name = "TITULO-INGLES")
    protected String tituloIngles;
}

package ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.partituramusical.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;

```

```

import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoPartituraXmlDto {

    // FORMACAO-INSTRUMENTAL CDATA #IMPLIED
    // EDITORA CDATA #IMPLIED
    // CIDADE-DA-EDITORA CDATA #IMPLIED
    // NUMERO-DE-PAGINAS CDATA #IMPLIED
    // NUMERO-DO-CATALOGO CDATA #IMPLIED

    @XmlAttribute(name = "FORMACAO-INSTRUMENTAL")
    protected String formacaoInstrumental;

    @XmlAttribute(name = "EDITORA")
    protected String editora;

    @XmlAttribute(name = "CIDADE-DA-EDITORA")
    protected String cidadeEditora;

    @XmlAttribute(name = "NUMERO-DE-PAGINAS")
    protected String numeroPaginas;

    @XmlAttribute(name = "NUMERO-DO-CATALOGO")
    protected String numeroCatalogo;

}
package ufsc.alpc.xml.dto.producao.bibliografica.demaistipos;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import lombok.*;

import ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.outra.OutraProducaoBibliograficaXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.partiturasmusical.PartituraMusicalXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.prefacioPosfacio.PrefacioPosfacioXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.traducao.TraducaoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DemaisTiposProducaoBibliograficaXmlDto {

    // OUTRA-PRODUCAO-BIBLIOGRAFICA*
    // PARTITURA-MUSICAL*
    // PREFACIO-POSFACIO*
    // TRADUCAO*

    @XmlElement(name = "OUTRA-PRODUCAO-BIBLIOGRAFICA")
    private List<OutraProducaoBibliograficaXmlDto> outraProducaoBibliografica;

    @XmlElement(name = "PARTITURA-MUSICAL")
    private List<PartituraMusicalXmlDto> partituraMusical;

    @XmlElement(name = "PREFACIO-POSFACIO")
    private List<PrefacioPosfacioXmlDto> prefacioPosfacio;

    @XmlElement(name = "TRADUCAO")
    private List<TraducaoXmlDto> traducao;

}
package ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.traducao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import lombok.*;

import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;

```

```

import ufsc.alpc.xml.dto.producao.bibliografica.Producao;
import
↳ ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.traducao.dadosbasicos.DadosBasicosTraducaoXmlDto
import
↳ ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.traducao.detalhamento.DetalhamentoTraducaoXmlDto

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class TraducaoXmlDto extends AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto
↳ implements Producao {

    // DADOS-BASICOS-DA-TRADUCAO?
    // DETALHAMENTO-DA-TRADUCAO?
    // AUTORES*
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlElement(name = "DADOS-BASICOS-DA-TRADUCAO")
    private DadosBasicosTraducaoXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DA-TRADUCAO")
    private DetalhamentoTraducaoXmlDto detalhamento;

}

package ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.traducao.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosProducaoBibliograficaXmlDto;
import ufsc.alpc.xml.dto.domain.NaturezaTraducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosTraducaoXmlDto extends DadosBasicosProducaoBibliograficaXmlDto {

    // NATUREZA (ARTIGO | LIVRO | OUTRO | NAO_INFORMADO) #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS-DE-PUBLICACAO CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
↳ MEIO_DIGITAL |
    // FILME | HIPERTEXTO
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaTraducao natureza;

    @XmlAttribute(name = "TITULO")
    protected String titulo;

    @XmlAttribute(name = "ANO")
    protected String ano;

    @XmlAttribute(name = "PAIS-DE-PUBLICACAO")
    protected String paisPublicacao;

    @XmlAttribute(name = "TITULO-INGLES")
    protected String tituloIngles;

}

package ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.traducao.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

```

```

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoTraducaoXmlDto {
    // NOME-DO-AUTOR-TRADUZIDO CDATA #IMPLIED
    // TITULO-DA-OBRA-ORIGINAL CDATA #IMPLIED
    // ISSN-ISBN CDATA #IMPLIED
    // IDIOMA-DA-OBRA-ORIGINAL CDATA #IMPLIED
    // EDITORA-DA-TRADUCAO CDATA #IMPLIED
    // CIDADE-DA-EDITORIA CDATA #IMPLIED
    // NUMERO-DE-PAGINAS CDATA #IMPLIED
    // NUMERO-DA-EDICAO-REVISAO CDATA #IMPLIED
    // VOLUME CDATA #IMPLIED
    // FASCICULO CDATA #IMPLIED
    // SERIE CDATA #IMPLIED

    @XmlAttribute(name = "NOME-DO-AUTOR-TRADUZIDO")
    protected String nomeAutorTraduzido;

    @XmlAttribute(name = "TITULO-DA-OBRA-ORIGINAL")
    protected String tituloObraOriginal;

    @XmlAttribute(name = "ISSN-ISBN")
    protected String issnIsbn;

    @XmlAttribute(name = "IDIOMA-DA-OBRA-ORIGINAL")
    protected String idiomaObraOriginal;

    @XmlAttribute(name = "EDITORIA-DA-TRADUCAO")
    protected String editoriaTraducao;

    @XmlAttribute(name = "CIDADE-DA-EDITORIA")
    protected String cidadeEditoria;

    @XmlAttribute(name = "NUMERO-DE-PAGINAS")
    protected String numeroPaginas;

    @XmlAttribute(name = "NUMERO-DA-EDICAO-REVISAO")
    protected String numeroEdicaoRevisao;

    @XmlAttribute(name = "VOLUME")
    protected String volume;

    @XmlAttribute(name = "FASCICULO")
    protected String fasciculo;

    @XmlAttribute(name = "SERIE")
    protected String serie;
}
package ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.outra;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;
import
    ↳ ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.outra.dadosbasicos.DadosBasicosDeOutraProducaoXmlDto;
import
    ↳ ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.outra.detalhamento.DetalhamentoOutraProducaoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OutraProducaoBibliograficaXmlDto extends
    ↳ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {
    // DADOS-BASICOS-DE-OUTRA-PRODUCAO?
    // DETALHAMENTO-DE-OUTRA-PRODUCAO?

```

```

// AUTORES*
// PALAVRAS-CHAVE?
// AREAS-DO-CONHECIMENTO?
// SETORES-DE-ATIVIDADE?
// INFORMACOES-ADICIONAIS?
// SEQUENCIA-PRODUCAO CDATA #IMPLIED

@XmlElement(name = "DADOS-BASICOS-DE-OUTRA-PRODUCAO")
private DadosBasicosDeOutraProducaoXmlDto dadosBasicos;

@XmlElement(name = "DETALHAMENTO-DE-OUTRA-PRODUCAO")
private DetalhamentoOutraProducaoXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.outra.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosProducaoBibliograficaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDeOutraProducaoXmlDto extends
↳ DadosBasicosProducaoBibliograficaXmlDto {

// NATUREZA CDATA #IMPLIED
// TITULO CDATA #IMPLIED
// ANO CDATA #IMPLIED
// PAIS-DE-PUBLICACAO CDATA #IMPLIED
// IDIOMA CDATA #IMPLIED
// MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
↳ MEIO_DIGITAL |
↳ FILME | HIPERTEXTO | OUTRO | VARIOS | NAO_INFORMADO)
↳ "NAO_INFORMADO"
// HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
// FLAG-RELEVANCIA (SIM | NAO) "NAO"
// DOI CDATA #IMPLIED
// TITULO-INGLES CDATA #IMPLIED
// NATUREZA-INGLES CDATA #IMPLIED
// FLAG-DIVULGACAO-CIENTIFICA CDATA #IMPLIED

@XmlAttribute(name = "NATUREZA")
protected String natureza;

@XmlAttribute(name = "TITULO")
protected String titulo;

@XmlAttribute(name = "ANO")
protected String ano;

@XmlAttribute(name = "PAIS-DE-PUBLICACAO")
protected String paisPublicacao;

@XmlAttribute(name = "TITULO-INGLES")
protected String tituloIngles;

@XmlAttribute(name = "NATUREZA-INGLES")
protected String naturezaIngles;
}
package ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.outra.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoOutraProducaoXmlDto {

```

```

// EDITORA CDATA #IMPLIED
// CIDADE-DA-EDITORA CDATA #IMPLIED
// NUMERO-DE-PAGINAS CDATA #IMPLIED
// ISSN-ISBN CDATA #IMPLIED

@XmlAttribute(name = "EDITORA")
protected String editora;

@XmlAttribute(name = "CIDADE-DA-EDITORA")
protected String cidadeEditora;

@XmlAttribute(name = "NUMERO-DE-PAGINAS")
protected String numeroPaginas;

@XmlAttribute(name = "ISSN-ISBN")
protected String issnIsbn;
}
package ufsc.alpc.xml.dto.producao.bibliografica.textosjornaisrevistas;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;
import ufsc.alpc.xml.dto.producao.bibliografica.textosjornaisrevistas.dadosbasicos.DadosBasicosTextoXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.textosjornaisrevistas.detalhamento.DetalhamentoTextoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class TextoEmJornalOuRevistaXmlDto extends
    AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {
    // DADOS-BASICOS-DO-TEXTO?
    // DETALHAMENTO-DO-TEXTO?
    // AUTORES*
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlElement(name = "DADOS-BASICOS-DO-TEXTO")
    private DadosBasicosTextoXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DO-TEXTO")
    private DetalhamentoTextoXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.bibliografica.textosjornaisrevistas.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosProducaoBibliograficaXmlDto;
import ufsc.alpc.xml.dto.domain.NaturezaTexto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosTextoXmlDto extends DadosBasicosProducaoBibliograficaXmlDto {
    // NATUREZA (JORNAL_DE_NOTICIAS | REVISTA_MAGAZINE |
    //     ↳ NAO_INFORMADO) #IMPLIED
    // TITULO-DO-TEXTO CDATA #IMPLIED
    // ANO-DO-TEXTO CDATA #IMPLIED
    // PAIS-DE-PUBLICACAO CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    //     ↳ MEIO_DIGITAL |

```



```

// FILME | HIPERTEXTO
// | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
// HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
// FLAG-RELEVANCIA (SIM | NAO) "NAO"
// DOI CDATA #IMPLIED
// TITULO-DO-TEXTO-INGLES CDATA #IMPLIED
// FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"

@XmlAttribute(name = "NATUREZA")
protected NaturezaTexto natureza;

@XmlAttribute(name = "TITULO-DO-TEXTO")
protected String titulo;

@XmlAttribute(name = "ANO-DO-TEXTO")
protected String ano;

@XmlAttribute(name = "PAIS-DE-PUBLICACAO")
protected String pais;

@XmlAttribute(name = "TITULO-DO-TEXTO-INGLES")
protected String tituloIngles;
}
package ufsc.alpc.xml.dto.producao.bibliografica.textosjornaisrevistas;
import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class TextosJornaisRevistasXmlDto {

    // TEXTO-EM-JORNAL-OU-REVISTA*

    @XmlElement(name = "TEXTO-EM-JORNAL-OU-REVISTA")
    private List<TextoEmJornalOuRevistaXmlDto> textoJornalRevista;
}
package ufsc.alpc.xml.dto.producao.bibliografica.textosjornaisrevistas.detalhamento;
import java.util.Date;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.DateXmlAdapter;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoTextoXmlDto {

    // TITULO-DO-JORNAL-OU-REVISTA CDATA #IMPLIED
    // ISSN CDATA #IMPLIED
    // FORMATO-DATA-DE-PUBLICACAO NMTOKEN #FIXED "DDMMAAA"
    // DATA-DE-PUBLICACAO CDATA #IMPLIED
    // VOLUME CDATA #IMPLIED
    // PAGINA-INICIAL CDATA #IMPLIED
    // PAGINA-FINAL CDATA #IMPLIED
    // LOCAL-DE-PUBLICACAO CDATA #IMPLIED

    @XmlAttribute(name = "TITULO-DO-JORNAL-OU-REVISTA")
    protected String tituloJornalRevista;

    @XmlAttribute(name = "ISSN")
    protected String issn;
}

```

```

@XmlJavaTypeAdapter(DateXmlAdapter.class)
@XmlAttribute(name = "DATA-DE-PUBLICACAO")
private Date dataPublicacao;

@XmlAttribute(name = "VOLUME")
protected String volume;

@XmlAttribute(name = "PAGINA-INICIAL")
protected String paginaInicial;

@XmlAttribute(name = "PAGINA-FINAL")
protected String paginaFinal;

@XmlAttribute(name = "LOCAL-DE-PUBLICACAO")
protected String localPublicacao;
}
package ufsc.alpc.xml.dto.producao.bibliografica;
import java.util.List;
import ufsc.alpc.xml.dto.common.AutorXmlDto;
public interface Producao {
    DadosBasicosProducao getDadosBasicos();
    List<AutorXmlDto> getAutores();
}
package ufsc.alpc.xml.dto.producao.bibliografica.livroscapitulos.livros.publicadosorganizados;
import java.util.List;
import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;
import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class LivrosPublicadosOuOrganizadosXmlDto {
    // LIVRO-PUBLICADO-OU-ORGANIZADO*
    @XmlElement(name = "LIVRO-PUBLICADO-OU-ORGANIZADO")
    private List<LivroPublicadoOuOrganizadoXmlDto> livrosPublicadosOrganizados;
}
package ufsc.alpc.xml.dto.producao.bibliografica.livroscapitulos.livros.publicadosorganizados;
import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;
import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;
import ↪ ufsc.alpc.xml.dto.producao.bibliografica.livroscapitulos.livros.publicadosorganizados.dadosbasicos.DadosBasicosLivros;
import ↪ ufsc.alpc.xml.dto.producao.bibliografica.livroscapitulos.livros.publicadosorganizados.detalhamento.DetalhamentoLivros;
@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class LivroPublicadoOuOrganizadoXmlDto extends ↪ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {
    // DADOS-BASICOS-DO-LIVRO?
    // DETALHAMENTO-DO-LIVRO?
    // AUTORES*
    // PALAVRAS-CHAVE?
}

```

```

// AREAS-DO-CONHECIMENTO?
// SETORES-DE-ATIVIDADE?
// INFORMACOES-ADICIONAIS?
// SEQUENCIA-PRODUCAO CDATA #IMPLIED

@XmlElement(name = "DADOS-BASICOS-DO-LIVRO")
private DadosBasicosLivroXmlDto dadosBasicos;

@XmlElement(name = "DETALHAMENTO-DO-LIVRO")
private DetalhamentoLivroXmlDto detalhamento;
}
package
↳ ufsc.alpc.xml.dto.producao.bibliografica.livroscapitulos.livros.publicadosorganizados.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosProducaoBibliograficaXmlDto;
import ufsc.alpc.xml.dto.domain.NaturezaLivro;
import ufsc.alpc.xml.dto.domain.TipoLivro;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosLivroXmlDto extends DadosBasicosProducaoBibliograficaXmlDto {

    // TIPO (LIVRO_PUBLICADO | LIVRO_ORGANIZADO_OU_EDICAO |
    // ↳ NAO_INFORMADO)
    // #IMPLIED
    // NATUREZA (COLETANEA | TEXTO_INTEGRAL | VERBETE | ANAIS | CATALOGO |
    // ENCICLOPEDIA | LIVRO
    // | OUTRA | PERIODICO | NAO_INFORMADO) "NAO_INFORMADO"
    // TITULO-DO-LIVRO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS-DE-PUBLICACAO CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    // ↳ MEIO_DIGITAL |
    // FILME | HIPERTEXTO
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-DO-LIVRO-INGLES CDATA #IMPLIED
    // FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"

    @XmlAttribute(name = "TIPO")
    protected TipoLivro tipo;

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaLivro natureza;

    @XmlAttribute(name = "TITULO-DO-LIVRO")
    protected String titulo;

    @XmlAttribute(name = "ANO")
    protected String ano;

    @XmlAttribute(name = "PAIS-DE-PUBLICACAO")
    protected String paisPublicacao;

    @XmlAttribute(name = "TITULO-DO-LIVRO-INGLES")
    protected String tituloLivroIngles;
}
package
↳ ufsc.alpc.xml.dto.producao.bibliografica.livroscapitulos.livros.publicadosorganizados.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

```

```

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoLivroXmlDto {

    // NUMERO-DE-VOLUMES CDATA #IMPLIED
    // NUMERO-DE-PAGINAS CDATA #IMPLIED
    // ISBN CDATA #IMPLIED
    // NUMERO-DA-EDICAO-REVISAO CDATA #IMPLIED
    // NUMERO-DA-SERIE CDATA #IMPLIED
    // CIDADE-DA-EDITORA CDATA #IMPLIED
    // NOME-DA-EDITORA CDATA #IMPLIED

    @XmlAttribute(name = "NUMERO-DE-VOLUMES")
    protected String numeroVolumes;

    @XmlAttribute(name = "NUMERO-DE-PAGINAS")
    protected String numeroPaginas;

    @XmlAttribute(name = "ISBN")
    protected String isbn;

    @XmlAttribute(name = "NUMERO-DA-EDICAO-REVISAO")
    protected String numeroEdicaoRevisao;

    @XmlAttribute(name = "NUMERO-DA-SERIE")
    protected String numeroSerie;

    @XmlAttribute(name = "CIDADE-DA-EDITORA")
    protected String cidadeEditora;

    @XmlAttribute(name = "NOME-DA-EDITORA")
    protected String nomeEditora;

}
package ufsc.alpc.xml.dto.producao.bibliografica.livroscapitulos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import lombok.RequiredArgsConstructor;

import ufsc.alpc.xml.dto.producao.bibliografica.livroscapitulos.capitulos.livrospublicados.CapitulosDeLivrosPublicadosXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.livroscapitulos.livros.publicadosorganizados.LivrosPublicadosOuOrganizadosXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class LivrosCapitulosXmlDto {

    // LIVROS-PUBLICADOS-OU-ORGANIZADOS?
    // CAPITULOS-DE-LIVROS-PUBLICADOS?

    @XmlElement(name = "LIVROS-PUBLICADOS-OU-ORGANIZADOS")
    private LivrosPublicadosOuOrganizadosXmlDto livrosPublicadosOrganizados;

    @XmlElement(name = "CAPITULOS-DE-LIVROS-PUBLICADOS")
    private CapitulosDeLivrosPublicadosXmlDto capitulosLivrosPublicados;

}
package ufsc.alpc.xml.dto.producao.bibliografica.livroscapitulos.capitulos.livrospublicados;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;
import ufsc.alpc.xml.dto.producao.bibliografica.livroscapitulos.capitulos.livrospublicados.dadosbasicos.DadosBasicosCapitulo;
import ufsc.alpc.xml.dto.producao.bibliografica.livroscapitulos.capitulos.livrospublicados.detalhamento.DetalhamentoCapitulo;

```

```

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class CapituloDeLivroPublicadoXmlDto extends
    ↳ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {
    // DADOS-BASICOS-DO-CAPITULO
    // DETALHAMENTO-DO-CAPITULO
    // AUTORES*
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlElement(name = "DADOS-BASICOS-DO-CAPITULO")
    private DadosBasicosCapituloXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DO-CAPITULO")
    private DetalhamentoCapituloXmlDto detalhamento;
}
package
    ↳ ufsc.alpc.xml.dto.producao.bibliografica.livroscapitulos.capitulos.livropublicados.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosProducaoBibliograficaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosCapituloXmlDto extends DadosBasicosProducaoBibliograficaXmlDto {
    // TIPO CDATA #IMPLIED
    // TITULO-DO-CAPITULO-DO-LIVRO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS-DE-PUBLICACAO CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    // ↳ MEIO_DIGITAL |
    // FILME | HIPERTEXTO
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-DO-CAPITULO-DO-LIVRO-INGLES CDATA #IMPLIED
    // FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"

    @XmlAttribute(name = "TIPO")
    protected String tipo;

    @XmlAttribute(name = "TITULO-DO-CAPITULO-DO-LIVRO")
    protected String titulo;

    @XmlAttribute(name = "ANO")
    protected String ano;

    @XmlAttribute(name = "PAIS-DE-PUBLICACAO")
    protected String paisPublicacao;

    @XmlAttribute(name = "TITULO-DO-CAPITULO-DO-LIVRO-INGLES")
    protected String tituloLivroIngles;
}
package ufsc.alpc.xml.dto.producao.bibliografica.livroscapitulos.capitulos.livropublicados;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

```

```

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class CapituloDeLivrosPublicadosXmlDto {

    // CAPITULO-DE-LIVRO-PUBLICADO*

    @XmlElement(name = "CAPITULO-DE-LIVRO-PUBLICADO")
    private List<CapituloDeLivroPublicadoXmlDto> capituloLivroPublicado;

}
package ↪ ufsc.alpc.xml.dto.producao.bibliografica.livros capitulos.livrospublicados.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoCapituloXmlDto {

    // TITULO-DO-LIVRO CDATA #IMPLIED
    // NUMERO-DE-VOLUMES CDATA #IMPLIED
    // PAGINA-INICIAL CDATA #IMPLIED
    // PAGINA-FINAL CDATA #IMPLIED
    // ISBN CDATA #IMPLIED
    // ORGANIZADORES CDATA #IMPLIED
    // NUMERO-DA-EDICAO-REVISAO CDATA #IMPLIED
    // NUMERO-DA-SERIE CDATA #IMPLIED
    // CIDADE-DA-EDITORIA CDATA #IMPLIED
    // NOME-DA-EDITORIA CDATA #IMPLIED

    @XmlAttribute(name = "TITULO-DO-LIVRO")
    protected String tituloLivro;

    @XmlAttribute(name = "NUMERO-DE-VOLUMES")
    protected String numeroVolumes;

    @XmlAttribute(name = "PAGINA-INICIAL")
    protected String paginaInicial;

    @XmlAttribute(name = "PAGINA-FINAL")
    protected String paginaFinal;

    @XmlAttribute(name = "ORGANIZADORES")
    protected String organizadores;

    @XmlAttribute(name = "NUMERO-DA-EDICAO-REVISAO")
    protected String numeroEdicaoRevisao;

    @XmlAttribute(name = "NUMERO-DA-SERIE")
    protected String numeroSerie;

    @XmlAttribute(name = "CIDADE-DA-EDITORIA")
    protected String cidadeEditoria;

    @XmlAttribute(name = "NOME-DA-EDITORIA")
    protected String nomeEditoria;

}
package ufsc.alpc.xml.dto.producao.bibliografica;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.producao.bibliografica.artigos.aceitos.ArtigosAceitosParaPublicacaoXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.artigos.publicados.ArtigosPublicadosXmlDto;
import ↪ ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.DemaisTiposProducaoBibliograficaXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.livros capitulos.livrospublicados.detalhamento;
import ufsc.alpc.xml.dto.producao.bibliografica.textosjornaisrevistas.TextosJornaisRevistasXmlDto;

```

```

import ufsc.alpc.xml.dto.producao.bibliografica.trabalhoseeventos.TrabalhosEmEventosXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ProducaoBibliograficaXmlDto {

    // TRABALHOS-EM-EVENTOS?
    // ARTIGOS-PUBLICADOS?
    // LIVROS-E-CAPITULOS?
    // TEXTOS-EM-JORNAIS-OU-REVISTAS?
    // DEMAIS-TIPOS-DE-PRODUCAO-BIBLIOGRAFICA?
    // ARTIGOS-ACEITOS-PARA-PUBLICACAO?

    @XmlElement(name = "TRABALHOS-EM-EVENTOS")
    private TrabalhosEmEventosXmlDto trabalhosEventos;

    @XmlElement(name = "ARTIGOS-PUBLICADOS")
    private ArtigosPublicadosXmlDto artigosPublicados;

    @XmlElement(name = "LIVROS-E-CAPITULOS")
    private LivrosCapitulosXmlDto livrosCapitulos;

    @XmlElement(name = "TEXTOS-EM-JORNAIS-OU-REVISTAS")
    private TextosJornaisRevistasXmlDto textosJornaisRevistas;

    @XmlElement(name = "ARTIGOS-ACEITOS-PARA-PUBLICACAO")
    private ArtigosAceitosParaPublicacaoXmlDto artigosAceitos;

    @XmlElement(name = "DEMAIS-TIPOS-DE-PRODUCAO-BIBLIOGRAFICA")
    private DemaisTiposProducaoBibliograficaXmlDto demaisTiposProducao;
}

package ufsc.alpc.xml.dto.producao.tecnica.software;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MeiodeDivulgacao;
import ufsc.alpc.xml.dto.domain.NaturezaDoSoftware;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDoSoftwareXmlDto implements DadosBasicosProducao {
    // NATUREZA (COMPUTACIONAL | MULTIMIDIA | OUTRO | NAO_INFORMADO)
    // ↪ #IMPLIED
    // TITULO-DO-SOFTWARE CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    // ↪ MEIO_DIGITAL |
    // FILME | HIPERTEXTO
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-DO-SOFTWARE-INGLES CDATA #IMPLIED
    // FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"
    // FLAG-POTENCIAL-INOVACAO (SIM | NAO) "NAO"

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaDoSoftware natureza;

    @XmlAttribute(name = "TITULO-DO-SOFTWARE")
    protected String titulo;

    @XmlAttribute(name = "ANO")
    protected String ano;

    @XmlAttribute(name = "PAIS")
    protected String pais;
}

```

```

@XmlAttribute(name = "IDIOMA")
protected String idioma;

@XmlAttribute(name = "MEIO-DE-DIVULGACAO")
protected MeioDeDivulgacao meioDeDivulgacao;

@XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
protected String homePageDoTrabalho;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-RELEVANCIA")
protected Boolean flagRelevancia;

@XmlAttribute(name = "DOI")
protected String DOI;

@XmlAttribute(name = "TITULO-DO-SOFTWARE-INGLES")
protected String tituloDoSoftwareIngles;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-DIVULGACAO-CIENTIFICA")
protected Boolean flagDivulgacaoCientifica;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-POTENCIAL-INOVACAO")
protected Boolean flagPotencialInovacao;
}
package ufsc.alpc.xml.dto.producao.tecnica.software;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class SoftwareXmlDto extends AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto
    ↪ implements Producao {
    // DADOS-BASICOS-DO-SOFTWARE?,
    // DETALHAMENTO-DO-SOFTWARE?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DO-SOFTWARE")
    private DadosBasicosDoSoftwareXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DO-SOFTWARE")
    private DetalhamentoDoSoftwareXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.tecnica.software;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.domain.Disponibilidade;
import ufsc.alpc.xml.dto.producao.tecnica.common.DetalhamentoRegistroPatenteXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoDoSoftwareXmlDto extends DetalhamentoRegistroPatenteXmlDto {
    // REGISTRO-OU-PATENTE*

```



```

// FINALIDADE CDATA #IMPLIED
// PLATAFORMA CDATA #IMPLIED
// AMBIENTE CDATA #IMPLIED
// DISPONIBILIDADE (RESTRITA | IRRESTRITA | NAO_INFORMADO)
// ↪ "NAO_INFORMADO"
// INSTITUICAO—FINANCIADORA CDATA #IMPLIED
// FINALIDADE—INGLES CDATA #IMPLIED

@XmlAttribute(name = "FINALIDADE")
protected String finalidade;

@XmlAttribute(name = "PLATAFORMA")
protected String plataforma;

@XmlAttribute(name = "AMBIENTE")
protected String ambiente;

@XmlAttribute(name = "DISPONIBILIDADE")
protected Disponibilidade disponibilidade;

@XmlAttribute(name = "INSTITUICAO—FINANCIADORA")
protected String instituicaoFinanciadora;

@XmlAttribute(name = "FINALIDADE—INGLES")
protected String finalidadeIngles;
}
package ufsc.alpc.xml.dto.producao.technica.processos.technicas;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ProcessosTechnicasXmlDto extends
    ↪ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {
    // DADOS—BASICOS—DO—PROCESSOS—OU—TECNICAS?,
    // DETALHAMENTO—DO—PROCESSOS—OU—TECNICAS?,
    // AUTORES*,
    // PALAVRAS—CHAVE?,
    // AREAS—DO—CONHECIMENTO?,
    // SETORES—DE—ATIVIDADE?,
    // INFORMACOES—ADICIONAIS?

    @XmlElement(name = "DADOS—BASICOS—DO—PROCESSOS—OU—TECNICAS")
    private DadosBasicosDoProcessosTechnicasXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO—DO—PROCESSOS—OU—TECNICAS")
    private DetalhamentoDoProcessosTechnicasXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.technica.processos.technicas;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MeiodeDivulgacao;
import ufsc.alpc.xml.dto.domain.NaturezaDoProcessosTechnicas;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDoProcessosTechnicasXmlDto implements DadosBasicosProducao{

    // NATUREZA (ANALITICA | INSTRUMENTAL | PEDAGOGICA | PROCESSUAL |
    // TERAPEUTICA | OUTRA |
    // NAO_INFORMADO) #IMPLIED

```

```

// TITULO-DO-PROCESSO CDATA #IMPLIED
// ANO CDATA #IMPLIED
// PAIS CDATA #IMPLIED
// IDIOMA CDATA #IMPLIED
// MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
// ↪ MEIO_DIGITAL |
// FILME | HIPERTEXTO
// | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
// HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
// FLAG-RELEVANCIA (SIM | NAO) "NAO"
// DOI CDATA #IMPLIED
// TITULO-DO-PROCESSO-INGLES CDATA #IMPLIED
// FLAG-POTENCIAL-INOVACAO (SIM | NAO) "NAO"

@XmlAttribute(name = "NATUREZA")
protected NaturezaDoProcessosTecnicas natureza;

@XmlAttribute(name = "TITULO-DO-PROCESSO")
protected String titulo;

@XmlAttribute(name = "ANO")
protected String ano;

@XmlAttribute(name = "PAIS")
protected String pais;

@XmlAttribute(name = "IDIOMA")
protected String idioma;

@XmlAttribute(name = "MEIO-DE-DIVULGACAO")
protected MeioDeDivulgacao meioDeDivulgacao;

@XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
protected String homePageDoTrabalho;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-RELEVANCIA")
protected Boolean flagRelevancia;

@XmlAttribute(name = "DOI")
protected String DOI;

@XmlAttribute(name = "TITULO-DO-PROCESSO-INGLES")
protected String tituloDoProcessoIngles;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-POTENCIAL-INOVACAO")
protected Boolean flagPotencialInovacao;
}
package ufsc.alpc.xml.dto.producao.tecnica.processos.tecnicas;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.producao.tecnica.common.DetalhamentoRegistroPatenteXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoDoProcessosTecnicasXmlDto extends DetalhamentoRegistroPatenteXmlDto
↪ {

// REGISTRO-OU-PATENTE*

// FINALIDADE CDATA #IMPLIED
// DISPONIBILIDADE CDATA #IMPLIED
// INSTITUICAO-FINANCIADORA CDATA #IMPLIED
// CIDADE-DO-PROCESSO CDATA #IMPLIED
// FINALIDADE-INGLES CDATA #IMPLIED

@XmlAttribute(name = "FINALIDADE")
private String finalidade;

@XmlAttribute(name = "DISPONIBILIDADE")
private String disponibilidade;

```

```

@XmlAttribute(name = "INSTITUICAO-FINANCIADORA")
private String instituicaoFinanciadora;

@XmlAttribute(name = "CIDADE-DO-PROCESSO")
private String cidadeDoProcesso;

@XmlAttribute(name = "FINALIDADE-INGLES")
private String finalidadeIngles;
}
package ufsc.alpc.xml.dto.producao.tecnica;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.producao.tecnica.cultivar.CultivarXmlDto;
import ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.DemaisTipoDeProducaoTecnicaXmlDto;
import ufsc.alpc.xml.dto.producao.tecnica.desenho.industrial.DesenhoIndustrialXmlDto;
import ufsc.alpc.xml.dto.producao.tecnica.marca.MarcaXmlDto;
import ufsc.alpc.xml.dto.producao.tecnica.patente.PatenteXmlDto;
import ufsc.alpc.xml.dto.producao.tecnica.processos.tecnicas.ProcessosTecnicasXmlDto;
import ufsc.alpc.xml.dto.producao.tecnica.produto.tecnologico.ProdutoTecnologicoXmlDto;
import ufsc.alpc.xml.dto.producao.tecnica.software.SoftwareXmlDto;
import ufsc.alpc.xml.dto.producao.tecnica.topografia.TopografiaDeCircuitoIntegradoXmlDto;
import ufsc.alpc.xml.dto.producao.tecnica.trabalho.tecnico.TrabalhoTecnicoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ProducaoTecnicaXmlDto {

    // CULTIVAR-REGISTRADA*,
    // SOFTWARE*,
    // PATENTE*,
    // CULTIVAR-PROTEGIDA*,
    // DESENHO-INDUSTRIAL*,
    // MARCA*,
    // TOPOGRAFIA-DE-CIRCUITO-INTEGRADO*,
    // PRODUTO-TECNOLOGICO*,
    // PROCESSOS-OU-TECNICAS*,
    // TRABALHO-TECNICO*,
    // DEMAIS-TIPOS-DE-PRODUCAO-TECNICA*

    @XmlElement(name = "CULTIVAR-REGISTRADA")
    private List<CultivarXmlDto> cultivarRegistrada;

    @XmlElement(name = "SOFTWARE")
    private List<SoftwareXmlDto> software;

    @XmlElement(name = "PATENTE")
    private List<PatenteXmlDto> patente;

    @XmlElement(name = "CULTIVAR-PROTEGIDA")
    private List<CultivarXmlDto> cultivarProtegida;

    @XmlElement(name = "DESENHO-INDUSTRIAL")
    private List<DesenhoIndustrialXmlDto> desenhoIndustrial;

    @XmlElement(name = "MARCA")
    private List<MarcaXmlDto> marca;

    @XmlElement(name = "TOPOGRAFIA-DE-CIRCUITO-INTEGRADO")
    private List<TopografiaDeCircuitoIntegradoXmlDto> topografiaDeCircuitoIntegrado;

    @XmlElement(name = "PRODUTO-TECNOLOGICO")
    private List<ProdutoTecnologicoXmlDto> produtoTecnologico;

    @XmlElement(name = "PROCESSOS-OU-TECNICAS")
    private List<ProcessosTecnicasXmlDto> processosTecnicas;

    @XmlElement(name = "TRABALHO-TECNICO")
    private List<TrabalhoTecnicoXmlDto> trabalhoTecnico;

    @XmlElement(name = "DEMAIS-TIPOS-DE-PRODUCAO-TECNICA")
    private List<DemaisTipoDeProducaoTecnicaXmlDto> demaisTiposDeProducaoTecnica;
}

```

```

package ufsc.alpc.xml.dto.producao.tecnica.desenho.industrial;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;
import ufsc.alpc.xml.dto.producao.tecnica.common.DadosBasicosDesenhoMarcaTopografiaXmlDto;
import ufsc.alpc.xml.dto.producao.tecnica.common.DetalhamentoDesenhoTopografiaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DesenhoIndustrialXmlDto extends
    ↪ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {
    // DADOS-BASICOS-DO-DESENHO-INDUSTRIAL?,
    // DETALHAMENTO-DO-DESENHO-INDUSTRIAL?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DO-DESENHO-INDUSTRIAL")
    private DadosBasicosDesenhoMarcaTopografiaXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DO-DESENHO-INDUSTRIAL")
    private DetalhamentoDesenhoTopografiaXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.tecnica.trabalho.tecnico;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class TrabalhoTecnicoXmlDto extends
    ↪ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {
    // DADOS-BASICOS-DO-TRABALHO-TECNICO?,
    // DETALHAMENTO-DO-TRABALHO-TECNICO?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DO-TRABALHO-TECNICO")
    private DadosBasicosDoTrabalhoTecnicoXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DO-TRABALHO-TECNICO")
    private DetalhamentoTrabalhoTecnicoXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.tecnica.trabalho.tecnico;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MeiodeDivulgacao;
import ufsc.alpc.xml.dto.domain.NaturezaDoTrabalhoTecnico;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter

```

```

@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDoTrabalhoTecnicoXmlDto implements DadosBasicosProducao {
    // NATUREZA (ASSESSORIA | CONSULTORIA | PARECER |
    //     ↪ ELABORACAO_DE_PROJETO |
    //     RELATORIO_TECNICO
    // | SERVICOS_NA_AREA_DA_SAUDE | OUTRA | NAO_INFORMADO) #IMPLIED
    // TITULO-DO-TRABALHO-TECNICO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    //     ↪ MEIO_DIGITAL |
    //     FILME | HIPERTEXTO
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-DO-TRABALHO-TECNICO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaDoTrabalhoTecnico natureza;

    @XmlAttribute(name = "TITULO-DO-TRABALHO-TECNICO")
    protected String titulo;

    @XmlAttribute(name = "ANO")
    protected String ano;

    @XmlAttribute(name = "PAIS")
    protected String pais;

    @XmlAttribute(name = "MEIO-DE-DIVULGACAO")
    protected MeioDeDivulgacao meioDeDivulgacao;

    @XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
    protected String homePageDoTrabalho;

    @XmlAttribute(name = "IDIOMA")
    protected String idioma;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-RELEVANCIA")
    protected Boolean flagRelevancia;

    @XmlAttribute(name = "DOI")
    protected String doi;

    @XmlAttribute(name = "TITULO-DO-TRABALHO-TECNICO-INGLES")
    protected String tituloDoTrabalhoTecnicoIngles;
}
package ufsc.alpc.xml.dto.producao.technica.trabalho.tecnico;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoTrabalhoTecnicoXmlDto {
    // FINALIDADE CDATA #IMPLIED
    // DURACAO-EM-MESES CDATA #IMPLIED
    // NUMERO-DE-PAGINAS CDATA #IMPLIED
    // DISPONIBILIDADE CDATA #IMPLIED
    // INSTITUICAO-FINANCIADORA CDATA #IMPLIED
    // CIDADE-DO-TRABALHO CDATA #IMPLIED
    // FINALIDADE-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "FINALIDADE")
    private String finalidade;

    @XmlAttribute(name = "DURACAO-EM-MESES")

```

```

private Integer duracaoEmMeses;

@XmlAttribute(name = "NUMERO-DE-PAGINAS")
private Integer numeroDePaginas;

@XmlAttribute(name = "DISPONIBILIDADE")
private String disponibilidade;

@XmlAttribute(name = "INSTITUICAO-FINANCIADORA")
private String instituicaoFinanciadora;

@XmlAttribute(name = "CIDADE-DO-TRABALHO")
private String cidadeDoTrabalho;

@XmlAttribute(name = "FINALIDADE-INGLES")
private String finalidadeIngles;
}
package ufsc.alpc.xml.dto.producao.tecnica.marca;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;
import ufsc.alpc.xml.dto.producao.tecnica.common.DadosBasicosDesenhoMarcaTopografiaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class MarcaXmlDto extends AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto
    ↪ implements Producao {
    // DADOS-BASICOS-DA-MARCA?,
    // DETALHAMENTO-DA-MARCA?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DA-MARCA")
    private DadosBasicosDesenhoMarcaTopografiaXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DA-MARCA")
    private DetalhamentoDaMarcaXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.tecnica.marca;

import javax.xml.bind.annotation.XmlAttribute;

public class DetalhamentoDaMarcaXmlDto {
    // REGISTRO-OU-PATENTE*

    // FINALIDADE CDATA #IMPLIED
    // FINALIDADE-INGLES CDATA #IMPLIED
    // NATUREZA CDATA #IMPLIED

    @XmlAttribute(name = "FINALIDADE")
    private String finalidade;

    @XmlAttribute(name = "FINALIDADE-INGLES")
    private String finalidadeIngles;

    @XmlAttribute(name = "NATUREZA")
    private String natureza;
}
package ufsc.alpc.xml.dto.producao.tecnica.patente;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

```

```

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class PatenteXmlDto extends AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto
    ↪ implements Producao {
    // DADOS-BASICOS-DA-PATENTE?,
    // DETALHAMENTO-DA-PATENTE?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DA-PATENTE")
    private DadosBasicosDaPatenteXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DA-PATENTE")
    private DetalhamentoDaPatenteXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.technica.patente;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.producao.technica.common.DetalhamentoRegistroPatenteXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoDaPatenteXmlDto extends DetalhamentoRegistroPatenteXmlDto {
    // REGISTRO-OU-PATENTE*
    // FINALIDADE CDATA #IMPLIED
    // INSTITUICAO-FINANCIADORA CDATA #IMPLIED
    // FINALIDADE-INGLES CDATA #IMPLIED
    // CATEGORIA CDATA #IMPLIED

    @XmlAttribute(name = "FINALIDADE")
    private String finalidade;

    @XmlAttribute(name = "INSTITUICAO-FINANCIADORA")
    private String instituicaoFinanciadora;

    @XmlAttribute(name = "FINALIDADE-INGLES")
    private String finalidadeIngles;

    @XmlAttribute(name = "CATEGORIA")
    private String categoria;
}
package ufsc.alpc.xml.dto.producao.technica.patente;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MeiodeDivulgacao;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDaPatenteXmlDto implements DadosBasicosProducao {
    // TITULO CDATA #IMPLIED
    // ANO-DESENVOLVIMENTO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // HOME-PAGE CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    ↪ MEIO_DIGITAL |

```

```

// FILME | HIPERTEXTO
// | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
// FLAG-RELEVANCIA (SIM | NAO) "NAO"
// TITULO-INGLES CDATA #IMPLIED
// FLAG-POTENCIAL-INOVACAO (SIM | NAO) "NAO"

@XmlAttribute(name = "TITULO")
protected String titulo;

@XmlAttribute(name = "ANO-DESENVOLVIMENTO")
protected String ano;

@XmlAttribute(name = "PAIS")
protected String pais;

@XmlAttribute(name = "HOME-PAGE")
protected String homePage;

@XmlAttribute(name = "MEIO-DE-DIVULGACAO")
protected MeioDeDivulgacao meioDeDivulgacao;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-RELEVANCIA")
protected Boolean flagRelevancia;

@XmlAttribute(name = "TITULO-INGLES")
protected String tituloDoProdutoIngles;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-POTENCIAL-INOVACAO")
protected Boolean flagPotencialInovacao;
}
package ufsc.alpc.xml.dto.producao.tecnica.common;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoRegistroPatenteXmlDto {

    @XmlElement(name = "REGISTRO-OU-PATENTE")
    protected List<RegistroPatenteXmlDto> registrosPatentes;
}
package ufsc.alpc.xml.dto.producao.tecnica.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDesenhoMarcaTopografiaXmlDto implements DadosBasicosProducao{

    // TITULO CDATA #IMPLIED
    // ANO-DESENVOLVIMENTO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // TITULO-INGLES CDATA #IMPLIED
    // FLAG-POTENCIAL-INOVACAO (SIM | NAO) "NAO"

    @XmlAttribute(name = "TITULO")
    protected String titulo;

```



```

@XmlAttribute(name = "ANO-DESENVOLVIMENTO")
protected String ano;

@XmlAttribute(name = "PAIS")
protected String pais;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-RELEVANCIA")
protected Boolean flagRelevancia;

@XmlAttribute(name = "TITULO-INGLES")
protected String tituloIngles;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-POTENCIAL-INOVACAO")
protected Boolean flagPotencialInovacao;
}
package ufsc.alpc.xml.dto.producao.technical.common;

import java.util.Date;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.DateXmlAdapter;
import ufsc.alpc.xml.dto.domain.TipoDePatente;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class RegistroPatenteXmlDto {
    // TIPO-PATENTE (PRIVILEGIO DE INOVACAO_PI |
    // ↳ MODELO DE UTILIDADE_MU |
    // DESENHO INDUSTRIAL_DI | MODELO INDUSTRIAL_MI |
    // ↳ PATENTE NO EXTERIOR_PE |
    // PROGRAMA DE COMPUTADOR_PC | OUTRO_OU | OUTRO_Ou |
    // CERTIFICADO DE ADICAO_CA | CULTIVAR_PROTEGIDA_CTV |
    // MARCA REGISTRADA DE PRODUTO_MPD |
    // MARCA REGISTRADA DE SERVICIO_MSV |
    // ↳ MARCA REGISTRADA COLETIVA_MCL |
    // MARCA REGISTRADA DE CERTIFICACAO_MCT |
    // TOPOGRAFIA DE CIRCUITO INTEGRADO_REGISTRADA_TCI |
    // MARCA REGISTRADA FIGURATIVA_MRF |
    // ↳ MARCA REGISTRADA NOMINATIVA_MRN |
    // MARCA REGISTRADA MISTA_MRM |
    // ↳ MARCA REGISTRADA TRIDIMENSIONAL_MRT
    // ) #IMPLIED
    // CODIGO-DO-REGISTRO-OU-PATENTE CDATA #IMPLIED
    // TITULO-PATENTE CDATA #IMPLIED
    // FORMATO-DATA-PEDIDO NMTOKEN #FIXED "DDMMAAAA"
    // DATA-PEDIDO-DE-DEPOSITO CDATA #IMPLIED
    // DATA-DE-PEDIDO-DE-EXAME CDATA #IMPLIED
    // DATA-DE-CONCESSAO CDATA #IMPLIED
    // INSTITUICAO-DEPOSITO-REGISTRO CDATA #IMPLIED
    // NUMERO-DEPOSITO-PCT CDATA #IMPLIED
    // FORMATO-DATA-DEPOSITO-PCT NMTOKEN #FIXED "DDMMAAAA"
    // DATA-DEPOSITO-PCT CDATA #IMPLIED
    // NOME-DO-TITULAR CDATA #IMPLIED

    @XmlAttribute(name = "TIPO-PATENTE")
    private TipoDePatente tipoDePatente;

    @XmlAttribute(name = "CODIGO-DO-REGISTRO-OU-PATENTE")
    private String codigoDoRegistroOuPatente;

    @XmlAttribute(name = "TITULO-PATENTE")
    private String tituloPatente;

    @XmlAttribute(name = "FORMATO-DATA-PEDIDO")
    private String formatoDataPedido;

    @XmlJavaTypeAdapter(DateXmlAdapter.class)
    @XmlAttribute(name = "DATA-PEDIDO-DE-DEPOSITO")
    private Date dataPedidoDeDeposito;

    @XmlJavaTypeAdapter(DateXmlAdapter.class)

```

```

    @XmlAttribute(name = "DATA-DE-PEDIDO-DE-EXAME")
    private Date dataPedidoDeExame;

    @XmlJavaTypeAdapter(DateXmlAdapter.class)
    @XmlAttribute(name = "DATA-DE-CONCESSAO")
    private Date dataDeConcessao;

    @XmlAttribute(name = "INSTITUICAO-DEPOSITO-REGISTRO")
    private String instituicaoDepositoRegistro;

    @XmlAttribute(name = "NUMERO-DEPOSITO-PCT")
    private String numeroDepositoPCT;

    @XmlAttribute(name = "FORMATO-DATA-DEPOSITO-PCT")
    private String formatoDataDepositoPCT;

    @XmlJavaTypeAdapter(DateXmlAdapter.class)
    @XmlAttribute(name = "DATA-DEPOSITO-PCT")
    private Date dataDepositoPCT;

    @XmlAttribute(name = "NOME-DO-TITULAR")
    private String nomeDoTitular;
}
package ufsc.alpc.xml.dto.producao.tecnica.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.producao.tecnica.common.DetalhamentoRegistroPatenteXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoDesenhoTopografiaXmlDto extends DetalhamentoRegistroPatenteXmlDto {

    // REGISTRO-OU-PATENTE*

    // FINALIDADE CDATA #IMPLIED
    // INSTITUICAO-FINANCIADORA CDATA #IMPLIED
    // FINALIDADE-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "FINALIDADE")
    private String finalidade;

    @XmlAttribute(name = "INSTITUICAO-FINANCIADORA")
    private String instituicaoFinanciadora;

    @XmlAttribute(name = "FINALIDADE-INGLES")
    private String finalidadeIngles;
}
package ufsc.alpc.xml.dto.producao.tecnica.produto.tecnologico;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MeiodeDivulgacao;
import ufsc.alpc.xml.dto.domain.NaturezaDaPatente;
import ufsc.alpc.xml.dto.domain.TipoProduto;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDoProdutoTecnologicoXmlDto implements DadosBasicosProducao {

    // TIPO-PRODUTO (PILOTO | PROJETO | PROTOTIPO | OUTRO |
    //     ↪ NAO_INFORMADO)
    // #IMPLIED
    // NATUREZA (APARELHO | EQUIPAMENTO | FARMACOS_E_SIMILARES |
    //     ↪ INSTRUMENTO |

```

```

// OUTRA |
// NAO_INFORMADO) "NAO_INFORMADO"
// TITULO-DO-PRODUTO CDATA #IMPLIED
// ANO CDATA #IMPLIED
// PAIS CDATA #IMPLIED
// IDIOMA CDATA #IMPLIED
// MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
// ↪ MEIO_DIGITAL |
// FILME | HIPERTEXTO
// | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
// HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
// FLAG-RELEVANCIA (SIM | NAO) "NAO"
// DOI CDATA #IMPLIED
// TITULO-DO-PRODUTO-INGLES CDATA #IMPLIED
// FLAG-POTENCIAL-INOVACAO (SIM | NAO) "NAO"

@XmlAttribute(name = "TIPO-PRODUTO")
protected TipoProduto tipoProduto;

@XmlAttribute(name = "NATUREZA")
protected NaturezaDaPatente natureza;

@XmlAttribute(name = "TITULO-DO-PRODUTO")
protected String titulo;

@XmlAttribute(name = "ANO")
protected String ano;

@XmlAttribute(name = "PAIS")
protected String pais;

@XmlAttribute(name = "IDIOMA")
protected String idioma;

@XmlAttribute(name = "MEIO-DE-DIVULGACAO")
protected MeioDeDivulgacao meioDeDivulgacao;

@XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
protected String homePageDoTrabalho;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-RELEVANCIA")
protected Boolean flagRelevancia;

@XmlAttribute(name = "DOI")
protected String DOI;

@XmlAttribute(name = "TITULO-DO-PRODUTO-INGLES")
protected String tituloDoProdutoIngles;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-POTENCIAL-INOVACAO")
protected Boolean flagPotencialInovacao;
}
package ufsc.alpc.xml.dto.producao.tecnica.produto.tecnologico;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.producao.tecnica.common.DetalhamentoRegistroPatenteXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoDoProdutoTecnologicoXmlDto extends
    ↪ DetalhamentoRegistroPatenteXmlDto {

// REGISTRO-OU-PATENTE*

// FINALIDADE CDATA #IMPLIED
// DISPONIBILIDADE CDATA #IMPLIED
// CIDADE-DO-PRODUTO CDATA #IMPLIED
// INSTITUICAO-FINANCIADORA CDATA #IMPLIED
// FINALIDADE-INGLES CDATA #IMPLIED

@XmlAttribute(name = "FINALIDADE")

```

```

private String finalidade;

@XmlAttribute(name = "DISPONIBILIDADE")
private String disponibilidade;

@XmlAttribute(name = "CIDADE-DO-PRODUTO")
private String cidadeDoProduto;

@XmlAttribute(name = "INSTITUICAO-FINANCIADORA")
private String instituicaoFinanciadora;

@XmlAttribute(name = "FINALIDADE-INGLES")
private String finalidadeIngles;
}
package ufsc.alpc.xml.dto.producao.tecnica.produto.tecnologico;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ProdutoTecnologicoXmlDto extends
↳ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {
    // DADOS-BASICOS-DO-PRODUTO-TECNOLOGICO,
    // DETALHAMENTO-DO-PRODUTO-TECNOLOGICO,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DO-PRODUTO-TECNOLOGICO")
private DadosBasicosDoProdutoTecnologicoXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DO-PRODUTO-TECNOLOGICO")
private DetalhamentoDoProdutoTecnologicoXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.editoracao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class EditoracaoXmlDto extends AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto
↳ implements Producao{
    // DADOS-BASICOS-DE-EDITORACAO?,
    // DETALHAMENTO-DE-EDITORACAO?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DE-EDITORACAO")
private DadosBasicosEditoracaoXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DE-EDITORACAO")
private DetalhamentoEditoracaoXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.editoracao;

```

```

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MeioDeDivulgacao;
import ufsc.alpc.xml.dto.domain.NaturezaLivro;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosEditoracaoXmlDto implements DadosBasicosProducao {

    /*
     * NATUREZA (LIVRO | ANAIS | CATALOGO | COLETANEA | ENCICLOPEDIA |
     *   ↳ PERIODICO | OUTRA |
     * NAO - INFORMADO) #IMPLIED
     * TITULO CDATA #IMPLIED
     * ANO CDATA #IMPLIED
     * PAIS CDATA #IMPLIED
     * IDIOMA CDATA #IMPLIED
     * MEIO - DE - DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
     *   ↳ MEIO_DIGITAL | FILME | HIPERTEXTO
     * | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
     * HOME - PAGE - DO - TRABALHO CDATA #IMPLIED
     * FLAG - RELEVANCIA (SIM | NAO) "NAO"
     * DOI CDATA #IMPLIED
     * TITULO - INGLES CDATA #IMPLIED
     */

    @XmlAttribute(name = "NATUREZA")
    private NaturezaLivro natureza;

    @XmlAttribute(name = "TITULO")
    private String titulo;

    @XmlAttribute(name = "ANO")
    private String ano;

    @XmlAttribute(name = "PAIS")
    private String pais;

    @XmlAttribute(name = "IDIOMA")
    private String idioma;

    @XmlAttribute(name = "MEIO-DE-DIVULGACAO")
    private MeioDeDivulgacao meioDivulgacao;

    @XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
    private String homePageTrabalho;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-RELEVANCIA")
    private Boolean flagRelevancia;

    @XmlAttribute(name = "DOI")
    private String doi;

    @XmlAttribute(name = "TITULO-INGLES")
    private String tituloIngles;
}
package ufsc.alpc.xml.dto.producao.technica.demais.producoes.editoracao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)

```

```

@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoEditoracaoXmlDto {

    /*
     * NUMERO-DE-PAGINAS CDATA #IMPLIED
     * INSTITUICAO-PROMOTORA CDATA #IMPLIED
     * EDITORA CDATA #IMPLIED
     * CIDADE CDATA #IMPLIED
     */

    @XmlAttribute(name = "NUMERO-DE-PAGINAS")
    private String numeroPaginas;

    @XmlAttribute(name = "INSTITUICAO-PROMOTORA")
    private String instituicaoPromotora;

    @XmlAttribute(name = "EDITORA")
    private String editora;

    @XmlAttribute(name = "CIDADE")
    private String cidade;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.programa.radio;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ProgramaDeRadioTVXmlDto extends
    ↳ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {

    // DADOS-BASICOS-DO-PROGRAMA-DE-RADIO-OU-TV?,
    // DETALHAMENTO-DO-PROGRAMA-DE-RADIO-OU-TV?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DO-PROGRAMA-DE-RADIO-OU-TV")
    private DadosBasicosProgramaRadioTvXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DO-PROGRAMA-DE-RADIO-OU-TV")
    private DetalhamentoProgramaRadioTvXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.programa.radio;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MeiDeDivulgacao;
import ufsc.alpc.xml.dto.domain.NaturezaProgramaRadioTv;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosProgramaRadioTvXmlDto implements DadosBasicosProducao {

    /*
     * NATUREZA (ENTREVISTA | MESA_REDONDA | COMENTARIO | PROGRAMA |
     * ↳ OUTRA | NAO_INFORMADO)
     * #IMPLIED
     * TITULO CDATA #IMPLIED
     * ANO CDATA #IMPLIED

```

```

* PAIS CDATA #IMPLIED
* IDIOMA CDATA #IMPLIED
* FLAG-RELEVANCIA (SIM | NAO) "NAO"
* DOI CDATA #IMPLIED
* TITULO-INGLES CDATA #IMPLIED
* HOME-PAGE CDATA #IMPLIED
* MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
  ↳ MEIO_DIGITAL | FILME | HIPERTEXTO
* | OUTRO | VÁRIOS | NAO_INFORMADO) "NAO_INFORMADO"
* FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"
*/

@XmlAttribute(name = "NATUREZA")
private NaturezaProgramaRadioTv natureza;

@XmlAttribute(name = "TITULO")
private String titulo;

@XmlAttribute(name = "ANO")
private String ano;

@XmlAttribute(name = "PAIS")
private String pais;

@XmlAttribute(name = "IDIOMA")
private String idioma;

@XmlAttribute(name = "MEIO-DE-DIVULGACAO")
private MeioDeDivulgacao meioDeDivulgacao;

@XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
private String homePage;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-RELEVANCIA")
private Boolean flagRelevancia;

@XmlAttribute(name = "DOI")
private String doi;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-DIVULGACAO-CIENTIFICA")
private Boolean flagDivulgacaoCientifica;

@XmlAttribute(name = "TITULO-INGLES")
private String tituloIngles;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.programa.radio;

import java.util.Date;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.DateXmlAdapter;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoProgramaRadioTvXmlDto {

    /*
    * EMISSORA CDATA #IMPLIED
    * TEMA CDATA #IMPLIED
    * FORMATO-DATA-DA-APRESENTACAO NMTOKEN #FIXED "DDMMAAAA"
    * DATA-DA-APRESENTACAO CDATA #IMPLIED
    * DURACAO-EM-MINUTOS CDATA #IMPLIED
    * CIDADE CDATA #IMPLIED
    * VEICULO-DE-DIVULGACAO CDATA #IMPLIED
    */

    @XmlAttribute(name = "EMISSORA")
    private String emissora;

    @XmlAttribute(name = "TEMA")

```

```

private String tema;

@XmlJavaTypeAdapter(DateXmlAdapter.class)
@XmlAttribute(name = "DATA-DA-APRESENTACAO")
private Date dataApresentacao;

@XmlAttribute(name = "DURACAO-EM-MINUTOS")
private String duracaoEmMinutos;

@XmlAttribute(name = "CIDADE")
private String cidade;

@XmlAttribute(name = "VEICULO-DE-DIVULGACAO")
private String veiculoDivulgacao;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.manutencao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.NaturezaManutencaoObraArtistica;
import ufsc.alpc.xml.dto.domain.TipoManutencaoObraArtistica;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosManutencaoObraArtisticaXmlDto implements DadosBasicosProducao {

    /*
    * TIPO (CONSERVACAO | RESTAURACAO | OUTRO | NAO_INFORMADO) #IMPLIED
    * NATUREZA (ARQUITETURA | DESENHO | ESCULTURA | FOTOGRAFIA |
    *   ↳ GRAVURA | OUTRA | PINTURA |
    * NAO_INFORMADO) "NAO_INFORMADO"
    * TITULO CDATA #IMPLIED
    * ANO CDATA #IMPLIED
    * PAIS CDATA #IMPLIED
    * IDIOMA CDATA #IMPLIED
    * FLAG-RELEVANCIA (SIM | NAO) "NAO"
    * DOI CDATA #IMPLIED
    * TITULO-INGLES CDATA #IMPLIED
    */

    @XmlAttribute(name = "TIPO")
    private TipoManutencaoObraArtistica tipo;

    @XmlAttribute(name = "NATUREZA")
    private NaturezaManutencaoObraArtistica natureza;

    @XmlAttribute(name = "TITULO")
    private String titulo;

    @XmlAttribute(name = "ANO")
    private String ano;

    @XmlAttribute(name = "PAIS")
    private String pais;

    @XmlAttribute(name = "IDIOMA")
    private String idioma;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-RELEVANCIA")
    private Boolean flagRelevancia;

    @XmlAttribute(name = "DOI")
    private String doi;

    @XmlAttribute(name = "TITULO-INGLES")
    private String tituloIngles;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.manutencao;

```



```

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ManutencaoDeObraArtisticaXmlDto extends
    ↳ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {
    // DADOS-BASICOS-DE-MANUTENCAO-DE-OBRA-ARTISTICA?,
    // DETALHAMENTO-DE-MANUTENCAO-DE-OBRA-ARTISTICA?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↳ "DADOS-BASICOS-DE-MANUTENCAO-DE-OBRA-ARTISTICA")
    private DadosBasicosManutencaoObraArtisticaXmlDto dadosBasicos;

    @XmlElement(name =
        ↳ "DETALHAMENTO-DE-MANUTENCAO-DE-OBRA-ARTISTICA")
    private DetalhamentoManutencaoObraArtisticaXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.technica.demais.producoes.manutencao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.domain.Acervo;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoManutencaoObraArtisticaXmlDto {

    /*
     * NOME-DA-OBRA CDATA #IMPLIED
     * AUTOR-DA-OBRA CDATA #IMPLIED
     * ANO-DA-OBRA CDATA #IMPLIED
     * ACERVO (PUBLICO | PRIVADO | NAO_INFORMADO) "NAO_INFORMADO"
     * LOCAL CDATA #IMPLIED
     * CIDADE CDATA #IMPLIED
     */

    @XmlAttribute(name = "NOME-DA-OBRA")
    private String nomeObra;

    @XmlAttribute(name = "AUTOR-DA-OBRA")
    private String autorObra;

    @XmlAttribute(name = "ANO-DA-OBRA")
    private String anoObra;

    @XmlAttribute(name = "ACERVO")
    private Acervo acervo;

    @XmlAttribute(name = "LOCAL")
    private String local;

    @XmlAttribute(name = "CIDADE")
    private String cidade;
}
package ufsc.alpc.xml.dto.producao.technica.demais.producoes.carta.mapa;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;

```

```

import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MeioDeDivulgacao;
import ufsc.alpc.xml.dto.domain.NaturezaCartaOuSimilar;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDaCartaOuSimilarXmlDto implements DadosBasicosProducao {

    /*
     * NATUREZA (AEROFOTOGRAMA | CARTA | FOTOGRAMA | MAPA | OUTRA |
     * ↔ NAO_INFORMADO) #IMPLIED
     * TITULO CDATA #IMPLIED
     * ANO CDATA #IMPLIED
     * PAIS CDATA #IMPLIED
     * IDIOMA CDATA #IMPLIED
     * MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
     * ↔ MEIO_DIGITAL | FILME | HIPERTEXTO
     * | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
     * HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
     * FLAG-RELEVANCIA (SIM | NAO) "NAO"
     * DOI CDATA #IMPLIED
     * TITULO-INGLES CDATA #IMPLIED
     */

    @XmlAttribute(name = "NATUREZA")
    private NaturezaCartaOuSimilar natureza;

    @XmlAttribute(name = "TITULO")
    private String titulo;

    @XmlAttribute(name = "ANO")
    private String ano;

    @XmlAttribute(name = "PAIS")
    private String pais;

    @XmlAttribute(name = "IDIOMA")
    private String idioma;

    @XmlAttribute(name = "MEIO-DE-DIVULGACAO")
    private MeioDeDivulgacao meioDivulgacao;

    @XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
    private String homePage;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-RELEVANCIA")
    private Boolean flagRelevancia;

    @XmlAttribute(name = "DOI")
    private String doi;

    @XmlAttribute(name = "TITULO-INGLES")
    private String tituloIngles;
}

package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.carta.mapa;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class CartaMapaSimilarXmlDto extends

```

```

↔ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao{
    // DADOS-BASICOS-DE-CARTA-MAPA-OU-SIMILAR?,
    // DETALHAMENTO-DA-APRESENTACAO-DE-TRABALHO?,
    // AUTORES+,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DE-CARTA-MAPA-OU-SIMILAR")
    private DadosBasicosDaCartaOuSimilarXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DE-CARTA-MAPA-OU-SIMILAR")
    private DetalhamentoDaCartaOuSimilarXmlDto detalhamento;
}

package ufsc.alpc.xml.dto.producao.technica.demais.producoes.carta.mapa;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoDaCartaOuSimilarXmlDto {

    /*
     * TEMA-DA-CARTA-MAPA-OU-SIMILAR CDATA #IMPLIED
     * TECNICA-UTILIZADA CDATA #IMPLIED
     * FINALIDADE CDATA #IMPLIED
     * AREA-REPRESENTADA CDATA #IMPLIED
     * INSTITUICAO-FINANCIADORA CDATA #IMPLIED
     * FINALIDADE-INGLES CDATA #IMPLIED
     */

    @XmlAttribute(name = "TEMA-DA-CARTA-MAPA-OU-SIMILAR")
    private String temaCartaMapaSimilar;

    @XmlAttribute(name = "TECNICA-UTILIZADA")
    private String tecnicaUtilizada;

    @XmlAttribute(name = "FINALIDADE")
    private String finalidade;

    @XmlAttribute(name = "AREA-REPRESENTADA")
    private String areaRepresentada;

    @XmlAttribute(name = "INSTITUICAO-FINANCIADORA")
    private String instituicaoFinanciadora;

    @XmlAttribute(name = "FINALIDADE-INGLES")
    private String finalidadeIngles;
}

package ufsc.alpc.xml.dto.producao.technica.demais.producoes;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.producao.technica.demais.producoes.apresentacao.trabalho.ApresentacaoDeTrabalhoXmlDto;
import ufsc.alpc.xml.dto.producao.technica.demais.producoes.carta.mapa.CartaMapaSimilarXmlDto;
import ufsc.alpc.xml.dto.producao.technica.demais.producoes.curso.CursoDeCurtaDuracaoMinistradoXmlDto;
import ufsc.alpc.xml.dto.producao.technica.demais.producoes.desenvolvimento.DesenvolvimentoDeMaterialDidatico;
import ufsc.alpc.xml.dto.producao.technica.demais.producoes.editoracao.EditoracaoXmlDto;
import ufsc.alpc.xml.dto.producao.technica.demais.producoes.manutencao.ManutencaoDeObraArtisticaXmlDto;

```

```

import ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.maquete.MaqueteXmlDto;
import
↳ ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.midia.social.MidiaSocialWebsiteBlogXmlDto;
import
↳ ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.organizacao.OrganizacaoDeEventoXmlDto;
import ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.outra.OutraProducaoTecnicaXmlDto;
import
↳ ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.programa.radio.ProgramaDeRadioTVXmlDto;
import
↳ ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.relatorio.pesquisa.RelatorioDePesquisaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DemaisTipoDeProducaoTecnicaXmlDto {
    // APRESENTACAO-DE-TRABALHO*,
    // CARTA-MAPA-OU-SIMILAR*,
    // CURSO-DE-CURTA-DURACAO-MINISTRADO*,
    // DESENVOLVIMENTO-DE-MATERIAL-DIDATICO-OU-INSTRUCIONAL*,
    // EDITORACAO*,
    // MANUTENCAO-DE-OBRA-ARTISTICA*,
    // MAQUETE*,
    // ORGANIZACAO-DE-EVENTO*,
    // PROGRAMA-DE-RADIO-OU-TV*,
    // RELATORIO-DE-PESQUISA*,
    // MIDIA-SOCIAL-WEBSITE-BLOG*,
    // OUTRA-PRODUCAO-TECNICA*

    @XmlElement(name = "APRESENTACAO-DE-TRABALHO")
    private List<ApresentacaoDeTrabalhoXmlDto> apresentacaoTrabalho;

    @XmlElement(name = "CARTA-MAPA-OU-SIMILAR")
    private List<CartaMapaSimilarXmlDto> cartaMapaSimilar;

    @XmlElement(name = "CURSO-DE-CURTA-DURACAO-MINISTRADO")
    private List<CursoDeCurtaDuracaoMinistradoXmlDto> cursoCurtaDuracao;

    @XmlElement(name =
↳ "DESENVOLVIMENTO-DE-MATERIAL-DIDATICO-OU-INSTRUCIONAL")
    private List<DesenvolvimentoDeMaterialDidaticoInstrucionalXmlDto>
↳ desenvolvimentoMaterialDidaticoInstrucional;

    @XmlElement(name = "EDITORACAO")
    private List<EditoracaoXmlDto> editoracao;

    @XmlElement(name = "MANUTENCAO-DE-OBRA-ARTISTICA")
    private List<ManutencaoDeObraArtisticaXmlDto> manutencaoObraArtistica;

    @XmlElement(name = "MAQUETE")
    private List<MaqueteXmlDto> maquete;

    @XmlElement(name = "ORGANIZACAO-DE-EVENTO")
    private List<OrganizacaoDeEventoXmlDto> organizacaoEvento;

    @XmlElement(name = "PROGRAMA-DE-RADIO-OU-TV")
    private List<ProgramaDeRadioTVXmlDto> programaRadioTv;

    @XmlElement(name = "RELATORIO-DE-PESQUISA")
    private List<RelatorioDePesquisaXmlDto> relatorioPesquisa;

    @XmlElement(name = "MIDIA-SOCIAL-WEBSITE-BLOG")
    private List<MidiaSocialWebsiteBlogXmlDto> midiaSocialWebsiteBlog;

    @XmlElement(name = "OUTRA-PRODUCAO-TECNICA")
    private List<OutraProducaoTecnicaXmlDto> outraProducaoTecnica;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.midia.social;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)

```

```

@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoMidiaSocialWebsiteXmlDto {

    // TEMA CDATA #IMPLIED

    @XmlAttribute(name = "TEMA")
    private String tema;

}

package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.midia.social;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class MidiaSocialWebsiteBlogXmlDto extends
    AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {

    // DADOS-BASICOS-DA-MIDIA-SOCIAL-WEBSITE-BLOG??
    // DETALHAMENTO-DA-MIDIA-SOCIAL-WEBSITE-BLOG?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DA-MIDIA-SOCIAL-WEBSITE-BLOG")
    private DadosBasicosMidiaSocialWebsiteXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DA-MIDIA-SOCIAL-WEBSITE-BLOG")
    private DetalhamentoMidiaSocialWebsiteXmlDto detalhamento;

}

package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.midia.social;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MedioDeDivulgacao;
import ufsc.alpc.xml.dto.domain.NaturezaMidiaSocialWebsite;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosMidiaSocialWebsiteXmlDto implements DadosBasicosProducao {

    /*
     * NATUREZA (REDE_SOCIAL | FORUM | BLOG | SITE) #IMPLIED
     * NATUREZA-INGLES CDATA #IMPLIED
     * TITULO CDATA #IMPLIED
     * TITULO-INGLES CDATA #IMPLIED
     * ANO CDATA #IMPLIED
     * PAIS CDATA #IMPLIED
     * IDIOMA CDATA #IMPLIED
     * HOME-PAGE CDATA #IMPLIED
     * FLAG-RELEVANCIA (SIM | NAO) "NAO"
     * FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"
     */
    @XmlAttribute(name = "NATUREZA")
    private NaturezaMidiaSocialWebsite natureza;

    @XmlAttribute(name = "NATUREZA-INGLES")
    private String naturezaIngles;

    @XmlAttribute(name = "TITULO")

```

```

private String titulo;

@XmlAttribute(name = "ANO")
private String ano;

@XmlAttribute(name = "PAIS")
private String pais;

@XmlAttribute(name = "IDIOMA")
private String idioma;

@XmlAttribute(name = "MEIO-DE-DIVULGACAO")
private MeioDeDivulgacao meioDeDivulgacao;

@XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
private String homepage;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-RELEVANCIA")
private Boolean flagRelevancia;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-DIVULGACAO-CIENTIFICA")
private Boolean flagDivulgacaoCientifica;

@XmlAttribute(name = "TITULO-INGLES")
private String tituloIngles;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.curso;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MeioDeDivulgacao;
import ufsc.alpc.xml.dto.domain.NivelCursoMinistrado;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosCursosCurtaDuracaoMinistradoXmlDto implements DadosBasicosProducao
↳ {

    /*
    * NIVEL-DO-CURSO (EXTENSAO | APERFEICOAMENTO | ESPECIALIZACAO |
    ↳ OUTRA | NAO_INFORMADO)
    * #IMPLIED
    * TITULO CDATA #IMPLIED
    * ANO CDATA #IMPLIED
    * PAIS CDATA #IMPLIED
    * IDIOMA CDATA #IMPLIED
    * MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    ↳ MEIO_DIGITAL | FILME | HIPERTEXTO
    * | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    * HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
    * FLAG-RELEVANCIA (SIM | NAO) "NAO"
    * DOI CDATA #IMPLIED
    * TITULO-INGLES CDATA #IMPLIED
    * FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"
    */

    @XmlAttribute(name = "NIVEL-DO-CURSO")
private NivelCursoMinistrado nivelCurso;

    @XmlAttribute(name = "TITULO")
private String titulo;

    @XmlAttribute(name = "ANO")
private String ano;

    @XmlAttribute(name = "PAIS")
private String pais;

    @XmlAttribute(name = "IDIOMA")

```

```

private String idioma;

@XmlAttribute(name = "MEIO-DE-DIVULGACAO")
private MeioDeDivulgacao meioDivulgacao;

@XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
private String homePage;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-RELEVANCIA ")
private Boolean flagRelevancia;

@XmlAttribute(name = "DOI")
private String doi;

@XmlAttribute(name = "TITULO-INGLES ")
private String tituloIngles;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-DIVULGACAO-CIENTIFICA")
private Boolean flagDivulgacaoCientifica;
}
package ufsc.alpc.xml.dto.producao.technica.demais.producoes.curso;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class CursoDeCurtaDuracaoMinistradoXmlDto extends
    ↳ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {

    // DADOS-BASICOS-DE-CURSOS-CURTA-DURACAO-MINISTRADO?,
    // DETALHAMENTO-DA-APRESENTACAO-DE-TRABALHO?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↳ "DADOS-BASICOS-DE-CURSOS-CURTA-DURACAO-MINISTRADO")
private DadosBasicosCursosCurtaDuracaoMinistradoXmlDto dadosBasicos;

    @XmlElement(name =
        ↳ "DETALHAMENTO-DE-CURSOS-CURTA-DURACAO-MINISTRADO")
private DetalhamentoCursosCurtaDuracaoMinistradoXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.technica.demais.producoes.curso;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.domain.ParticipacaoAutores;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoCursosCurtaDuracaoMinistradoXmlDto {

    /*
    * PARTICIPACAO-DOS-AUTORES (DOCENTE | ORGANIZADOR | OUTRA |
    ↳ NAO_INFORMADO) #IMPLIED
    * INSTITUICAO-PROMOTORA-DO-CURSO CDATA #IMPLIED
    * LOCAL-DO-CURSO CDATA #IMPLIED
    * CIDADE CDATA #IMPLIED

```

```

    * DURACAO CDATA #IMPLIED
    * UNIDADE CDATA #IMPLIED
    * UNIDADE-INGLES CDATA #IMPLIED
    */

    @XmlAttribute(name = "PARTICIPACAO-DOS-AUTORES")
    private ParticipacaoAutores participacaoAutores;

    @XmlAttribute(name = "INSTITUICAO-PROMOTORA-DO-CURSO")
    private String instituicaoPromotoraCurso;

    @XmlAttribute(name = "LOCAL-DO-CURSO")
    private String localCurso;

    @XmlAttribute(name = "CIDADE")
    private String cidade;

    @XmlAttribute(name = "DURACAO")
    private String duracao;

    @XmlAttribute(name = "UNIDADE")
    private String unidade;

    @XmlAttribute(name = "UNIDADE-INGLES")
    private String unidadeIngles;
}

package ufsc.alpc.xml.dto.producao.technica.demais.producoes.organizacao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OrganizacaoDeEventoXmlDto extends
    AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {

    // DADOS-BASICOS-DA-ORGANIZACAO-DE-EVENTO?,
    // DETALHAMENTO-DA-ORGANIZACAO-DE-EVENTO?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DA-ORGANIZACAO-DE-EVENTO")
    private DadosBasicosOrganizacaoEventoXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DA-ORGANIZACAO-DE-EVENTO")
    private DetalhamentoOrganizacaoEventoXmlDto detalhamento;
}

package ufsc.alpc.xml.dto.producao.technica.demais.producoes.organizacao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoOrganizacaoEventoXmlDto {
    /*
    * INSTITUICAO-PROMOTORA CDATA #IMPLIED
    * DURACAO-EM-SEMANAS CDATA #IMPLIED
    * FLAG-EVENTO-ITINERANTE CDATA #IMPLIED
    */

```



```

* FLAG-CATALOGO CDATA #IMPLIED
* LOCAL CDATA #IMPLIED
* CIDADE CDATA #IMPLIED
*/

@XmlAttribute(name = "INSTITUICAO-PROMOTORA")
private String instituicaoPromotora;

@XmlAttribute(name = "DURACAO-EM-SEMANAS")
private String duracaoSemanas;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-EVENTO-ITINERANTE")
private Boolean flagEventoItinerante;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-CATALOGO")
private Boolean flagCatalogo;

@XmlAttribute(name = "LOCAL")
private String local;

@XmlAttribute(name = "CIDADE")
private String cidade;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.organizacao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MeloDeDivulgacao;
import ufsc.alpc.xml.dto.domain.NaturezaOrganizacaoEvento;
import ufsc.alpc.xml.dto.domain.TipoOrganizacaoEvento;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosOrganizacaoEventoXmlDto implements DadosBasicosProducao {

    /*
    * TIPO (CONCERTO | CONCURSO | CONGRESSO | EXPOSICAO | FESTIVAL | FEIRA
    *   ↪ | OLIMPADA | OUTRO
    * | NAO_INFORMADO) #IMPLIED
    * NATUREZA (CURADORIA | MONTAGEM | MUSEOLOGIA | ORGANIZACAO |
    *   ↪ NAO_INFORMADO) "NAO_INFORMADO"
    * TITULO CDATA #IMPLIED
    * ANO CDATA #IMPLIED
    * PAIS CDATA #IMPLIED
    * IDIOMA CDATA #IMPLIED
    * MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    *   ↪ MEIO_DIGITAL | FILME | HIPERTEXTO
    * | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    * HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
    * FLAG-RELEVANCIA (SIM | NAO) "NAO"
    * DOI CDATA #IMPLIED
    * TITULO-INGLES CDATA #IMPLIED
    * FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"
    */

    @XmlAttribute(name = "TIPO")
    private TipoOrganizacaoEvento tipo;

    @XmlAttribute(name = "NATUREZA")
    private NaturezaOrganizacaoEvento natureza;

    @XmlAttribute(name = "TITULO")
    private String titulo;

    @XmlAttribute(name = "ANO")
    private String ano;

    @XmlAttribute(name = "PAIS")
    private String pais;

```

```

@XmlAttribute(name = "IDIOMA")
private String idioma;

@XmlAttribute(name = "MEIO-DE-DIVULGACAO")
private MeioDeDivulgacao meioDeDivulgacao;

@XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
private String homePage;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-RELEVANCIA")
private Boolean flagRelevancia;

@XmlAttribute(name = "DOI")
private String doi;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-DIVULGACAO-CIENTIFICA")
private Boolean flagDivulgacaoCientifica;

@XmlAttribute(name = "TITULO-INGLES")
private String tituloIngles;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.apresentacao.trabalho;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.NaturezaApresentacaoTrabalho;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDaApresentacaoXmlDto implements DadosBasicosProducao {

    /*
     * NATUREZA (COMUNICACAO | CONFERENCIA | CONGRESSO | SEMINARIO |
     * ↳ SIMPOSIO | OUTRA | NAO_INFORMADO) #IMPLIED
     * TITULO CDATA #IMPLIED
     * ANO CDATA #IMPLIED
     * PAIS CDATA #IMPLIED
     * IDIOMA CDATA #IMPLIED
     * FLAG-RELEVANCIA (SIM | NAO) "NAO"
     * DOI CDATA #IMPLIED
     * TITULO-INGLES CDATA #IMPLIED
     * FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"
     */

    @XmlAttribute(name = "NATUREZA")
    private NaturezaApresentacaoTrabalho natureza;

    @XmlAttribute(name = "TITULO")
    private String titulo;

    @XmlAttribute(name = "ANO")
    private String ano;

    @XmlAttribute(name = "PAIS")
    private String pais;

    @XmlAttribute(name = "IDIOMA")
    private String idioma;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-RELEVANCIA")
    private Boolean flagRelevancia;

    @XmlAttribute(name = "DOI")
    private String doi;

    @XmlAttribute(name = "TITULO-INGLES")
    private String tituloIngles;
}

```

```

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-DIVULGACAO-CIENTIFICA")
private Boolean flagDivulgacaoCientifica;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.apresentacao.trabalho;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoDaApresentacaoXmlDto {

    /*
     * NOME-DO-EVENTO CDATA #IMPLIED
     * INSTITUICAO-PROMOTORA CDATA #IMPLIED
     * LOCAL-DA-APRESENTACAO CDATA #IMPLIED
     * CIDADE-DA-APRESENTACAO CDATA #IMPLIED
     * NOME-DO-EVENTO-INGLES CDATA #IMPLIED
     */

    @XmlAttribute(name = "NOME-DO-EVENTO")
    private String nomeEvento;

    @XmlAttribute(name = "INSTITUICAO-PROMOTORA")
    private String instituicaoPromotora;

    @XmlAttribute(name = "LOCAL-DA-APRESENTACAO")
    private String localApresentacao;

    @XmlAttribute(name = "CIDADE-DA-APRESENTACAO")
    private String cidadeApresentacao;

    @XmlAttribute(name = "NOME-DO-EVENTO-INGLES")
    private String nomeEventoIngles;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.apresentacao.trabalho;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ApresentacaoDeTrabalhoXmlDto extends
    ↪ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {

    // DADOS-BASICOS-DA-APRESENTACAO-DE-TRABALHO?,
    // DETALHAMENTO-DA-APRESENTACAO-DE-TRABALHO?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DA-APRESENTACAO-DE-TRABALHO")
    private DadosBasicosDaApresentacaoXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DA-APRESENTACAO-DE-TRABALHO")
    private DetalhamentoDaApresentacaoXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.outra;

import javax.xml.bind.annotation.XmlAccessType;

```

```

import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoOutraProducaoTecnicaXmlDto {

    /*
     * FINALIDADE CDATA #IMPLIED
     * INSTITUICAO-PROMOTORA CDATA #IMPLIED
     * LOCAL CDATA #IMPLIED
     * CIDADE CDATA #IMPLIED
     * FINALIDADE-INGLES CDATA #IMPLIED
     */

    @XmlAttribute(name = "FINALIDADE")
    private String finalidade;

    @XmlAttribute(name = "INSTITUICAO-PROMOTORA")
    private String instituicaoPromotora;

    @XmlAttribute(name = "LOCAL")
    private String local;

    @XmlAttribute(name = "CIDADE")
    private String cidade;

    @XmlAttribute(name = "FINALIDADE-INGLES")
    private String finalidadeIngles;

}

package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.outra;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OutraProducaoTecnicaXmlDto extends
    ↳ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {

    // DADOS-BASICOS-DE-OUTRA-PRODUCAO-TECNICA?,
    // DETALHAMENTO-DE-OUTRA-PRODUCAO-TECNICA?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DE-OUTRA-PRODUCAO-TECNICA")
    private DadosBasicosOutraProducaoTecnicaXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DE-OUTRA-PRODUCAO-TECNICA")
    private DetalhamentoOutraProducaoTecnicaXmlDto detalhamento;

}

package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.outra;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MedioDeDivulgacao;

```

```

import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosOutraProducaoTecnicaXmlDto implements DadosBasicosProducao {
    // NATUREZA CDATA #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    // ↔ MEIO_DIGITAL | FILME | HIPERTEXTO
    // | OUTRO | VÁRIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // IDIOMA CDATA #IMPLIED
    // HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED
    // NATUREZA-INGLES CDATA #IMPLIED
    // FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"

    @XmlAttribute(name = "NATUREZA")
    private String natureza;

    @XmlAttribute(name = "TITULO")
    private String titulo;

    @XmlAttribute(name = "ANO")
    private String ano;

    @XmlAttribute(name = "PAIS")
    private String pais;

    @XmlAttribute(name = "IDIOMA")
    private String idioma;

    @XmlAttribute(name = "MEIO-DE-DIVULGACAO")
    private MeioDeDivulgacao meioDeDivulgacao;

    @XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
    private String homePage;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-RELEVANCIA")
    private Boolean flagRelevancia;

    @XmlAttribute(name = "DOI")
    private String doi;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-DIVULGACAO-CIENTIFICA")
    private Boolean flagDivulgacaoCientifica;

    @XmlAttribute(name = "NATUREZA-INGLES")
    private String naturezaIngles;

    @XmlAttribute(name = "TITULO-INGLES")
    private String tituloIngles;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.maquete;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MeioDeDivulgacao;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosMaqueteXmlDto implements DadosBasicosProducao {

```

```

/*
 * TITULO CDATA #IMPLIED
 * ANO CDATA #IMPLIED
 * PAIS CDATA #IMPLIED
 * IDIOMA CDATA #IMPLIED
 * MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
 * ↳ MEIO_DIGITAL | FILME | HIPERTEXTO
 * | OUTRO | VÁRIOS | NAO_INFORMADO) "NAO_INFORMADO"
 * HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
 * FLAG-RELEVANCIA (SIM | NAO) "NAO"
 * DOI CDATA #IMPLIED
 * TITULO-INGLES CDATA #IMPLIED
 */

@XmlAttribute(name = "TITULO")
private String titulo;

@XmlAttribute(name = "ANO")
private String ano;

@XmlAttribute(name = "PAIS")
private String pais;

@XmlAttribute(name = "IDIOMA")
private String idioma;

@XmlAttribute(name = "MEIO-DE-DIVULGACAO")
private MeioDeDivulgacao meioDeDivulgacao;

@XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
private String homePage;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-RELEVANCIA")
private Boolean flagRelevancia;

@XmlAttribute(name = "DOI")
private String doi;

@XmlAttribute(name = "TITULO-INGLES")
private String tituloIngles;
}
package ufsc.alpc.xml.dto.producao.technica.demais.producoes.maquete;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoMaqueteXmlDto {

    /*
     * FINALIDADE CDATA #IMPLIED
     * OBJETO-REPRESENTADO CDATA #IMPLIED
     * MATERIAL-UTILIZADO CDATA #IMPLIED
     * INSTITUICAO-FINANCIADORA CDATA #IMPLIED
     * FINALIDADE-INGLES CDATA #IMPLIED
     */

    @XmlAttribute(name = "FINALIDADE")
    private String finalidade;

    @XmlAttribute(name = "OBJETO-REPRESENTADO")
    private String objetoRepresentado;

    @XmlAttribute(name = "MATERIAL-UTILIZADO")
    private String materialUtilizado;

    @XmlAttribute(name = "INSTITUICAO-FINANCIADORA")
    private String instituicaoFinanciadora;

    @XmlAttribute(name = "FINALIDADE-INGLES")
    private String finalidadeIngles;
}

```

```

}
package ufsc.alpc.xml.dto.producao.technica.demais.producoes.maquete;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class MaqueteXmlDto extends AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto
    ↪ implements Producao {

    // DADOS-BASICOS-DA-MAQUETE?,
    // DETALHAMENTO-DA-MAQUETE?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DA-MAQUETE")
    private DadosBasicosMaqueteXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DA-MAQUETE")
    private DetalhamentoMaqueteXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.technica.demais.producoes.desenvolvimento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DesenvolvimentoDeMaterialDidaticoInstrucionalXmlDto extends
    ↪ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {

    // DADOS-BASICOS-DO-MATERIAL-DIDATICO-OU-INSTRUCIONAL??,
    // DETALHAMENTO-DO-MATERIAL-DIDATICO-OU-INSTRUCIONAL?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↪ "DADOS-BASICOS-DO-MATERIAL-DIDATICO-OU-INSTRUCIONAL")
    private DadosBasicosMaterialDidaticoInstrucionalXmlDto dadosBasicos;

    @XmlElement(name =
        ↪ "DETALHAMENTO-DO-MATERIAL-DIDATICO-OU-INSTRUCIONAL")
    private DetalhamentoMaterialDidaticoInstrucionalXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.technica.demais.producoes.desenvolvimento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MedioDeDivulgacao;

```

```

import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosMaterialDidaticoInstrucionalXmlDto implements DadosBasicosProducao {

    /*
     * NATUREZA CDATA #IMPLIED
     * TITULO CDATA #IMPLIED
     * ANO CDATA #IMPLIED
     * PAIS CDATA #IMPLIED
     * IDIOMA CDATA #IMPLIED
     * MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
     * ↪ MEIO_DIGITAL | FILME | HIPERTEXTO
     * | OUTRO | VÁRIOS | NÃO_INFORMADO) "NÃO_INFORMADO"
     * HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
     * FLAG-RELEVANCIA (SIM | NÃO) "NÃO"
     * DOI CDATA #IMPLIED
     * TITULO-INGLES CDATA #IMPLIED
     * NATUREZA-INGLES CDATA #IMPLIED
     * FLAG-DIVULGACAO-CIENTIFICA (SIM | NÃO) "NÃO"
     */

    @XmlAttribute(name = "NATUREZA")
    private String natureza;

    @XmlAttribute(name = "TITULO")
    private String titulo;

    @XmlAttribute(name = "ANO")
    private String ano;

    @XmlAttribute(name = "PAIS")
    private String pais;

    @XmlAttribute(name = "IDIOMA")
    private String idioma;

    @XmlAttribute(name = "MEIO-DE-DIVULGACAO")
    private MeioDeDivulgacao meioDivulgacao;

    @XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
    private String homePage;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-RELEVANCIA")
    private Boolean flagRelevancia;

    @XmlAttribute(name = "DOI")
    private String doi;

    @XmlAttribute(name = "TITULO-INGLES")
    private String tituloIngles;

    @XmlAttribute(name = "NATUREZA-INGLES")
    private String naturezaIngles;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-DIVULGACAO-CIENTIFICA")
    private Boolean flagDivulgacaoCientifica;
}
package ufsc.alpc.xml.dto.producao.tecnica.demais.producoes.desenvolvimento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoMaterialDidaticoInstrucionalXmlDto {

    /*
     * FINALIDADE CDATA #IMPLIED
     * FINALIDADE-INGLES CDATA #IMPLIED

```



```

        */
        @XmlAttribute(name = "FINALIDADE")
        private String finalidade;

        @XmlAttribute(name = "FINALIDADE-INGLES")
        private String finalidadeIngles;
    }
}
package ufsc.alpc.xml.dto.producao.technica.demais.producoes.relatorio.pesquisa;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoRelatorioPesquisaXmlDto {
    /*
     * NOME-DO-PROJETO CDATA #IMPLIED
     * NUMERO-DE-PAGINAS CDATA #IMPLIED
     * DISPONIBILIDADE CDATA #IMPLIED
     * INSTITUICAO-FINANCIADORA CDATA #IMPLIED
     */

    @XmlAttribute(name = "NOME-DO-PROJETO")
    private String nomeProjeto;

    @XmlAttribute(name = "NUMERO-DE-PAGINAS")
    private String numeroPaginas;

    @XmlAttribute(name = "DISPONIBILIDADE")
    private String disponibilidade;

    @XmlAttribute(name = "INSTITUICAO-FINANCIADORA")
    private String instituicaoFinanciadora;
}
package ufsc.alpc.xml.dto.producao.technica.demais.producoes.relatorio.pesquisa;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class RelatorioDePesquisaXmlDto extends
    ↳ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {
    // DADOS-BASICOS-DA-APRESENTACAO-DE-TRABALHO?,
    // DETALHAMENTO-DA-APRESENTACAO-DE-TRABALHO?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DO-RELATORIO-DE-PESQUISA")
    private DadosBasicosRelatorioPesquisaXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DO-RELATORIO-DE-PESQUISA")
    private DetalhamentoRelatorioPesquisaXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.technica.demais.producoes.relatorio.pesquisa;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

```

```

import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MeiDeDivulgacao;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosRelatorioPesquisaXmlDto implements DadosBasicosProducao {

    /*
     * TITULO CDATA #IMPLIED
     * ANO CDATA #IMPLIED
     * PAIS CDATA #IMPLIED
     * IDIOMA CDATA #IMPLIED
     * MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
     * ← MEIO_DIGITAL | FILME | HIPERTEXTO
     * | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
     * HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
     * FLAG-RELEVANCIA (SIM | NAO) "NAO"
     * DOI CDATA #IMPLIED
     * TITULO-INGLES CDATA #IMPLIED
     */

    @XmlAttribute(name = "TITULO")
    private String titulo;

    @XmlAttribute(name = "ANO")
    private String ano;

    @XmlAttribute(name = "PAIS")
    private String pais;

    @XmlAttribute(name = "IDIOMA")
    private String idioma;

    @XmlAttribute(name = "MEIO-DE-DIVULGACAO")
    private MeiDeDivulgacao meioDeDivulgacao;

    @XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
    private String homePage;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-RELEVANCIA")
    private Boolean flagRelevancia;

    @XmlAttribute(name = "DOI")
    private String doi;

    @XmlAttribute(name = "TITULO-INGLES")
    private String tituloIngles;
}

package ufsc.alpc.xml.dto.producao.tecnica.cultivar;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.producao.tecnica.common.DetalhamentoRegistroPatenteXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoDaCultivarXmlDto extends DetalhamentoRegistroPatenteXmlDto {

    // REGISTRO-OU-PATENTE*

    // FINALIDADE CDATA #IMPLIED
    // INSTITUICAO-FINANCIADORA CDATA #IMPLIED
    // FINALIDADE-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "FINALIDADE")

```

```

        private String finalidade;

        @XmlAttribute(name = "INSTITUICAO-FINANCIADORA")
        private String instituicaoFinanciadora;

        @XmlAttribute(name = "FINALIDADE-INGLES")
        private String finalidadeIngles;
    }
}
package ufsc.alpc.xml.dto.producao.tecnica.cultivar;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class CultivarXmlDto extends AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto
    ↪ implements Producao {

    // DADOS-BASICOS-DA-CULTIVAR?,
    // DETALHAMENTO-DA-CULTIVAR?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DA-CULTIVAR")
    private DadosBasicosDaCultivarXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DA-CULTIVAR")
    private DetalhamentoDaCultivarXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.producao.tecnica.cultivar;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDaCultivarXmlDto implements DadosBasicosProducao {

    // DENOMINACAO CDATA #IMPLIED
    // ANO-SOLICITACAO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DENOMINACAO-INGLES CDATA #IMPLIED
    // FLAG-POTENCIAL-INOVACAO (SIM | NAO) "NAO"

    @XmlAttribute(name = "DENOMINACAO")
    private String denominacao;

    @XmlAttribute(name = "ANO-SOLICITACAO")
    private String ano;

    @XmlAttribute(name = "PAIS")
    private String pais;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-RELEVANCIA")
    protected Boolean flagRelevancia;

    @XmlAttribute(name = "DENOMINACAO-INGLES")
    private String denominacaoIngles;
}

```

```

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-POTENCIAL-INOVACAO")
    protected Boolean flagPotencialInovacao;

    @Override
    public String getTitulo() {
        return denominacao;
    }
}

package ufsc.alpc.xml.dto.producao.technical.topografia;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;
import ufsc.alpc.xml.dto.producao.technical.common.DadosBasicosDesenhoMarcaTopografiaXmlDto;
import ufsc.alpc.xml.dto.producao.technical.common.DetalhamentoDesenhoTopografiaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class TopografiaDeCircuitoIntegradoXmlDto extends
    ↪ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto implements Producao {
    // DADOS-BASICOS-DA-TOPOGRAFIA-DE-CIRCUITO-INTEGRADO?,
    // DETALHAMENTO-DA-TOPOGRAFIA-DE-CIRCUITO-INTEGRADO?,
    // AUTORES*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↪ "DADOS-BASICOS-DA-TOPOGRAFIA-DE-CIRCUITO-INTEGRADO")
    private DadosBasicosDesenhoMarcaTopografiaXmlDto dadosBasicos;

    @XmlElement(name =
        ↪ "DETALHAMENTO-DA-TOPOGRAFIA-DE-CIRCUITO-INTEGRADO")
    private DetalhamentoDesenhoTopografiaXmlDto detalhamento;
}

package ufsc.alpc.xml.dto.dados.gerais.idiomas;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.domain.Proficiencia;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class IdiomaXmlDto {

    // IDIOMA CDATA #IMPLIED
    // DESCRICAO-DO-IDIOMA CDATA #IMPLIED
    // PROFICIENCIA-DE-LEITURA (POUCO | RAZOAVELMENTE | BEM |
    // ↪ NAO_INFORMADO)
    // #IMPLIED
    // PROFICIENCIA-DE-FALA (POUCO | RAZOAVELMENTE | BEM |
    // ↪ NAO_INFORMADO)
    // #IMPLIED
    // PROFICIENCIA-DE-ESCRITA (POUCO | RAZOAVELMENTE | BEM |
    // ↪ NAO_INFORMADO)
    // #IMPLIED
    // PROFICIENCIA-DE-COMPREENSAO (POUCO | RAZOAVELMENTE | BEM |
    // ↪ NAO_INFORMADO)
    // #IMPLIED

    @XmlAttribute(name = "IDIOMA")
    private String idioma;

    @XmlAttribute(name = "DESCRICAO-DO-IDIOMA")

```

```

private String descricaoIdioma;

@XmlAttribute(name = "PROFICIENCIA-DE-LEITURA")
private Proficiencia proficienciaLeitura;

@XmlAttribute(name = "PROFICIENCIA-DE-FALA")
private Proficiencia proficienciaFala;

@XmlAttribute(name = "PROFICIENCIA-DE-ESCRITA")
private Proficiencia proficienciaEscrita;

@XmlAttribute(name = "PROFICIENCIA-DE-COMPREENSAO")
private Proficiencia proficienciaComprensao;
}
package ufsc.alpc.xml.dto.dados.gerais.idiomas;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class IdiomasXmlDto {

    // IDIOMA*

    @XmlElement(name = "IDIOMA")
    private List<IdiomaXmlDto> idiomas;
}
package ufsc.alpc.xml.dto.dados.gerais.areasatuacao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.domain.GrandeAreaDoConhecimento;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AreaAtuacaoXmlDto {

    // SEQUENCIA-AREA-DE-ATUACAO CDATA #IMPLIED
    // NOME-GRANDE-AREA-DO-CONHECIMENTO (OUTROS |
    //     ↳ LINGUISTICA LETRAS E ARTES |
    //     CIENCIAS_HUMANAS | CIENCIAS_SOCIAIS_APLICADAS | CIENCIAS_AGRARIAS |
    //     CIENCIAS_DA_SAUDE |
    //     ENGENHARIAS | CIENCIAS_BIOLOGICAS | CIENCIAS_EXATAS_E_DA_TERRA)
    //     ↳ #IMPLIED
    // NOME-DA-AREA-DO-CONHECIMENTO CDATA #IMPLIED
    // NOME-DA-SUB-AREA-DO-CONHECIMENTO CDATA #IMPLIED
    // NOME-DA-ESPECIALIDADE CDATA #IMPLIED

    @XmlAttribute(name = "SEQUENCIA-AREA-DE-ATUACAO")
    private String sequenciaAreaAtuacao;

    @XmlAttribute(name = "NOME-GRANDE-AREA-DO-CONHECIMENTO")
    private GrandeAreaDoConhecimento nomeGrandeAreaConhecimento;

    @XmlAttribute(name = "NOME-DA-AREA-DO-CONHECIMENTO")
    private String nomeAreaConhecimento;

    @XmlAttribute(name = "NOME-DA-SUB-AREA-DO-CONHECIMENTO")
    private String nomeSubAreaConhecimento;

    @XmlAttribute(name = "NOME-DA-ESPECIALIDADE")
    private String nomeEspecialidade;
}
package ufsc.alpc.xml.dto.dados.gerais.areasatuacao;

```

```

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AreasDeAtuacaoXmlDto {

    // AREA-DE-ATUACAO*

    @XmlElement(name = "AREA-DE-ATUACAO")
    private List<AreaAtuacaoXmlDto> atuacoes;

}

package ufsc.alpc.xml.dto.dados.gerais.formacao;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.dados.gerais.formacao.aperfeicoamento.AperfeicoamentoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.doutorado.DoutoradoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.especializacao.EspecializacaoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.graduacao.GraduacaoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.livredocencia.LivreDocenciaXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.mestrado.MestradoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.posdoutorado.PosDoutoradoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.residencia.ResidenciaMedicaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class FormacaoAcademicaTitulacaoXmlDto {

    // GRADUACAO*
    // ESPECIALIZACAO*
    // MESTRADO*
    // DOUTORADO*
    // POS-DOUTORADO*
    // LIVRE-DOCENCIA*
    // CURSO-TECNICO-PROFISSIONALIZANTE*
    // MESTRADO-PROFISSIONALIZANTE*
    // ENSINO-FUNDAMENTAL-PRIMEIRO-GRAU*
    // ENSINO-MEDIO-SEGUNDO-GRAU*
    // RESIDENCIA-MEDICA*
    // APERFEICOAMENTO*

    @XmlElement(name = "GRADUACAO")
    private List<GraduacaoXmlDto> graduacao;

    @XmlElement(name = "ESPECIALIZACAO")
    private List<EspecializacaoXmlDto> especializacao;

    @XmlElement(name = "MESTRADO")
    private List<MestradoXmlDto> mestrado;

    @XmlElement(name = "DOUTORADO")
    private List<DoutoradoXmlDto> doutorado;

    @XmlElement(name = "POS-DOUTORADO")
    private List<PosDoutoradoXmlDto> posDoutorado;

    @XmlElement(name = "LIVRE-DOCENCIA")
    private List<LivreDocenciaXmlDto> livreDocencia;

    @XmlElement(name = "CURSO-TECNICO-PROFISSIONALIZANTE")
    private List<CursoXmlDto> cursoTecnicoProfissionalizante;

```

```

@XmlElement(name = "MESTRADO--PROFISSIONALIZANTE")
private List<PosXmlDto> mestradoProfissionalizante;

@XmlElement(name = "ENSINO--FUNDAMENTAL--PRIMEIRO--GRAU")
private List<FormacaoBaseXmlDto> ensinoFundamental;

@XmlElement(name = "ENSINO--MEDIO--SEGUNDO--GRAU")
private List<FormacaoBaseXmlDto> ensinoMedio;

@XmlElement(name = "RESIDENCIA--MEDICA")
private List<ResidenciaMedicaXmlDto> residenciaMedica;

@XmlElement(name = "APERFEICOAMENTO")
private List<AperfeicoamentoXmlDto> aperfeicoamento;

public boolean isSetGraduacao() {
    return this.graduacao != null;
}

public boolean isSetMestrado() {
    return this.mestrado != null;
}

public boolean isSetDoutorado() {
    return this.doutorado != null;
}

public boolean isSetPosDoutorado() {
    return this.posDoutorado != null;
}
}

package ufsc.alpc.xml.dto.dados.gerais.formacao.mestrado;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.dados.gerais.formacao.PosXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class MestradoXmlDto extends PosXmlDto {

    // SEQUENCIA--FORMACAO CDATA #IMPLIED
    // NIVEL CDATA #IMPLIED
    // CODIGO--INSTITUICAO CDATA #IMPLIED
    // NOME--INSTITUICAO CDATA #IMPLIED
    // CODIGO--ORGAO CDATA #IMPLIED
    // NOME--ORGAO CDATA #IMPLIED
    // CODIGO--CURSO CDATA #IMPLIED
    // NOME--CURSO CDATA #IMPLIED
    // STATUS--DO--CURSO (EM _ ANDAMENTO | CONCLUIDO | INCOMPLETO)
    // ↪ #IMPLIED
    // ANO--DE--INICIO CDATA #IMPLIED
    // ANO--DE--CONCLUSAO CDATA #IMPLIED
    // FLAG--BOLSA CDATA #IMPLIED
    // CODIGO--AGENCIA--FINANCIADORA CDATA #IMPLIED
    // NOME--AGENCIA CDATA #IMPLIED
    // NOME--CURSO--INGLES CDATA #IMPLIED

    // CODIGO--AREA--CURSO CDATA #IMPLIED
    // ANO--DE--OBTENCAO--DO--TITULO CDATA #IMPLIED
    // TITULO--DA--DISSERTACAO--TESE CDATA #IMPLIED
    // NOME--COMPLETO--DO--ORIENTADOR CDATA #IMPLIED
    // NUMERO--ID--ORIENTADOR CDATA #IMPLIED
    // CODIGO--CURSO--CAPES CDATA #IMPLIED
    // TITULO--DA--DISSERTACAO--TESE--INGLES CDATA #IMPLIED
    // NOME--DO--CO--ORIENTADOR CDATA #IMPLIED

    // TIPO--MESTRADO CDATA #IMPLIED
    // CODIGO--INSTITUICAO--DOUT CDATA #IMPLIED
    // NOME--INSTITUICAO--DOUT CDATA #IMPLIED
    // CODIGO--INSTITUICAO--OUTRA--DOUT CDATA #IMPLIED
    // NOME--INSTITUICAO--OUTRA--DOUT CDATA #IMPLIED
    // NOME--ORIENTADOR--DOUT CDATA #IMPLIED

```

```

@XmlAttribute(name = "TIPO-MESTRADO")
protected String tipoMestrado;

@XmlAttribute(name = "CODIGO-INSTITUICAO-DOUT")
protected String codigoInstituicaoDout;

@XmlAttribute(name = "NOME-INSTITUICAO-DOUT")
protected String nomeInstituicaoDout;

@XmlAttribute(name = "CODIGO-INSTITUICAO-OUTRA-DOUT")
protected String codigoInstituicaoOutraDout;

@XmlAttribute(name = "NOME-INSTITUICAO-OUTRA-DOUT")
protected String nomeInstituicaoOutraDot;

@XmlAttribute(name = "NOME-ORIENTADOR-DOUT")
protected String nomeOrientadorDout;
}

package ufsc.alpc.xml.dto.dados.gerais.formacao.graduacao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.dados.gerais.formacao.CursoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class GraduacaoXmlDto extends CursoXmlDto {

    // SEQUENCIA-FORMACAO CDATA #IMPLIED
    // NIVEL CDATA #IMPLIED
    // TITULO-DO-TRABALHO-DE-CONCLUSAO-DE-CURSO CDATA #IMPLIED
    // NOME-DO-ORIENTADOR CDATA #IMPLIED
    // CODIGO-INSTITUICAO CDATA #IMPLIED
    // NOME-INSTITUICAO CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-CURSO CDATA #IMPLIED
    // NOME-CURSO CDATA #IMPLIED
    // CODIGO-AREA-CURSO CDATA #IMPLIED
    // STATUS-DO-CURSO (EM_ANDAMENTO | CONCLUIDO | INCOMPLETO)
    // ↪ #IMPLIED
    // ANO-DE-INICIO CDATA #IMPLIED
    // ANO-DE-CONCLUSAO CDATA #IMPLIED
    // FLAG-BOLSA CDATA #IMPLIED
    // CODIGO-AGENCIA-FINANCIADORA CDATA #IMPLIED
    // NOME-AGENCIA CDATA #IMPLIED
    // NUMERO-ID-ORIENTADOR CDATA #IMPLIED
    // CODIGO-CURSO-CAPIES CDATA #IMPLIED
    // TITULO-DO-TRABALHO-DE-CONCLUSAO-DE-CURSO-INGLES CDATA
    // ↪ #IMPLIED
    // NOME-CURSO-INGLES CDATA #IMPLIED
    // FORMACAO-ACADEMICA-TITULACAO CDATA #IMPLIED
    // TIPO-GRADUACAO CDATA #IMPLIED
    // CODIGO-INSTITUICAO-GRAD CDATA #IMPLIED
    // NOME-INSTITUICAO-GRAD CDATA #IMPLIED
    // CODIGO-INSTITUICAO-OUTRA-GRAD CDATA #IMPLIED
    // NOME-INSTITUICAO-OUTRA-GRAD CDATA #IMPLIED
    // NOME-ORIENTADOR-GRAD CDATA #IMPLIED

    @XmlAttribute(name = "TITULO-DO-TRABALHO-DE-CONCLUSAO-DE-CURSO")
    protected String tituloTcc;

    @XmlAttribute(name = "NOME-DO-ORIENTADOR")
    protected String nomeOrientador;

    @XmlAttribute(name = "CODIGO-AREA-CURSO")
    protected String codigoAreaCurso;

    @XmlAttribute(name = "NUMERO-ID-ORIENTADOR")
    protected String numeroIdOrientador;

    @XmlAttribute(name = "CODIGO-CURSO-CAPIES")
    protected String codigoCursoCapes;
}

```



```

@XmlAttribute(name =
    ↪ "TITULO-DO-TRABALHO-DE-CONCLUSAO-DE-CURSO-INGLES")
protected String tituloTecIngles;

@XmlAttribute(name = "FORMACAO-ACADEMICA-TITULACAO")
protected String formacaoAcademicaTitulacao;

@XmlAttribute(name = "TIPO-GRADUACAO")
protected String tipoGraduacao;

@XmlAttribute(name = "CODIGO-INSTITUICAO-GRAD")
protected String codigoInstituicaoGrad;

@XmlAttribute(name = "NOME-INSTITUICAO-GRAD")
protected String nomeInstituicaoGrad;

@XmlAttribute(name = "CODIGO-INSTITUICAO-OUTRA-GRAD")
protected String codigoInstituicaoOutraGrad;

@XmlAttribute(name = "NOME-INSTITUICAO-OUTRA-GRAD")
protected String nomeInstituicaoOutraGrad;

@XmlAttribute(name = "NOME-ORIENTADOR-GRAD")
protected String nomeOrientadorGrad;
}
package ufsc.alpc.xml.dto.dados.gerais.formacao.livredocencia;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.PalavrasChaveXmlDto;
import ufsc.alpc.xml.dto.common.SetoresDeAtividadeXmlDto;
import ufsc.alpc.xml.dto.common.area.conhecimento.AreasDoConhecimentoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class LivreDocenciaXmlDto {

    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // SEQUENCIA-FORMACAO CDATA #IMPLIED
    // NIVEL CDATA #IMPLIED
    // CODIGO-INSTITUICAO CDATA #IMPLIED
    // NOME-INSTITUICAO CDATA #IMPLIED
    // ANO-DE-OBTENCAO-DO-TITULO CDATA #IMPLIED
    // TITULO-DO-TRABALHO CDATA #IMPLIED
    // TITULO-DO-TRABALHO-INGLES CDATA #IMPLIED

    @XmlElement(name = "PALAVRAS-CHAVE")
    protected PalavrasChaveXmlDto palavraChave;

    @XmlElement(name = "AREAS-DO-CONHECIMENTO")
    protected AreasDoConhecimentoXmlDto areasDoConhecimento;

    @XmlElement(name = "SETORES-DE-ATIVIDADE")
    protected SetoresDeAtividadeXmlDto setoresDeAtividade;

    @XmlAttribute(name = "SEQUENCIA-FORMACAO")
    protected String sequenciaFormacao;

    @XmlAttribute(name = "NIVEL")
    protected String nivel;

    @XmlAttribute(name = "CODIGO-INSTITUICAO")
    protected String codigoInstituicao;

    @XmlAttribute(name = "NOME-INSTITUICAO")
    protected String nomeInstituicao;

    @XmlAttribute(name = "ANO-DE-OBTENCAO-DO-TITULO")
    protected String anoObtencao;
}

```

```

    @XmlAttribute(name = "TITULO-DO-TRABALHO")
    protected String tituloTrabalho;

    @XmlAttribute(name = "TITULO-DO-TRABALHO-INGLES")
    protected String tituloTrabalhoIngles;
}
package ufsc.alpc.xml.dto.dados.gerais.formacao.doutorado;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.dados.gerais.formacao.PosXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DoutoradoXmlDto extends PosXmlDto {

    // SEQUENCIA-FORMACAO CDATA #IMPLIED
    // NIVEL CDATA #IMPLIED
    // CODIGO-INSTITUICAO CDATA #IMPLIED
    // NOME-INSTITUICAO CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-CURSO CDATA #IMPLIED
    // NOME-CURSO CDATA #IMPLIED
    // STATUS-DO-CURSO (EM_ANDAMENTO | CONCLUIDO | INCOMPLETO)
    // ← #IMPLIED
    // ANO-DE-INICIO CDATA #IMPLIED
    // ANO-DE-CONCLUSAO CDATA #IMPLIED
    // FLAG-BOLSA CDATA #IMPLIED
    // CODIGO-AGENCIA-FINANCIADORA CDATA #IMPLIED
    // NOME-AGENCIA CDATA #IMPLIED
    // NOME-CURSO-INGLES CDATA #IMPLIED

    // ANO-DE-OBTENCAO-DO-TITULO CDATA #IMPLIED
    // CODIGO-AREA-CURSO CDATA #IMPLIED
    // TITULO-DA-DISSERTACAO-TESE CDATA #IMPLIED
    // NOME-COMPLETO-DO-ORIENTADOR CDATA #IMPLIED
    // NUMERO-ID-ORIENTADOR CDATA #IMPLIED
    // CODIGO-CURSO-CAPES CDATA #IMPLIED
    // TITULO-DA-DISSERTACAO-TESE-INGLES CDATA #IMPLIED
    // NOME-DO-CO-ORIENTADOR CDATA #IMPLIED

    // TIPO-DOUTORADO CDATA #IMPLIED
    // CODIGO-INSTITUICAO-DOUT CDATA #IMPLIED
    // NOME-INSTITUICAO-DOUT CDATA #IMPLIED
    // CODIGO-INSTITUICAO-OUTRA-DOUT CDATA #IMPLIED
    // NOME-INSTITUICAO-OUTRA-DOUT CDATA #IMPLIED
    // NOME-ORIENTADOR-DOUT CDATA #IMPLIED
    // NOME-DO-ORIENTADOR-CO-TUTELA CDATA #IMPLIED
    // CODIGO-INSTITUICAO-OUTRA-CO-TUTELA CDATA #IMPLIED
    // CODIGO-INSTITUICAO-CO-TUTELA CDATA #IMPLIED
    // NOME-DO-ORIENTADOR-SANDUICHE CDATA #IMPLIED
    // CODIGO-INSTITUICAO-OUTRA-SANDUICHE CDATA #IMPLIED
    // CODIGO-INSTITUICAO-SANDUICHE CDATA #IMPLIED

    @XmlAttribute(name = "TIPO-DOUTORADO")
    protected String tipoDoutorado;

    @XmlAttribute(name = "CODIGO-INSTITUICAO-DOUT")
    protected String codigoInstituicaoDout;

    @XmlAttribute(name = "NOME-INSTITUICAO-DOUT")
    protected String nomeInstituicaoDout;

    @XmlAttribute(name = "CODIGO-INSTITUICAO-OUTRA-DOUT")
    protected String codigoInstituicaoOutraDout;

    @XmlAttribute(name = "NOME-INSTITUICAO-OUTRA-DOUT")
    protected String nomeInstituicaoOutraDot;

    @XmlAttribute(name = "NOME-ORIENTADOR-DOUT")
    protected String nomeOrientadorDout;

    @XmlAttribute(name = "NOME-DO-ORIENTADOR-CO-TUTELA")
    protected String nomeOrientadorCoTutela;
}

```

```

@XmlAttribute(name = "CODIGO-INSTITUICAO-OUTRA-CO-TUTELA")
protected String codigoInstituicaoOutraCoTutela;

@XmlAttribute(name = "CODIGO-INSTITUICAO-CO-TUTELA")
protected String codigoInstituicaoCoTutela;

@XmlAttribute(name = "NOME-DO-ORIENTADOR-SANDUICHE")
protected String nomeOrientadorSanduiche;

@XmlAttribute(name = "CODIGO-INSTITUICAO-OUTRA-SANDUICHE")
protected String codigoInstituicaoOutraSanduiche;

@XmlAttribute(name = "CODIGO-INSTITUICAO-SANDUICHE")
protected String codigoInstituicaoSanduiche;
}
package ufsc.alpc.xml.dto.dados.gerais.formacao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class CursoXmlDto extends FormacaoBaseXmlDto {
    // SEQUENCIA-FORMACAO CDATA #IMPLIED
    // NIVEL CDATA #IMPLIED
    // CODIGO-INSTITUICAO CDATA #IMPLIED
    // NOME-INSTITUICAO CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-CURSO CDATA #IMPLIED
    // NOME-CURSO CDATA #IMPLIED
    // STATUS-DO-CURSO (EM_ANDAMENTO | CONCLUIDO | INCOMPLETO)
    // → #IMPLIED
    // ANO-DE-INICIO CDATA #IMPLIED
    // ANO-DE-CONCLUSAO CDATA #IMPLIED
    // FLAG-BOLSA CDATA #IMPLIED
    // CODIGO-AGENCIA-FINANCIADORA CDATA #IMPLIED
    // NOME-AGENCIA CDATA #IMPLIED
    // NOME-CURSO-INGLES CDATA #IMPLIED - 15

    @XmlAttribute(name = "CODIGO-ORGAO")
    protected String codigoOrgao;

    @XmlAttribute(name = "NOME-ORGAO")
    protected String nomeOrgao;

    @XmlAttribute(name = "CODIGO-CURSO")
    protected String codigoCurso;

    @XmlAttribute(name = "NOME-CURSO")
    protected String nomeCurso;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-BOLSA")
    protected Boolean flagBolsa;

    @XmlAttribute(name = "CODIGO-AGENCIA-FINANCIADORA")
    protected String codigoAgenciaFinanciadora;

    @XmlAttribute(name = "NOME-AGENCIA")
    protected String nomeAgencia;

    @XmlAttribute(name = "NOME-CURSO-INGLES")
    protected String nomeCursoIngles;
}
package ufsc.alpc.xml.dto.dados.gerais.formacao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;

```

```

import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.PalavraAreasSetoresXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class PosXmlDto extends PalavraAreasSetoresXmlDto {
    // SEQUENCIA-FORMACAO CDATA #IMPLIED
    // NIVEL CDATA #IMPLIED
    // CODIGO-INSTITUICAO CDATA #IMPLIED
    // NOME-INSTITUICAO CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-CURSO CDATA #IMPLIED
    // NOME-CURSO CDATA #IMPLIED
    // STATUS-DO-CURSO (EM_ANDAMENTO | CONCLUIDO | INCOMPLETO)
    // ← #IMPLIED
    // ANO-DE-INICIO CDATA #IMPLIED
    // ANO-DE-CONCLUSAO CDATA #IMPLIED
    // FLAG-BOLSA CDATA #IMPLIED
    // CODIGO-AGENCIA-FINANCIADORA CDATA #IMPLIED
    // NOME-AGENCIA CDATA #IMPLIED
    // NOME-CURSO-INGLES CDATA #IMPLIED

    // CODIGO-AREA-CURSO CDATA #IMPLIED
    // ANO-DE-OBTENCAO-DO-TITULO CDATA #IMPLIED
    // TITULO-DA-DISSERTACAO-TESE CDATA #IMPLIED
    // NOME-COMPLETO-DO-ORIENTADOR CDATA #IMPLIED
    // NUMERO-ID-ORIENTADOR CDATA #IMPLIED
    // CODIGO-CURSO-CAPES CDATA #IMPLIED
    // TITULO-DA-DISSERTACAO-TESE-INGLES CDATA #IMPLIED
    // NOME-DO-CO-ORIENTADOR CDATA #IMPLIED

    @XmlAttribute(name = "CODIGO-AREA-CURSO")
    protected String codigoAreaCurso;

    @XmlAttribute(name = "ANO-DE-OBTENCAO-DO-TITULO")
    protected String anoObtencaoTitulo;

    @XmlAttribute(name = "TITULO-DA-DISSERTACAO-TESE")
    protected String tituloDissertacaoTese;

    @XmlAttribute(name = "NOME-COMPLETO-DO-ORIENTADOR")
    protected String nomeCompletoOrientador;

    @XmlAttribute(name = "NUMERO-ID-ORIENTADOR")
    protected String numeroIdOrientador;

    @XmlAttribute(name = "CODIGO-CURSO-CAPES")
    protected String codigoCursoCapes;

    @XmlAttribute(name = "TITULO-DA-DISSERTACAO-TESE-INGLES")
    protected String tituloDissertacaoTeseIngles;

    @XmlAttribute(name = "NOME-DO-CO-ORIENTADOR")
    protected String nomeCoOrientador;
}
package ufsc.alpc.xml.dto.dados.gerais.formacao.posdoutorado;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.PalavraAreasSetoresXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class PosDoutoradoXmlDto extends PalavraAreasSetoresXmlDto {
    // PALAVRAS-CHAVE?

```

```

// AREAS-DO-CONHECIMENTO?
// SETORES-DE-ATIVIDADE?

// SEQUENCIA-FORMACAO CDATA #IMPLIED
// NIVEL CDATA #IMPLIED
// CODIGO-INSTITUICAO CDATA #IMPLIED
// NOME-INSTITUICAO CDATA #IMPLIED
// ANO-DE-INICIO CDATA #IMPLIED
// ANO-DE-CONCLUSAO CDATA #IMPLIED
// FLAG-BOLSA CDATA #IMPLIED
// CODIGO-AGENCIA-FINANCIADORA CDATA #IMPLIED
// NOME-AGENCIA CDATA #IMPLIED
// STATUS-DO-CURSO CDATA #IMPLIED
// NOME-CURSO-INGLES CDATA #IMPLIED

// ANO-DE-OBTENCAO-DO-TITULO CDATA #IMPLIED
// STATUS-DO-ESTAGIO CDATA #IMPLIED
// NUMERO-ID-ORIENTADOR CDATA #IMPLIED
// CODIGO-CURSO-CAPIES CDATA #IMPLIED
// TITULO-DO-TRABALHO CDATA #IMPLIED
// TITULO-DO-TRABALHO-INGLES CDATA #IMPLIED

@XmlAttribute(name = "ANO-DE-OBTENCAO-DO-TITULO")
protected String anoObtencao;

@XmlAttribute(name = "STATUS-DO-ESTAGIO")
protected String statusEstagio;

@XmlAttribute(name = "NUMERO-ID-ORIENTADOR")
protected String numeroIdOrientador;

@XmlAttribute(name = "CODIGO-CURSO-CAPIES")
protected String codigoCursoCapes;

@XmlAttribute(name = "TITULO-DO-TRABALHO")
protected String tituloTrabalho;

@XmlAttribute(name = "TITULO-DO-TRABALHO-INGLES")
protected String tituloTrabalhoIngles;
}
package ufsc.alpc.xml.dto.dados.gerais.formacao.especializacao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.dados.gerais.formacao.CursoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class EspecializacaoXmlDto extends CursoXmlDto {

// SEQUENCIA-FORMACAO CDATA #IMPLIED
// NIVEL CDATA #IMPLIED
// TITULO-DA-MONOGRRAFIA CDATA #IMPLIED
// NOME-DO-ORIENTADOR CDATA #IMPLIED
// CODIGO-INSTITUICAO CDATA #IMPLIED
// NOME-INSTITUICAO CDATA #IMPLIED
// CODIGO-ORGAO CDATA #IMPLIED
// NOME-ORGAO CDATA #IMPLIED
// CODIGO-CURSO CDATA #IMPLIED
// NOME-CURSO CDATA #IMPLIED
// STATUS-DO-CURSO (EM_ANDAMENTO | CONCLUIDO | INCOMPLETO)
// ANO-DE-INICIO CDATA #IMPLIED
// ANO-DE-CONCLUSAO CDATA #IMPLIED
// FLAG-BOLSA CDATA #IMPLIED
// CODIGO-AGENCIA-FINANCIADORA CDATA #IMPLIED
// NOME-AGENCIA CDATA #IMPLIED
// CARGA-HORARIA CDATA #IMPLIED
// TITULO-DA-MONOGRRAFIA-INGLES CDATA #IMPLIED
// NOME-CURSO-INGLES CDATA #IMPLIED

@XmlAttribute(name = "TITULO-DA-MONOGRRAFIA")
protected String tituloMonografia;

@XmlAttribute(name = "NOME-DO-ORIENTADOR")
protected String nomeOrientador;

```

```

        @XmlAttribute(name = "CARGA-HORARIA")
        protected String cargaHoraria;

        @XmlAttribute(name = "TITULO-DA-MONOGRAFIA-INGLES")
        protected String tituloMonografiaIngles;
    }
}
package ufsc.alpc.xml.dto.dados.gerais.formacao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.domain.StatusCurso;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class FormacaoBaseXmlDto {

    // SEQUENCIA-FORMACAO CDATA #IMPLIED
    // NIVEL CDATA #IMPLIED
    // CODIGO-INSTITUICAO CDATA #IMPLIED
    // NOME-INSTITUICAO CDATA #IMPLIED
    // STATUS-DO-CURSO (EM_ANDAMENTO | CONCLUIDO | INCOMPLETO)
    // ↔ #IMPLIED
    // ANO-DE-INICIO CDATA #IMPLIED
    // ANO-DE-CONCLUSAO CDATA #IMPLIED - 7

    @XmlAttribute(name = "SEQUENCIA-FORMACAO")
    protected String sequenciaFormacao;

    @XmlAttribute(name = "NIVEL")
    protected String nivel;

    @XmlAttribute(name = "CODIGO-INSTITUICAO")
    protected String codigoInstituicao;

    @XmlAttribute(name = "NOME-INSTITUICAO")
    protected String nomeInstituicao;

    @XmlAttribute(name = "STATUS-DO-CURSO")
    protected StatusCurso statusCurso;

    @XmlAttribute(name = "ANO-DE-INICIO")
    protected Integer anoInicio;

    @XmlAttribute(name = "ANO-DE-CONCLUSAO")
    protected Integer anoConclusao;
}
package ufsc.alpc.xml.dto.dados.gerais.formacao.residencia;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.common.PalavrasChaveXmlDto;
import ufsc.alpc.xml.dto.common.SetoresDeAtividadeXmlDto;
import ufsc.alpc.xml.dto.common.area.conhecimento.AreasDoConhecimentoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.FormacaoBaseXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ResidenciaMedicaXmlDto extends FormacaoBaseXmlDto {

    // SEQUENCIA-FORMACAO CDATA #IMPLIED
    // NIVEL CDATA #IMPLIED
    // CODIGO-INSTITUICAO CDATA #IMPLIED

```

```

// NOME-INSTITUICAO CDATA #IMPLIED
// STATUS-DO-CURSO (EM_ANDAMENTO | CONCLUIDO | INCOMPLETO)
// ↪ #IMPLIED
// ANO-DE-INICIO CDATA #IMPLIED
// ANO-DE-CONCLUSAO CDATA #IMPLIED

// FLAG-BOLSA CDATA #IMPLIED
// CODIGO-AGENCIA-FINANCIADORA CDATA #IMPLIED
// NOME-AGENCIA CDATA #IMPLIED
// TITULO-DA-RESIDENCIA-MEDICA CDATA #IMPLIED
// NUMERO-DO-REGISTRO CDATA #IMPLIED
// TITULO-DA-RESIDENCIA-MEDICA-INGLES CDATA #IMPLIED

@XmlElement(name = "PALAVRAS-CHAVE")
protected PalavrasChaveXmlDto palavraChave;

@XmlElement(name = "AREAS-DO-CONHECIMENTO")
protected AreasDoConhecimentoXmlDto areasDoConhecimento;

@XmlElement(name = "SETORES-DE-ATIVIDADE")
protected SetoresDeAtividadeXmlDto setoresDeAtividade;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-BOLSA")
protected Boolean flagBolsa;

@XmlAttribute(name = "CODIGO-AGENCIA-FINANCIADORA")
protected String codigoAgenciaFinanciadora;

@XmlAttribute(name = "NOME-AGENCIA")
protected String nomeAgencia;

@XmlAttribute(name = "TITULO-DA-RESIDENCIA-MEDICA")
protected String tituloResidenciaMedica;

@XmlAttribute(name = "NUMERO-DO-REGISTRO")
protected String numeroRegistro;

@XmlAttribute(name = "TITULO-DA-RESIDENCIA-MEDICA-INGLES")
protected String tituloResidenciaMedicaIngles;
}
package ufsc.alpc.xml.dto.dados.gerais.formacao.aperfeicoamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.dados.gerais.formacao.especializacao.EspecializacaoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AperfeicoamentoXmlDto extends EspecializacaoXmlDto {

// SEQUENCIA-FORMACAO CDATA #IMPLIED
// NIVEL CDATA #IMPLIED
// TITULO-DA-MONOGRAFIA CDATA #IMPLIED
// NOME-DO-ORIENTADOR CDATA #IMPLIED
// CODIGO-INSTITUICAO CDATA #IMPLIED
// NOME-INSTITUICAO CDATA #IMPLIED
// CODIGO-ORGAO CDATA #IMPLIED
// NOME-ORGAO CDATA #IMPLIED
// CODIGO-CURSO CDATA #IMPLIED
// NOME-CURSO CDATA #IMPLIED
// CODIGO-AREA-CURSO CDATA #IMPLIED
// STATUS-DO-CURSO (EM_ANDAMENTO | CONCLUIDO | INCOMPLETO)
// ↪ #IMPLIED
// ANO-DE-INICIO CDATA #IMPLIED
// ANO-DE-CONCLUSAO CDATA #IMPLIED
// FLAG-BOLSA CDATA #IMPLIED
// CODIGO-AGENCIA-FINANCIADORA CDATA #IMPLIED
// NOME-AGENCIA CDATA #IMPLIED
// CARGA-HORARIA CDATA #IMPLIED
// TITULO-DA-MONOGRAFIA-INGLES CDATA #IMPLIED
// NOME-CURSO-INGLES CDATA #IMPLIED

@XmlAttribute(name = "CODIGO-AREA-CURSO")

```

```

        protected String codigoAreaCurso;
    }
}
package ufsc.alpc.xml.dto.dados.gerais.outrasinformacoes;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OutrasInformacoesRelevantesXmlDto {

    // OUTRAS-INFORMACOES-RELEVANTES CDATA #IMPLIED

    @XmlAttribute(name = "OUTRAS-INFORMACOES-RELEVANTES")
    private String outrasInformacoes;

}
package ufsc.alpc.xml.dto.dados.gerais.resumocv;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ResumoCvXmlDto {

    // TEXTO-RESUMO-CV-RH CDATA #IMPLIED
    // TEXTO-RESUMO-CV-RH-EN CDATA #IMPLIED

    @XmlAttribute(name = "TEXTO-RESUMO-CV-RH")
    private String textoResumoCvRh;

    @XmlAttribute(name = "TEXTO-RESUMO-CV-RH-EN")
    private String textoResumoCvRhEn;

}
package ufsc.alpc.xml.dto.dados.gerais;

import java.util.Date;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.DateXmlAdapter;
import ufsc.alpc.xml.dto.dados.gerais.areasatuacao.AreasDeAtuacaoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.AtuacoesProfissionaisXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.endereco.EnderecoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.FormacaoAcademicaTitulacaoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.idiomas.IdiomasXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.outrasinformacoes.OutrasInformacoesRelevantesXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.premiostitulos.PremiosTitulosXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.resumocv.ResumoCvXmlDto;
import ufsc.alpc.xml.dto.domain.Sexo;
import ufsc.alpc.xml.dto.domain.SimNao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosGeraisXmlDto {

```



```

// RESUMO-CV?
// OUTRAS-INFORMACOES-RELEVANTES?
// ENDERECO?
// FORMACAO-ACADEMICA-TITULACAO?
// ATUACOES-PROFISSIONAIS?
// AREAS-DE-ATUACAO?
// IDIOMAS?
// PREMIOS-TITULOS?

@XmlElement(name = "RESUMO-CV")
private ResumoCvXmlDto resumo;

@XmlElement(name = "OUTRAS-INFORMACOES-RELEVANTES")
private OutrasInformacoesRelevantesXmlDto outrasInformacoes;

@XmlElement(name = "ENDERECO")
private EnderecoXmlDto endereco;

@XmlElement(name = "FORMACAO-ACADEMICA-TITULACAO")
private FormacaoAcademicaTitulacaoXmlDto formacaoAcademica;

@XmlElement(name = "ATUACOES-PROFISSIONAIS")
private AtuacoesProfissionaisXmlDto atuacoesProfissionais;

@XmlElement(name = "AREAS-DE-ATUACAO")
private AreasDeAtuacaoXmlDto areasAtuacao;

@XmlElement(name = "IDIOMAS")
private IdiomasXmlDto dadosGerais;

@XmlElement(name = "PREMIOS-TITULOS")
private PremiosTitulosXmlDto premiosTitulos;

public boolean isSetFormacaoAcademica() {
    return this.formacaoAcademica != null;
}

// NOME-COMPLETO CDATA #REQUIRED
// NOME-EM-CITACOES-BIBLIOGRAFICAS CDATA #REQUIRED
// NACIONALIDADE CDATA #REQUIRED
// CPF CDATA #IMPLIED
// NUMERO-DO-PASSAPORTE CDATA #IMPLIED
// PAIS-DE-NASCIMENTO CDATA #IMPLIED
// UF-NASCIMENTO CDATA #IMPLIED
// CIDADE-NASCIMENTO CDATA #IMPLIED
// FORMATO-DATA-DE-NASCIMENTO NMTOKEN #FIXED "DDMMAAAA"
// DATA-NASCIMENTO CDATA #IMPLIED
// SEXO (MASCULINO | FEMININO) #REQUIRED
// NUMERO-IDENTIDADE CDATA #IMPLIED
// ORGAO-EMISSOR CDATA #IMPLIED
// UF-ORGAO-EMISSOR CDATA #IMPLIED
// FORMATO-DATA-DE-EMISSAO NMTOKEN #FIXED "DDMMAAAA"
// DATA-DE-EMISSAO CDATA #IMPLIED
// NOME-DO-PAI CDATA #IMPLIED
// NOME-DA-MAE CDATA #IMPLIED
// PERMISSAO-DE-DIVULGACAO (SIM | NAO) #REQUIRED
// NOME-DO-ARQUIVO-DE-FOTO CDATA #IMPLIED
// TEXTO-RESUMO-CV-RH CDATA #IMPLIED
// OUTRAS-INFORMACOES-RELEVANTES CDATA #IMPLIED
// DATA-FALECIMENTO CDATA #IMPLIED
// SIGLA-PAIS-NACIONALIDADE CDATA #IMPLIED
// PAIS-DE-NACIONALIDADE CDATA #IMPLIED
// RACA-OU-COR CDATA #IMPLIED

@XmlAttribute(name = "NOME-COMPLETO", required = true)
private String nomeCompleto;

@XmlAttribute(name = "NOME-EM-CITACOES-BIBLIOGRAFICAS", required = true)
private String nomeCitacoesBibliograficas;

@XmlAttribute(name = "NACIONALIDADE", required = true)
private String nacionalidade;

@XmlAttribute(name = "CPF")
private String cpf;

@XmlAttribute(name = "NUMERO-DO-PASSAPORTE")
private String numeroPassaporte;

@XmlAttribute(name = "PAIS-DE-NASCIMENTO")
private String paisNascimento;

```

```

@XmlAttribute(name = "UF-NASCIMENTO")
private String ufNascimento;

@XmlAttribute(name = "CIDADE-NASCIMENTO")
private String cidadeNascimento;

@XmlJavaTypeAdapter(DateXmlAdapter.class)
@XmlAttribute(name = "DATA-NASCIMENTO")
private Date dataNascimento;

@XmlAttribute(name = "SEXO", required = true)
private Sexo sexo;

@XmlAttribute(name = "NUMERO-IDENTIDADE")
private String numeroIdentidade;

@XmlAttribute(name = "ORGAO-EMISSOR")
private String orgaoEmissor;

@XmlAttribute(name = "UF-ORGAO-EMISSOR")
private String ufOrgaoEmissor;

@XmlJavaTypeAdapter(DateXmlAdapter.class)
@XmlAttribute(name = "DATA-DE-EMISSAO")
private Date dataEmissao;

@XmlAttribute(name = "NOME-DO-PAI")
private String nomePai;

@XmlAttribute(name = "NOME-DA-MAE")
private String nomeMae;

@XmlAttribute(name = "PERMISSAO-DE-DIVULGACAO")
private SimNao permissaoDivulgacao;

@XmlAttribute(name = "NOME-DO-ARQUIVO-DE-FOTO")
private String nomeArquivoFoto;

@XmlAttribute(name = "TEXTO-RESUMO-CV-RH")
private String textoResumoCvRh;

@XmlAttribute(name = "OUTRAS-INFORMACOES-RELEVANTES")
private String outrasInformacoesRelevantes;

@XmlJavaTypeAdapter(DateXmlAdapter.class)
@XmlAttribute(name = "DATA-FALECIMENTO")
private Date dataFalecimento;

@XmlAttribute(name = "SIGLA-PAIS-NACIONALIDADE")
private String siglaPaisNacionalidade;

@XmlAttribute(name = "PAIS-DE-NACIONALIDADE")
private String paisNacionalidade;

@XmlAttribute(name = "RACA-OU-COR")
private String racaCor;
}
package ufsc.alpc.xml.dto.dados.gerais.atuacoes;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.conselhocomissao.AtividadesConselhoComissaoConsultoriaXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.direcaoadministracao.AtividadesDirecaoAdministracaoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.ensino.AtividadesEnsinoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.estagio.AtividadesEstagioXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.extensaouniversitaria.AtividadesExtensaoUniversitariaXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.AtividadesParticipacaoProjetoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.pesquisadesenvolvimento.AtividadesPesquisaDesenvolvimentoXmlDto;

```

```

import
↳ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.servicotecnico.AtividadesServicoTecnicoEspecializadoXm
import
↳ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.treinamentoministrado.AtividadesTreinamentoMinistrado
import
↳ ufsc.alpc.xml.dto.dados.gerais.atuacoes.outras.atividadestecnicocientifica.OutrasAtividadesTecnicoCientifica
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.vinculos.VinculosXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AtuacaoProfissionalXmlDto {

    // VINCULOS*
    // ATIVIDADES-DE-DIRECAO-E-ADMINISTRACAO?
    // ATIVIDADES-DE-PESQUISA-E-DESENVOLVIMENTO?
    // ATIVIDADES-DE-ENSINO?
    // ATIVIDADES-DE-ESTAGIO?
    // ATIVIDADES-DE-SERVICO-TECNICO-ESPECIALIZADO?
    // ATIVIDADES-DE-EXTENSAO-UNIVERSITARIA?
    // ATIVIDADES-DE-TREINAMENTO-MINISTRADO?
    // OUTRAS-ATIVIDADES-TECNICO-CIENTIFICA?
    // ATIVIDADES-DE-CONSELHO-COMISSAO-E-CONSULTORIA?
    // ATIVIDADES-DE-PARTICIPACAO-EM-PROJETO?

    @XmlElement(name = "VINCULOS")
    private List<VinculosXmlDto> vinculos;

    @XmlElement(name = "ATIVIDADES-DE-DIRECAO-E-ADMINISTRACAO")
    private AtividadesDirecaoAdministracaoXmlDto atividadesDirecaoAdministracao;

    @XmlElement(name = "ATIVIDADES-DE-PESQUISA-E-DESENVOLVIMENTO")
    private AtividadesPesquisaDesenvolvimentoXmlDto atividadesPesquisaDesenvolvimento;

    @XmlElement(name = "ATIVIDADES-DE-ENSINO")
    private AtividadesEnsinoXmlDto atividadesEnsino;

    @XmlElement(name = "ATIVIDADES-DE-ESTAGIO")
    private AtividadesEstagioXmlDto atividadesEstagio;

    @XmlElement(name = "ATIVIDADES-DE-SERVICO-TECNICO-ESPECIALIZADO")
    private AtividadesServicoTecnicoEspecializadoXmlDto atividadesServicoTecnicoEspecializado;

    @XmlElement(name = "ATIVIDADES-DE-EXTENSAO-UNIVERSITARIA")
    private AtividadesExtensaoUniversitariaXmlDto atividadesExtensaoUniversitaria;

    @XmlElement(name = "ATIVIDADES-DE-TREINAMENTO-MINISTRADO")
    private AtividadesTreinamentoMinistradoXmlDto atividadesTreinamentoMinistrado;

    @XmlElement(name = "OUTRAS-ATIVIDADES-TECNICO-CIENTIFICA")
    private OutrasAtividadesTecnicoCientificaXmlDto outrasAtividadesTecnicoCientifica;

    @XmlElement(name =
        ↳ "ATIVIDADES-DE-CONSELHO-COMISSAO-E-CONSULTORIA")
    private AtividadesConselhoComissaoConsultoriaXmlDto
        ↳ atividadesConselhoComissaoConsultoria;

    @XmlElement(name = "ATIVIDADES-DE-PARTICIPACAO-EM-PROJETO")
    private AtividadesParticipacaoProjetoXmlDto atividadesParticipacaoProjeto;

    // CODIGO-INSTITUICAO CDATA #IMPLIED
    // NOME-INSTITUICAO CDATA #IMPLIED
    // SEQUENCIA-ATIVIDADE CDATA #IMPLIED
    // SEQUENCIA-IMPORTANCIA CDATA #IMPLIED

    @XmlAttribute(name = "CODIGO-INSTITUICAO")
    private String codigoInstituicao;

    @XmlAttribute(name = "NOME-INSTITUICAO")
    private String nomeInstituicao;

    @XmlAttribute(name = "SEQUENCIA-ATIVIDADE")
    private String sequenciaAtividade;

    @XmlAttribute(name = "SEQUENCIA-IMPORTANCIA")
    private String sequenciaImportancia;
}
package ufsc.alpc.xml.dto.dados.gerais.atuacoes.outras.atividadestecnicocientifica;

import java.util.List;

```

```

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OutrasAtividadesTecnicoCientificaXmlDto {

    // OUTRA-ATIVIDADE-TECNICO-CIENTIFICA+

    @XmlElement(name = "OUTRA-ATIVIDADE-TECNICO-CIENTIFICA", required =
        ↵ true)
    private List<OutraAtividadeTecnicoCientificaXmlDto> outrasAtividadesTecnicoCientifica;

}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.outras.atividadestecnicocientifica;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.AtividadeUnidadeXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OutraAtividadeTecnicoCientificaXmlDto extends AtividadeUnidadeXmlDto {

    // SEQUENCIA-FUNCAO-ATIVIDADE CDATA #IMPLIED
    // FLAG-PERODO (ANTERIOR | ATUAL) #IMPLIED
    // MES-INICIO CDATA #IMPLIED
    // ANO-INICIO CDATA #IMPLIED
    // MES-FIM CDATA #IMPLIED
    // ANO-FIM CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-UNIDADE CDATA #IMPLIED
    // NOME-UNIDADE CDATA #IMPLIED

    // ATIVIDADE-REALIZADA CDATA #IMPLIED

    @XmlAttribute(name = "ATIVIDADE-REALIZADA")
    protected String atividadeRealizada;

}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AtuacoesProfissionaisXmlDto {

    @XmlElement(name = "ATUACAO-PROFISSIONAL")
    private List<AtuacaoProfissionalXmlDto> atuacoes;

}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.extensaouniversitaria;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;

```

```

import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AtividadesExtensaoUniversitariaXmlDto {

    // EXTENSAO-UNIVERSITARIA+

    @XmlElement(name = "EXTENSAO-UNIVERSITARIA", required = true)
    private List<ExtensaoUniversitariaXmlDto> extensoesUniversitarias;

}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.extensaouniversitaria;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.AtividadeUnidadeXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ExtensaoUniversitariaXmlDto extends AtividadeUnidadeXmlDto {

    // SEQUENCIA-FUNCAO-ATIVIDADE CDATA #IMPLIED
    // FLAG-PERODO (ANTERIOR | ATUAL) #IMPLIED
    // MES-INICIO CDATA #IMPLIED
    // ANO-INICIO CDATA #IMPLIED
    // MES-FIM CDATA #IMPLIED
    // ANO-FIM CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-UNIDADE CDATA #IMPLIED
    // NOME-UNIDADE CDATA #IMPLIED

    // ATIVIDADE-DE-EXTENSAO-REALIZADA CDATA #IMPLIED

    @XmlAttribute(name = "ATIVIDADE-DE-EXTENSAO-REALIZADA")
    protected String atividadeExtensaoRealizada;

}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.treinamentoministrado;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AtividadesTreinamentoMinistradoXmlDto {

    // TREINAMENTO-MINISTRADO+

    @XmlElement(name = "TREINAMENTO-MINISTRADO", required = true)
    private List<TreinamentoMinistradoXmlDto> treinamentosMinistrados;

}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.treinamentoministrado;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;

```

```

import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufs.alpc.xml.dto.dados.gerais.atuacoes.atividades.AtividadeUnidadeXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class TreinamentoMinistradoXmlDto extends AtividadeUnidadeXmlDto {
    // TREINAMENTO+

    @XmlElement(name = "TREINAMENTO", required = true)
    private List<TreinamentoXmlDto> treinamentos;
}
package ufs.alpc.xml.dto.dados.gerais.atuacoes.atividades.treinamentoministrado;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlValue;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class TreinamentoXmlDto {
    // SEQUENCIA-ESPECIFICACAO CDATA #IMPLIED

    @XmlAttribute(name = "SEQUENCIA-ESPECIFICACAO")
    private String sequenciaEspecificacao;

    @XmlValue
    private String pcdata;
}
package ufs.alpc.xml.dto.dados.gerais.atuacoes.atividades.ensino;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufs.alpc.xml.dto.dados.gerais.atuacoes.atividades.pesquisadesenvolvimento.ensino.EnsinoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AtividadesEnsinoXmlDto {
    // ENSINO*

    @XmlElement(name = "ENSINO")
    private List<EnsinoXmlDto> ensino;
}
package ufs.alpc.xml.dto.dados.gerais.atuacoes.atividades;

import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import ufs.alpc.xml.adapter.PeriodoXmlAdapter;

public class AtividadeXmlDto {
    // SEQUENCIA-FUNCAO-ATIVIDADE CDATA #IMPLIED
    // FLAG-PERODO (ANTERIOR | ATUAL) #IMPLIED

```

```

// MES-INICIO CDATA #IMPLIED
// ANO-INICIO CDATA #IMPLIED
// MES-FIM CDATA #IMPLIED
// ANO-FIM CDATA #IMPLIED
// CODIGO-ORGAO CDATA #IMPLIED
// NOME-ORGAO CDATA #IMPLIED

@XmlAttribute(name = "SEQUENCIA-FUNCAO-ATIVIDADE")
protected String sequenciaFuncaoAtividade;

@XmlJavaTypeAdapter(PeriodoXmlAdapter.class)
@XmlAttribute(name = "FLAG-PERIODO")
protected Boolean flagPeriodo;

@XmlAttribute(name = "MES-INICIO")
protected String mesInicio;

@XmlAttribute(name = "ANO-INICIO")
protected String anoInicio;

@XmlAttribute(name = "MES-FIM")
protected String mesFim;

@XmlAttribute(name = "ANO-FIM")
protected String anoFim;

@XmlAttribute(name = "CODIGO-ORGAO")
protected String codigoOrgao;

@XmlAttribute(name = "NOME-ORGAO")
protected String nomeOrgao;

public String getAnoFim() {
    return anoFim;
}

public String getAnoInicio() {
    return anoInicio;
}

public String getMesFim() {
    return mesFim;
}

public String getMesInicio() {
    return mesInicio;
}

public Boolean getFlagPeriodo() {
    return flagPeriodo;
}
}
package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades;

import javax.xml.bind.annotation.XmlAttribute;

public class AtividadeUnidadeXmlDto extends AtividadeXmlDto {

    // SEQUENCIA-FUNCAO-ATIVIDADE CDATA #IMPLIED
    // FLAG-PERIODO (ANTERIOR | ATUAL) #IMPLIED
    // MES-INICIO CDATA #IMPLIED
    // ANO-INICIO CDATA #IMPLIED
    // MES-FIM CDATA #IMPLIED
    // ANO-FIM CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-UNIDADE CDATA #IMPLIED
    // NOME-UNIDADE CDATA #IMPLIED

    @XmlAttribute(name = "CODIGO-UNIDADE")
    protected String codigoUnidade;

    @XmlAttribute(name = "NOME-UNIDADE")
    protected String nomeUnidade;
}
package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.estagio;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

```

```

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.AtividadeUnidadeXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class EstagioXmlDto extends AtividadeUnidadeXmlDto {

    // SEQUENCIA-FUNCAO-ATIVIDADE CDATA #IMPLIED
    // FLAG-PERIODO (ANTERIOR | ATUAL) #IMPLIED
    // MES-INICIO CDATA #IMPLIED
    // ANO-INICIO CDATA #IMPLIED
    // MES-FIM CDATA #IMPLIED
    // ANO-FIM CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-UNIDADE CDATA #IMPLIED
    // NOME-UNIDADE CDATA #IMPLIED

    // ESTAGIO-REALIZADO CDATA #IMPLIED

    @XmlAttribute(name = "ESTAGIO-REALIZADO")
    private String estagioRealizado;

}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.estagio;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AtividadesEstagioXmlDto {

    // ESTAGIO+

    @XmlElement(name = "ESTAGIO", required = true)
    private List<EstagioXmlDto> estagio;

}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.direcaoadministracao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.AtividadeUnidadeXmlDto;
import ufsc.alpc.xml.dto.domain.FormatoCargoFuncao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DirecaoAdministracaoXmlDto extends AtividadeUnidadeXmlDto {

    // SEQUENCIA-FUNCAO-ATIVIDADE CDATA #IMPLIED
    // FLAG-PERIODO (ANTERIOR | ATUAL) #IMPLIED
    // MES-INICIO CDATA #IMPLIED
    // ANO-INICIO CDATA #IMPLIED
    // MES-FIM CDATA #IMPLIED
    // ANO-FIM CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-UNIDADE CDATA #IMPLIED
    // NOME-UNIDADE CDATA #IMPLIED

```



```

// FORMATO-CARGO-OU-FUNCAO (CHEFE_DE_DEPARTAMENTO |
// ↳ COORDENADOR_DE_CURSO |
// COORDENADOR_DE_PROGRAMA | DECANO_DE_CENTRO |
// ↳ DIRETOR_DE_UNIDADE |
// MEMBRO_DE_COLEGIADO_SUPERIOR |
// ↳ MEMBRO_DE_COMISSAO_PERMANENTE |
// MEMBRO_DE_COMISSAO_TEMPORARIA |
// ↳ MEMBRO_DE_CONSELHO_DE_CENTRO |
// MEMBRO_DE_CONSELHO_DE_UNIDADE | REITOR |
// ↳ VICE_REITOR_OU_PRO_REITOR |
// OUTRO | LIVRE) #IMPLIED
// CARGO-OU-FUNCAO CDATA #IMPLIED
// CARGO-OU-FUNCAO-INGLES CDATA #IMPLIED

@XmlAttribute(name = "FORMATO-CARGO-OU-FUNCAO")
private FormatoCargoFuncao formatoCargoFuncao;

@XmlAttribute(name = "CARGO-OU-FUNCAO")
private String cargoFuncao;

@XmlAttribute(name = "CARGO-OU-FUNCAO-INGLES")
private String cargoFuncaoIngles;
}
package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.direcaoadministracao;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AtividadesDirecaoAdministracaoXmlDto {

    // DIRECAO-E-ADMINISTRACAO+

    @XmlElement(name = "DIRECAO-E-ADMINISTRACAO", required = true)
    private List<DirecaoAdministracaoXmlDto> direcaoAdministracao;
}
package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.pesquisadesenvolvimento;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AtividadesPesquisaDesenvolvimentoXmlDto {

    // PESQUISA-E-DESENVOLVIMENTO+*

    @XmlElement(name = "PESQUISA-E-DESENVOLVIMENTO", required = true)
    private List<PesquisaDesenvolvimentoXmlDto> pesquisasDesenvolvimentos;
}
package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.pesquisadesenvolvimento.ensino.disciplina;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlValue;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

```

```

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DisciplinaXmlDto {

    // SEQUENCIA-ESPECIFICACAO CDATA #IMPLIED

    @XmlAttribute(name = "SEQUENCIA-ESPECIFICACAO")
    private String sequenciaEspecificacao;

    @XmlValue
    private String pcdata;

}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.pesquisadesenvolvimento.ensino;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.AtividadeXmlDto;
import ↪ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.pesquisadesenvolvimento.ensino.disciplina.DisciplinaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class EnsinoXmlDto extends AtividadeXmlDto {

    // DISCIPLINA*

    @XmlElement(name = "DISCIPLINA")
    private List<DisciplinaXmlDto> disciplina;

    // SEQUENCIA-FUNCAO-ATIVIDADE CDATA #IMPLIED
    // FLAG-PERODO (ANTERIOR | ATUAL) #IMPLIED
    // MES-INICIO CDATA #IMPLIED
    // ANO-INICIO CDATA #IMPLIED
    // MES-FIM CDATA #IMPLIED
    // ANO-FIM CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED

    // TIPO-ENSINO (GRADUACAO | POS-GRADUACAO | ESPECIALIZACAO |
    // ↪ APERFEICOAMENTO
    // | ENSINO-FUNDAMENTAL | ENSINO-MEDIO | OUTRO) #IMPLIED
    // CODIGO-CURSO CDATA #IMPLIED
    // NOME-CURSO CDATA #IMPLIED
    // NOME-CURSO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "TIPO-ENSINO")
    protected String tipoEnsino;

    @XmlAttribute(name = "CODIGO-CURSO")
    protected String codigoCurso;

    @XmlAttribute(name = "NOME-CURSO")
    protected String nomeCurso;

    @XmlAttribute(name = "NOME-CURSO-INGLES")
    protected String nomeCursoIngles;

}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.pesquisadesenvolvimento;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

```

```

import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.AtividadeUnidadeXmlDto;
import
↳ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.pesquisadesenvolvimento.linhapesquisa.LinhaPesquisaX

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class PesquisaDesenvolvimentoXmlDto extends AtividadeUnidadeXmlDto {

    // LINHA-DE-PESQUISA*

    // SEQUENCIA-FUNCAO-ATIVIDADE CDATA #IMPLIED
    // FLAG-PERIODO (ANTERIOR | ATUAL) #IMPLIED
    // MES-INICIO CDATA #IMPLIED
    // ANO-INICIO CDATA #IMPLIED
    // MES-FIM CDATA #IMPLIED
    // ANO-FIM CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-UNIDADE CDATA #IMPLIED
    // NOME-UNIDADE CDATA #IMPLIED

    @XmlElement(name = "LINHA-DE-PESQUISA")
    private List<LinhaPesquisaXmlDto> linhaPesquisa;

}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.pesquisadesenvolvimento.linhapesquisa;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.common.PalavraAreasSetoresXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class LinhaPesquisaXmlDto extends PalavraAreasSetoresXmlDto {

    // PALAVRAS-CHAVE?, AREAS-DO-CONHECIMENTO?,
    ↳ SETORES-DE-ATIVIDADE?

    // SEQUENCIA-LINHA CDATA #IMPLIED
    // TITULO-DA-LINHA-DE-PESQUISA CDATA #IMPLIED
    // FLAG-LINHA-DE-PESQUISA-ATIVA (SIM | NAO) #IMPLIED
    // OBJETIVOS-LINHA-DE-PESQUISA CDATA #IMPLIED
    // TITULO-DA-LINHA-DE-PESQUISA-INGLES CDATA #IMPLIED
    // OBJETIVOS-LINHA-DE-PESQUISA-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "SEQUENCIA-LINHA")
    private String sequenciaLinha;

    @XmlAttribute(name = "TITULO-DA-LINHA-DE-PESQUISA")
    private String tituloLinhaPesquisa;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-LINHA-DE-PESQUISA-ATIVA")
    private Boolean flagLinhaPesquisaAtiva;

    @XmlAttribute(name = "OBJETIVOS-LINHA-DE-PESQUISA")
    private String objetivosLinhaPesquisa;

    @XmlAttribute(name = "TITULO-DA-LINHA-DE-PESQUISA-INGLES")
    private String tituloLinhaPesquisaIngles;

    @XmlAttribute(name = "OBJETIVOS-LINHA-DE-PESQUISA-INGLES")
    private String objetivosLinhaPesquisaIngles;

}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;

```

```

import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.AtividadeUnidadeXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.ProjetoPesquisaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoProjetoXmlDto extends AtividadeUnidadeXmlDto {
    // PROJETO-DE-PESQUISA*

    @XmlElement(name = "PROJETO-DE-PESQUISA")
    private List<ProjetoPesquisaXmlDto> projetoPesquisa;
}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AtividadesParticipacaoProjetoXmlDto {
    // PARTICIPACAO-EM-PROJETO*

    @XmlElement(name = "PARTICIPACAO-EM-PROJETO")
    private List<ParticipacaoProjetoXmlDto> participacaoProjeto;
}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.producao;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.producao.ProducaoCtProjeto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ProducoesCtDoProjetoXmlDto {
    // PRODUCAO-CT-DO-PROJETO+

    @XmlElement(name = "PRODUCAO-CT-DO-PROJETO", required = true)
    private List<ProducaoCtProjetoXmlDto> producoesCtProjeto;
}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.producao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

```

```

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ProducaoCtProjetoXmlDto {
    // SEQUENCIA-PRODUCAO-CT CDATA #IMPLIED
    // TITULO-DA-PRODUCAO-CT CDATA #IMPLIED
    // TIPO-PRODUCAO-CT CDATA #IMPLIED
    // TITULO-DA-PRODUCAO-CT-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "SEQUENCIA-PRODUCAO-CT")
    private String sequenciaProducaoCt;

    @XmlAttribute(name = "TITULO-DA-PRODUCAO-CT")
    private String tituloProducaoCt;

    @XmlAttribute(name = "TIPO-PRODUCAO-CT")
    private String tipoProducaoCt;

    @XmlAttribute(name = "TITULO-DA-PRODUCAO-CT-INGLES")
    private String tituloProducaoCtIngles;
}
package
↪ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.orientacao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OrientacaoXmlDto {
    // SEQUENCIA-ORIENTACAO CDATA #IMPLIED
    // TITULO-ORIENTACAO CDATA #IMPLIED
    // TIPO-ORIENTACAO CDATA #IMPLIED
    // TITULO-ORIENTACAO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "SEQUENCIA-ORIENTACAO")
    private String sequenciaOrientacao;

    @XmlAttribute(name = "TITULO-ORIENTACAO")
    private String tituloOrientacao;

    @XmlAttribute(name = "TIPO-ORIENTACAO")
    private String tipoOrientacao;

    @XmlAttribute(name = "TITULO-ORIENTACAO-INGLES")
    private String tituloOrientacaoIngles;
}
package
↪ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.orientacao;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OrientacoesXmlDto {
    // ORIENTACAO+

    @XmlElement(name = "ORIENTACAO", required = true)
    private List<OrientacaoXmlDto> orientacoes;
}

```

```

}
package
↳ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.equipe;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class EquipeDoProjetoXmlDto {

    // INTEGRANTES-DO-PROJETO*
    @XmlElement(name = "INTEGRANTES-DO-PROJETO")
    private List<IntegrantesProjetoXmlDto> integrantesProjeto;

}
package
↳ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.equipe;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlValue;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class IntegrantesProjetoXmlDto {

    // NOME-COMPLETO CDATA #IMPLIED
    // NOME-PARA-CITACAO CDATA #IMPLIED
    // ORDEM-DE-INTEGRACAO CDATA #IMPLIED
    // FLAG-RESPONSAVEL (SIM | NAO) #IMPLIED
    // NRO-ID-CNPQ CDATA #IMPLIED

    @XmlAttribute(name = "NOME-COMPLETO")
    private String nomeCompleto;

    @XmlAttribute(name = "NOME-PARA-CITACAO")
    private String nomeCitacao;

    @XmlAttribute(name = "ORDEM-DE-INTEGRACAO")
    private String ordemIntegracao;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-RESPONSAVEL")
    private Boolean flagResponsavel;

    @XmlAttribute(name = "NRO-ID-CNPQ")
    private String nroIdCnpq;

    @XmlValue
    private String pData;

}
package
↳ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.financiador;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

```

```

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class FinanciadoresDoProjetoXmlDto {

    // FINANCIADOR-DO-PROJETO+

    @XmlElement(name = "FINANCIADOR-DO-PROJETO", required = true)
    private List<FinanciadorProjetoXmlDto> integrantesProjeto;
}
package
↳ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.financiador;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.domain.NaturezaFinanciador;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class FinanciadorProjetoXmlDto {

    // SEQUENCIA-FINANCIADOR CDATA #IMPLIED
    // CODIGO-INSTITUICAO CDATA #IMPLIED
    // NOME-INSTITUICAO CDATA #IMPLIED
    // NATUREZA (BOLSA | AUXILIO_FINANCEIRO | REMUNERACAO | OUTRO |
    ↳ COOPERACAO |
    // NAO_INFORMADO) #IMPLIED

    @XmlAttribute(name = "SEQUENCIA-FINANCIADOR")
    private String sequenciaFinanciador;

    @XmlAttribute(name = "CODIGO-INSTITUICAO")
    private String codigoInstituicao;

    @XmlAttribute(name = "NOME-INSTITUICAO")
    private String nomeInstituicao;

    @XmlAttribute(name = "NATUREZA")
    private NaturezaFinanciador natureza;
}
package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa;

import java.util.Date;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.adapter.DateXmlAdapter;
import
↳ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.equipe.EquipeDoPr
import
↳ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.financiador.Financiador
import
↳ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.orientacao.Orientacao
import
↳ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.producao.Producao
import ufsc.alpc.xml.dto.domain.NaturezaProjeto;
import ufsc.alpc.xml.dto.domain.SituacaoProjeto;

@Getter
@Setter

```

```

@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ProjetoPesquisaXmlDto {

    // EQUIPE-DO-PROJETO?
    // FINANCIADORES-DO-PROJETO?
    // PRODUCOES-CT-DO-PROJETO?
    // ORIENTACOES?

    @XmlElement(name = "EQUIPE-DO-PROJETO")
    private EquipeDoProjetoXmlDto equipeProjeto;

    @XmlElement(name = "FINANCIADORES-DO-PROJETO")
    private FinanciadoresDoProjetoXmlDto financiadoresProjeto;

    @XmlElement(name = "PRODUCOES-CT-DO-PROJETO")
    private ProducoesCtDoProjetoXmlDto producoesCtProjeto;

    @XmlElement(name = "ORIENTACOES")
    private OrientacoesXmlDto orientacoes;

    // SEQUENCIA-PROJETO CDATA #IMPLIED
    // ANO-INICIO CDATA #IMPLIED
    // ANO-FIM CDATA #IMPLIED
    // NOME-DO-PROJETO CDATA #IMPLIED
    // SITUACAO (EM-ANDAMENTO | DESATIVADO | CONCLUIDO) #IMPLIED
    // NATUREZA (DESENVOLVIMENTO | EXTENSAO | PESQUISA | OUTRA) #IMPLIED
    // NUMERO-GRADUACAO CDATA #IMPLIED
    // NUMERO-ESPECIALIZACAO CDATA #IMPLIED
    // NUMERO-MESTRADO-ACADEMICO CDATA #IMPLIED
    // NUMERO-MESTRADO-PROF CDATA #IMPLIED
    // NUMERO-DOCTORADO CDATA #IMPLIED
    // DESCRICAO-DO-PROJETO CDATA #IMPLIED
    // IDENTIFICADOR-PROJETO CDATA #IMPLIED
    // DESCRICAO-DO-PROJETO-INGLES CDATA #IMPLIED
    // NOME-DO-PROJETO-INGLES CDATA #IMPLIED
    // FLAG-POTENCIAL-INOVACAO (SIM | NAO) "NAO"
    // FLAG-PROJETO-CERTIFICADO (SIM | NAO) "NAO"
    // NOME-COORDENADOR-CERTIFICACAO CDATA #IMPLIED
    // DATA-CERTIFICACAO CDATA #IMPLIED
    // NUMERO_TECNICO_NIVEL_MEDIO CDATA #IMPLIED

    @XmlAttribute(name = "SEQUENCIA-PROJETO")
    private String sequenciaProjeto;

    @XmlAttribute(name = "ANO-INICIO")
    private String anoInicio;

    @XmlAttribute(name = "ANO-FIM")
    private String anoFim;

    @XmlAttribute(name = "NOME-DO-PROJETO")
    private String nomeProjeto;

    @XmlAttribute(name = "SITUACAO")
    private SituacaoProjeto situacao;

    @XmlAttribute(name = "NATUREZA")
    private NaturezaProjeto natureza;

    @XmlAttribute(name = "NUMERO-GRADUACAO")
    private String numeroGraduacao;

    @XmlAttribute(name = "NUMERO-ESPECIALIZACAO")
    private String numeroEspecializacao;

    @XmlAttribute(name = "NUMERO-MESTRADO-ACADEMICO")
    private String numeroMestradoAcademico;

    @XmlAttribute(name = "NUMERO-MESTRADO-PROF")
    private String numeroMestradoProf;

    @XmlAttribute(name = "NUMERO-DOCTORADO")
    private String numeroDoutorado;

    @XmlAttribute(name = "DESCRICAO-DO-PROJETO")
    private String descricaoProjeto;

    @XmlAttribute(name = "IDENTIFICADOR-PROJETO")
    private String identificadorProjeto;

    @XmlAttribute(name = "DESCRICAO-DO-PROJETO-INGLES")

```



```

private String descricaoProjetoIngles;

@XmlAttribute(name = "NOME-DO-PROJETO-INGLES")
private String nomeProjetoIngles;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-POTENCIAL-INOVACAO")
private Boolean flagPotencialInovacao;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-PROJETO-CERTIFICADO")
private Boolean flagProjetoCertificado;

@XmlAttribute(name = "NOME-COORDENADOR-CERTIFICACAO")
private String nomeCoordenadorCertificacao;

@XmlJavaTypeAdapter(DateXmlAdapter.class)
@XmlAttribute(name = "DATA-CERTIFICACAO")
private Date dataCertificacao;

@XmlAttribute(name = "NUMERO_TECNICO_NIVEL_MEDIO")
private String numeroTecnicoNivelMedio;
}
package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.conselhocomissao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.AtividadeUnidadeXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ConselhoComissaoConsultoriaXmlDto extends AtividadeUnidadeXmlDto {

    // SEQUENCIA-FUNCAO-ATIVIDADE CDATA #IMPLIED
    // FLAG-PERODO (ANTERIOR | ATUAL)
    // MES-INICIO CDATA #IMPLIED
    // ANO-INICIO CDATA #IMPLIED
    // MES-FIM CDATA #IMPLIED
    // ANO-FIM CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-UNIDADE CDATA #IMPLIED
    // NOME-UNIDADE CDATA #IMPLIED

    // ESPECIFICACAO CDATA #IMPLIED

    @XmlAttribute(name = "ESPECIFICACAO")
    private String especificacao;
}
package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.conselhocomissao;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AtividadesConselhoComissaoConsultoriaXmlDto {

    // CONSELHO-COMISSAO-E-CONSULTORIA*

    @XmlElement(name = "CONSELHO-COMISSAO-E-CONSULTORIA")
    private List<ConselhoComissaoConsultoriaXmlDto> conselhoComissaoConsultoria;
}
package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.servicotecnico;

```

```

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.AtividadeUnidadeXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ServicoTecnicoEspecializadoXmlDto extends AtividadeUnidadeXmlDto {

    // SEQUENCIA-FUNCAO-ATIVIDADE CDATA #IMPLIED
    // FLAG-PERÍODO (ANTERIOR | ATUAL) #IMPLIED
    // MES-INÍCIO CDATA #IMPLIED
    // ANO-INÍCIO CDATA #IMPLIED
    // MES-FIM CDATA #IMPLIED
    // ANO-FIM CDATA #IMPLIED
    // CÓDIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-UNIDADE CDATA #IMPLIED
    // NOME-UNIDADE CDATA #IMPLIED

    // SERVICIO-REALIZADO CDATA #IMPLIED

    @XmlAttribute(name = "SERVICIO-REALIZADO")
    private String servicoRealizado;

}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.servicotecnico;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AtividadesServicoTecnicoEspecializadoXmlDto {

    // SERVICIO-TECNICO-ESPECIALIZADO+

    @XmlElement(name = "SERVICIO-TECNICO-ESPECIALIZADO", required = true)
    private List<ServicoTecnicoEspecializadoXmlDto> servicosTecnicoEspecializado;

}

package ufsc.alpc.xml.dto.dados.gerais.atuacoes.vinculos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.EnquadramentoFuncional;
import ufsc.alpc.xml.dto.domain.TipoVinculo;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class VinculosXmlDto {

    // SEQUENCIA-HISTORICO CDATA #IMPLIED
    // TIPO-DE-VINCULO (SERVIDOR_PUBLICO_OR_COLETISTA |
    //   ↳ SERVIDOR_PUBLICO |
    //   ↳ COLETISTA | PROFESSOR_VISITANTE | COLABORADOR |
    //   ↳ BOLSISTA_RECEM_DOUTOR |

```

```

// OUTRO | LIVRE) #IMPLIED
// ENQUADRAMENTO-FUNCIONAL (PROFESSOR_TITULAR | OUTRO | LIVRE)
// ↪ #IMPLIED
// CARGA-HORARIA-SEMANAL CDATA #IMPLIED
// FLAG-DEDICACAO-EXCLUSIVA (SIM | NAO) #IMPLIED
// MES-INICIO CDATA #IMPLIED
// ANO-INICIO CDATA #IMPLIED
// MES-FIM CDATA #IMPLIED
// ANO-FIM CDATA #IMPLIED
// OUTRAS-INFORMACOES CDATA #IMPLIED
// FLAG-VINCULO-EMPREGATICIO (SIM | NAO) #IMPLIED
// OUTRO-VINCULO-INFORMADO CDATA #IMPLIED
// OUTRO-ENQUADRAMENTO-FUNCIONAL-INFORMADO CDATA #IMPLIED
// OUTRO-ENQUADRAMENTO-FUNCIONAL-INFORMADO-INGLES CDATA
// ↪ #IMPLIED
// OUTRAS-INFORMACOES-INGLES CDATA #IMPLIED

@XmlAttribute(name = "SEQUENCIA-HISTORICO")
private String sequenciaHistorico;

@XmlAttribute(name = "TIPO-DE-VINCULO")
private TipoVinculo tipoVinculo;

@XmlAttribute(name = "ENQUADRAMENTO-FUNCIONAL")
private EnquadramentoFuncional enquadramentoFuncional;

@XmlAttribute(name = "CARGA-HORARIA-SEMANAL")
private String cargaHorariaSemanal;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-DEDICACAO-EXCLUSIVA-RELEVANCIA")
private Boolean flagDedicacaoExclusiva;

@XmlAttribute(name = "MES-INICIO")
private String mesInicio;

@XmlAttribute(name = "ANO-INICIO")
private String anoInicio;

@XmlAttribute(name = "MES-FIM")
private String mesFim;

@XmlAttribute(name = "ANO-FIM")
private String anoFim;

@XmlAttribute(name = "OUTRAS-INFORMACOES")
private String outrasInformacoes;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-VINCULO-EMPREGATICIO")
private Boolean flagVinculoEmpregaticio;

@XmlAttribute(name = "OUTRO-VINCULO-INFORMADO")
private String outroVinculoInformado;

@XmlAttribute(name = "OUTRO-ENQUADRAMENTO-FUNCIONAL-INFORMADO")
private String outroEnquadramentoFuncionalInformado;

@XmlAttribute(name =
    ↪ "OUTRO-ENQUADRAMENTO-FUNCIONAL-INFORMADO-INGLES")
private String outroEnquadramentoFuncionalInformadoIngles;

@XmlAttribute(name = "OUTRAS-INFORMACOES-INGLES")
private String outrasInformacoesIngles;
}
package ufsc.alpc.xml.dto.dados.gerais.premiostitulos;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class PremiosTitulosXmlDto {

```

```

// PREMIO-TITULO*
@XmlElement(name = "PREMIO-TITULO")
private List<PremioTituloXmlDto> premiosTitulos;
}
package ufsc.alpc.xml.dto.dados.gerais.premiostitulos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class PremioTituloXmlDto {

    // NOME-DO-PREMIO-OU-TITULO CDATA #IMPLIED
    // NOME-DA-ENTIDADE-PROMOTORA CDATA #IMPLIED
    // ANO-DA-PREMIACAO CDATA #IMPLIED
    // NOME-DO-PREMIO-OU-TITULO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "NOME-DO-PREMIO-OU-TITULO")
    private String nomePremioTitulo;

    @XmlAttribute(name = "NOME-DA-ENTIDADE-PROMOTORA")
    private String nomeEntidadePromotora;

    @XmlAttribute(name = "ANO-DA-PREMIACAO")
    private String anoPremiacao;

    @XmlAttribute(name = "NOME-DO-PREMIO-OU-TITULO-INGLES")
    private String nomePremioTituloIngles;
}package ufsc.alpc.xml.dto.dados.gerais.endereco;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.EnderecoXmlAdapter;
import ufsc.alpc.xml.dto.dados.gerais.endereco.profissional.EnderecoProfissionalXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.endereco.residencial.EnderecoResidencialXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class EnderecoXmlDto {

    // ENDERECO-PROFISSIONAL?
    // ENDERECO-RESIDENCIAL?
    // FLAG-DE-PREFERENCIA (ENDERECO_INSTITUCIONAL |
    // ↪ ENDERECO_RESIDENCIAL)

    @XmlElement(name = "ENDERECO-PROFISSIONAL")
    private EnderecoProfissionalXmlDto enderecoProfissional;

    @XmlElement(name = "ENDERECO-RESIDENCIAL")
    private EnderecoResidencialXmlDto enderecoResidencial;

    @XmlJavaTypeAdapter(EnderecoXmlAdapter.class)
    @XmlAttribute(name = "FLAG-DE-PREFERENCIA")
    protected Boolean flagPreferencia;
}
package ufsc.alpc.xml.dto.dados.gerais.endereco.profissional;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

```

```

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufs.alpc.xml.dto.dados.gerais.endereco.EnderecoBaseXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class EnderecoProfissionalXmlDto extends EnderecoBaseXmlDto {

    // CODIGO-INSTITUICAO-EMPRESA CDATA #IMPLIED
    // NOME-INSTITUICAO-EMPRESA CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-UNIDADE CDATA #IMPLIED
    // NOME-UNIDADE CDATA #IMPLIED
    // LOGRADOURO-COMPLEMENTO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // UF CDATA #IMPLIED
    // CEP CDATA #IMPLIED
    // CIDADE CDATA #IMPLIED
    // BAIRRO CDATA #IMPLIED
    // DDD CDATA #IMPLIED
    // TELEFONE CDATA #IMPLIED
    // RAMAL CDATA #IMPLIED
    // FAX CDATA #IMPLIED
    // CAIXA-POSTAL CDATA #IMPLIED
    // E-MAIL CDATA #IMPLIED
    // HOME-PAGE CDATA #IMPLIED

    @XmlAttribute(name = "CODIGO-INSTITUICAO-EMPRESA")
    private String codigoInstituicaoEmpresa;

    @XmlAttribute(name = "NOME-INSTITUICAO-EMPRESA")
    private String nomeInstituicaoEmpresa;

    @XmlAttribute(name = "CODIGO-ORGAO")
    private String codigoOrgao;

    @XmlAttribute(name = "NOME-ORGAO")
    private String nomeOrgao;

    @XmlAttribute(name = "CODIGO-UNIDADE")
    private String codigoUnidade;

    @XmlAttribute(name = "NOME-UNIDADE")
    private String nomeUnidade;

    @XmlAttribute(name = "LOGRADOURO-COMPLEMENTO")
    private String logradouroComplemento;
}
package ufs.alpc.xml.dto.dados.gerais.endereco.residencial;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufs.alpc.xml.dto.dados.gerais.endereco.EnderecoBaseXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class EnderecoResidencialXmlDto extends EnderecoBaseXmlDto {

    // LOGRADOURO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // UF CDATA #IMPLIED
    // CEP CDATA #IMPLIED
    // CIDADE CDATA #IMPLIED
    // BAIRRO CDATA #IMPLIED
    // DDD CDATA #IMPLIED
    // TELEFONE CDATA #IMPLIED
    // RAMAL CDATA #IMPLIED
    // FAX CDATA #IMPLIED
    // CAIXA-POSTAL CDATA #IMPLIED

```

```

// E-MAIL CDATA #IMPLIED
// HOME-PAGE CDATA #IMPLIED

@XmlAttribute(name = "LOGRADOURO")
private String logradouro;

}
package ufsc.alpc.xml.dto.dados.gerais.endereco;

import javax.xml.bind.annotation.XmlAttribute;

public class EnderecoBaseXmlDto {

// PAIS CDATA #IMPLIED
// UF CDATA #IMPLIED
// CEP CDATA #IMPLIED
// CIDADE CDATA #IMPLIED
// BAIRRO CDATA #IMPLIED
// DDD CDATA #IMPLIED
// TELEFONE CDATA #IMPLIED
// RAMAL CDATA #IMPLIED
// FAX CDATA #IMPLIED
// CAIXA-POSTAL CDATA #IMPLIED
// E-MAIL CDATA #IMPLIED
// HOME-PAGE CDATA #IMPLIED

@XmlAttribute(name = "PAIS")
protected String pais;

@XmlAttribute(name = "UF")
protected String uf;

@XmlAttribute(name = "CEP")
protected String cep;

@XmlAttribute(name = "CIDADE")
protected String cidade;

@XmlAttribute(name = "BAIRRO")
protected String bairro;

@XmlAttribute(name = "DDD")
protected String ddd;

@XmlAttribute(name = "TELEFONE")
protected String telefone;

@XmlAttribute(name = "RAMAL")
protected String ramal;

@XmlAttribute(name = "FAX")
protected String fax;

@XmlAttribute(name = "CAIXA-POSTAL")
protected String caixaPostal;

@XmlAttribute(name = "E-MAIL")
protected String email;

@XmlAttribute(name = "HOME-PAGE")
protected String homePage;

}
package ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.mestrado;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoXmlDto;
import ufsc.alpc.xml.dto.common.SequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.DadosBasicosDaOrientacaoEmAndamentoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OrientacaoEmAndamentoDeMestradoXmlDto extends

```

```

↪ SequenciaPalavraAreaSetorInfoAgrupadorXmlDto {
    // DADOS-BASICOS-DA-ORIENTACAO-EM-ANDAMENTO-DE-MESTRADO?,
    // DETALHAMENTO-DA-ORIENTACAO-EM-ANDAMENTO-DE-MESTRADO?,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↪ "DADOS-BASICOS-DA-ORIENTACAO-EM-ANDAMENTO-DE-MESTRADO")
    private DadosBasicosDaOrientacaoEmAndamentoXmlDto dadosBasicos;

    @XmlElement(name =
        ↪ "DETALHAMENTO-DA-ORIENTACAO-EM-ANDAMENTO-DE-MESTRADO")
    private DetalhamentoDaOrientacaoXmlDto detalhamento;
}

package ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosNaturezaIdiomaHomePageXmlDto;
import ufsc.alpc.xml.dto.domain.Tipo;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDaOrientacaoEmAndamentoXmlDto extends
    ↪ DadosBasicosNaturezaIdiomaHomePageXmlDto {

    // NATUREZA CDATA #IMPLIED
    // TIPO (ACADEMICO | PROFISSIONALIZANTE | NAO_INFORMADO)
    ↪ "NAO_INFORMADO"
    // TITULO-DO-TRABALHO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // HOME-PAGE CDATA #IMPLIED
    // DOI CDATA #IMPLIED
    // TITULO-DO-TRABALHO-INGLES CDATA #IMPLIED

    // apenas para OrientacaoEmAndamentoDeMestradoXmlDto
    @XmlAttribute(name = "TIPO")
    private Tipo tipo;

    @XmlAttribute(name = "TITULO-DO-TRABALHO")
    protected String tituloDoTrabalho;

    @XmlAttribute(name = "TITULO-DO-TRABALHO-INGLES")
    protected String tituloDoTrabalhoIngles;
}

package ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.iniciacao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoXmlDto;
import ufsc.alpc.xml.dto.common.SequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.DadosBasicosDaOrientacaoEmAndamento;

↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.DadosBasicosDaOrientacaoEmAndamento;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OrientacaoEmAndamentoDeIniciacaoCientificaXmlDto extends
    ↪ SequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

    //
    ↪ DADOS-BASICOS-DA-ORIENTACAO-EM-ANDAMENTO-DE-INICIACAO-CIENTIFIC
    //

```

```

        ↪ DETALHAMENTO-DA-ORIENTACAO-EM-ANDAMENTO-DE-INICIACAO-CIENTIFICA?,
        // PALAVRAS-CHAVE?,
        // AREAS-DO-CONHECIMENTO?,
        // SETORES-DE-ATIVIDADE?,
        // INFORMACOES-ADICIONAIS?

        @XmlElement(name =
            ↪ "DADOS-BASICOS-DA-ORIENTACAO-EM-ANDAMENTO-DE-INICIACAO-CIENTIFICA")
        private DadosBasicosDaOrientacaoEmAndamentoXmlDto dadosBasicos;

        @XmlElement(name =
            ↪ "DETALHAMENTO-DA-ORIENTACAO-EM-ANDAMENTO-DE-INICIACAO-CIENTIFICA")
        private DetalhamentoDaOrientacaoXmlDto detalhamento;
    }

    package ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.graduacao;

    import javax.xml.bind.annotation.XmlAccessType;
    import javax.xml.bind.annotation.XmlAccessorType;
    import javax.xml.bind.annotation.XmlElement;

    import lombok.Getter;
    import lombok.Setter;
    import lombok.ToString;
    import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoXmlDto;
    import ufsc.alpc.xml.dto.common.SequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
    import ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.DadosBasicosDaOrientacaoEmAndamentoXmlDto;

    ↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.DadosBasicosDaOrientacaoEmAndamentoXmlDto;

    @Getter
    @Setter
    @ToString(callSuper = true)
    @XmlAccessorType(XmlAccessType.FIELD)
    public class OrientacaoEmAndamentoDeGraduacaoXmlDto extends
        ↪ SequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

        // DADOS-BASICOS-DA-ORIENTACAO-EM-ANDAMENTO-DE-GRADUACAO?,
        // DETALHAMENTO-DA-ORIENTACAO-EM-ANDAMENTO-DE-GRADUACAO?,
        // PALAVRAS-CHAVE?,
        // AREAS-DO-CONHECIMENTO?,
        // SETORES-DE-ATIVIDADE?,
        // INFORMACOES-ADICIONAIS?

        @XmlElement(name =
            ↪ "DADOS-BASICOS-DA-ORIENTACAO-EM-ANDAMENTO-DE-GRADUACAO")
        private DadosBasicosDaOrientacaoEmAndamentoXmlDto dadosBasicos;

        @XmlElement(name =
            ↪ "DETALHAMENTO-DA-ORIENTACAO-EM-ANDAMENTO-DE-GRADUACAO")
        private DetalhamentoDaOrientacaoXmlDto detalhamento;
    }

    package ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.doutorado;

    import javax.xml.bind.annotation.XmlAccessType;
    import javax.xml.bind.annotation.XmlAccessorType;
    import javax.xml.bind.annotation.XmlElement;

    import lombok.Getter;
    import lombok.Setter;
    import lombok.ToString;
    import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoXmlDto;
    import ufsc.alpc.xml.dto.common.SequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
    import ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.DadosBasicosDaOrientacaoEmAndamentoXmlDto;

    ↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.DadosBasicosDaOrientacaoEmAndamentoXmlDto;

    @Getter
    @Setter
    @ToString(callSuper = true)
    @XmlAccessorType(XmlAccessType.FIELD)
    public class OrientacaoEmAndamentoDeDoutoradoXmlDto extends
        ↪ SequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

        // DADOS-BASICOS-DA-ORIENTACAO-EM-ANDAMENTO-DE-DOCTORADO?,
        // DETALHAMENTO-DA-ORIENTACAO-EM-ANDAMENTO-DE-DOCTORADO?,
        // PALAVRAS-CHAVE?,
        // AREAS-DO-CONHECIMENTO?,
        // SETORES-DE-ATIVIDADE?,
        // INFORMACOES-ADICIONAIS?

        @XmlElement(name =
            ↪ "DADOS-BASICOS-DA-ORIENTACAO-EM-ANDAMENTO-DE-DOCTORADO")
        private DadosBasicosDaOrientacaoEmAndamentoXmlDto dadosBasicos;
    }

```



```

import
    @XmlElement(name =
        ↪ "DETALHAMENTO-DA-ORIENTACAO-EM-ANDAMENTO-DE-DOCTORADO")
        private DetalhamentoDaOrientacaoXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

import org.apache.commons.collections.CollectionUtils;

import
    ↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.doutorado.OrientacaoEmAndamentoDeDoutorado;
import
    ↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.especializado.OrientacaoEmAndamentoDeEspecializacao;
import
    ↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.graduacao.OrientacaoEmAndamentoDeGraduacao;
import
    ↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.iniciacao.OrientacaoEmAndamentoDeIniciacao;
import
    ↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.mestrado.OrientacaoEmAndamentoDeMestrado;
import
    ↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.outros.OutrasOrientacoesEmAndamentoDeOutras;
import
    ↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.posdoutorado.OrientacaoEmAndamentoDePosDoutorado;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OrientacoesEmAndamentoXmlDto {

    // ORIENTACAO-EM-ANDAMENTO-DE-MESTRADO*,
    // ORIENTACAO-EM-ANDAMENTO-DE-DOCTORADO*,
    // ORIENTACAO-EM-ANDAMENTO-DE-POS-DOCTORADO*,
    //
    ↪ ORIENTACAO-EM-ANDAMENTO-DE-APERFEICOAMENTO-ESPECIALIZACAO*,
    // ORIENTACAO-EM-ANDAMENTO-DE-GRADUACAO*,
    // ORIENTACAO-EM-ANDAMENTO-DE-INICIACAO-CIENTIFICA*,
    // OUTRAS-ORIENTACOES-EM-ANDAMENTO*

    @XmlElement(name = "ORIENTACAO-EM-ANDAMENTO-DE-MESTRADO")
    private List<OrientacaoEmAndamentoDeMestradoXmlDto> orientacoesMestrado;

    @XmlElement(name = "ORIENTACAO-EM-ANDAMENTO-DE-DOCTORADO")
    private List<OrientacaoEmAndamentoDeDoutoradoXmlDto> orientacoesDoutorado;

    @XmlElement(name = "ORIENTACAO-EM-ANDAMENTO-DE-POS-DOCTORADO")
    private List<OrientacaoEmAndamentoDePosDoutoradoXmlDto> orientacoesPosDoutorado;

    @XmlElement(name =
        ↪ "ORIENTACAO-EM-ANDAMENTO-DE-APERFEICOAMENTO-ESPECIALIZACAO")
    private List<OrientacaoEmAndamentoDeAperfeiçoamentoEspecializadoXmlDto>
        ↪ orientacoesDeAperfeiçoamentoEspecializacao;

    @XmlElement(name = "ORIENTACAO-EM-ANDAMENTO-DE-GRADUACAO")
    private List<OrientacaoEmAndamentoDeGraduacaoXmlDto> orientacoesDeGraduacao;

    @XmlElement(name =
        ↪ "ORIENTACAO-EM-ANDAMENTO-DE-INICIACAO-CIENTIFICA")
    private List<OrientacaoEmAndamentoDeIniciacaoCientificaXmlDto>
        ↪ orientacoesDeIniciacaoCientifica;

    @XmlElement(name = "OUTRAS-ORIENTACOES-EM-ANDAMENTO")
    private List<OutrasOrientacoesEmAndamentoXmlDto> outrasOrientacoes;

    public boolean isSetOrientacoesMestrado() {
        return !CollectionUtils.isEmpty(this.orientacoesMestrado);
    }

    public boolean isSetOrientacoesDoutorado() {
        return !CollectionUtils.isEmpty(this.orientacoesDoutorado);
    }
}

```

```

}
package ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.outros;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoXmlDto;
import ufsc.alpc.xml.dto.common.SequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import
↳ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.DadosBasicosDaOrientacaoEmAndamentoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OutrasOrientacoesEmAndamentoXmlDto extends
↳ SequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

    // DADOS-BASICOS-DE-OUTRAS-ORIENTACOES-EM-ANDAMENTO?,
    // DETALHAMENTO-DE-OUTRAS-ORIENTACOES-EM-ANDAMENTO?,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
↳ "DADOS-BASICOS-DE-OUTRAS-ORIENTACOES-EM-ANDAMENTO")
    private DadosBasicosDaOrientacaoEmAndamentoXmlDto dadosBasicos;

    @XmlElement(name =
↳ "DETALHAMENTO-DE-OUTRAS-ORIENTACOES-EM-ANDAMENTO")
    private DetalhamentoDaOrientacaoXmlDto detalhamento;
}

package ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.especializado;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoXmlDto;
import ufsc.alpc.xml.dto.common.SequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import
↳ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.DadosBasicosDaOrientacaoEmAndamentoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OrientacaoEmAndamentoDeAperfeicoamentoEspecializadoXmlDto extends
↳ SequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

    //
↳ DADOS-BASICOS-DA-ORIENTACAO-EM-ANDAMENTO-DE-APERFEICOAMENTO-ESPECIAL
    //
↳ DETALHAMENTO-DA-ORIENTACAO-EM-ANDAMENTO-DE-APERFEICOAMENTO-ESPECIAL
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
↳ "DADOS-BASICOS-DA-ORIENTACAO-EM-ANDAMENTO-DE-APERFEICOAMENTO-ESPECIAL")
    private DadosBasicosDaOrientacaoEmAndamentoXmlDto dadosBasicos;

    @XmlElement(name =
↳ "DETALHAMENTO-DA-ORIENTACAO-EM-ANDAMENTO-DE-APERFEICOAMENTO-ESPECIAL")
    private DetalhamentoDaOrientacaoXmlDto detalhamento;
}

package ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.posdoutorado;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

```

```

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoXmlDto;
import ufsc.alpc.xml.dto.common.SequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.DadosBasicosDaOrientacaoEmAndamento;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OrientacaoEmAndamentoDePosDoutoradoXmlDto extends SequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

    // DADOS-BASICOS-DA-ORIENTACAO-EM-ANDAMENTO-DE-POS-DOCTORADO?,
    // DETALHAMENTO-DA-ORIENTACAO-EM-ANDAMENTO-DE-POS-DOCTORADO?,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DA-ORIENTACAO-EM-ANDAMENTO-DE-POS-DOCTORADO")
    private DadosBasicosDaOrientacaoEmAndamentoXmlDto dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DA-ORIENTACAO-EM-ANDAMENTO-DE-POS-DOCTORADO")
    private DetalhamentoDaOrientacaoXmlDto detalhamento;
}

package ufsc.alpc.xml.dto.dados.complementares.participacao.eventos.congressos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmCongressoXmlDto extends EventosCongressosXmlDto {

    // DADOS-BASICOS-DA-PARTICIPACAO-EM-CONGRESSO?,
    // DETALHAMENTO-DA-PARTICIPACAO-EM-CONGRESSO?,
    // PARTICIPANTE-DE-EVENTOS-CONGRESSOS*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DA-PARTICIPACAO-EM-CONGRESSO")
    private DadosBasicosDaParticipacaoEmEventosCongressosXmlDto dados;

    @XmlElement(name = "DETALHAMENTO-DA-PARTICIPACAO-EM-CONGRESSO")
    private DetalhamentoDaParticipacaoEmEventosCongressosXmlDto detalhamento;
}

package ufsc.alpc.xml.dto.dados.complementares.participacao.eventos.congressos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmFeiraXmlDto extends EventosCongressosXmlDto {

    // DADOS-BASICOS-DA-PARTICIPACAO-EM-FEIRA?,
    // DETALHAMENTO-DA-PARTICIPACAO-EM-FEIRA?,

```

```

// PARTICIPANTE-DE-EVENTOS-CONGRESSOS*,
// PALAVRAS-CHAVE?,
// AREAS-DO-CONHECIMENTO?,
// SETORES-DE-ATIVIDADE?,
// INFORMACOES-ADICIONAIS?

@XmlElement(name = "DADOS-BASICOS-DA-PARTICIPACAO-EM-FEIRA")
private DadosBasicosDaParticipacaoEmEventosCongressosXmlDto dados;

@XmlElement(name = "DETALHAMENTO-DA-PARTICIPACAO-EM-FEIRA")
private DetalhamentoDaParticipacaoEmEventosCongressosXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.eventos.congressos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoDaParticipacaoEmEventosCongressosXmlDto {
    // NOME-DO-EVENTO CDATA #IMPLIED
    // CODIGO-INSTITUICAO CDATA #IMPLIED
    // NOME-INSTITUICAO CDATA #IMPLIED
    // LOCAL-DO-EVENTO CDATA #IMPLIED
    // CIDADE-DO-EVENTO CDATA #IMPLIED
    // NOME-DO-EVENTO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "NOME-DO-EVENTO")
    protected String nomeDoEvento;

    @XmlAttribute(name = "CODIGO-INSTITUICAO")
    protected String codigoInstituicao;

    @XmlAttribute(name = "NOME-INSTITUICAO")
    protected String nomeInstituicao;

    @XmlAttribute(name = "LOCAL-DO-EVENTO")
    protected String localDoEvento;

    @XmlAttribute(name = "CIDADE-DO-EVENTO")
    protected String cidadeDoEvento;

    @XmlAttribute(name = "NOME-DO-EVENTO-INGLES")
    protected String nomeDoEventoIngles;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.eventos.congressos;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.SequenciaPalavraAreaSetorInfoAgrupadorXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class EventosCongressosXmlDto extends SequenciaPalavraAreaSetorInfoAgrupadorXmlDto {
    @XmlElement(name = "PARTICIPANTE-DE-EVENTOS-CONGRESSOS")
    private List<ParticipanteDeEventosCongressosXmlDto> participantesDeEventosCongressos;
}package ufsc.alpc.xml.dto.dados.complementares.participacao.eventos.congressos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

```

```

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmOlimpiadaXmlDto extends EventosCongressosXmlDto {
    // DADOS-BASICOS-DA-PARTICIPACAO-EM-OLIMPIADA?,
    // DETALHAMENTO-DA-PARTICIPACAO-EM-OLIMPIADA?,
    // PARTICIPANTE-DE-EVENTOS-CONGRESSOS*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DA-PARTICIPACAO-EM-OLIMPIADA")
    private DadosBasicosDaParticipacaoEmEventosCongressosXmlDto dados;

    @XmlElement(name = "DETALHAMENTO-DA-PARTICIPACAO-EM-OLIMPIADA")
    private DetalhamentoDaParticipacaoEmEventosCongressosXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.eventos.congressos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipanteDeEventosCongressosXmlDto {
    // NOME-COMPLETO-DO-PARTICIPANTE-DE-EVENTOS-CONGRESSOS CDATA
    // ↪ #IMPLIED
    // NOME-PARA-CITACAO-DO-PARTICIPANTE-DE-EVENTOS-CONGRESSOS
    // ↪ CDATA #IMPLIED
    // ORDEM-PARTICIPANTE CDATA #IMPLIED
    // CPF-DO-PARTICIPANTE-PARTICIPANTE-DE-EVENTOS-CONGRESSOS
    // ↪ CDATA #IMPLIED
    // NRO-ID-CNPQ CDATA #IMPLIED

    @XmlAttribute(name =
        ↪ "NOME-COMPLETO-DO-PARTICIPANTE-DE-EVENTOS-CONGRESSOS")
    private String nomeCompletoDoParticipanteDeEventosCongressos;

    @XmlAttribute(name =
        ↪ "NOME-PARA-CITACAO-DO-PARTICIPANTE-DE-EVENTOS-CONGRESSOS")
    private String nomeParaCitacaoDoParticipanteDeEventosCongressos;

    @XmlAttribute(name = "ORDEM-PARTICIPANTE")
    private String ordemParticipante;

    @XmlAttribute(name =
        ↪ "CPF-DO-PARTICIPANTE-PARTICIPANTE-DE-EVENTOS-CONGRESSOS")
    private String cpfDoParticipanteDeEventosCongressos;

    @XmlAttribute(name = "NRO-ID-CNPQ")
    private String nroIdCnpq;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.eventos.congressos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmSeminarioXmlDto extends EventosCongressosXmlDto {
    // DADOS-BASICOS-DA-PARTICIPACAO-EM-SEMINARIO?,

```

```

// DETALHAMENTO-DA-PARTICIPACAO-EM-SEMINARIO?,
// PARTICIPANTE-DE-EVENTOS-CONGRESSOS*,
// PALAVRAS-CHAVE?,
// AREAS-DO-CONHECIMENTO?,
// SETORES-DE-ATIVIDADE?,
// INFORMACOES-ADICIONAIS?

@XmlElement(name = "DADOS-BASICOS-DA-PARTICIPACAO-EM-SEMINARIO")
private DadosBasicosDaParticipacaoEmEventosCongressosXmlDto dados;

@XmlElement(name = "DETALHAMENTO-DA-PARTICIPACAO-EM-SEMINARIO")
private DetalhamentoDaParticipacaoEmEventosCongressosXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.eventos.congressos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.common.DadosBasicosNaturezaIdiomaHomePageXmlDto;
import ufsc.alpc.xml.dto.domain.MeiodeDivulgacao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDaParticipacaoEmEventosCongressosXmlDto extends
↳ DadosBasicosNaturezaIdiomaHomePageXmlDto {
// NATUREZA CDATA #IMPLIED
// TITULO CDATA #IMPLIED
// ANO CDATA #IMPLIED
// PAIS CDATA #IMPLIED
// IDIOMA CDATA #IMPLIED
// MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
↳ MEIO_DIGITAL |
// FILME | HIPERTEXTO
// | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
// HOME-PAGE-DO-TRABALHO CDATA #IMPLIED
// FLAG-RELEVANCIA (SIM | NAO) "NAO"
// TIPO-PARTICIPACAO CDATA #IMPLIED
// FORMA-PARTICIPACAO CDATA #IMPLIED
// DOI CDATA #IMPLIED
// TITULO-INGLES CDATA #IMPLIED
// FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"

@XmlAttribute(name = "TITULO")
protected String titulo;

@XmlAttribute(name = "MEIO-DE-DIVULGACAO")
protected MeiodeDivulgacao meiodeDivulgacao;

@XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
protected String homePageDoTrabalho;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-RELEVANCIA")
protected Boolean flagRelevancia;

@XmlAttribute(name = "TIPO-PARTICIPACAO")
protected String tipoParticipacao;

@XmlAttribute(name = "FORMA-PARTICIPACAO")
protected String formaParticipacao;

@XmlAttribute(name = "TITULO-INGLES")
protected String tituloIngles;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-DIVULGACAO-CIENTIFICA")
protected Boolean flagDivulgacaoCientifica;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.eventos.congressos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

```

```

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmOficinaXmlDto extends EventosCongressosXmlDto {

    // DADOS-BASICOS-DA-PARTICIPACAO-EM-OFICINA?,
    // DETALHAMENTO-DA-PARTICIPACAO-EM-OFICINA?,
    // PARTICIPANTE-DE-EVENTOS-CONGRESSOS*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DA-PARTICIPACAO-EM-OFICINA")
    private DadosBasicosDaParticipacaoEmEventosCongressosXmlDto dados;

    @XmlElement(name = "DETALHAMENTO-DA-PARTICIPACAO-EM-OFICINA")
    private DetalhamentoDaParticipacaoEmEventosCongressosXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.eventos.congressos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OutrasParticipacoesEmEventosCongressosXmlDto extends EventosCongressosXmlDto {

    // ↪ DADOS-BASICOS-DE-OUTRAS-PARTICIPACOES-EM-EVENTOS-CONGRESSOS?,
    // ↪ DETALHAMENTO-DE-OUTRAS-PARTICIPACOES-EM-EVENTOS-CONGRESSOS?,
    // PARTICIPANTE-DE-EVENTOS-CONGRESSOS*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↪ "DADOS-BASICOS-DE-OUTRAS-PARTICIPACOES-EM-EVENTOS-CONGRESSOS")
    private DadosBasicosDaParticipacaoEmEventosCongressosXmlDto dados;

    @XmlElement(name =
        ↪ "DETALHAMENTO-DE-OUTRAS-PARTICIPACOES-EM-EVENTOS-CONGRESSOS")
    private DetalhamentoDaParticipacaoEmEventosCongressosXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.eventos.congressos;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmEventosCongressosXmlDto {

    // PARTICIPACAO-EM-CONGRESSO*,
    // PARTICIPACAO-EM-FEIRA*,
    // PARTICIPACAO-EM-SEMINARIO*,
    // PARTICIPACAO-EM-SIMPOSIO*,
    // PARTICIPACAO-EM-OFICINA*,
    // PARTICIPACAO-EM-ENCONTRO*,

```

```

// PARTICIPACAO-EM-EXPOSICAO*,
// PARTICIPACAO-EM-OLIMPIADA*,
// OUTRAS-PARTICIPACOES-EM-EVENTOS-CONGRESSOS*

@XmlElement(name = "PARTICIPACAO-EM-CONGRESSO")
private List<ParticipacaoEmCongressoXmlDto> congressos;

@XmlElement(name = "PARTICIPACAO-EM-FEIRA")
private List<ParticipacaoEmFeiraXmlDto> feiras;

@XmlElement(name = "PARTICIPACAO-EM-SEMINARIO")
private List<ParticipacaoEmSeminarioXmlDto> seminarios;

@XmlElement(name = "PARTICIPACAO-EM-SIMPOSIO")
private List<ParticipacaoEmSimposioXmlDto> simposios;

@XmlElement(name = "PARTICIPACAO-EM-OFICINA")
private List<ParticipacaoEmOficinaXmlDto> oficinas;

@XmlElement(name = "PARTICIPACAO-EM-ENCONTRO")
private List<ParticipacaoEmEncontroXmlDto> encontros;

@XmlElement(name = "PARTICIPACAO-EM-EXPOSICAO")
private List<ParticipacaoEmExposicaoXmlDto> exposicoes;

@XmlElement(name = "PARTICIPACAO-EM-OLIMPIADA")
private List<ParticipacaoEmOlimpiadaXmlDto> olimpíadas;

@XmlElement(name = "OUTRAS-PARTICIPACOES-EM-EVENTOS-CONGRESSOS")
private List<OutrasParticipacoesEmEventosCongressosXmlDto> outras;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.eventos.congressos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmEncontroXmlDto extends EventosCongressosXmlDto {

    // DADOS-BASICOS-DA-PARTICIPACAO-EM-ENCONTRO?,
    // DETALHAMENTO-DA-PARTICIPACAO-EM-ENCONTRO?,
    // PARTICIPANTE-DE-EVENTOS-CONGRESSOS*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DA-PARTICIPACAO-EM-ENCONTRO")
    private DadosBasicosDaParticipacaoEmEventosCongressosXmlDto dados;

    @XmlElement(name = "DETALHAMENTO-DA-PARTICIPACAO-EM-ENCONTRO")
    private DetalhamentoDaParticipacaoEmEventosCongressosXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.eventos.congressos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmSimposioXmlDto extends EventosCongressosXmlDto {

    // DADOS-BASICOS-DA-PARTICIPACAO-EM-SIMPOSIO?,
    // DETALHAMENTO-DA-PARTICIPACAO-EM-SIMPOSIO?,
    // PARTICIPANTE-DE-EVENTOS-CONGRESSOS*,
    // PALAVRAS-CHAVE?,

```



```

// AREAS-DO-CONHECIMENTO?,
// SETORES-DE-ATIVIDADE?,
// INFORMACOES-ADICIONAIS?

@XmlElement(name = "DADOS-BASICOS-DA-PARTICIPACAO-EM-SIMPOSIO")
private DadosBasicosDaParticipacaoEmEventosCongressosXmlDto dados;

@XmlElement(name = "DETALHAMENTO-DA-PARTICIPACAO-EM-SIMPOSIO")
private DetalhamentoDaParticipacaoEmEventosCongressosXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.eventos.congressos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmExposicaoXmlDto extends EventosCongressosXmlDto {

// DADOS-BASICOS-DA-PARTICIPACAO-EM-EXPOSICAO?,
// DETALHAMENTO-DA-PARTICIPACAO-EM-EXPOSICAO?,
// PARTICIPANTE-DE-EVENTOS-CONGRESSOS*,
// PALAVRAS-CHAVE?,
// AREAS-DO-CONHECIMENTO?,
// SETORES-DE-ATIVIDADE?,
// INFORMACOES-ADICIONAIS?

@XmlElement(name = "DADOS-BASICOS-DA-PARTICIPACAO-EM-EXPOSICAO")
private DadosBasicosDaParticipacaoEmEventosCongressosXmlDto dados;

@XmlElement(name = "DETALHAMENTO-DA-PARTICIPACAO-EM-EXPOSICAO")
private DetalhamentoDaParticipacaoEmEventosCongressosXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.SequenciaPalavraAreaSetorInfoAgrupadorXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmBancaXmlDto extends SequenciaPalavraAreaSetorInfoAgrupadorXmlDto
↪ {

@XmlElement(name = "PARTICIPANTE-BANCA")
private List<ParticipanteBancaXmlDto> participantesBanca;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao;

import javax.xml.bind.annotation.XmlAttribute;

public class ParticipanteBancaXmlDto {

// NOME-COMPLETO-DO-PARTICIPANTE-DA-BANCA CDATA #IMPLIED
// NOME-PARA-CITACAO-DO-PARTICIPANTE-DA-BANCA CDATA #IMPLIED
// ORDEM-PARTICIPANTE CDATA #IMPLIED
// CPF-DO-PARTICIPANTE-DA-BANCA CDATA #IMPLIED
// NRO-ID-CNPQ CDATA #IMPLIED

@XmlAttribute(name = "NOME-COMPLETO-DO-PARTICIPANTE-DA-BANCA")
protected String nomeCompletoDoParticipanteDaBanca;

@XmlAttribute(name =
↪ "NOME-PARA-CITACAO-DO-PARTICIPANTE-DA-BANCA")
protected String nomeParaCitacaoDoParticipanteDaBanca;

@XmlAttribute(name = "ORDEM-PARTICIPANTE")

```

```

    protected String ordemParticipante;

    @XmlAttribute(name = "CPF-DO-PARTICIPANTE-DA-BANCA")
    protected String cpfDoParticipanteDaBanca;

    @XmlAttribute(name = "NRO-ID-CNPQ")
    protected String nroIdCnpq;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.trabalhos.conclusao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoTipoXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.participacao.ParticipacaoEmBancaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OutrasParticipacoesEmBancaXmlDto extends ParticipacaoEmBancaXmlDto {

    // DADOS-BASICOS-DE-OUTRAS-PARTICIPACOES-EM-BANCA?,
    // DETALHAMENTO-DE-OUTRAS-PARTICIPACOES-EM-BANCA?,
    // PARTICIPANTE-BANCA*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↪ "DADOS-BASICOS-DA-PARTICIPACAO-EM-BANCA-DE-APERFEICOAMENTO-ESPECIALIZADA")
    private DadosBasicosDaParticipacaoTipoXmlDto dadosBasicos;

    @XmlElement(name =
        ↪ "DETALHAMENTO-DA-PARTICIPACAO-EM-BANCA-DE-APERFEICOAMENTO-ESPECIALIZADA")
    private DetalhamentoDaParticipacaoEmBancaXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.trabalhos.conclusao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.participacao.ParticipacaoEmBancaXmlDto;
import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmBancaDeGraduacaoXmlDto extends ParticipacaoEmBancaXmlDto {

    // DADOS-BASICOS-DA-PARTICIPACAO-EM-BANCA-DE-GRADUACAO?,
    // DETALHAMENTO-DA-PARTICIPACAO-EM-BANCA-DE-GRADUACAO?,
    // PARTICIPANTE-BANCA*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↪ "DADOS-BASICOS-DA-PARTICIPACAO-EM-BANCA-DE-GRADUACAO")
    private DadosBasicosDaParticipacaoXmlDto dadosBasicos;

    @XmlElement(name =
        ↪ "DETALHAMENTO-DA-PARTICIPACAO-EM-BANCA-DE-GRADUACAO")
    private DetalhamentoDaParticipacaoEmBancaXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.trabalhos.conclusao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

```

```

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufs.alpc.xml.dto.dados.complementares.DetalhamentoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoDaParticipacaoEmBancaXmlDto extends DetalhamentoXmlDto {
    // NOME-DO-CANDIDATO CDATA #IMPLIED
    // CODIGO-INSTITUICAO CDATA #IMPLIED
    // NOME-INSTITUICAO CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-CURSO CDATA #IMPLIED
    // NOME-CURSO CDATA #IMPLIED
    // NOME-CURSO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "NOME-DO-CANDIDATO")
    protected String nomeDoCandidato;
}

package ufs.alpc.xml.dto.dados.complementares.participacao.trabalhos.conclusao;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlRootElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlRootElement(name = "PARTICIPACAO-EM-BANCA-TRABALHOS-CONCLUSAO")
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmBancaTrabalhosConclusaoXmlDto {
    // PARTICIPACAO-EM-BANCA-DE-MESTRADO*,
    // PARTICIPACAO-EM-BANCA-DE-DOCTORADO*,
    // PARTICIPACAO-EM-BANCA-DE-EXAME-QUALIFICACAO*,
    // PARTICIPACAO-EM-BANCA-DE-APERFEICOAMENTO-ESPECIALIZACAO*,
    // PARTICIPACAO-EM-BANCA-DE-GRADUACAO*,
    // OUTRAS-PARTICIPACOES-EM-BANCA*

    @XmlElement(name = "PARTICIPACAO-EM-BANCA-DE-MESTRADO")
    private List<ParticipacaoEmBancaDeMestradoXmlDto> mestrado;

    @XmlElement(name = "PARTICIPACAO-EM-BANCA-DE-DOCTORADO")
    private List<ParticipacaoEmBancaDeDoutoradoXmlDto> doutorado;

    @XmlElement(name = "PARTICIPACAO-EM-BANCA-DE-EXAME-QUALIFICACAO")
    private List<ParticipacaoEmBancaDeExameQualificacaoXmlDto> exameQualificacao;

    @XmlElement(name =
        ↵ "PARTICIPACAO-EM-BANCA-DE-APERFEICOAMENTO-ESPECIALIZACAO")
    private List<ParticipacaoEmBancaDeAperfeicoamentoEspecializacaoXmlDto>
        ↵ aperfeicoamentoEspecializacao;

    @XmlElement(name = "PARTICIPACAO-EM-BANCA-DE-GRADUACAO")
    private List<ParticipacaoEmBancaDeGraduacaoXmlDto> graduacao;

    @XmlElement(name = "OUTRAS-PARTICIPACOES-EM-BANCA")
    private List<OutrasParticipacoesEmBancaXmlDto> outrasParticipacoes;
}

package ufs.alpc.xml.dto.dados.complementares.participacao.trabalhos.conclusao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufs.alpc.xml.dto.common.DadosBasicosDaParticipacaoXmlDto;
import ufs.alpc.xml.dto.dados.complementares.participacao.ParticipacaoEmBancaXmlDto;

```

```

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmBancaDeDoutoradoXmlDto extends ParticipacaoEmBancaXmlDto {
    // DADOS-BASICOS-DA-PARTICIPACAO-EM-BANCA-DE-DOCTORADO?,
    // DETALHAMENTO-DA-PARTICIPACAO-EM-BANCA-DE-DOCTORADO?,
    // PARTICIPANTE-BANCA*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↪ "DADOS-BASICOS-DA-PARTICIPACAO-EM-BANCA-DE-DOCTORADO")
    private DadosBasicosDaParticipacaoXmlDto dadosBasicos;

    @XmlElement(name =
        ↪ "DETALHAMENTO-DA-PARTICIPACAO-EM-BANCA-DE-DOCTORADO")
    private DetalhamentoDaParticipacaoEmBancaXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.trabalhos.conclusao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.participacao.ParticipacaoEmBancaXmlDto;
import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmBancaDeAperfeiçoamentoEspecializacaoXmlDto extends
    ↪ ParticipacaoEmBancaXmlDto {
    //
    // ↪ DADOS-BASICOS-DA-PARTICIPACAO-EM-BANCA-DE-APERFEICAMENTO-ESPECIALIZACAO?,
    // ↪ DETALHAMENTO-DA-PARTICIPACAO-EM-BANCA-DE-APERFEICAMENTO-ESPECIALIZACAO?,
    // PARTICIPANTE-BANCA*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↪ "DADOS-BASICOS-DA-PARTICIPACAO-EM-BANCA-DE-APERFEICAMENTO-ESPECIALIZACAO")
    private DadosBasicosDaParticipacaoXmlDto dadosBasicos;

    @XmlElement(name =
        ↪ "DETALHAMENTO-DA-PARTICIPACAO-EM-BANCA-DE-APERFEICAMENTO-ESPECIALIZACAO")
    private DetalhamentoDaParticipacaoEmBancaXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.trabalhos.conclusao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoTipoXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.participacao.ParticipacaoEmBancaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmBancaDeMestradoXmlDto extends ParticipacaoEmBancaXmlDto {
    // DADOS-BASICOS-DA-PARTICIPACAO-EM-BANCA-DE-MESTRADO?,
    // DETALHAMENTO-DA-PARTICIPACAO-EM-BANCA-DE-MESTRADO?,
    // PARTICIPANTE-BANCA*,
    // PALAVRAS-CHAVE?,

```

```

// AREAS-DO-CONHECIMENTO?,
// SETORES-DE-ATIVIDADE?,
// INFORMACOES-ADICIONAIS?

@XmlElement(name =
    ↪ "DADOS-BASICOS-DA-PARTICIPACAO-EM-BANCA-DE-MESTRADO")
private DadosBasicosDaParticipacaoTipoXmlDto dadosBasicos;

@XmlElement(name =
    ↪ "DETALHAMENTO-DA-PARTICIPACAO-EM-BANCA-DE-MESTRADO")
private DetalhamentoDaParticipacaoEmBancaXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.trabalhos.conclusao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.participacao.ParticipacaoEmBancaXmlDto;
import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmBancaDeExameQualificacaoXmlDto extends
    ↪ ParticipacaoEmBancaXmlDto {

    //
    ↪ DADOS-BASICOS-DA-PARTICIPACAO-EM-BANCA-DE-EXAME-QUALIFICACAO?,
    //
    ↪ DETALHAMENTO-DA-PARTICIPACAO-EM-BANCA-DE-EXAME-QUALIFICACAO?,
    // PARTICIPANTE-BANCA*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↪ "DADOS-BASICOS-DA-PARTICIPACAO-EM-BANCA-DE-EXAME-QUALIFICACAO")
    private DadosBasicosDaParticipacaoXmlDto dadosBasicos;

    @XmlElement(name =
        ↪ "DETALHAMENTO-DA-PARTICIPACAO-EM-BANCA-DE-EXAME-QUALIFICACAO")
    private DetalhamentoDaParticipacaoEmBancaXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.banca.julgadora;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.participacao.ParticipacaoEmBancaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class BancaJulgadoraParaProfessorTitularXmlDto extends ParticipacaoEmBancaXmlDto {

    // DADOS-BASICOS-DA-BANCA-JULGADORA-PARA-PROFESSOR-TITULAR?,
    // DETALHAMENTO-DA-BANCA-JULGADORA-PARA-PROFESSOR-TITULAR?,
    // PARTICIPANTE-BANCA*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↪ "DADOS-BASICOS-DA-BANCA-JULGADORA-PARA-PROFESSOR-TITULAR")
    private DadosBasicosDaParticipacaoXmlDto dadosBasicos;

    @XmlElement(name =
        ↪ "DETALHAMENTO-DA-BANCA-JULGADORA-PARA-PROFESSOR-TITULAR")
    private DetalhamentoDaBancaJulgadoraXmlDto detalhamento;
}

```

```

}
package ufsc.alpc.xml.dto.dados.complementares.participacao.banca.julgadora;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoDaBancaJulgadoraXmlDto {
    // CODIGO-INSTITUICAO CDATA #IMPLIED
    // NOME-INSTITUICAO CDATA #IMPLIED

    @XmlAttribute(name = "CODIGO-INSTITUICAO")
    protected String codigoInstituicao;

    @XmlAttribute(name = "NOME-INSTITUICAO")
    protected String nomeInstituicao;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.banca.julgadora;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.participacao.ParticipacaoEmBancaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class BancaJulgadoraParaLivreDocenciaXmlDto extends ParticipacaoEmBancaXmlDto {

    // DADOS-BASICOS-DA-BANCA-JULGADORA-PARA-LIVRE-DOCENCIA?,
    // DETALHAMENTO-DA-BANCA-JULGADORA-PARA-LIVRE-DOCENCIA?,
    // PARTICIPANTE-BANCA*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↪ "DADOS-BASICOS-DA-BANCA-JULGADORA-PARA-LIVRE-DOCENCIA")
    private DadosBasicosDaParticipacaoXmlDto dadosBasicos;

    @XmlElement(name =
        ↪ "DETALHAMENTO-DA-BANCA-JULGADORA-PARA-LIVRE-DOCENCIA")
    private DetalhamentoDaBancaJulgadoraXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.banca.julgadora;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoTipoInglesXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.participacao.ParticipacaoEmBancaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OutrasBancasJulgadorasXmlDto extends ParticipacaoEmBancaXmlDto {

    // DADOS-BASICOS-DE-OUTRAS-BANCAS-JULGADORAS?,
    // DETALHAMENTO-DE-OUTRAS-BANCAS-JULGADORAS?,
    // PARTICIPANTE-BANCA*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,

```

```

// SETORES-DE-ATIVIDADE?,
// INFORMACOES-ADICIONAIS?

@XmlElement(name = "DADOS-BASICOS-DE-OUTRAS-BANCAS-JULGADORAS")
private DadosBasicosDaParticipacaoTipoInglesXmlDto dadosBasicos;

@XmlElement(name = "DETALHAMENTO-DE-OUTRAS-BANCAS-JULGADORAS")
private DetalhamentoDaBancaJulgadoraXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.banca.julgadora;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ParticipacaoEmBancaJulgadoraXmlDto {

    // BANCA-JULGADORA-PARA-PROFESSOR-TITULAR*,
    // BANCA-JULGADORA-PARA-CONCURSO-PUBLICO*,
    // BANCA-JULGADORA-PARA-LIVRE-DOCENCIA*,
    // BANCA-JULGADORA-PARA-AVALIACAO-CURSOS*,
    // OUTRAS-BANCAS-JULGADORAS*

    @XmlElement(name = "BANCA-JULGADORA-PARA-PROFESSOR-TITULAR")
    private List<BancaJulgadoraParaProfessorTitularXmlDto> professoresTitulares;

    @XmlElement(name = "BANCA-JULGADORA-PARA-CONCURSO-PUBLICO")
    private List<BancaJulgadoraParaConcursoPublicoXmlDto> concursosPublicos;

    @XmlElement(name = "BANCA-JULGADORA-PARA-LIVRE-DOCENCIA")
    private List<BancaJulgadoraParaLivreDocenciaXmlDto> livreDocencia;

    @XmlElement(name = "BANCA-JULGADORA-PARA-AVALIACAO-CURSOS")
    private List<BancaJulgadoraParaAvaliacaoCursosXmlDto> avaliacaoCurso;

    @XmlElement(name = "OUTRAS-BANCAS-JULGADORAS")
    private List<OutrasBancasJulgadorasXmlDto> outras;
}
package ufsc.alpc.xml.dto.dados.complementares.participacao.banca.julgadora;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.participacao.ParticipacaoEmBancaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class BancaJulgadoraParaAvaliacaoCursosXmlDto extends ParticipacaoEmBancaXmlDto {

    // DADOS-BASICOS-DA-BANCA-JULGADORA-PARA-AVALIACAO-CURSOS?,
    // DETALHAMENTO-DA-BANCA-JULGADORA-PARA-AVALIACAO-CURSOS?,
    // PARTICIPANTE-BANCA*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↪ "DADOS-BASICOS-DA-BANCA-JULGADORA-PARA-AVALIACAO-CURSOS")
    private DadosBasicosDaParticipacaoXmlDto dadosBasicos;

    @XmlElement(name =
        ↪ "DETALHAMENTO-DA-BANCA-JULGADORA-PARA-AVALIACAO-CURSOS")
    private DetalhamentoDaBancaJulgadoraXmlDto detalhamento;
}

```

```

}
package ufsc.alpc.xml.dto.dados.complementares.participacao.banca.julgadora;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.participacao.ParticipacaoEmBancaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class BancaJulgadoraParaConcursoPublicoXmlDto extends ParticipacaoEmBancaXmlDto {
    // DADOS-BASICOS-DA-BANCA-JULGADORA-PARA-CONCURSO-PUBLICO?,
    // DETALHAMENTO-DA-BANCA-JULGADORA-PARA-CONCURSO-PUBLICO?,
    // PARTICIPANTE-BANCA*,
    // PALAVRAS-CHAVE?,
    // AREAS-DO-CONHECIMENTO?,
    // SETORES-DE-ATIVIDADE?,
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↪ "DADOS-BASICOS-DA-BANCA-JULGADORA-PARA-CONCURSO-PUBLICO")
    private DadosBasicosDaParticipacaoXmlDto dadosBasicos;

    @XmlElement(name =
        ↪ "DETALHAMENTO-DA-BANCA-JULGADORA-PARA-CONCURSO-PUBLICO")
    private DetalhamentoDaBancaJulgadoraXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.dados.complementares;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.dados.complementares.informacoes.InformacoesAdicionaisCursosXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.informacoes.InformacoesAdicionaisInstituicoesXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.OrientacoesEmAndamentoXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.participacao.banca.julgadora.ParticipacaoEmBancaJulgadoraXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.participacao.eventos.congressos.ParticipacaoEmEventosCongressosXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.participacao.trabalhos.conclusao.ParticipacaoEmBancaTrabalhosConclusao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosComplementaresXmlDto {
    // FORMACAO-COMPLEMENTAR*,
    // PARTICIPACAO-EM-BANCA-TRABALHOS-CONCLUSAO?,
    // PARTICIPACAO-EM-BANCA-JULGADORA?,
    // PARTICIPACAO-EM-EVENTOS-CONGRESSOS?,
    // ORIENTACOES-EM-ANDAMENTO?,
    // INFORMACOES-ADICIONAIS-INSTITUICOES?,
    // INFORMACOES-ADICIONAIS-CURSOS?

    @XmlElement(name = "PARTICIPACAO-EM-BANCA-TRABALHOS-CONCLUSAO")
    private ParticipacaoEmBancaTrabalhosConclusaoXmlDto partBancaTrabalhosConclusao;

    @XmlElement(name = "PARTICIPACAO-EM-BANCA-JULGADORA")
    private ParticipacaoEmBancaJulgadoraXmlDto partBancaJulgadora;

    @XmlElement(name = "PARTICIPACAO-EM-EVENTOS-CONGRESSOS")
    private ParticipacaoEmEventosCongressosXmlDto partEmEventosCongresso;

    @XmlElement(name = "ORIENTACOES-EM-ANDAMENTO")
    private OrientacoesEmAndamentoXmlDto orientacoesEmAndamento;
}

```



```

@XmlElement(name = "INFORMACOES-ADICIONAIS-INSTITUICOES")
private InformacoesAdicionaisInstituicoesXmlDto infoAdicionaisInstituicoes;

@XmlElement(name = "INFORMACOES-ADICIONAIS-CURSOS")
private InformacoesAdicionaisCursosXmlDto infoAdicionaisCursos;

public boolean isSetOrientacoesEmAndamento() {
    return this.orientacoesEmAndamento != null;
}

public boolean isSetInfoAdicionaisInstituicoes() {
    return this.infoAdicionaisInstituicoes != null;
}

public boolean isSetInfoAdicionaisCursos() {
    return this.infoAdicionaisCursos != null;
}
}
package ufsc.alpc.xml.dto.dados.complementares;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoXmlDto {

    // CODIGO-INSTITUICAO CDATA #IMPLIED
    // NOME-INSTITUICAO CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-CURSO CDATA #IMPLIED
    // NOME-CURSO CDATA #IMPLIED
    // NOME-CURSO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "CODIGO-INSTITUICAO")
    protected String codigoInstituicao;

    @XmlAttribute(name = "NOME-INSTITUICAO")
    protected String nomeInstituicao;

    @XmlAttribute(name = "CODIGO-ORGAO")
    protected String codigoOrgao;

    @XmlAttribute(name = "NOME-ORGAO")
    protected String nomeOrgao;

    @XmlAttribute(name = "CODIGO-CURSO")
    protected String codigoCurso;

    @XmlAttribute(name = "NOME-CURSO")
    protected String nomeCurso;

    @XmlAttribute(name = "NOME-CURSO-INGLES")
    protected String nomeCursoIngles;
}
package ufsc.alpc.xml.dto.dados.complementares.informacoes;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.domain.GrandeAreaDoConhecimento;
import ufsc.alpc.xml.dto.domain.NivelCurso;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class InformacaoAdicionalCursoXmlDto {

    // CODIGO-CURSO CDATA #IMPLIED

```

```

// CODIGO-ORGAO CDATA #IMPLIED
// NOME-ORGAO CDATA #IMPLIED
// CODIGO-INSTITUICAO CDATA #IMPLIED
// NOME-INSTITUICAO CDATA #IMPLIED
// NOME-GRANDE-AREA-DO-CONHECIMENTO (OUTROS |
//   ↳ LINGUISTICA LETRAS E ARTES |
// CIENCIAS_HUMANAS | CIENCIAS_SOCIAIS APLICADAS | CIENCIAS_AGRARIAS |
// CIENCIAS_DA_SAUDE | ENGENHARIAS | CIENCIAS_BIOLÓGICAS |
// CIENCIAS_EXATAS E DA TERRA | NAO_INFORMADO)"NAO_INFORMADO"
// NOME-DA-AREA-DO-CONHECIMENTO CDATA #IMPLIED
// NOME-DA-SUB-AREA-DO-CONHECIMENTO CDATA #IMPLIED
// NOME-DA-ESPECIALIDADE CDATA #IMPLIED
// NIVEL-CURSO (APERFEICOAMENTO ESPECIALIZACAO | APERFEICOAMENTO |
// CURSO_DE_CURTA_DURACAO | ESPECIALIZACAO | ESTAGIO |
// EXTENSAO_UNIVERSITARIA | DOUTORADO | GRADUACAO | MESTRADO |
// MESTRADO_DOUTORADO | OT |OUTRO) #IMPLIED

@XmlAttribute(name = "CODIGO-CURSO")
private String codigoCurso;

@XmlAttribute(name = "CODIGO-ORGAO")
private String codigoOrgao;

@XmlAttribute(name = "NOME-ORGAO")
private String nomeOrgao;

@XmlAttribute(name = "CODIGO-INSTITUICAO")
private String codigoInstituicao;

@XmlAttribute(name = "NOME-INSTITUICAO")
private String nomeInstituicao;

@XmlAttribute(name = "NOME-GRANDE-AREA-DO-CONHECIMENTO")
private GrandeAreaDoConhecimento grandeAreaDoConhecimento;

@XmlAttribute(name = "NOME-DA-AREA-DO-CONHECIMENTO")
private String nomeDaAreaDoConhecimento;

@XmlAttribute(name = "NOME-DA-SUB-AREA-DO-CONHECIMENTO")
private String nomeDaSubAreaDoConhecimento;

@XmlAttribute(name = "NOME-DA-ESPECIALIDADE")
private String nomeDaEspecialidade;

@XmlAttribute(name = "NIVEL-CURSO")
private NivelCurso nivelCurso;
}
package ufsc.alpc.xml.dto.dados.complementares.informacoes;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlRootElement;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;

@Getter
@Setter
@ToString(callSuper = true)
@XmlRootElement(name = "INFORMACAO-ADICIONAL-INSTITUICAO")
@XmlAccessorType(XmlAccessType.FIELD)
public class InformacaoAdicionalInstituicaoXmlDto {
// CODIGO-INSTITUICAO CDATA #IMPLIED
// SIGLA-INSTITUICAO CDATA #IMPLIED
// SIGLA-UF-INSTITUICAO CDATA #IMPLIED
// SIGLA-PAIS-INSTITUICAO CDATA #IMPLIED
// NOME-PAIS-INSTITUICAO CDATA #IMPLIED
// FLAG-AGENCIA-FOMENTO (SIM | NAO) #IMPLIED
// FLAG-INSTITUICAO-DE-ENSINO (SIM | NAO) #IMPLIED

@XmlAttribute(name = "CODIGO-INSTITUICAO")
private String codigoInstituicao;

@XmlAttribute(name = "SIGLA-INSTITUICAO")
private String siglaInstituicao;

@XmlAttribute(name = "SIGLA-UF-INSTITUICAO")
private String siglaUFInstituicao;
}

```

```

@XmlAttribute(name = "SIGLA-PAIS-INSTITUICAO")
private String siglaPaisInstituicao;

@XmlAttribute(name = "NOME-PAIS-INSTITUICAO")
private String nomePaisInstituicao;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-AGENCIA-FOMENTO")
private Boolean flagAgenciaFomento;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-INSTITUICAO-DE-ENSINO")
private Boolean flagInstituicaoDeEnsino;
}
package ufsc.alpc.xml.dto.dados.complementares.informacoes;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

import org.apache.commons.collections.CollectionUtils;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class InformacoesAdicionaisCursosXmlDto {
    // INFORMACAO-ADICIONAL-CURSO*
    @XmlElement(name = "INFORMACAO-ADICIONAL-CURSO")
    private List<InformacaoAdicionalCursoXmlDto> infAdicionalCurso;

    public boolean isSetInfAdicionalCurso() {
        return !CollectionUtils.isEmpty(this.infAdicionalCurso);
    }
}
package ufsc.alpc.xml.dto.dados.complementares.informacoes;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlRootElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

import org.apache.commons.collections.CollectionUtils;

@Getter
@Setter
@ToString(callSuper = true)
@XmlRootElement(name = "INFORMACOES-ADICIONAIS-CURSOS")
@XmlAccessorType(XmlAccessType.FIELD)
public class InformacoesAdicionaisInstituicoesXmlDto {
    // INFORMACAO-ADICIONAL-INSTITUICAO*
    @XmlElement(name = "INFORMACAO-ADICIONAL-INSTITUICAO")
    private List<InformacaoAdicionalInstituicaoXmlDto> infoAdicionalInstituicao;

    public boolean isSetInfoAdicionalInstituicao() {
        return !CollectionUtils.isEmpty(this.infoAdicionalInstituicao);
    }
}package ufsc.alpc.xml.dto;

import java.util.Date;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.XmlRootElement;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

```

```

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.DateXmlAdapter;
import ufsc.alpc.xml.dto.dados.complementares.DadosComplementaresXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.DadosGeraisXmlDto;
import ufsc.alpc.xml.dto.outra.producao.OutraProducaoXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.ProducaoBibliograficaXmlDto;
import ufsc.alpc.xml.dto.producao.tecnica.ProducaoTecnicaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlRootElement(name = "CURRICULO-VITAE")
@XmlAccessorType(XmlAccessType.FIELD)
public class CurriculoVitaeXmlDto {

    // DADOS-GERAIS,
    // PRODUCAO-BIBLIOGRAFICA?,
    // PRODUCAO-TECNICA?,
    // OUTRA-PRODUCAO?,
    // DADOS-COMPLEMENTARES?

    @XmlElement(name = "DADOS-GERAIS")
    private DadosGeraisXmlDto dadosGerais;

    @XmlElement(name = "PRODUCAO-BIBLIOGRAFICA")
    private ProducaoBibliograficaXmlDto producaoBibliografica;

    @XmlElement(name = "PRODUCAO-TECNICA")
    private ProducaoTecnicaXmlDto producaoTecnica;

    @XmlElement(name = "OUTRA-PRODUCAO")
    private OutraProducaoXmlDto outraProducao;

    @XmlElement(name = "DADOS-COMPLEMENTARES")
    private DadosComplementaresXmlDto dadosComplementares;

    // SISTEMA-ORIGEM-XML CDATA #REQUIRED
    // NUMERO-IDENTIFICADOR CDATA #IMPLIED
    // FORMATO-DATA-ATUALIZACAO NMTOKEN #FIXED "DDMMAAAA"
    // DATA-ATUALIZACAO CDATA #IMPLIED
    // FORMATO-HORA-ATUALIZACAO NMTOKEN #FIXED "HHMMSS"
    // HORA-ATUALIZACAO CDATA #IMPLIED
    // xmlns:lattes CDATA #IMPLIED

    @XmlJavaTypeAdapter(DateXmlAdapter.class)
    @XmlAttribute(name = "DATA-ATUALIZACAO")
    private Date dataAtualizacao;

    @XmlAttribute(name = "NUMERO-IDENTIFICADOR")
    private String numeroIdentificador;

    @XmlAttribute(name = "HORA-ATUALIZACAO")
    private String horaAtualizacao;

    public boolean isSetProducaoBibliografica() {
        return this.producaoBibliografica != null;
    }

    public boolean isSetProducaoTecnica() {
        return this.producaoTecnica != null;
    }

    public boolean isSetOutraProducao() {
        return this.outraProducao != null;
    }

    public boolean isSetDadosComplementares() {
        return this.dadosComplementares != null;
    }
}

package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

```

```

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosNaturezaIdiomaHomePageXmlDto extends DadosBasicosXmlDto {
    // NATUREZA CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // HOME-PAGE CDATA #IMPLIED
    // DOI CDATA #IMPLIED

    @XmlAttribute(name = "NATUREZA")
    protected String natureza;

    @XmlAttribute(name = "IDIOMA")
    protected String idioma;

    @XmlAttribute(name = "HOME-PAGE")
    protected String homePage;
}
package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.domain.MeiodeDivulgacao;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;

public abstract class DadosBasicosProducaoBibliograficaXmlDto implements DadosBasicosProducao {

    @XmlAttribute(name = "IDIOMA")
    protected String idioma;

    @XmlAttribute(name = "MEIO-DE-DIVULGACAO")
    protected MeioDeDivulgacao meioDivulgacao;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-RELEVANCIA")
    protected Boolean flagRelevancia = Boolean.FALSE;

    @XmlAttribute(name = "DOI")
    protected String DOI;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-DIVULGACAO-CIENTIFICA")
    protected Boolean flagDivulgacaoCientifica = Boolean.FALSE;

    @XmlAttribute(name = "HOME-PAGE-DO-TRABALHO")
    protected String homePage;
}
package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class SequenciaPalavraAreaSetorInfoAgrupadorXmlDto extends
    PalavraAreaSetorInfoAgrupadorXmlDto {

    @XmlAttribute(name = "SEQUENCIA-PRODUCAO")
    protected String sequenciaProducao;
}
package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;

```

```

import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AutorXmlDto {

    // NOME-COMPLETO-DO-AUTOR CDATA #IMPLIED
    // NOME-PARA-CITACAO CDATA #IMPLIED
    // ORDEM-DE-AUTORIA CDATA #IMPLIED
    // CPF CDATA #IMPLIED
    // NRO-ID-CNPQ CDATA #IMPLIED

    @XmlAttribute(name = "NOME-COMPLETO-DO-AUTOR")
    private String nomeCompleto;

    @XmlAttribute(name = "NOME-PARA-CITACAO")
    private String nomeCitacao;

    @XmlAttribute(name = "ORDEM-DE-AUTORIA")
    private String ordemAutoria;

    @XmlAttribute(name = "CPF")
    private String cpf;

    @XmlAttribute(name = "NRO-ID-CNPQ")
    private String nroIdCnpq;

}

package ufsc.alpc.xml.dto.common.area.conhecimento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AreasDoConhecimentoXmlDto {

    // AREA-DO-CONHECIMENTO-1?
    // AREA-DO-CONHECIMENTO-2?
    // AREA-DO-CONHECIMENTO-3?

    @XmlElement(name = "AREA-DO-CONHECIMENTO-1")
    private AreaDoConhecimentoXmlDto areaDoConhecimento1;

    @XmlElement(name = "AREA-DO-CONHECIMENTO-2")
    private AreaDoConhecimentoXmlDto areaDoConhecimento2;

    @XmlElement(name = "AREA-DO-CONHECIMENTO-3")
    private AreaDoConhecimentoXmlDto areaDoConhecimento3;

    public boolean isSetareaDoConhecimento1() {
        return this.areaDoConhecimento1 != null;
    }

    public boolean isSetareaDoConhecimento2() {
        return this.areaDoConhecimento2 != null;
    }

    public boolean isSetareaDoConhecimento3() {
        return this.areaDoConhecimento3 != null;
    }

}

package ufsc.alpc.xml.dto.common.area.conhecimento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.domain.GrandeAreaDoConhecimento;

```

```

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AreaDoConhecimentoXmlDto {
    // NOME-GRANDE-AREA-DO-CONHECIMENTO (OUTROS |
    // ↪ LINGUISTICA LETRAS_E_ARTES |
    // CIENCIAS_HUMANAS
    // | CIENCIAS_SOCIAIS_APLICADAS | CIENCIAS_AGRARIAS |
    // ↪ CIENCIAS_DA_SAUDE |
    // ENGENHARIAS |
    // CIENCIAS_BIOLÓGICAS | CIENCIAS_EXATAS_E_DA_TERRA) #IMPLIED
    // NOME-DA-AREA-DO-CONHECIMENTO CDATA #IMPLIED
    // NOME-DA-SUB-AREA-DO-CONHECIMENTO CDATA #IMPLIED
    // NOME-DA-ESPECIALIDADE CDATA #IMPLIED

    @XmlAttribute(name = "NOME-GRANDE-AREA-DO-CONHECIMENTO")
    private GrandeAreaDoConhecimento grandeAreaDoConhecimento;

    @XmlAttribute(name = "NOME-DA-AREA-DO-CONHECIMENTO")
    private String nomeDaAreaDoConhecimento;

    @XmlAttribute(name = "NOME-DA-SUB-AREA-DO-CONHECIMENTO")
    private String nomeDaSubAreaDoConhecimento;

    @XmlAttribute(name = "NOME-DA-ESPECIALIDADE")
    private String nomeDaEspecialidade;
}
package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

import ufsc.alpc.xml.adapter.BooleanXmlAdapter;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDaParticipacaoFlagRelevanciaXmlDto extends
    ↪ DadosBasicosDaParticipacaoXmlDto {
    // NATUREZA CDATA #IMPLIED
    // TIPO (ACADEMICO | PROFISSIONALIZANTE | NAO_INFORMADO) #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // HOME-PAGE CDATA #IMPLIED
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-RELEVANCIA")
    protected Boolean flagRelevancia = Boolean.FALSE;
}
package ufsc.alpc.xml.dto.common;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto extends
    ↪ SequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

```

```

        @XmlElement(name = "AUTORES")
        private List<AutorXmlDto> autores;
    }
package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class InformacoesAdicionaisXmlDto {

    // DESCRICAO-INFORMACOES-ADICIONAIS CDATA #IMPLIED
    // DESCRICAO-INFORMACOES-ADICIONAIS-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "DESCRICAO-INFORMACOES-ADICIONAIS")
    private String descricaoInformacoesAdicionais;

    @XmlAttribute(name = "DESCRICAO-INFORMACOES-ADICIONAIS-INGLES")
    private String descricaoInformacoesAdicionaisIngles;
}
package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosXmlDto {

    @XmlAttribute(name = "ANO")
    protected Integer ano;

    @XmlAttribute(name = "PAIS")
    protected String pais;

    @XmlAttribute(name = "DOI")
    protected String DOI;
}
package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class SetoresDeAtividadeXmlDto {

    // SETOR-DE-ATIVIDADE-1 CDATA #IMPLIED
    // SETOR-DE-ATIVIDADE-2 CDATA #IMPLIED
    // SETOR-DE-ATIVIDADE-3 CDATA #IMPLIED

    @XmlAttribute(name = "SETOR-DE-ATIVIDADE-1")
    private String setorDeAtividade1;

    @XmlAttribute(name = "SETOR-DE-ATIVIDADE-2")
    private String setorDeAtividade2;

    @XmlAttribute(name = "SETOR-DE-ATIVIDADE-3")

```



```

        private String setorDeAtividade3;
    }
    package ufsc.alpc.xml.dto.common;

    import javax.xml.bind.annotation.XmlAttribute;
    import ufsc.alpc.xml.dto.domain.MeiDeDivulgacao;

    public class DadosBasicosOutraProducaoDivulgado extends DadosBasicosOutraProducaoXmlDto {

        @XmlAttribute(name = "MEIO-DE-DIVULGACAO")
        protected MeiDeDivulgacao meioDeDivulgacao;

        @XmlAttribute(name = "HOME-PAGE")
        protected String homePage;
    }
    package ufsc.alpc.xml.dto.common;

    import javax.xml.bind.annotation.XmlAccessType;
    import javax.xml.bind.annotation.XmlAccessorType;
    import javax.xml.bind.annotation.XmlAttribute;
    import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

    import lombok.Getter;
    import lombok.Setter;
    import lombok.ToString;
    import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
    import ufsc.alpc.xml.dto.domain.TipoDeOrientacao;

    @Getter
    @Setter
    @ToString(callSuper = true)
    @XmlAccessorType(XmlAccessType.FIELD)
    public class DetalhamentoDaOrientacaoConcluidaXmlDto {

        // TIPO-DE-ORIENTACAO (ORIENTADOR_PRINCIPAL | CO_ORIENTADOR)
        // ↪ #IMPLIED
        // NOME-DO-ORIENTADO CDATA #IMPLIED
        // CODIGO-INSTITUICAO CDATA #IMPLIED
        // NOME-DA-INSTITUICAO CDATA #IMPLIED
        // CODIGO-ORGAO CDATA #IMPLIED
        // NOME-ORGAO CDATA #IMPLIED
        // CODIGO-CURSO CDATA #IMPLIED
        // NOME-DO-CURSO CDATA #IMPLIED
        // FLAG-BOLSA CDATA #IMPLIED
        // CODIGO-AGENCIA-FINANCIADORA CDATA #IMPLIED
        // NOME-DA-AGENCIA CDATA #IMPLIED
        // NUMERO-DE-PAGINAS CDATA #IMPLIED
        // NUMERO-ID-ORIENTADO CDATA #IMPLIED
        // NOME-DO-CURSO-INGLES CDATA #IMPLIED

        @XmlAttribute(name = "TIPO-DE-ORIENTACAO")
        protected TipoDeOrientacao tipoOrientacao;

        @XmlAttribute(name = "NOME-DO-ORIENTADO")
        protected String nomeOrientando;

        @XmlAttribute(name = "CODIGO-INSTITUICAO")
        protected String codigoInstituicao;

        @XmlAttribute(name = "NOME-DA-INSTITUICAO")
        protected String nomeInstituicao;

        @XmlAttribute(name = "CODIGO-ORGAO")
        protected String codigoOrgao;

        @XmlAttribute(name = "NOME-ORGAO")
        protected String nomeOrgao;

        @XmlAttribute(name = "CODIGO-CURSO")
        protected String codigoCurso;

        @XmlAttribute(name = "NOME-DO-CURSO")
        protected String nomeCurso;

        @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
        @XmlAttribute(name = "FLAG-BOLSA")
        protected Boolean flagBolsa;

        @XmlAttribute(name = "CODIGO-AGENCIA-FINANCIADORA")
        protected String codigoAgenciaFinanciadora;
    }

```

```

    @XmlAttribute(name = "NOME-DA-AGENCIA")
    protected String nomeDaAgencia;

    @XmlAttribute(name = "NUMERO-DE-PAGINAS")
    protected String numeroPaginas;

    @XmlAttribute(name = "NUMERO-ID-ORIENTADO")
    protected String numeroIDorientado;

    @XmlAttribute(name = "NOME-DO-CURSO-INGLES")
    protected String nomeCursoIngles;
}

package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.dados.complementares.DetalhamentoXmlDto;
import ufsc.alpc.xml.dto.domain.TipoDeOrientacao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoDaOrientacaoXmlDto extends DetalhamentoXmlDto {

    // TIPO-DE-ORIENTACAO (ORIENTADOR_PRINCIPAL | CO_ORIENTADOR)
    // ↔ #IMPLIED
    // NOME-DO-ORIENTANDO CDATA #IMPLIED
    // CODIGO-INSTITUICAO CDATA #IMPLIED
    // NOME-INSTITUICAO CDATA #IMPLIED
    // CODIGO-ORGAO CDATA #IMPLIED
    // NOME-ORGAO CDATA #IMPLIED
    // CODIGO-CURSO CDATA #IMPLIED
    // NOME-CURSO CDATA #IMPLIED
    // FLAG-BOLSA CDATA #IMPLIED
    // CODIGO-AGENCIA-FINANCIADORA CDATA #IMPLIED
    // NOME-DA-AGENCIA CDATA #IMPLIED
    // NUMERO-ID-ORIENTANDO CDATA #IMPLIED
    // NOME-CURSO-INGLES CDATA #IMPLIED

    // apenas para mestrado, doutorado e pos doutorado
    @XmlAttribute(name = "TIPO-DE-ORIENTACAO")
    protected TipoDeOrientacao tipoOrientacao;

    @XmlAttribute(name = "NOME-DO-ORIENTANDO")
    protected String nomeOrientando;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-BOLSA")
    protected Boolean flagBolsa;

    @XmlAttribute(name = "CODIGO-AGENCIA-FINANCIADORA")
    protected String codigoAgenciaFinanciadora;

    @XmlAttribute(name = "NOME-DA-AGENCIA")
    protected String nomeDaAgencia;

    @XmlAttribute(name = "NUMERO-ID-ORIENTANDO")
    protected String numeroIDorientado;

    /**
     * apenas para OrientacaoEmAndamentoDeMestradoXmlDto
     */
    @XmlAttribute(name = "NUMERO-ID-ORIENTADO")
    protected String numeroIDorientado;
}

package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

```

```

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDaParticipacaoXmlDto extends
    ↵ DadosBasicosNaturezaIdiomaHomePageXmlDto {
    // NATUREZA CDATA #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // HOME-PAGE CDATA #IMPLIED
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "TITULO")
    protected String titulo;

    @XmlAttribute(name = "TITULO-INGLES")
    protected String tituloIngles;
}
package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

import ufsc.alpc.xml.dto.common.area.conhecimento.AreasDoConhecimentoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.CursoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class PalavraAreasSetoresXmlDto extends CursoXmlDto {
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?

    @XmlElement(name = "PALAVRAS-CHAVE")
    protected PalavrasChaveXmlDto palavraChave;

    @XmlElement(name = "AREAS-DO-CONHECIMENTO")
    protected AreasDoConhecimentoXmlDto areasDoConhecimento;

    @XmlElement(name = "SETORES-DE-ATIVIDADE")
    protected SetoresDeAtividadeXmlDto setoresDeAtividade;
}
package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import ufsc.alpc.xml.adapter.BooleanXmlAdapter;

public class DadosBasicosOutraProducaoXmlDto extends DadosBasicosXmlDto {
    @XmlAttribute(name = "TITULO")
    protected String titulo;

    @XmlAttribute(name = "TITULO-INGLES")
    protected String tituloIngles;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-RELEVANCIA")
    protected Boolean flagRelevancia = Boolean.FALSE;
}
package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;

```

```

import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

import ufsc.alpc.xml.dto.common.area.conhecimento.AreasDoConhecimentoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class PalavraAreaSetorInfoAgrupadorXmlDto {

    @XmlElement(name = "PALAVRAS-CHAVE")
    protected PalavrasChaveXmlDto palavraChave;

    @XmlElement(name = "AREAS-DO-CONHECIMENTO")
    protected AreasDoConhecimentoXmlDto areasDoConhecimento;

    @XmlElement(name = "SETORES-DE-ATIVIDADE")
    protected SetoresDeAtividadeXmlDto setoresDeAtividade;

    @XmlElement(name = "INFORMACOES-ADICIONAIS")
    protected InformacoesAdicionaisXmlDto informacoesAdicionais;
}

package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDaParticipacaoTipoInglesXmlDto extends
↳ DadosBasicosDaParticipacaoTipoXmlDto {
    // NATUREZA CDATA #IMPLIED
    // TIPO (ACADEMICO | PROFISSIONALIZANTE | NAO_INFORMADO) #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // HOME-PAGE CDATA #IMPLIED
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED
    // TIPO (ACADEMICO | PROFISSIONALIZANTE | NAO_INFORMADO) #IMPLIED
    // TIPO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "TIPO-INGLES")
    protected String tipoIngles;
}

package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

import ufsc.alpc.xml.dto.domain.Tipo;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDaParticipacaoTipoXmlDto extends DadosBasicosDaParticipacaoXmlDto {
    // NATUREZA CDATA #IMPLIED
    // TIPO (ACADEMICO | PROFISSIONALIZANTE | NAO_INFORMADO) #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // HOME-PAGE CDATA #IMPLIED
    // DOI CDATA #IMPLIED

```

```

// TITULO-INGLES CDATA #IMPLIED
// TIPO (ACADEMICO | PROFISSIONALIZANTE | NAO_INFORMADO) #IMPLIED

@XmlAttribute(name = "TIPO")
private Tipo tipo;
}
package ufsc.alpc.xml.dto.common;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class PalavrasChaveXmlDto {

    // PALAVRA-CHAVE-1 CDATA #IMPLIED
    // PALAVRA-CHAVE-2 CDATA #IMPLIED
    // PALAVRA-CHAVE-3 CDATA #IMPLIED
    // PALAVRA-CHAVE-4 CDATA #IMPLIED
    // PALAVRA-CHAVE-5 CDATA #IMPLIED
    // PALAVRA-CHAVE-6 CDATA #IMPLIED

    @XmlAttribute(name = "PALAVRA-CHAVE-1")
    private String palavraChave1;
    @XmlAttribute(name = "PALAVRA-CHAVE-2")
    private String palavraChave2;
    @XmlAttribute(name = "PALAVRA-CHAVE-3")
    private String palavraChave3;
    @XmlAttribute(name = "PALAVRA-CHAVE-4")
    private String palavraChave4;
    @XmlAttribute(name = "PALAVRA-CHAVE-5")
    private String palavraChave5;
    @XmlAttribute(name = "PALAVRA-CHAVE-6")
    private String palavraChave6;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.artes.visuais.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.common.DadosBasicosOutraProducaoDivulgado;
import ufsc.alpc.xml.dto.domain.NaturezaArtesVisuais;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosArtesVisuais extends DadosBasicosOutraProducaoDivulgado {

    // NATUREZA (INTERVENCAO_URBANA | LIVRO_DE_ARTISTA | PERFORMANCE |
    // ↳ PINTURA |
    // PROGRAMACAO_VISUAL | VIDEO | WEBART | ANIMACAO | INSTALACAO |
    // COMPUTACAO_GRAFICA | DESENHO | DIVERSAS | ESCULTURA | FILME |
    // ↳ FOTOGRAFIA
    // | GRAVURA | ILUSTRACAO | OUTRA ) #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // TITULO-INGLES CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    // ↳ MEIO_DIGITAL |
    // FILME | HIPERTEXTO
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE CDATA #IMPLIED
    // FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"

```

```

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaArtesVisuais natureza;

    @XmlAttribute(name = "IDIOMA")
    protected String idioma;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-DIVULGACAO-CIENTIFICA")
    protected Boolean flagDivulgacaoCientifica = Boolean.FALSE;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.artes.visuais;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.artes.visuais.dadosbasicos.DadosBasicosArtesVisuais;
import ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.artes.visuais.detalhamento.DetalhamentoArtesVisuais;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ArtesVisuaisXmlDto extends AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

    // DADOS-BASICOS-DE-ARTES-VISUAIS?
    // DETALHAMENTO-DE-ARTES-VISUAIS?
    // AUTORES*,
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlElement(name = "DADOS-BASICOS-DE-ARTES-VISUAIS")
    private DadosBasicosArtesVisuais dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DE-ARTES-VISUAIS")
    private DetalhamentoArtesVisuais detalhamento;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.artes.visuais.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoArtesVisuais {

    // PREMIACAO CDATA #IMPLIED
    // ATIVIDADE-DOS-AUTORES CDATA #IMPLIED
    // INSTITUICAO-PROMOTORA-DO-EVENTO CDATA #IMPLIED
    // LOCAL-DO-EVENTO CDATA #IMPLIED
    // CIDADE-DO-EVENTO CDATA #IMPLIED
    // TEMPORADA CDATA #IMPLIED

    @XmlAttribute(name = "PREMIACAO")
    protected String premiacao;

    @XmlAttribute(name = "ATIVIDADE-DOS-AUTORES")
    protected String atividadeAutores;

    @XmlAttribute(name = "INSTITUICAO-PROMOTORA-DO-EVENTO")
    protected String instituicaoPromotoraEvento;

    @XmlAttribute(name = "LOCAL-DO-EVENTO")
    protected String localEvento;
}

```

```

@XmlAttribute(name = "CIDADE-DO-EVENTO")
protected String cidadeEvento;

@XmlAttribute(name = "TEMPORADA")
protected String temporada;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.artes.cenicas.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.common.DadosBasicosOutraProducaoDivulgado;
import ufsc.alpc.xml.dto.domain.NaturezaArtesCenicas;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosArtesCenicas extends DadosBasicosOutraProducaoDivulgado {

    // NATUREZA (AUDIOVISUAL | CIRCENSE | COREOGRAFICA | DIVERSAS |
    // ↳ OPERÍSTICA |
    // PERFORMÁTICA |
    // RADIALÍSTICA | TEATRAL | OUTRA ) #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // TITULO-INGLES CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    // ↳ MEIO_DIGITAL |
    // FILME | HIPERTEXTO
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE CDATA #IMPLIED
    // FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaArtesCenicas natureza;

    @XmlAttribute(name = "IDIOMA")
    protected String idioma;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-DIVULGACAO-CIENTIFICA")
    protected Boolean flagDivulgacaoCientifica = Boolean.FALSE;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.artes.cenicas.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoArtesCenicas {

    // TIPO-DE-EVENTO CDATA #IMPLIED
    // ATIVIDADE-DOS-AUTORES CDATA #IMPLIED
    // DATA-ESTREIA CDATA #IMPLIED
    // DATA-ENCERRAMENTO CDATA #IMPLIED
    // LOCAL-DE-ESTREIA CDATA #IMPLIED
    // PREMIAÇÃO CDATA #IMPLIED
    // INSTITUIÇÃO-PROMOTORA-DO-PREMIO CDATA #IMPLIED
    // OBRA-DE-REFERENCIA CDATA #IMPLIED
    // AUTOR-DA-OBRA-DE-REFERENCIA CDATA #IMPLIED
    // ANO-DA-OBRA-DE-REFERENCIA CDATA #IMPLIED
    // DURACAO CDATA #IMPLIED

```

```

// TEMPORADA CDATA #IMPLIED
// INSTITUICAO-PROMOTORA-DO-EVENTO CDATA #IMPLIED
// LOCAL-DO-EVENTO CDATA #IMPLIED
// CIDADE-DO-EVENTO CDATA #IMPLIED
// FLAG-ITINERANTE CDATA #IMPLIED

@XmlAttribute(name = "TIPO-DE-EVENTO")
protected String tipoEvento;

@XmlAttribute(name = "ATIVIDADE-DOS-AUTORES")
protected String atividadeAutores;

@XmlAttribute(name = "DATA-ESTREIA")
protected String dataEstreia;

@XmlAttribute(name = "DATA-ENCERRAMENTO")
protected String dataEncerramento;

@XmlAttribute(name = "LOCAL-DE-ESTREIA")
protected String localEstreia;

@XmlAttribute(name = "PREMIACAO")
protected String premiacao;

@XmlAttribute(name = "INSTITUICAO-PROMOTORA-DO-PREMIO")
protected String instituicaoPromotoraPremio;

@XmlAttribute(name = "OBRA-DE-REFERENCIA")
protected String obraReferencia;

@XmlAttribute(name = "AUTOR-DA-OBRA-DE-REFERENCIA")
protected String autorObraReferencia;

@XmlAttribute(name = "ANO-DA-OBRA-DE-REFERENCIA")
protected String anoObraReferencia;

@XmlAttribute(name = "DURACAO")
protected String duracao;

@XmlAttribute(name = "TEMPORADA")
protected String temporada;

@XmlAttribute(name = "INSTITUICAO-PROMOTORA-DO-EVENTO")
protected String instituicaoPromotoraEvento;

@XmlAttribute(name = "LOCAL-DO-EVENTO")
protected String localEvento;

@XmlAttribute(name = "CIDADE-DO-EVENTO")
protected String cidadeEvento;

@XmlAttribute(name = "FLAG-ITINERANTE")
protected String flagItinerante;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.artes.cenicass;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.artes.cenicass.dadosbasicos.DadosBasicosArtesCenicass;
import ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.artes.cenicass.detalhamento.DetalhamentoArtesCenicass;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ArtesCenicassXmlDto extends AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

    // DADOS-BASICOS-DE-ARTES-CENICAS?
    // DETALHAMENTO-DE-ARTES-CENICAS?
    // AUTORES*,
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?

```



```

// INFORMACOES-ADICIONAIS?
// SEQUENCIA-PRODUCAO CDATA #IMPLIED

@XmlElement(name = "DADOS-BASICOS-DE-ARTES-CENICAS")
private DadosBasicosArtesCenicas dadosBasicos;

@XmlElement(name = "DETALHAMENTO-DE-ARTES-CENICAS")
private DetalhamentoArtesCenicas detalhamento;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.obra.artesvisuais.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosOutraProducaoDivulgado;
import ufsc.alpc.xml.dto.domain.NaturezaObraArtesVisuais;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosObraArtesVisuais extends DadosBasicosOutraProducaoDivulgado {

    // NATUREZA (CINEMA | DESENHO | ESCULTURA | FOTOGRAFIA | GRAVURA |
    // INSTALACAO | PINTURA | TELEVISAO | VIDEO | OUTRA) #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    // ↪ MEIO_DIGITAL |
    // FILME | HIPERTEXTO | OUTRO | VARIOS | NAO_INFORMADO)
    // ↪ "NAO_INFORMADO"
    // HOME-PAGE CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaObraArtesVisuais natureza;

    @XmlAttribute(name = "IDIOMA")
    protected String idioma;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.obra.artesvisuais.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.domain.Acervo;
import ufsc.alpc.xml.dto.domain.TipoEventoArteVisual;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoObraArtesVisuais {

    // MATERIAL-EMPREGADO CDATA #IMPLIED
    // TIPO-DE-EVENTO (APRESENTACAO | CONCURSO | CRIACAO | EXPOSICAO |
    // ↪ FESTIVAL
    // | OUTRO | NAO_INFORMADO) #IMPLIED
    // EVENTO CDATA #IMPLIED
    // PREMIACAO CDATA #IMPLIED
    // ACERVO (PUBLICO | PRIVADO | NAO_INFORMADO) "NAO_INFORMADO"
    // INSTITUICAO-PROMOTORA CDATA #IMPLIED

    @XmlAttribute(name = "MATERIAL-EMPREGADO")
    protected String materialEmpregado;

    @XmlAttribute(name = "TIPO-DE-EVENTO")
    protected TipoEventoArteVisual tipoEvento;
}

```

```

    @XmlAttribute(name = "EVENTO")
    protected String evento;

    @XmlAttribute(name = "PREMIACAO")
    protected String premiacao;

    @XmlAttribute(name = "ACERVO")
    protected Acervo acervo;

    @XmlAttribute(name = "INSTITUICAO-PROMOTORA")
    protected String instituicaoPromotora;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.obra.artesvisuais;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.obra.artesvisuais.dadosbasicos.DadosBasicosObraArtesVisuais;
import ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.obra.artesvisuais.detalhamento.DetalhamentoObraArtesVisuais;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ObraArtesVisuaisXmlDto extends AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto {
    // DADOS-BASICOS-DA-OBRA-DE-ARTES-VISUAIS?
    // DETALHAMENTO-DA-OBRA-DE-ARTES-VISUAIS?
    // AUTORES*
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name = "DADOS-BASICOS-DA-OBRA-DE-ARTES-VISUAIS")
    private DadosBasicosObraArtesVisuais dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DA-OBRA-DE-ARTES-VISUAIS")
    private DetalhamentoObraArtesVisuais detalhamento;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.composicao musical;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.composicaomusical.dadosbasicos.DadosBasicosComposicaoMusical;
import ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.composicaomusical.detalhamento.DetalhamentoComposicaoMusical;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ComposicaoMusicalXmlDto extends AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto {
    // DADOS-BASICOS-DA-COMPOSICAO-MUSICAL?
    // DETALHAMENTO-DA-COMPOSICAO-MUSICAL?
    // AUTORES*,
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

```

```

    @XmlElement(name = "DADOS-BASICOS-DA-COMPOSICAO-MUSICAL")
    private DadosBasicosComposicaoMusical dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DA-COMPOSICAO-MUSICAL")
    private DetalhamentoComposicaoMusical detalhamento;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.composicaomusical.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosOutraProducaoDivulgado;
import ufsc.alpc.xml.dto.domain.NaturezaArranjoMusical;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosComposicaoMusical extends DadosBasicosOutraProducaoDivulgado {

    // NATUREZA (CANTO | CORAL | ORQUESTRA | OUTRA | NAO_INFORMADO)
    // ↪ #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    // ↪ MEIO_DIGITAL |
    // FILME | HIPERTEXTO
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaArranjoMusical natureza;

    @XmlAttribute(name = "IDIOMA")
    protected String idioma;
}
package ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.composicaomusical.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoComposicaoMusical {

    // FORMACAO-INSTRUMENTAL CDATA #IMPLIED
    // NUMERO-DE-PAGINAS CDATA #IMPLIED
    // REGISTRO-DE-DIREITO-AUTORAL CDATA #IMPLIED
    // PREMIACAO CDATA #IMPLIED

    @XmlAttribute(name = "FORMACAO-INSTRUMENTAL")
    protected String formacaoInstrumental;

    @XmlAttribute(name = "NUMERO-DE-PAGINAS")
    protected String numPaginas;

    @XmlAttribute(name = "REGISTRO-DE-DIREITO-AUTORAL")
    protected String registroDireitoAutoral;

    @XmlAttribute(name = "PREMIACAO")
    protected String premio;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.arranjomusical.dadosbasicos;

```

```

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosOutraProducaoDivulgado;
import ufsc.alpc.xml.dto.domain.NaturezaArranjoMusical;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosArranjoMusical extends DadosBasicosOutraProducaoDivulgado {
    // ATTLIST DADOS-BASICOS-DO-ARRANJO-MUSICAL
    // NATUREZA (CANTO | CORAL | ORQUESTRA | OUTRA | NAO_INFORMADO)
    //     ↔ #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    //     ↔ MEIO_DIGITAL |
    // FILME | HIPERTEXTO
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaArranjoMusical natureza;
}

package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.arranjomusical.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoArranjoMusical {

    // AUTOR-DA-OBRA-DE-REFERENCIA CDATA #IMPLIED
    // ANO-DA-OBRA-DE-REFERENCIA CDATA #IMPLIED
    // FORMACAO-INSTRUMENTAL CDATA #IMPLIED
    // REGISTRO-DE-DIREITO-AUTORAL CDATA #IMPLIED
    // PREMIACAO CDATA #IMPLIED

    @XmlElement(name = "AUTOR-DA-OBRA-DE-REFERENCIA")
    private String autorObraReferencia;

    @XmlElement(name = "ANO-DA-OBRA-DE-REFERENCIA")
    private String anoObraReferencia;

    @XmlElement(name = "FORMACAO-INSTRUMENTAL")
    private String formacaoInstrumental;

    @XmlElement(name = "REGISTRO-DE-DIREITO-AUTORAL")
    private String registroDireitoAutoral;

    @XmlElement(name = "PREMIACAO")
    private String premio;
}

package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.arranjomusical;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;

```

```

import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.arranjomusical.dadosbasicos.DadosBasicosArranjoMusical;
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.arranjomusical.detalhamento.DetalhamentoArranjoMusical;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ArranjoMusicalXmlDto extends
    ↪ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

    // DADOS-BASICOS-DO-ARRANJO-MUSICAL?
    // DETALHAMENTO-DO-ARRANJO-MUSICAL?
    // AUTORES*
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlElement(name = "DADOS-BASICOS-DO-ARRANJO-MUSICAL")
    private DadosBasicosArranjoMusical dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DO-ARRANJO-MUSICAL")
    private DetalhamentoArranjoMusical detalhamento;

} package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.curso.curtaduracao;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.curso.curtaduracao.dadosbasicos.DadosBasicosCursoCurtaDuracao;
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.curso.curtaduracao.detalhamento.DetalhamentoCursoCurtaDuracao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class CursoCurtaDuracaoXmlDto extends
    ↪ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

    // DADOS-BASICOS-DO-CURSO-DE-CURTA-DURACAO?
    // DETALHAMENTO-DO-CURSO-DE-CURTA-DURACAO?
    // AUTORES*
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlElement(name = "DADOS-BASICOS-DO-CURSO-DE-CURTA-DURACAO")
    private DadosBasicosCursoCurtaDuracao dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DO-CURSO-DE-CURTA-DURACAO")
    private DetalhamentoCursoCurtaDuracao detalhamento;

}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.curso.curtaduracao.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosOutraProducaoDivulgado;
import ufsc.alpc.xml.dto.domain.NaturezaCursoCurtaDuracao;

@Getter
@Setter
@ToString(callSuper = true)

```

```

@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosCursoCurtaDuracao extends DadosBasicosOutraProducaoDivulgado {
    // NATUREZA (EXTENSAO | APERFEICOAMENTO | ESPECIALIZACAO | OUTRA |
    // NAO_INFORMADO) #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    // ↪ MEIO_DIGITAL |
    // FILME | HIPERTEXTO |
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaCursoCurtaDuracao natureza;

    @XmlAttribute(name = "IDIOMA")
    protected String idioma;
}
package ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.curso.curtaduracao.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.domain.Unidade;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoCursoCurtaDuracao {
    // DURACAO CDATA #IMPLIED
    // UNIDADE (SEMANAS | DIAS | HORAS | NAO_INFORMADO) "NAO_INFORMADO"
    // INSTITUICAO-PROMOTORA-DO-EVENTO CDATA #IMPLIED
    // LOCAL-DO-EVENTO CDATA #IMPLIED
    // CIDADE CDATA #IMPLIED

    @XmlAttribute(name = "DURACAO")
    protected String duracao;

    @XmlAttribute(name = "UNIDADE")
    protected Unidade unidade;

    @XmlAttribute(name = "INSTITUICAO-PROMOTORA-DO-EVENTO")
    protected String instituicaoPromotoraEvento;

    @XmlAttribute(name = "LOCAL-DO-EVENTO")
    protected String localEvento;

    @XmlAttribute(name = "CIDADE")
    protected String cidade;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.sonoplastia;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.sonoplastia.dadosbasicos.DadosBasicosSonoplastia;
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.sonoplastia.detalhamento.DetalhamentoSonoplastia;

@Getter
@Setter

```

```

@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class SonoplastiaXmlDto extends AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

    // DADOS-BASICOS-DE-SONOPLASTIA?
    // DETALHAMENTO-DE-SONOPLASTIA?
    // AUTORES*
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlElement(name = "DADOS-BASICOS-DE-SONOPLASTIA")
    private DadosBasicosSonoplastia dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DE-SONOPLASTIA")
    private DetalhamentoSonoplastia detalhamento;
}

package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.sonoplastia.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosOutraProducaoDivulgado;
import ufsc.alpc.xml.dto.domain.NaturezaSonoplastia;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosSonoplastia extends DadosBasicosOutraProducaoDivulgado {

    // NATUREZA (CINEMA | MUSICA | RADIO | TELEVISAO | TEATRO | OUTRA)
    ↪ #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    ↪ MEIO_DIGITAL |
    // FILME | HIPERTEXTO )
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaSonoplastia natureza;
}

package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.sonoplastia.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoSonoplastia {

    // FINALIDADE CDATA #IMPLIED
    // PREMIACAO CDATA #IMPLIED

    @XmlAttribute(name = "FINALIDADE")
    protected String finalidade;

    @XmlAttribute(name = "PREMIACAO")
    protected String premio;
}

package
↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.apresentacao.obra.artistica.dadosbasicos;

```

```

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosOutraProducaoDivulgado;
import ufsc.alpc.xml.dto.domain.NaturezaObraArtistica;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosApresentacaoObraArtisticaXmlDto extends
    ↪ DadosBasicosOutraProducaoDivulgado {

    // NATUREZA (COREOGRAFICA | LITERARIA | MUSICAL | TEATRAL | OUTRA |
    // NAO_INFORMADO) #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    // ↪ MEIO_DIGITAL |
    // FILME | HIPERTEXTO
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaObraArtistica natureza;

    @XmlAttribute(name = "IDIOMA")
    protected String idioma;

}
package
    ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.apresentacao.obra.artistica.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.domain.TipoEventoMusical;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoApresentacaoObraArtisticaXmlDto {

    // TIPO-DE-EVENTO (CONCERTO | CONCURSO | FESTIVAL | GRAVACAO |
    // ↪ RECITAL | OUTRO | NAO_INFORMADO) #IMPLIED
    // ATIVIDADE-DOS-AUTORES (CANTO | CRIACAO | DANCA | DIRECAO |
    // ↪ ENCENACAO | INSTRUMENTO_MUSICAL | REGENCIA | ROTEIRO | OUTRA
    // ↪ | VARIAS | NAO_INFORMADO) #IMPLIED
    // FLAG-INEDITISMO-DA-OBRA CDATA #IMPLIED
    // PREMIACAO CDATA #IMPLIED
    // OBRA-DE-REFERENCIA CDATA #IMPLIED
    // AUTOR-DA-OBRA-DE-REFERENCIA CDATA #IMPLIED
    // ANO-DA-OBRA-DE-REFERENCIA CDATA #IMPLIED
    // DURACAO-EM-MINUTOS CDATA #IMPLIED
    // INSTITUICAO-PROMOTORA-DO-EVENTO CDATA #IMPLIED
    // LOCAL-DO-EVENTO CDATA #IMPLIED
    // CIDADE CDATA #IMPLIED

    @XmlAttribute(name = "TIPO-DE-EVENTO")
    private TipoEventoMusical tipo;

    @XmlAttribute(name = "FLAG-INEDITISMO-DA-OBRA")
    protected String flagIneditismoObra;

    @XmlAttribute(name = "PREMIACAO")
    protected String premiacao;

```



```

    @XmlAttribute(name = "OBRA-DE-REFERENCIA")
    protected String obraReferencia;

    @XmlAttribute(name = "AUTOR-DA-OBRA-DE-REFERENCIA")
    protected String autorObraReferencia;

    @XmlAttribute(name = "ANO-DA-OBRA-DE-REFERENCIA")
    protected String anoObraReferencia;

    @XmlAttribute(name = "DURACAO-EM-MINUTOS")
    protected String duracaoMinutos;

    @XmlAttribute(name = "INSTITUICAO-PROMOTORA-DO-EVENTO")
    protected String instituicaoPromotora;

    @XmlAttribute(name = "LOCAL-DO-EVENTO")
    protected String local;

    @XmlAttribute(name = "CIDADE")
    protected String cidade;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.apresentacao.obra.artistica;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import
    ↳ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.apresentacao.obra.artistica.dadosbasicos.DadosBasicos;
import
    ↳ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.apresentacao.obra.artistica.detalhamento.Detalhamento;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ApresentacaoObraArtisticaXmlDto extends
    ↳ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto {
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED
    // DADOS-BASICOS-DA-APRESENTACAO-DE-OBRA-ARTISTICA?
    // DETALHAMENTO-DA-APRESENTACAO-DE-OBRA-ARTISTICA?
    // AUTORES*
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?

    @XmlElement(name =
        ↳ "DADOS-BASICOS-DA-APRESENTACAO-DE-OBRA-ARTISTICA")
    private DadosBasicosApresentacaoObraArtisticaXmlDto dadosBasicos;

    @XmlElement(name =
        ↳ "DETALHAMENTO-DA-APRESENTACAO-DE-OBRA-ARTISTICA")
    private DetalhamentoApresentacaoObraArtisticaXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.apresentacao.radiotv;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import
    ↳ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.apresentacao.radiotv.dadosbasicos.DadosBasicos;
import
    ↳ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.apresentacao.radiotv.detalhamento.Detalhamento;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ApresentacaoRadioTvXmlDto extends
    ↳ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

```

```

// DADOS-BASICOS-DA-APRESENTACAO-EM-RADIO-OU-TV?
// DETALHAMENTO-DA-APRESENTACAO-EM-RADIO-OU-TV?
// AUTORES*
// PALAVRAS-CHAVE?,
// AREAS-DO-CONHECIMENTO?
// SETORES-DE-ATIVIDADE?
// INFORMACOES-ADICIONAIS?

@XmlElement(name =
    ↪ "DADOS-BASICOS-DA-APRESENTACAO-EM-RADIO-OU-TV")
private DadosBasicosApresentacaoRadioTv dadosBasicos;

@XmlElement(name =
    ↪ "DETALHAMENTO-DA-APRESENTACAO-EM-RADIO-OU-TV")
private DetalhamentoApresentacaoRadioTv detalhamento;
}
package
    ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.apresentacao.radiotv.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.common.DadosBasicosOutraProducaoXmlDto;
import ufsc.alpc.xml.dto.domain.NaturezaRadioTv;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosApresentacaoRadioTv extends DadosBasicosOutraProducaoXmlDto {
    // NATUREZA (DANCA | MUSICA | TEATRO | OUTRA | NAO_INFORMADO)
    ↪ #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED
    // FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"

    @XmlAttribute(name = "NATUREZA")
    protected NaturezaRadioTv natureza;

    @XmlAttribute(name = "IDIOMA")
    protected String idioma;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-DIVULGACAO-CIENTIFICA")
    private Boolean flagDivulgacaoCientifica = Boolean.FALSE;
}
package
    ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.apresentacao.radiotv.detalhamento;

import java.util.Date;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.XmlElement;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.DateXmlAdapter;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoApresentacaoRadioTv {

```

```

// EMISSORA CDATA #IMPLIED
// FORMATO-DATA-DE-APRESENTACAO NMTOKEN #FIXED "DDMMAAAA"
// DATA-DE-APRESENTACAO CDATA #IMPLIED
// DURACAO-EM-MINUTOS CDATA #IMPLIED
// CIDADE CDATA #IMPLIED

    @XmlElement(name = "EMISSORA")
    private String emissora;

    @XmlJavaTypeAdapter(DateXmlAdapter.class)
    @XmlAttribute(name = "DATA-DE-APRESENTACAO")
    private Date DataAtualizacao;

    @XmlElement(name = "DURACAO-EM-MINUTOS")
    private String duracao;

    @XmlElement(name = "CIDADE")
    private String cidade;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.outra.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.common.DadosBasicosOutraProducaoDivulgado;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosOutraProducaoArtisticaCultural extends
    ↪ DadosBasicosOutraProducaoDivulgado {

    // NATUREZA CDATA #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    ↪ MEIO_DIGITAL |
    // FILME | HIPERTEXTO
    // | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
    // HOME-PAGE CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED
    // NATUREZA-INGLES CDATA #IMPLIED
    // FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"

    @XmlAttribute(name = "NATUREZA")
    protected String natureza;

    @XmlAttribute(name = "IDIOMA")
    protected String idioma;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-DIVULGACAO-CIENTIFICA")
    protected Boolean flagDivulgacaoCientifica = Boolean.FALSE;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.outra;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.outra.dadosbasicos.DadosBasicosOutraProducaoDivulgado;
import ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.outra.detalhamento.DetalhamentoOutraProducaoDivulgado;

```

```

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OutraProducaoArtisticaCulturalXmlDto extends
    ↳ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto {
    // DADOS-BASICOS-DE-OUTRA-PRODUCAO-ARTISTICA-CULTURAL?
    // DETALHAMENTO-DE-OUTRA-PRODUCAO-ARTISTICA-CULTURAL?
    // AUTORES*
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlElement(name =
        ↳ "DADOS-BASICOS-DE-OUTRA-PRODUCAO-ARTISTICA-CULTURAL")
    private DadosBasicosOutraProducaoArtisticaCultural dadosBasicos;

    @XmlElement(name =
        ↳ "DETALHAMENTO-DE-OUTRA-PRODUCAO-ARTISTICA-CULTURAL")
    private DetalhamentoOutraProducaoArtisticaCultural detalhamento;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.outra.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoOutraProducaoArtisticaCultural {
    // INSTITUICAO-PROMOTORA-DO-EVENTO CDATA #IMPLIED
    // LOCAL-DO-EVENTO CDATA #IMPLIED
    // CIDADE CDATA #IMPLIED
    // EXPOSICAO CDATA #IMPLIED
    // PREMIACAO CDATA #IMPLIED

    @XmlAttribute(name = "INSTITUICAO-PROMOTORA-DO-EVENTO")
    protected String instituicaoPromotoraEvento;

    @XmlAttribute(name = "LOCAL-DO-EVENTO")
    protected String localEvento;

    @XmlAttribute(name = "CIDADE")
    protected String cidade;

    @XmlAttribute(name = "EXPOSICAO")
    protected String exposicao;

    @XmlAttribute(name = "PREMIACAO")
    protected String premio;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.musica.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.common.DadosBasicosOutraProducaoDivulgado;
import ufsc.alpc.xml.dto.domain.NaturezaMusica;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosMusica extends DadosBasicosOutraProducaoDivulgado {
    // NATUREZA (APRESENTACAO_DE_OBRA | ARRANJO | AUDIOVISUAL |

```

```

        ↪ COMPOSICAO |
// DIVERSAS | INTERPRETACAO | PUBLICACAO_DE_PARTITURA |
        ↪ REGISTRO_FONOGRAFICO
// | TRILHA_SONORA | OUTRA ) #IMPLIED
// TITULO CDATA #IMPLIED
// ANO CDATA #IMPLIED
// PAIS CDATA #IMPLIED
// IDIOMA CDATA #IMPLIED
// MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
        ↪ MEIO_DIGITAL |
// FILME | HIPERTEXTO
// | OUTRO | VARIOS | NAO_INFORMADO) "NAO_INFORMADO"
// HOME-PAGE CDATA #IMPLIED
// FLAG-RELEVANCIA (SIM | NAO) "NAO"
// TITULO-INGLES CDATA #IMPLIED
// FLAG-DIVULGACAO-CIENTIFICA (SIM | NAO) "NAO"

@XmlAttribute(name = "NATUREZA")
protected NaturezaMusica natureza;

@XmlAttribute(name = "IDIOMA")
protected String idioma;

@XmlJavaTypeAdapter(BooleanXmlAdapter.class)
@XmlAttribute(name = "FLAG-DIVULGACAO-CIENTIFICA")
protected Boolean flagDivulgacaoCientifica = Boolean.FALSE;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.musica;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.musica.dadosbasicos.DadosBasicosMusica;
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.musica.detalhamento.DetalhamentoMusica;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class MusicaXmlDto extends AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto {
    // DADOS-BASICOS-DA-MUSICA?
    // DETALHAMENTO-DA-MUSICA?
    // AUTORES*,
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlElement(name = "DADOS-BASICOS-DA-MUSICA")
    private DadosBasicosMusica dadosBasicos;

    @XmlElement(name = "DETALHAMENTO-DA-MUSICA")
    private DetalhamentoMusica detalhamento;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.musica.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoMusica {
    // TIPO-DE-EVENTO CDATA #IMPLIED

```

```

// ATIVIDADE-DOS-AUTORES CDATA #IMPLIED
// FORMACAO-INSTRUMENTAL CDATA #IMPLIED
// FLAG-INEDITISMO-DA-OBRA CDATA #IMPLIED
// DATA-ESTREIA CDATA #IMPLIED
// DATA-ENCERRAMENTO CDATA #IMPLIED
// LOCAL-DE-ESTREIA CDATA #IMPLIED
// PREMIACAO CDATA #IMPLIED
// INSTITUICAO-PROMOTORA-DO-PREMIO CDATA #IMPLIED
// OBRA-DE-REFERENCIA CDATA #IMPLIED
// AUTOR-DA-OBRA-DE-REFERENCIA CDATA #IMPLIED
// ANO-DA-OBRA-DE-REFERENCIA CDATA #IMPLIED
// DURACAO CDATA #IMPLIED
// TEMPORADA CDATA #IMPLIED
// INSTITUICAO-PROMOTORA-DO-EVENTO CDATA #IMPLIED
// LOCAL-DO-EVENTO CDATA #IMPLIED
// CIDADE-DO-EVENTO CDATA #IMPLIED

@XmlAttribute(name = "TIPO-DE-EVENTO")
protected String tipoEvento;

@XmlAttribute(name = "ATIVIDADE-DOS-AUTORES")
protected String atividadeAutores;

@XmlAttribute(name = "FORMACAO-INSTRUMENTAL")
protected String formacaoInstrumental;

@XmlAttribute(name = "FLAG-INEDITISMO-DA-OBRA")
protected String flagIneditismoObra;

@XmlAttribute(name = "DATA-ESTREIA")
protected String dataEstreia;

@XmlAttribute(name = "DATA-ENCERRAMENTO")
protected String dataEncerramento;

@XmlAttribute(name = "LOCAL-DE-ESTREIA")
protected String localEstreia;

@XmlAttribute(name = "PREMIACAO")
protected String premiacao;

@XmlAttribute(name = "INSTITUICAO-PROMOTORA-DO-PREMIO")
protected String instituicaoPromotoraPremio;

@XmlAttribute(name = "OBRA-DE-REFERENCIA")
protected String obraReferencia;

@XmlAttribute(name = "AUTOR-DA-OBRA-DE-REFERENCIA")
protected String autorObraReferencia;

@XmlAttribute(name = "ANO-DA-OBRA-DE-REFERENCIA")
protected String anoObraReferencia;

@XmlAttribute(name = "DURACAO")
protected String duracao;

@XmlAttribute(name = "TEMPORADA")
protected String temporada;

@XmlAttribute(name = "INSTITUICAO-PROMOTORA-DO-EVENTO")
protected String instituicaoPromotoraEvento;

@XmlAttribute(name = "LOCAL-DO-EVENTO")
protected String localEvento;

@XmlAttribute(name = "CIDADE-DO-EVENTO")
protected String cidadeEvento;
}
package ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import org.springframework.data.jpa.domain.support.AuditingEntityListener;

import ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.apresentacao.obra.artistica.ApresentacaoObraArtistica;
import

```

```

import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.apresentacao.radiotv.ApresentacaoRadioTvXm
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.arranjomusical.ArranjoMusicalXmlDto;
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.artes.cenicas.ArtesCenicasXmlDto;
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.artes.visuais.ArtesVisuaisXmlDto;
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.composicaomusical.ComposicaoMusicalXmlDto;
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.curso.curtaduracao.CursoCurtaDuracaoXmlDt
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.musica.MusicaXmlDto;
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.obra.artesvisuais.ObraArtesVisuaisXmlDto;
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.outra.OutraProducaoArtisticaCulturalXmlDt
import ↪ ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.sonoplastia.SonoplastiaXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class ProducaoArtisticaCulturalXmlDto {

    // APRESENTACAO-DE-OBRA-ARTISTICA*
    // APRESENTACAO-EM-RADIO-OU-TV*
    // ARRANJO-MUSICAL*
    // COMPOSICAO-MUSICAL*
    // CURSO-DE-CURTA-DURACAO*
    // OBRA-DE-ARTES-VISUAIS*
    // OUTRA-PRODUCAO-ARTISTICA-CULTURAL*
    // SONOPLASTIA*
    // ARTES-CENICAS*
    // ARTES-VISUAIS*
    // MUSICA*

    @XmlElement(name = "APRESENTACAO-DE-OBRA-ARTISTICA")
    private List<ApresentacaoObraArtisticaXmlDto> apresentacaoObraArtistica;

    @XmlElement(name = "APRESENTACAO-EM-RADIO-OU-TV")
    private List<ApresentacaoRadioTvXmlDto> apresentacaoRadioTv;

    @XmlElement(name = "ARRANJO-MUSICAL")
    private List<ArranjoMusicalXmlDto> arranjoMusical;

    @XmlElement(name = "COMPOSICAO-MUSICAL")
    private List<ComposicaoMusicalXmlDto> composicaoMusical;

    @XmlElement(name = "CURSO-DE-CURTA-DURACAO")
    private List<CursoCurtaDuracaoXmlDto> cursoCurtaDuracao;

    @XmlElement(name = "OBRA-DE-ARTES-VISUAIS")
    private List<ObraArtesVisuaisXmlDto> obraArtesVisuais;

    @XmlElement(name = "OUTRA-PRODUCAO-ARTISTICA-CULTURAL")
    private List<OutraProducaoArtisticaCulturalXmlDto> outraProducaoArtistica;

    @XmlElement(name = "SONOPLASTIA")
    private List<SonoplastiaXmlDto> sonoplastia;

    @XmlElement(name = "ARTES-CENICAS")
    private List<ArtesCenicasXmlDto> artesCenicas;

    @XmlElement(name = "ARTES-VISUAIS")
    private List<ArtesVisuaisXmlDto> artesVisuais;

    @XmlElement(name = "MUSICA")
    private List<MusicaXmlDto> musica;

}
package ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas;

import java.util.List;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

import org.apache.commons.collections.CollectionUtils;

```

```

import
↳ ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.doutorado.OrientacaoConcluidaDoutoradoXmlDto;
import
↳ ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.mestrado.OrientacaoConcluidaMestradoXmlDto;
import
↳ ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.outra.OutraOrientacaoConcluidaXmlDto;
import
↳ ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.posdoutorado.OrientacaoConcluidaPosDoutoradoXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OrientacoesConcluidasXmlDto {

    // ORIENTACOES-CONCLUIDAS-PARA-MESTRADO*
    // ORIENTACOES-CONCLUIDAS-PARA-DOCTORADO*
    // ORIENTACOES-CONCLUIDAS-PARA-POS-DOCTORADO*
    // OUTRAS-ORIENTACOES-CONCLUIDAS*

    @XmlElement(name = "ORIENTACOES-CONCLUIDAS-PARA-MESTRADO")
    private List<OrientacaoConcluidaMestradoXmlDto> orientacoesConcluidasMestrado;

    @XmlElement(name = "ORIENTACOES-CONCLUIDAS-PARA-DOCTORADO")
    private List<OrientacaoConcluidaDoutoradoXmlDto> orientacoesConcluidasDoutorado;

    @XmlElement(name = "ORIENTACOES-CONCLUIDAS-PARA-POS-DOCTORADO")
    private List<OrientacaoConcluidaPosDoutoradoXmlDto>
        ↳ orientacoesConcluidasPosDoutorado;

    @XmlElement(name = "OUTRAS-ORIENTACOES-CONCLUIDAS")
    private List<OutraOrientacaoConcluidaXmlDto> outrasOrientacoesConcluidas;

    public boolean isSetOrientacoesConcluidasMestrado() {
        return !CollectionUtils.isEmpty(this.orientacoesConcluidasMestrado);
    }

    public boolean isSetOrientacoesConcluidasDoutorado() {
        return !CollectionUtils.isEmpty(this.orientacoesConcluidasDoutorado);
    }

}
package ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.mestrado;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoTipoXmlDto;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoConcluidaXmlDto;
import ufsc.alpc.xml.dto.common.SequenciaPalavraAreaSetorInfoAgrupadorXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OrientacaoConcluidaMestradoXmlDto extends
↳ SequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

    // DADOS-BASICOS-DE-ORIENTACOES-CONCLUIDAS-PARA-MESTRADO?
    // DETALHAMENTO-DE-ORIENTACOES-CONCLUIDAS-PARA-MESTRADO?
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlElement(name =
        ↳ "DADOS-BASICOS-DE-ORIENTACOES-CONCLUIDAS-PARA-MESTRADO")
    private DadosBasicosDaParticipacaoTipoXmlDto dadosBasicos;

    @XmlElement(name =
        ↳ "DETALHAMENTO-DE-ORIENTACOES-CONCLUIDAS-PARA-MESTRADO")
    private DetalhamentoDaOrientacaoConcluidaXmlDto detalhamento;

}
package ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.doutorado;

```



```

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoFlagRelevanciaXmlDto;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoConcluidaXmlDto;
import ufsc.alpc.xml.dto.common.SequenciaPalavraAreaSetorInfoAgrupadorXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OrientacaoConcluidaDoutoradoXmlDto extends
    ↪ SequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

    // DADOS-BASICOS-DE-ORIENTACOES-CONCLUIDAS-PARA-DOCTORADO?
    // DETALHAMENTO-DE-ORIENTACOES-CONCLUIDAS-PARA-DOCTORADO?
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlElement(name =
        ↪ "DADOS-BASICOS-DE-ORIENTACOES-CONCLUIDAS-PARA-DOCTORADO")
    private DadosBasicosDaParticipacaoFlagRelevanciaXmlDto dadosBasicos;

    @XmlElement(name =
        ↪ "DETALHAMENTO-DE-ORIENTACOES-CONCLUIDAS-PARA-DOCTORADO")
    private DetalhamentoDaOrientacaoConcluidaXmlDto detalhamento;
}

package ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.posdoutorado;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoFlagRelevanciaXmlDto;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoConcluidaXmlDto;
import ufsc.alpc.xml.dto.common.SequenciaPalavraAreaSetorInfoAgrupadorXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OrientacaoConcluidaPosDoutoradoXmlDto extends
    ↪ SequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

    // ↪ DADOS-BASICOS-DE-ORIENTACOES-CONCLUIDAS-PARA-POS-DOCTORADO?
    // ↪ DETALHAMENTO-DE-ORIENTACOES-CONCLUIDAS-PARA-POS-DOCTORADO?
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlElement(name =
        ↪ "DADOS-BASICOS-DE-ORIENTACOES-CONCLUIDAS-PARA-POS-DOCTORADO")
    private DadosBasicosDaParticipacaoFlagRelevanciaXmlDto dadosBasicos;

    @XmlElement(name =
        ↪ "DETALHAMENTO-DE-ORIENTACOES-CONCLUIDAS-PARA-POS-DOCTORADO")
    private DetalhamentoDaOrientacaoConcluidaXmlDto detalhamento;
}

package ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.outra.dadosbasicos;

import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoTipoInglesXmlDto;

public class DadosBasicosOutrasOrientacoesConcluidasXmlDto extends

```

```

↳ DadosBasicosDaParticipacaoTipoInglesXmlDto {
    //
    // ↳ NATUREZA(MONOGRRAFIA_DE_CONCLUSAO_DE_CURSO_APERFEICOAMENTO_E_ESPECIALIZ
    // |TRABALHO_DE_CONCLUSAO_DE_CURSO_GRADUACAO |
    // ↳ INICIACAO_CIENTIFICA |
    // ORIENTACAO-DE-OUTRA-NATUREZA) #IMPLIED
    // TIPO CDATA #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // HOME-PAGE CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED
    // TIPO-INGLES CDATA #IMPLIED

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-RELEVANCIA")
    protected Boolean flagRelevancia = Boolean.FALSE;
}
package ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.outra;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoConcluidaXmlDto;
import ufsc.alpc.xml.dto.common.SequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.outra.dadosbasicos.DadosBasicosOutrasOrientacoesConcluidasXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class OutraOrientacaoConcluidaXmlDto extends
↳ SequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

    // DADOS-BASICOS-DE-OUTRAS-ORIENTACOES-CONCLUIDAS?
    // DETALHAMENTO-DE-OUTRAS-ORIENTACOES-CONCLUIDAS?
    // PALAVRAS-CHAVE?
    // AREAS-DO-CONHECIMENTO?
    // SETORES-DE-ATIVIDADE?
    // INFORMACOES-ADICIONAIS?
    // SEQUENCIA-PRODUCAO CDATA #IMPLIED

    @XmlElement(name =
↳ "DADOS-BASICOS-DE-OUTRAS-ORIENTACOES-CONCLUIDAS")
    private DadosBasicosOutrasOrientacoesConcluidasXmlDto dadosBasicos;

    @XmlElement(name =
↳ "DETALHAMENTO-DE-OUTRAS-ORIENTACOES-CONCLUIDAS")
    private DetalhamentoDaOrientacaoConcluidaXmlDto detalhamento;
}
package ufsc.alpc.xml.dto.outra.producao.demais.trabalhos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlElement;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.dto.common.AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto;
import ufsc.alpc.xml.dto.outra.producao.demais.trabalhos.dadosbasicos.DadosBasicosDemaisTrabalhosXmlDto;
import ufsc.alpc.xml.dto.outra.producao.demais.trabalhos.detalhamento.DetalhamentoDemaisTrabalhosXmlDto;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DemaisTrabalhosXmlDto extends
↳ AutoresSequenciaPalavraAreaSetorInfoAgrupadorXmlDto {

```

```

// DADOS-BASICOS-DE-DEMAIS-TRABALHOS?
// DETALHAMENTO-DE-DEMAIS-TRABALHOS?
// AUTORES+
// PALAVRAS-CHAVE?
// AREAS-DO-CONHECIMENTO?
// SETORES-DE-ATIVIDADE?
// INFORMACOES-ADICIONAIS?
// SEQUENCIA-PRODUCAO CDATA #IMPLIED

@XmlElement(name = "DADOS-BASICOS-DE-DEMAIS-TRABALHOS")
private DadosBasicosDemaisTrabalhosXmlDto dadosBasicos;

@XmlElement(name = "DETALHAMENTO-DE-DEMAIS-TRABALHOS")
private DetalhamentoDemaisTrabalhosXmlDto detalhamento;

}package ufsc.alpc.xml.dto.outra.producao.demais.trabalhos.dadosbasicos;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;
import javax.xml.bind.annotation.adapters.XmlJavaTypeAdapter;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;
import ufsc.alpc.xml.adapter.BooleanXmlAdapter;
import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoXmlDto;
import ufsc.alpc.xml.dto.domain.MedioDeDivulgacao;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DadosBasicosDemaisTrabalhosXmlDto extends DadosBasicosDaParticipacaoXmlDto {

    // NATUREZA CDATA #IMPLIED
    // TITULO CDATA #IMPLIED
    // ANO CDATA #IMPLIED
    // PAIS CDATA #IMPLIED
    // IDIOMA CDATA #IMPLIED
    // MEIO-DE-DIVULGACAO (IMPRESSO | WEB | MEIO_MAGNETICO |
    // ↔ MEIO_DIGITAL |
    // FILME | HIPERTEXTO | OUTRO | VARIOS | NAO_INFORMADO)
    // ↔ "NAO_INFORMADO"
    // HOME-PAGE CDATA #IMPLIED
    // FLAG-RELEVANCIA (SIM | NAO) "NAO"
    // DOI CDATA #IMPLIED
    // TITULO-INGLES CDATA #IMPLIED
    // NATUREZA-INGLES CDATA// #IMPLIED

    @XmlAttribute(name = "MEIO-DE-DIVULGACAO")
    protected MedioDeDivulgacao meioDeDivulgacao;

    @XmlJavaTypeAdapter(BooleanXmlAdapter.class)
    @XmlAttribute(name = "FLAG-RELEVANCIA")
    protected Boolean flagRelevancia = Boolean.FALSE;

    @XmlAttribute(name = "NATUREZA-INGLES")
    protected String naturezaIngles;

}
package ufsc.alpc.xml.dto.outra.producao.demais.trabalhos.detalhamento;

import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlAttribute;

import lombok.Getter;
import lombok.Setter;
import lombok.ToString;

@Getter
@Setter
@ToString(callSuper = true)
@XmlAccessorType(XmlAccessType.FIELD)
public class DetalhamentoDemaisTrabalhosXmlDto {

    // FINALIDADE CDATA #IMPLIED
    // LOCAL CDATA #IMPLIED

    @XmlAttribute(name = "FINALIDADE")
    protected String finalidade;

```

```

        @XmlAttribute(name = "LOCAL")
        protected String local;
    }
    package ufsc.alpc.xml.dto.outra.producao;

    import java.util.List;

    import javax.xml.bind.annotation.XmlAccessType;
    import javax.xml.bind.annotation.XmlAccessorType;
    import javax.xml.bind.annotation.XmlElement;

    import lombok.Getter;
    import lombok.Setter;
    import lombok.ToString;

    import org.apache.commons.collections.CollectionUtils;

    import ufsc.alpc.xml.dto.outra.producao.demais.trabalhos.DemaisTrabalhosXmlDto;
    import ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.OrientacoesConcluidasXmlDto;
    import ufsc.alpc.xml.dto.outra.producao.producao.artistica.cultural.ProducaoArtisticaCulturalXmlDto;

    @Getter
    @Setter
    @ToString(callSuper = true)
    @XmlAccessorType(XmlAccessType.FIELD)
    public class OutraProducaoXmlDto {

        // PRODUCAO-ARTISTICA-CULTURAL*
        // ORIENTACOES-CONCLUIDAS*,
        // DEMAIS-TRABALHOS*

        @XmlElement(name = "PRODUCAO-ARTISTICA-CULTURAL")
        private List<ProducaoArtisticaCulturalXmlDto> producaoArtisticaCultural;

        @XmlElement(name = "ORIENTACOES-CONCLUIDAS")
        private List<OrientacoesConcluidasXmlDto> orientacoesConcluidas;

        @XmlElement(name = "DEMAIS-TRABALHOS")
        private List<DemaisTrabalhosXmlDto> demaisTrabalhos;

        public boolean isSetOrientacoesConcluidas() {
            return !CollectionUtils.isEmpty(this.orientacoesConcluidas);
        }
    }
    package ufsc.alpc.xml.dto.domain;

    public enum FormatoCargoFuncao {

        CHEFE_DE_DEPARTAMENTO, COORDENADOR_DE_CURSO,
        ↳ COORDENADOR_DE_PROGRAMA, DECAÑO_DE_CENTRO,
        ↳ DIRETOR_DE_UNIDADE, MEMBRO_DE_COLEGIADO_SUPERIOR,
        ↳ MEMBRO_DE_COMISSAO_PERMANENTE,
        ↳ MEMBRO_DE_COMISSAO_TEMPORARIA,
        ↳ MEMBRO_DE_CONSELHO_DE_CENTRO,
        ↳ MEMBRO_DE_CONSELHO_DE_UNIDADE, REITOR,
        ↳ VICE_REITOR_OU_PRO_REITOR, OUTRO, LIVRE;

    }
    package ufsc.alpc.xml.dto.domain;

    public enum TipoLivro {

        LIVRO_PUBLICADO, LIVRO_ORGANIZADO_OU_EDICAO, NAO_INFORMADO;

    }
    package ufsc.alpc.xml.dto.domain;

    public enum NaturezaManutencaoObraArtistica {

        ARQUITETURA,
        DESENHO,
        ESCULTURA,
        FOTOGRAFIA,
        GRAVURA,
        OUTRA,
        PINTURA,
        NAO_INFORMADO

    }
}

```

```

package ufsc.alpc.xml.dto.domain;

public enum NivelCursoMinistrado {

    EXTENSAO,
    APERFEICOAMENTO,
    ESPECIALIZACAO,
    OUTRA, NAO_INFORMADO

}

package ufsc.alpc.xml.dto.domain;

public enum TipoEventoMusical {

    CONCERTO, CONCURSO, FESTIVAL, GRAVACAO, RECITAL, OUTRO,
    ↪ NAO_INFORMADO;

}

package ufsc.alpc.xml.dto.domain;

public enum NivelCurso {

    APERFEICOAMENTO_ESPECIALIZACAO,
    APERFEICOAMENTO,
    CURSO_DE_CURTA_DURACAO,
    ESPECIALIZACAO,
    ESTAGIO,
    EXTENSAO_UNIVERSITARIA,
    DOUTORADO,
    GRADUACAO,
    MESTRADO,
    MESTRADO_DOUTORADO,
    OT,
    OUTRO;

}

package ufsc.alpc.xml.dto.domain;

public enum SituacaoProjeto {

    EM_ANDAMENTO, DESATIVADO, CONCLUIDO;

}

package ufsc.alpc.xml.dto.domain;

public enum NaturezaLivro {

    COLETANEA, TEXTO_INTEGRAL, VERBETE, ANAIS, CATALOGO, ENCICLOPEDIA,
    ↪ LIVRO, OUTRA, PERIODICO, NAO_INFORMADO;

}

package ufsc.alpc.xml.dto.domain;

public enum Disponibilidade {

    RESTRITA,
    IRRESTRITA,
    NAO_INFORMADO;

}

package ufsc.alpc.xml.dto.domain;

public enum Acervo {

    PUBLICO, PRIVADO, NAO_INFORMADO;

}

package ufsc.alpc.xml.dto.domain;

public enum EnquadramentoFuncional {

    PROFESSOR_TITULAR, OUTRO, LIVRE;

}

package ufsc.alpc.xml.dto.domain;

public enum NaturezaMusica {

    APRESENTACAO_DE_OBRA, ARRANJO, AUDIOVISUAL, COMPOSICAO, DIVERSAS,
    ↪ INTERPRETACAO, PUBLICACAO_DE_PARTITURA,
    ↪ REGISTRO_FONOGRAFICO, TRILHA_SONORA, OUTRA;

}

package ufsc.alpc.xml.dto.domain;

public enum TipoEventoArteVisual {

    APRESENTACAO, CONCURSO, CRIACAO, EXPOSICAO, FESTIVAL, OUTRO,
    ↪ NAO_INFORMADO;

}

```

```

}
package ufsc.alpc.xml.dto.domain;

import lombok.AllArgsConstructor;
import lombok.Getter;

@Getter
@AllArgsConstructor
public enum StatusCurso {

    EM_ANDAMENTO("Em andamento"),
    CONCLUIDO("Concluído"),
    INCOMPLETO("Incompleto");

    private String descricao;
}
package ufsc.alpc.xml.dto.domain;

public enum NaturezaFinanciador {

    BOLSA, AUXILIO_FINANCEIRO, REMUNERACAO, OUTRO, COOPERACAO,
    ↪ NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;

public enum NaturezaObraArtesVisuais {

    CINEMA, DESENHO, ESCULTURA, FOTOGRAFIA, GRAVURA, INSTALACAO,
    ↪ PINTURA, TELEVISAO, VIDEO, OUTRA;
}
package ufsc.alpc.xml.dto.domain;

public enum ParticipacaoAutores {

    DOCENTE,
    ORGANIZADOR,
    OUTRA, NAO_INFORMADO
}
package ufsc.alpc.xml.dto.domain;

public enum NaturezaArtesVisuais {

    INTERVENCAO_URBANA, LIVRO_DE_ARTISTA, PERFORMANCE, PINTURA,
    ↪ PROGRAMACAO_VISUAL, VIDEO, WEBART, ANIMACAO, INSTALACAO,
    ↪ COMPUTACAO_GRAFICA, DESENHO, DIVERSAS, ESCULTURA, FILME,
    ↪ FOTOGRAFIA, GRAVURA, ILUSTRACAO, OUTRA;
}
package ufsc.alpc.xml.dto.domain;

public enum NaturezaRadioTv {

    DANCA , MUSICA , TEATRO , OUTRA , NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;

public enum NaturezaMidiaSocialWebsite {

    REDE_SOCIAL,
    FORUM,
    BLOG, SITE
}
package ufsc.alpc.xml.dto.domain;

public enum Tipo {

    ACADEMICO,
    PROFISSIONALIZANTE,
    NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;

public enum NaturezaSonoplastia {

    CINEMA, MUSICA, RADIO, TELEVISAO, TEATRO, OUTRA;
}
}

```

```

package ufsc.alpc.xml.dto.domain;
public enum NaturezaProgramaRadioTv {
    ENTREVISTA,
    MESA_REDONDA,
    COMENTARIO,
    PROGRAMA,
    OUTRA, NAO_INFORMADO
}
package ufsc.alpc.xml.dto.domain;
public enum NaturezaOrganizacaoEvento {
    CURADORIA,
    MONTAGEM,
    MUSEOLOGIA,
    ORGANIZACAO, NAO_INFORMADO
}
package ufsc.alpc.xml.dto.domain;
public enum Proficiencia {
    POUCO, RAZOAVELMENTE, BEM, NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;
public enum NaturezaTexto {
    JORNAL_DE_NOTICIAS, REVISTA_MAGAZINE, NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;
public enum NaturezaArtesCenicas {
    AUDIOVISUAL, CIRCENSE, COREOGRAFICA, DIVERSAS, OPERISTICA,
    ↪ PERFORMATICA, RADIALISTICA, TEATRAL, OUTRA;
}
package ufsc.alpc.xml.dto.domain;
public enum Sexo {
    MASCULINO, FEMININO;
}
package ufsc.alpc.xml.dto.domain;
public enum NaturezaArtigo {
    COMPLETO, RESUMO, NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;
public enum TipoManutencaoObraArtistica {
    CONSERVACAO,
    RESTAURACAO,
    OUTRO, NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;
public enum NaturezaProjeto {
    DESENVOLVIMENTO, EXTENSAO, PESQUISA, OUTRA;
}
package ufsc.alpc.xml.dto.domain;
public enum TipoOrganizacaoEvento {
    CONCERTO,
    CONCURSO,
    CONGRESSO,
    EXPOSICAO,
    FESTIVAL,
    FEIRA,
    OLIMPADA,
    OUTRO,
    NAO_INFORMADO
}

```

```

}
package ufsc.alpc.xml.dto.domain;

public enum NaturezaDoProcessosTecnincas {
    ANALITICA,
    INSTRUMENTAL,
    PEDAGOGICA,
    PROCESSUAL,
    TERAPEUTICA,
    OUTRA,
    NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;

public enum NaturezaTraducao {
    ARTIGO, LIVRO, OUTRO, NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;

public enum Unidade {
    SEMANAS, DIAS, HORAS, NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;

public enum NaturezaDoTrabalhoTecnico {
    ASSESSORIA,
    CONSULTORIA,
    PARECER,
    ELABORACAO_DE_PROJETO,
    RELATORIO_TECNICO,
    SERVICOS_NA_AREA_DA_SAUDE,
    OUTRA,
    NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;

public enum MeioDeDivulgacao {
    IMPRESSO,
    WEB,
    MEIO_MAGNETICO,
    MEIO_DIGITAL,
    FILME,
    HIPERTEXTO,
    OUTRO,
    VARIOS,
    NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;

public enum TipoVinculo {
    SERVIDOR_PUBLICO_OU_CELETISTA, SERVIDOR_PUBLICO, CELETISTA,
    ↪ PROFESSOR_VISITANTE, COLABORADOR, BOLSISTA_RECEM_DOUTOR,
    ↪ OUTRO, LIVRE;
}
package ufsc.alpc.xml.dto.domain;

public enum NaturezaTrabalho {
    COMPLETO, RESUMO, RESUMO_EXPANDIDO;
}
package ufsc.alpc.xml.dto.domain;

public enum ClassificacaoEvento {
    INTERNACIONAL, NACIONAL, REGIONAL, LOCAL, NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;

public enum NaturezaApresentacaoTrabalho {
    COMUNICACAO,
    CONFERENCIA,
    CONGRESSO,
    SEMINARIO,
    SIMPOSIO,
    OUTRA, NAO_INFORMADO
}

```



```

}
package ufsc.alpc.xml.dto.domain;
public enum NaturezaPrefacioPosfacio {
    LIVRO, OUTRA, REVISTAS_OU_PERIODICOS, NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;
public enum AtividadeAutores {
    CANTO, CRIACAO, DANCA, DIRECAO, ENCENACAO, INSTRUMENTO_MUSICAL,
    ↪ REGENCIA, ROTEIRO, OUTRA, VARIAS, NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;
public enum NaturezaObraArtistica {
    COREOGRAFICA, LITERARIA, MUSICAL, TEATRAL, OUTRA, NAO_INFORMADO;
}
package ufsc.alpc.xml.dto.domain;
public enum TipoDePatente {
    PRIVILEGIO_DE_INOVACAO_PI,
    MODELO_DE_UTILIDADE_MU,
    DESENHO_INDUSTRIAL_DI,
    MODELO_INDUSTRIAL_MI,
    PATENTE_NO_EXTERIOR_PE,
    PROGRAMA_DE_COMPUTADOR_PC,
    OUTRO_OU,
    OUTRO_Ou,
    CERTIFICADO_DE_ADICAO_CA,
    CULTIVAR_PROTEGIDA_CTV,
    MARCA_REGISTRADA_DE_PRODUTO_MPD,
    MARCA_REGISTRADA_DE_SERVICO_MSV,
    MARCA_REGISTRADA_COLETIVA_MCL,
    MARCA_REGISTRADA_DE_CERTIFICACAO_MCT,
    TOPOGRAFIA_DE_CIRCUITO_INTEGRADO_REGISTRADA_TCI,
    MARCA_REGISTRADA_FIGURATIVA_MRF,
    MARCA_REGISTRADA_NOMINATIVA_MRN,
    MARCA_REGISTRADA_MISTA_MRM,
    MARCA_REGISTRADA_TRIDIMENSIONAL_MRT;
}
package ufsc.alpc.xml.dto.domain;
public enum NaturezaCartaOuSimilar {
    AEROFOTOGRAMA,
    CARTA,
    FOTOGRAMA,
    MAPA,
    OUTRA, NAO_INFORMADO
}
package ufsc.alpc.xml.dto.domain;
public enum TipoPrefacioPosfacio {
    PREFACIO, POSFACIO, APRESENTACAO, INTRODUCAO;
}
package ufsc.alpc.xml.dto.domain;
import lombok.AllArgsConstructor;
import lombok.Getter;

@Getter
@AllArgsConstructor
public enum TipoDeOrientacao {
    ORIENTADOR_PRINCIPAL("Orientao principal"),
    CO_ORIENTADOR("Coorientador");

    private String descricao;
}
package ufsc.alpc.xml.dto.domain;
public enum NaturezaDaPatente {
    APARELHO,
    EQUIPAMENTO,
    FARMACOS_E_SIMILARES,

```

```

        INSTRUMENTO,
        OUTRA,
        NAO_INFORMADO;
    }
    package ufsc.alpc.xml.dto.domain;

    public enum SimNao {

        SIM, NAO;

    }
    package ufsc.alpc.xml.dto.domain;

    public enum NaturezaArranjoMusical {

        CANTO, CORAL, ORQUESTRA, OUTRA, NAO_INFORMADO;

    }
    package ufsc.alpc.xml.dto.domain;

    import lombok.AllArgsConstructor;
    import lombok.Getter;

    @Getter
    @AllArgsConstructor
    public enum GrandeAreaDoConhecimento {
        CIENCIAS_AGRARIAS(1L, "Cincias agrarias"),
        CIENCIAS_BIOLOGICAS(2L, "Cincias biologicas"),
        CIENCIAS_DA_SAUDE(3L, "Cincias da sade"),
        CIENCIAS_EXATAS_E_DA_TERRA(4L, "Cincias exatas e da terra"),
        CIENCIAS_HUMANAS(5L, "Cincias humanas"),
        CIENCIAS_SOCIAIS_APLICADAS(6L, "Cincias sociais aplicadas"),
        ENGENHARIAS(7L, "Engenharias"),
        LINGUISTICA_LETRAS_E_ARTES(8L, "Lingustica letras e artes"),
        NAO_INFORMADO(9L, "No informado"),
        OUTROS(10L, "Outros");

        private Long id;
        private String descricao;

        public static GrandeAreaDoConhecimento getById(Long id) {
            for (GrandeAreaDoConhecimento area : GrandeAreaDoConhecimento.values()) {
                if (area.getId().equals(id)) {
                    return area;
                }
            }
            return null;
        }
    }
    package ufsc.alpc.xml.dto.domain;

    public enum TipoProduto {

        PILOTO,
        PROJETO,
        PROTOTIPO,
        OUTRO,
        NAO_INFORMADO;

    }
    package ufsc.alpc.xml.dto.domain;

    public enum NaturezaCursoCurtaDuracao {

        EXTENSAO, APERFEICOAMENTO, ESPECIALIZACAO, OUTRA, NAO_INFORMADO;

    }
    package ufsc.alpc.xml.dto.domain;

    public enum NaturezaDoSoftware {

        COMPUTACIONAL,
        MULTIMIDIA,
        OUTRO,
        NAO_INFORMADO;

    }
    package ufsc.alpc.xml.adapter;

    import java.text.SimpleDateFormat;
    import java.util.Date;

    import javax.xml.bind.annotation.adapters.XmlAdapter;

    public class DateXmlAdapter extends XmlAdapter<String, Date> {

```

```

@Override
public Date unmarshal(String v) throws Exception {
    if (v != null && !v.isEmpty()) {
        return new SimpleDateFormat("ddMMyyyy").parse(v);
    }
    return null;
}

@Override
public String marshal(Date v) throws Exception {
    if (v != null) {
        return new SimpleDateFormat("ddMMyyyy").format(v);
    }
    return null;
}
}

package ufsc.alpc.xml.adapter;

import javax.xml.bind.annotation.adapters.XmlAdapter;
import org.apache.commons.lang3.StringUtils;

public class PeriodoXmlAdapter extends XmlAdapter<String, Boolean> {

    @Override
    public Boolean unmarshal(String v) throws Exception {
        if (!StringUtils.isBlank(v)) {
            return v.equals("ANTERIOR") ? false : v.equals("ATUAL") ? true : null;
        }
        return null;
    }

    @Override
    public String marshal(Boolean v) throws Exception {
        return v == null ? null : v ? "ANTERIOR" : "ATUAL";
    }
}

package ufsc.alpc.xml.adapter;

import javax.xml.bind.annotation.adapters.XmlAdapter;
import org.apache.commons.lang3.StringUtils;

public class EnderecoXmlAdapter extends XmlAdapter<String, Boolean> {

    @Override
    public Boolean unmarshal(String v) throws Exception {
        if (!StringUtils.isBlank(v)) {
            return v.equals("ENDERECO_INSTITUCIONAL") ? true :
                ↪ v.equals("ENDERECO_RESIDENCIAL") ? false : null;
        }
        return null;
    }

    @Override
    public String marshal(Boolean v) throws Exception {
        return v == null ? null : v ? "ENDERECO_INSTITUCIONAL" :
            ↪ "ENDERECO_RESIDENCIAL";
    }
}

package ufsc.alpc.xml.adapter;

import javax.xml.bind.annotation.adapters.XmlAdapter;
import org.apache.commons.lang3.StringUtils;

public class BooleanXmlAdapter extends XmlAdapter<String, Boolean> {

    @Override
    public Boolean unmarshal(String v) throws Exception {
        if (!StringUtils.isBlank(v)) {
            return v.equals("SIM") ? true : v.equals("NAO") ? false : null;
        }
        return null;
    }

    @Override
    public String marshal(Boolean v) throws Exception {
        return v == null ? null : v ? "SIM" : "NAO";
    }
}

```

```

    }
}
package alpc.ufsc;

import org.springframework.boot.builder.SpringApplicationBuilder;
import org.springframework.boot.context.web.SpringBootServletInitializer;

import alpc.ufsc.Application;

public class SpringBootWebStarter extends SpringBootServletInitializer {

    @Override
    protected SpringApplicationBuilder configure(SpringApplicationBuilder application) {
        return application.sources(Application.class);
    }
}

package alpc.ufsc.entity.programa;

import java.util.Date;
import java.util.List;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.OneToMany;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import org.hibernate.search.annotations.Analyze;
import org.hibernate.search.annotations.Field;
import org.hibernate.search.annotations.Store;

import alpc.ufsc.entity.domain.instituicao.InstituicaoEntity;
import alpc.ufsc.entity.login.user.ProgramaUserEntity;
import alpc.ufsc.entity.util.QueryUtils;

@Entity
@Getter
@Setter
//@formatter:off
@Table(name = "tb_programa", indexes = {
    @Index(name="in_programa_nome_filtro", columnList = "nome_filtro"),
    @Index(name="in_programa_data_inicio", columnList = "data_inicio"),
    @Index(name = "in_programa_id_instituicao", columnList = "id_instituicao")})
//@formatter:on
@NoArgsConstructor
public class ProgramaEntity {

    private static final String sequenceName = "sq_programa";

    public static final Long DEFAULT = 1L;

    @Id
    @Field(analyze = Analyze.NO, store = Store.YES)
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName =
        ↳ ProgramaEntity.sequenceName, name = ProgramaEntity.sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator =
        ↳ ProgramaEntity.sequenceName)
    @Column(name = "id_programa")
    private Long id;

    @Column(name = "nome", length = 400, nullable = false)
    private String nome;

    @Column(name = "nome_filtro", length = 400, nullable = false)
    private String nomeFiltro;

    @Temporal(value = TemporalType.DATE)

```

```

@Column(name = "data_inicio", nullable = false)
private Date dataInicio;

@ManyToOne
@JoinColumn(name = "id_instituicao", foreignKey = @ForeignKey(name =
    ↪ "fk_programa_instituicao"), nullable = false)
private InstituicaoEntity instituicao;

@OneToMany(mappedBy = "programa", fetch = FetchType.LAZY)
private List<ProgramaUserEntity> users;

public void setNome(String nome) {
    this.nome = nome;
    this.nomeFiltro = QueryUtils.lowerCaseStripAccents(nome);
}
}

package alpc.ufsc.entity.pessoa.producao;

import java.io.Serializable;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.IndexColumn;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.Setter;

import alpc.ufsc.entity.pessoa.docente.DocenteEntity;

@Entity
@Getter
@Setter
@Table(name = "rl_docente_producao", indexes = {
    @Index(name = "in_docente_producao_id_docente", columnList = "id_docente"),
    @Index(name = "in_docente_producao_id_producao", columnList =
        ↪ "id_producao")
})
public class DocenteProducaoEntity implements Serializable {

    private static final long serialVersionUID = -807000358386362097L;

    private static final String sequenceName = "sq_docente_producao";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName = sequenceName,
        ↪ name = sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator = sequenceName)
    @Column(name = "id_docente_producao")
    private Long id;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_docente", foreignKey = @ForeignKey(name =
        ↪ "fk_docenteproducao_docente"), nullable = false)
    private DocenteEntity docente;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_producao", foreignKey = @ForeignKey(name =
        ↪ "fk_docenteproducao_producao"), nullable = false)
    private ProducaoEntity producao;
}

package alpc.ufsc.entity.pessoa.producao;

import java.io.Serializable;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

```

```

import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import org.hibernate.search.annotations.Analyze;
import org.hibernate.search.annotations.Analyzer;
import org.hibernate.search.annotations.Field;
import org.hibernate.search.annotations.Index;
import org.hibernate.search.annotations.IndexedEmbedded;
import org.hibernate.search.annotations.Store;

import alpc.ufsc.entity.programa.ProgramaEntity;

@Entity
@Getter
@Setter
@Table(name = "tb_producao", indexes = {
    @javax.persistence.Index(name = "in_producao_tipo_producao", columnList =
        ↳ "tipo_producao"),
    @javax.persistence.Index(name = "in_producao_id_programa", columnList =
        ↳ "id_programa")
})
@NoArgsConstructor
public class ProducaoEntity implements Serializable {

    private static final long serialVersionUID = -2316853521107707372L;

    private static final String sequenceName = "sq_producao";

    @Id
    @Field(index = Index.NO, store = Store.YES)
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName =
        ↳ ProducaoEntity.sequenceName, name = ProducaoEntity.sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator =
        ↳ ProducaoEntity.sequenceName)
    @Column(name = "id_producao")
    private Long id;

    @Field(name = "titulo", analyzer = @Analyzer(definition = "ngram-titulo"), store =
        ↳ Store.YES)
    @Column(name = "titulo", length = 400, nullable = false)
    private String tituloDaProducao;

    @Field(name = "ano", analyze = Analyze.NO)
    @Column(name = "ano")
    private Integer ano;

    @Field(name = "tipo_producao", analyze = Analyze.NO)
    @Enumerated(EnumType.STRING)
    @Column(name = "tipo_producao")
    private TipoProducao tipoProducao;

    @IndexedEmbedded
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_programa", foreignKey = @ForeignKey(name =
        ↳ "fk_producao_programa"), nullable = false)
    private ProgramaEntity programa;
}

package alpc.ufsc.entity.pessoa.producao;

import java.util.Arrays;
import java.util.List;

public enum TipoProducao {

    // Bibliográfica
    ARTIGO_PUBLICADO,
    ARTIGO_ACEITO,
    TRABALHO_EM_EVENTO,
    LIVRO_PUBLICADO_OU_ORGANIZADO,
    CAPITULO_DE_LIVRO,
    TRADUÇÃO,
    PARTITURA,
    PREFACIO_POSFACIO,
    TEXTO_JORNAL_OU_REVISTA,
    OUTRA_PRODUCÃO_BIBLIOGRAFICA,
}

```

```

// Tecnica
CULTIVAR_REGISTRADA,
CULTIVAR_PROTEGIDA,
DESENHO_INDUSTRIAL,
MARCA,
PATENTE,
PROCESSO_TECNICA,
PRODUTO_TECNOLOGICO,
SOFTWARE,
TOPOGRAFIA_CIRCUITO_INTEGRADO,
TRABALHO_TECNICO,
APRESENTACAO_TRABALHO,
CARTA_MAPA_SIMILAR,
CURSO_CURTA_DURACAO,
DESENVOLVIMENTO_MATERIAL_DIDATICO_INSTRUCIONAL,
EDITORACAO,
MANUTENCAO_OBRA_ARTISTICA,
MAQUETE,
MIDIA_SOCIAL_WEBSITE_BLOG,
ORGANIZACAO_EVENTO,
OUTRA_PRODUCAO_TECNICA,
PROGRAMA_RADIO_TV,
RELATORIO_PESQUISA;

public static List<String> getProducaoBibliografica() {
    return Arrays.asList(
        ARTIGO_PUBLICADO.name(),
        ARTIGO_ACEITO.name(),
        TRABALHO_EM_EVENTO.name(),
        LIVRO_PUBLICADO_OU_ORGANIZADO.name(),
        CAPITULO_DE_LIVRO.name(),
        TRADUCAO.name(),
        PARTITURA.name(),
        PREFACIO_POSFACIO.name(),
        TEXTO_JORNAL_OU_REVISTA.name(),
        OUTRA_PRODUCAO_BIBLIOGRAFICA.name());
}

public static List<String> getProducaoTecnica() {
    return Arrays.asList(
        CULTIVAR_REGISTRADA.name(),
        CULTIVAR_PROTEGIDA.name(),
        DESENHO_INDUSTRIAL.name(),
        MARCA.name(),
        PATENTE.name(),
        PROCESSO_TECNICA.name(),
        PRODUTO_TECNOLOGICO.name(),
        SOFTWARE.name(),
        TOPOGRAFIA_CIRCUITO_INTEGRADO.name(),
        TRABALHO_TECNICO.name(),
        APRESENTACAO_TRABALHO.name(),
        CARTA_MAPA_SIMILAR.name(),
        CURSO_CURTA_DURACAO.name(),
        DESENVOLVIMENTO_MATERIAL_DIDATICO_INSTRUCIONAL.name(),
        EDITORACAO.name(),
        MANUTENCAO_OBRA_ARTISTICA.name(),
        MAQUETE.name(),
        MIDIA_SOCIAL_WEBSITE_BLOG.name(),
        ORGANIZACAO_EVENTO.name(),
        OUTRA_PRODUCAO_TECNICA.name(),
        PROGRAMA_RADIO_TV.name(),
        RELATORIO_PESQUISA.name());
}
}

package alpc.ufsc.entity.pessoa.producao;

import java.io.Serializable;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

```

```

import lombok.Getter;
import lombok.Setter;

import org.hibernate.search.annotations.Indexed;
import org.hibernate.search.annotations.IndexedEmbedded;

import alpc.ufsc.entity.pessoa.AutorEntity;

@Entity
@Getter
@Setter
@Indexed(index = "autores_producoes")
@Table(name = "rl_autor_producao", indexes = {
    @Index(name = "in_autor_producao_id_autor", columnList = "id_autor"),
    @Index(name = "in_autor_producao_id_producao", columnList = "id_producao")
})
public class AutorProducaoEntity implements Serializable {

    private static final long serialVersionUID = -807000358386362097L;

    private static final String sequenceName = "sq_autor_producao";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName = sequenceName,
        ↪ name = sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator = sequenceName)
    @Column(name = "id_autor_producao")
    private Long id;

    @IndexedEmbedded
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_autor", foreignKey = @ForeignKey(name =
        ↪ "fk_autorproducao_autor"), nullable = false)
    private AutorEntity autor;

    @IndexedEmbedded
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_producao", foreignKey = @ForeignKey(name =
        ↪ "fk_autorproducao_producao"), nullable = false)
    private ProducaoEntity producao;
}

package alpc.ufsc.entity.pessoa.orientacao.mestrado;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import org.apache.solr.analysis.BrazilianStemFilterFactory;
import org.apache.solr.analysis.LowerCaseFilterFactory;
import org.apache.solr.analysis.NGramFilterFactory;
import org.apache.solr.analysis.StopFilterFactory;
import org.apache.solr.analysis.WhitespaceTokenizerFactory;
import org.hibernate.search.annotations.Analyze;
import org.hibernate.search.annotations.Analyzer;
import org.hibernate.search.annotations.AnalyzerDef;
import org.hibernate.search.annotations.Field;
import org.hibernate.search.annotations.Indexed;
import org.hibernate.search.annotations.IndexedEmbedded;
import org.hibernate.search.annotations.Parameter;
import org.hibernate.search.annotations.Store;
import org.hibernate.search.annotations.TokenFilterDef;
import org.hibernate.search.annotations.TokenizerDef;

import alpc.ufsc.entity.domain.StatusOrientacao;
import alpc.ufsc.entity.domain.area.AreaConhecimentoEntity;
import alpc.ufsc.entity.domain.instituicao.InstituicaoEntity;
import alpc.ufsc.entity.pessoa.discente.DiscenteEntity;

```



```

import alpc.ufsc.entity.pessoa.docente.DocenteEntity;
import alpc.ufsc.entity.pessoa.orientacao.BaseOrientacaoEntity;
import alpc.ufsc.entity.programa.ProgramaEntity;
import alpc.ufsc.entity.util.QueryUtils;

@Entity
@Getter
@Setter
//@@formatter:off
@Table(name = "tb_orientacao_mestrado", indexes = {
    @Index(name = "in_mestrado_id_orientador", columnList = "id_orientador"),
    @Index(name = "in_mestrado_id_coorientador", columnList = "id_coorientador"),
    @Index(name = "in_mestrado_id_discente", columnList = "id_discente"),
    @Index(name = "in_mestrado_titulo_do_trabalho_filtro", columnList =
        ↳ "titulo_do_trabalho_filtro"),
    @Index(name = "in_mestrado_id_area_conhecimento_um", columnList =
        ↳ "id_area_conhecimento_um"),
    @Index(name = "in_mestrado_id_area_conhecimento_dois", columnList =
        ↳ "id_area_conhecimento_dois"),
    @Index(name = "in_mestrado_id_area_conhecimento_tres", columnList =
        ↳ "id_area_conhecimento_tres"),
    @Index(name = "in_mestrado_id_instituicao", columnList = "id_instituicao"),
    @Index(name = "in_mestrado_id_programa_deletado", columnList =
        ↳ "id_programa_deletado"),
    @Index(name = "in_mestrado_status_orientacao", columnList =
        ↳ "status_orientacao"))
//@@formatter:on
@NoArgsConstructor
@Indexed(index = "mestrados")
@AnalyzerDef(name = "ngram-titulo", tokenizer = @TokenizerDef(factory =
    ↳ WhitespaceTokenizerFactory.class), filters = {
    @TokenFilterDef(factory = StopFilterFactory.class, params = { @Parameter(name
        ↳ = "words", value = "stopwords.txt"), @Parameter(name = "ignoreCase",
        ↳ value = "true") }),
    @TokenFilterDef(factory = BrazilianStemFilterFactory.class),
        ↳ @TokenFilterDef(factory = LowerCaseFilterFactory.class),
    @TokenFilterDef(factory = NGramFilterFactory.class, params = {
        ↳ @Parameter(name = "minGramSize", value = "4"), @Parameter(name =
        ↳ "maxGramSize", value = "4") }) })
public class OrientacaoMestradoEntity implements BaseOrientacaoEntity {

    private static final String sequenceName = "sq_orientacao_mestrado";

    @Id
    @Field(analyze = Analyze.NO, index = org.hibernate.search.annotations.Index.NO, store =
        ↳ Store.YES)
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName =
        ↳ OrientacaoMestradoEntity.sequenceName, name =
        ↳ OrientacaoMestradoEntity.sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator =
        ↳ OrientacaoMestradoEntity.sequenceName)
    @Column(name = "id_orientacao_mestrado")
    private Long id;

    @IndexedEmbedded
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_orientador", foreignKey = @ForeignKey(name =
        ↳ "fk_orientacao_mestrado_orientador"))
    private DocenteEntity orientador;

    @Column(name = "orientador_ativo")
    private Boolean orientadorAtivo = false;

    @IndexedEmbedded
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_coorientador", foreignKey = @ForeignKey(name =
        ↳ "fk_orientacao_mestrado_coorientador"))
    private DocenteEntity coorientador;

    @Column(name = "coorientador_ativo")
    private Boolean coorientadorAtivo = false;

    @IndexedEmbedded
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_discente", foreignKey = @ForeignKey(name =
        ↳ "fk_orientacao_mestrado_discente"), nullable = false)
    private DiscenteEntity discente;

    /**
     * Indica o programa em que a orientao ou formao
     * foi importada no necessariamente queira dizer que
     * ainda pertence ao mesmo
     */
}

```

```

*/
@ManyToOne(fetch = FetchType.LAZY)
@JoinColumn(name = "id_programa", foreignKey = @ForeignKey(name =
    ↪ "fk_orientacao_mestrado_programa"), nullable = false)
private ProgramaEntity programa;

@Column(name = "ano_inicio")
private Integer anoInicio;

@Column(name = "ano_fim")
private Integer anoFim;

@Field(name = "titulo", analyzer = @Analyzer(definition = "ngram-titulo"), store =
    ↪ Store.YES)
@Column(name = "titulo_do_trabalho", length = 400, nullable = false)
private String tituloDoTrabalho;

@Column(name = "titulo_do_trabalho_filtro", length = 400, nullable = false)
private String tituloDoTrabalhoFiltro;

@ManyToOne(fetch = FetchType.LAZY)
@JoinColumn(name = "id_instituicao", foreignKey = @ForeignKey(name =
    ↪ "fk_orientacao_mestrado_instituicao"))
private InstituicaoEntity instituicao;

@Column
private Boolean bolsa;

@Enumerated(EnumType.STRING)
@Column(name = "status_orientacao")
private StatusOrientacao statusOrientacao;

@ManyToOne(fetch = FetchType.LAZY)
@JoinColumn(name = "id_area_conhecimento_um", foreignKey = @ForeignKey(name =
    ↪ "fk_orientacao_mestrado_area_conhecimento_um"))
private AreaConhecimentoEntity areaConhecimentoUm;

@ManyToOne(fetch = FetchType.LAZY)
@JoinColumn(name = "id_area_conhecimento_dois", foreignKey = @ForeignKey(name =
    ↪ "fk_orientacao_mestrado_area_conhecimento_dois"))
private AreaConhecimentoEntity areaConhecimentoDois;

@ManyToOne(fetch = FetchType.LAZY)
@JoinColumn(name = "id_area_conhecimento_tres", foreignKey = @ForeignKey(name =
    ↪ "fk_orientacao_mestrado_area_conhecimento_tres"))
private AreaConhecimentoEntity areaConhecimentoTres;

// no atualizar com update
@Field(analyze = Analyze.NO, index = org.hibernate.search.annotations.Index.NO, store =
    ↪ Store.YES)
@Column(name = "discente_completo")
private Boolean discenteCompleto = false;

@Column
private Boolean deletado = false;

public void setTituloDoTrabalho(String tituloDoTrabalho) {
    this.tituloDoTrabalho = tituloDoTrabalho;
    this.tituloDoTrabalhoFiltro = QueryUtils.lowerCaseStripAccents(tituloDoTrabalho);
}
}
package alpc.ufsc.entity.pessoa.orientacao.graduacao;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import org.hibernate.search.annotations.Analyzer;
import org.hibernate.search.annotations.Field;

```

```

import org.hibernate.search.annotations.Indexed;
import org.hibernate.search.annotations.IndexedEmbedded;
import org.hibernate.search.annotations.Store;

import alpc.ufsc.entity.domain.instituicao.InstituicaoEntity;
import alpc.ufsc.entity.pessoa.discente.DiscenteEntity;
import alpc.ufsc.entity.programa.ProgramaEntity;

@Entity
@Getter
@Setter
//@formatter:off
@Table(name = "tb_graduacao", indexes = {
    @Index(name = "in_graduacao_id_discente", columnList = "id_discente"),
    @Index(name = "in_graduacao_id_programa_deletado", columnList =
        ↳ "id_programa,deletado")})
//@formatter:on
@NoArgsConstructor
@Indexed(index = "graduacoes")
public class GraduacaoEntity {

    private static final String sequenceName = "sq_graduacao";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName =
        ↳ GraduacaoEntity.sequenceName, name = GraduacaoEntity.sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator =
        ↳ GraduacaoEntity.sequenceName)
    @Column(name = "id_graduacao")
    private Long id;

    @IndexedEmbedded
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_discente", foreignKey = @ForeignKey(name =
        ↳ "fk_graduacao_discente"), nullable = false)
    private DiscenteEntity discente;

    @IndexedEmbedded
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_programa", foreignKey = @ForeignKey(name =
        ↳ "fk_graduacao_programa"), nullable = false)
    private ProgramaEntity programa;

    @Field(name = "titulo", analyzer = @Analyzer(definition = "ngram-titulo"), store =
        ↳ Store.YES)
    @Column(name = "titulo", length = 400)
    private String titulo;

    @Column(name = "ano_inicio")
    private Integer anoInicio;

    @Column(name = "ano_fim")
    private Integer anoFim;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_instituicao", foreignKey = @ForeignKey(name =
        ↳ "fk_orientacao_mestrado_instituicao"))
    private InstituicaoEntity instituicao;

    @Column
    private Boolean deletado = false;
}
package alpc.ufsc.entity.pessoa.orientacao.doutorado;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

```

```

import org.hibernate.search.annotations.Analyze;
import org.hibernate.search.annotations.Analyzer;
import org.hibernate.search.annotations.Field;
import org.hibernate.search.annotations.Indexed;
import org.hibernate.search.annotations.IndexedEmbedded;
import org.hibernate.search.annotations.Store;

import alpc.ufsc.entity.domain.StatusOrientacao;
import alpc.ufsc.entity.domain.area.AreaConhecimentoEntity;
import alpc.ufsc.entity.domain.instituicao.InstituicaoEntity;
import alpc.ufsc.entity.pessoa.discente.DiscenteEntity;
import alpc.ufsc.entity.pessoa.docente.DocenteEntity;
import alpc.ufsc.entity.pessoa.orientacao.BaseOrientacaoEntity;
import alpc.ufsc.entity.programa.ProgramaEntity;
import alpc.ufsc.entity.util.QueryUtils;

@Entity
@Getter
@Setter
//@formatter:off
@Table(name = "tb_orientacao_doutorado", indexes = {
    @Index(name = "in_doutorado_id_orientador", columnList = "id_orientador"),
    @Index(name = "in_doutorado_id_coorientador", columnList =
        ↳ "id_coorientador"),
    @Index(name = "in_doutorado_id_discente", columnList = "id_discente"),
    @Index(name = "in_doutorado_titulo_do_trabalho_filtro", columnList =
        ↳ "titulo_do_trabalho_filtro"),
    @Index(name = "in_doutorado_id_area_conhecimento_um", columnList =
        ↳ "id_area_conhecimento_um"),
    @Index(name = "in_doutorado_id_area_conhecimento_dois", columnList =
        ↳ "id_area_conhecimento_dois"),
    @Index(name = "in_doutorado_id_area_conhecimento_tres", columnList =
        ↳ "id_area_conhecimento_tres"),
    @Index(name = "in_doutorado_id_instituicao", columnList = "id_instituicao"),
    @Index(name = "in_doutorado_id_programa_deletado", columnList =
        ↳ "id_programa_deletado"),
    @Index(name = "in_doutorado_status_orientacao", columnList =
        ↳ "status_orientacao")})
//@formatter:on
@NoArgsConstructor
@Indexed(index = "doutorados")
public class OrientacaoDoutoradoEntity implements BaseOrientacaoEntity {

    private static final String sequenceName = "sq_orientacao_doutorado";

    @Id
    @Field(analyze = Analyze.NO, index = org.hibernate.search.annotations.Index.NO, store =
        ↳ Store.YES)
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName = sequenceName,
        ↳ name = sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator = sequenceName)
    @Column(name = "id_orientacao_doutorado")
    private Long id;

    @IndexedEmbedded
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_orientador", foreignKey = @ForeignKey(name =
        ↳ "fk_orientacao_doutorado_orientador"))
    private DocenteEntity orientador;

    @Column(name = "orientador_ativo")
    private Boolean orientadorAtivo = false;

    @IndexedEmbedded
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_coorientador", foreignKey = @ForeignKey(name =
        ↳ "fk_orientacao_doutorado_coorientador"))
    private DocenteEntity coorientador;

    @Column(name = "coorientador_ativo")
    private Boolean coorientadorAtivo = false;

    @IndexedEmbedded
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_discente", foreignKey = @ForeignKey(name =
        ↳ "fk_orientacao_doutorado_discente"), nullable = false)
    private DiscenteEntity discente;

    /**
     * Indica o programa em que a orientao ou formao
     * foi importada no necessariamente queira dizer que

```

```

    * ainda pertence ao mesmo
    */
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_programa", foreignKey = @ForeignKey(name =
        ↳ "fk_orientacao_doutorado_programa"), nullable = false)
    private ProgramaEntity programa;

    @Column(name = "ano_inicio")
    private Integer anoInicio;

    @Column(name = "ano_fim")
    private Integer anoFim;

    @Field(name = "titulo", analyzer = @Analyzer(definition = "ngram-titulo"), store =
        ↳ Store.YES)
    @Column(name = "titulo_do_trabalho", length = 400, nullable = false)
    private String tituloDoTrabalho;

    @Column(name = "titulo_do_trabalho_filtro", length = 400, nullable = false)
    private String tituloDoTrabalhoFiltro;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_instituicao", foreignKey = @ForeignKey(name =
        ↳ "fk_orientacao_doutorado_instituicao"))
    private InstituicaoEntity instituicao;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_instituicao_sanduiche", foreignKey = @ForeignKey(name =
        ↳ "fk_orientacao_doutorado_instituicao_sanduiche"))
    private InstituicaoEntity instituicaoSanduiche;

    @Column(name = "nome_orientador_sanduiche", length = 400)
    private String nomeOrientadorSanduiche;

    @Column(name = "nome_orientador_sanduiche_filtro", length = 400)
    private String nomeOrientadorSanduicheFiltro;

    @Column
    private Boolean bolsa;

    @Enumerated(EnumType.STRING)
    @Column(name = "status_orientacao")
    private StatusOrientacao statusOrientacao;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_area_conhecimento_um", foreignKey = @ForeignKey(name =
        ↳ "fk_orientacao_doutorado_area_conhecimento_um"))
    private AreaConhecimentoEntity areaConhecimentoUm;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_area_conhecimento_dois", foreignKey = @ForeignKey(name =
        ↳ "fk_orientacao_doutorado_area_conhecimento_dois"))
    private AreaConhecimentoEntity areaConhecimentoDois;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_area_conhecimento_tres", foreignKey = @ForeignKey(name =
        ↳ "fk_orientacao_doutorado_area_conhecimento_tres"))
    private AreaConhecimentoEntity areaConhecimentoTres;

    @Field(analyze = Analyze.NO, index = org.hibernate.search.annotations.Index.NO, store =
        ↳ Store.YES)
    @Column(name = "discente_completo")
    private Boolean discenteCompleto = false;

    @Column
    private Boolean deletado = false;

    public void setTituloDoTrabalho(String tituloDoTrabalho) {
        this.tituloDoTrabalho = tituloDoTrabalho;
        this.tituloDoTrabalhoFiltro = QueryUtils.lowerCaseStripAccents(tituloDoTrabalho);
    }

    public void setNomeOrientadorSanduiche(String nomeOrientadorSanduiche) {
        this.nomeOrientadorSanduiche = nomeOrientadorSanduiche;
        this.nomeOrientadorSanduicheFiltro =
            ↳ QueryUtils.lowerCaseStripAccents(nomeOrientadorSanduiche);
    }
}
package alpc.ufsc.entity.pessoa.orientacao;

import alpc.ufsc.entity.domain.area.AreaConhecimentoEntity;
import alpc.ufsc.entity.domain.instituicao.InstituicaoEntity;

```

```

import alpc.ufsc.entity.pessoa.discente.DiscenteEntity;
import alpc.ufsc.entity.pessoa.docente.DocenteEntity;
import alpc.ufsc.entity.programa.ProgramaEntity;

public interface BaseOrientacaoEntity {

    Long getId();

    void setDiscente(DiscenteEntity discente);

    default void setOrientador(DocenteEntity docente){}

    default void setOrientadorAtivo(Boolean ativo){}

    default void setCoorientador(DocenteEntity docente){}

    default void setCoorientadorAtivo(Boolean ativo){}

    void setPrograma(ProgramaEntity programa);

    void setInstituicao(InstituicaoEntity instituicao);

    void setAreaConhecimentoUm(AreaConhecimentoEntity areaConhecimento);

    void setAreaConhecimentoDois(AreaConhecimentoEntity areaConhecimento);

    void setAreaConhecimentoTres(AreaConhecimentoEntity areaConhecimento);

    void setDiscenteCompleto(Boolean discenteCompleto);

    default void setDeletado(Boolean deletado) {}
}

package alpc.ufsc.entity.pessoa.orientacao.posdoutorado;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import org.hibernate.search.annotations.Analyze;
import org.hibernate.search.annotations.Analyzer;
import org.hibernate.search.annotations.Field;
import org.hibernate.search.annotations.Indexed;
import org.hibernate.search.annotations.IndexedEmbedded;
import org.hibernate.search.annotations.Store;

import alpc.ufsc.entity.domain.StatusOrientacao;
import alpc.ufsc.entity.domain.area.AreaConhecimentoEntity;
import alpc.ufsc.entity.domain.instituicao.InstituicaoEntity;
import alpc.ufsc.entity.pessoa.discente.DiscenteEntity;
import alpc.ufsc.entity.pessoa.orientacao.BaseOrientacaoEntity;
import alpc.ufsc.entity.programa.ProgramaEntity;
import alpc.ufsc.entity.util.QueryUtils;

@Entity
@Getter
@Setter
// @formatter:off
@Table(name = "tb_orientacao_pos_doutorado", indexes = {
    @Index(name = "in_pos_doutorado_id_discente", columnList = "id_discente"),
    @Index(name = "in_pos_doutorado_titulo_do_trabalho_filtro", columnList =
        ↳ "titulo_do_trabalho_filtro"),
    @Index(name = "in_pos_doutorado_id_area_conhecimento_um", columnList =
        ↳ "id_area_conhecimento_um"),
    @Index(name = "in_pos_doutorado_id_area_conhecimento_dois", columnList =
        ↳ "id_area_conhecimento_dois"),
    @Index(name = "in_pos_doutorado_id_area_conhecimento_tres", columnList =
        ↳ "id_area_conhecimento_tres"),

```

```

        @Index(name = "in_pos_doutorado_id_instituicao", columnList =
            ↳ "id_instituicao"),
        @Index(name = "in_pos_doutorado_id_programa_deletado", columnList =
            ↳ "id_programa_deletado"))
    //@@formatter:on
    @NoArgsConstructor
    @Indexed(index = "posdoutorados")
    public class OrientacaoPosDoutoradoEntity implements BaseOrientacaoEntity {

        private static final String sequenceName = "sq_orientacao_pos_doutorado";

        @Id
        @Field(analyze = Analyze.NO, index = org.hibernate.search.annotations.Index.NO, store =
            ↳ Store.YES)
        @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName =
            ↳ OrientacaoPosDoutoradoEntity.sequenceName, name =
            ↳ OrientacaoPosDoutoradoEntity.sequenceName)
        @GeneratedValue(strategy = GenerationType.AUTO, generator =
            ↳ OrientacaoPosDoutoradoEntity.sequenceName)
        @Column(name = "id_orientacao_pos_doutorado")
        private Long id;

        @IndexedEmbedded
        @ManyToOne(fetch = FetchType.LAZY)
        @JoinColumn(name = "id_discente", foreignKey = @ForeignKey(name =
            ↳ "fk_orientacao_pos_doutorado_discente"), nullable = false)
        private DiscenteEntity discente;

        /**
         * Indica o programa em que a orientao ou formao
         * foi importada no necessariamente queira dizer que
         * ainda pertence ao mesmo
         */
        @ManyToOne(fetch = FetchType.LAZY)
        @JoinColumn(name = "id_programa", foreignKey = @ForeignKey(name =
            ↳ "fk_orientacao_pos_doutorado_programa"), nullable = false)
        private ProgramaEntity programa;

        @Column(name = "ano_inicio")
        private Integer anoInicio;

        @Column(name = "ano_fim")
        private Integer anoFim;

        @Field(name = "titulo", analyzer = @Analyzer(definition = "ngram-titulo"), store =
            ↳ Store.YES)
        @Column(name = "titulo_do_trabalho", length = 400, nullable = false)
        private String tituloDoTrabalho;

        @Column(name = "titulo_do_trabalho_filtro", length = 400, nullable = false)
        private String tituloDoTrabalhoFiltro;

        @ManyToOne(fetch = FetchType.LAZY)
        @JoinColumn(name = "id_instituicao", foreignKey = @ForeignKey(name =
            ↳ "fk_orientacao_pos_doutorado_instituicao"))
        private InstituicaoEntity instituicao;

        @Column
        private Boolean bolsa;

        @Enumerated(EnumType.STRING)
        @Column(name = "status_orientacao")
        private StatusOrientacao statusOrientacao;

        @ManyToOne(fetch = FetchType.LAZY)
        @JoinColumn(name = "id_area_conhecimento_um", foreignKey = @ForeignKey(name =
            ↳ "fk_orientacao_pos_doutorado_area_conhecimento_um"))
        private AreaConhecimentoEntity areaConhecimentoUm;

        @ManyToOne(fetch = FetchType.LAZY)
        @JoinColumn(name = "id_area_conhecimento_dois", foreignKey = @ForeignKey(name =
            ↳ "fk_orientacao_pos_doutorado_area_conhecimento_dois"))
        private AreaConhecimentoEntity areaConhecimentoDois;

        @ManyToOne(fetch = FetchType.LAZY)
        @JoinColumn(name = "id_area_conhecimento_tres", foreignKey = @ForeignKey(name =
            ↳ "fk_orientacao_pos_doutorado_area_conhecimento_tres"))
        private AreaConhecimentoEntity areaConhecimentoTres;

        @Field(analyze = Analyze.NO, index = org.hibernate.search.annotations.Index.NO, store =
            ↳ Store.YES)
        @Column(name = "discente_completo")

```

```

    private Boolean discenteCompleto = false;

    @Column
    private Boolean deletado = false;

    public void setTituloDoTrabalho(String tituloDoTrabalho) {
        this.tituloDoTrabalho = tituloDoTrabalho;
        this.tituloDoTrabalhoFiltro = QueryUtils.lowerCaseStripAccents(tituloDoTrabalho);
    }
}
package alpc.ufsc.entity.pessoa.docente;

import java.io.Serializable;
import java.util.List;

import javax.persistence.Column;
import javax.persistence.Convert;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.OneToMany;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import org.hibernate.search.annotations.Analyze;
import org.hibernate.search.annotations.ContainedIn;
import org.hibernate.search.annotations.Field;
import org.hibernate.search.annotations.IndexedEmbedded;
import org.hibernate.search.annotations.Store;

import alpc.ufsc.entity.domain.produtividade.Categoria;
import alpc.ufsc.entity.domain.produtividade.CategoriaJpaConverter;
import alpc.ufsc.entity.domain.produtividade.Modalidade;
import alpc.ufsc.entity.domain.produtividade.ModalidadeJpaConverter;
import alpc.ufsc.entity.pessoa.PessoaEntity;
import alpc.ufsc.entity.pessoa.orientacao.doutorado.OrientacaoDoutoradoEntity;
import alpc.ufsc.entity.pessoa.orientacao.mestrado.OrientacaoMestradoEntity;

@Entity
@Getter
@Setter
//@formatter:off
@Table(name = "tb_docente", indexes = {
    @Index(name = "in_docente_id_pessoa", columnList = "id_pessoa")})
//@formatter:on
@NoArgsConstructor
public class DocenteEntity implements Serializable {

    private static final long serialVersionUID = -2316853521107707372L;

    private static final String sequenceName = "sq_docente";

    @Id
    @Field(analyze = Analyze.NO, index = org.hibernate.search.annotations.Index.NO, store =
        ↳ Store.YES)
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName =
        ↳ DocenteEntity.sequenceName, name = DocenteEntity.sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator =
        ↳ DocenteEntity.sequenceName)
    @Column(name = "id_docente")
    private Long id;

    @IndexedEmbedded
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_pessoa", foreignKey = @ForeignKey(name =
        ↳ "fk_docente_pessoa"), nullable = false)
    private PessoaEntity pessoa;

    @Column
    @Convert(converter = CategoriaJpaConverter.class)
    private Categoria categoria;

```



```

        @Column
        @Convert(converter = ModalidadeJpaConverter.class)
        private Modalidade modalidade;

        @ContainedIn
        @OneToMany(mappedBy = "orientador", fetch = FetchType.LAZY)
        private List<OrientacaoMestradoEntity> orientacoesPrincipaisMestrado;

        @ContainedIn
        @OneToMany(mappedBy = "orientador", fetch = FetchType.LAZY)
        private List<OrientacaoDoutoradoEntity> orientacoesPrincipaisDoutorado;
    }
}
package alpc.ufsc.entity.pessoa;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import org.hibernate.search.annotations.Analyze;
import org.hibernate.search.annotations.ContainedIn;
import org.hibernate.search.annotations.Field;
import org.hibernate.search.annotations.Store;

import lombok.Getter;
import lombok.Setter;

@Entity
@Getter
@Setter
@Table(name = "tb_nome_citacao")
public class NomeCitacaoEntity {

    private static final String sequenceName = "sq_nome_citacao";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName =
        ↳ NomeCitacaoEntity.sequenceName, name = NomeCitacaoEntity.sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator =
        ↳ NomeCitacaoEntity.sequenceName)
    @Column(name = "id_nome_pessoa")
    private Long id;

    @Field(name = "citacao", analyze = Analyze.NO, store = Store.YES)
    @Column(name = "citacao", length = 400)
    private String citacao;

    @ContainedIn
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_pessoa", foreignKey = @ForeignKey(name =
        ↳ "fk_nomecitacao_pessoa"), nullable = false)
    private PessoaEntity pessoa;
}
package alpc.ufsc.entity.pessoa;

import java.io.Serializable;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import org.hibernate.search.annotations.Analyze;
import org.hibernate.search.annotations.Analyzer;
import org.hibernate.search.annotations.Field;

import lombok.Getter;
import lombok.Setter;

@Entity

```

```

@Getter
@Setter
@Table(name = "tb_autor")
public class AutorEntity implements Serializable {

    private static final long serialVersionUID = 1L;

    private static final String sequenceName = "sq_autor";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName =
        ↳ AutorEntity.sequenceName, name = AutorEntity.sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator =
        ↳ AutorEntity.sequenceName)
    @Column(name = "id_autor_producao")
    private Long id;

    @Field(name = "nome", analyzer = @Analyzer(definition = "ngram-nome"))
    @Column(name = "nome", length = 400, nullable = false)
    private String nome;

    @Field(name = "citacao", analyze = Analyze.NO)
    @Column(name = "citacao", length = 400)
    private String citacao;

    @Field(name = "numero_identificador", analyze = Analyze.NO)
    @Column(name = "numero_identificador", length = 400)
    private String numeroIdentificador;

}

package alpc.ufsc.entity.pessoa.discente;

import java.io.Serializable;
import java.util.List;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.OneToMany;
import javax.persistence.OneToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import org.hibernate.search.annotations.Analyze;
import org.hibernate.search.annotations.ContainedIn;
import org.hibernate.search.annotations.Field;
import org.hibernate.search.annotations.IndexedEmbedded;
import org.hibernate.search.annotations.Store;

import alpc.ufsc.entity.pessoa.PessoaEntity;
import alpc.ufsc.entity.pessoa.orientacao.doutorado.OrientacaoDoutoradoEntity;
import alpc.ufsc.entity.pessoa.orientacao.mestrado.OrientacaoMestradoEntity;
import alpc.ufsc.entity.pessoa.orientacao.posdoutorado.OrientacaoPosDoutoradoEntity;

@Entity
@Getter
@Setter
//@formatter:off
@Table(name = "tb_discente", indexes = {
    @Index(name = "in_discente_id_pessoa", columnList = "id_pessoa" )})
//@formatter:on
@NoArgsConstructor
public class DiscenteEntity implements Serializable {

    private static final long serialVersionUID = -4633210405649809197L;

    private static final String sequenceName = "sq_discente";

    @Id
    @Field(analyze = Analyze.NO, index = org.hibernate.search.annotations.Index.NO, store =
        ↳ Store.YES)
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName =

```

```

        ↪ DiscenteEntity.sequenceName, name = DiscenteEntity.sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator =
        ↪ DiscenteEntity.sequenceName)
    @Column(name = "id_discente")
    private Long id;

    @IndexedEmbedded
    @OneToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_pessoa", foreignKey = @ForeignKey(name =
        ↪ "fk_discente_pessoa"), nullable = false)
    private PessoaEntity pessoa;

    @ContainedIn
    @OneToMany(mappedBy = "discente", fetch = FetchType.LAZY)
    private List<OrientacaoMestradoEntity> orientacoesMestrado;

    @ContainedIn
    @OneToMany(mappedBy = "discente", fetch = FetchType.LAZY)
    private List<OrientacaoDoutoradoEntity> orientacoesDoutorado;

    @ContainedIn
    @OneToMany(mappedBy = "discente", fetch = FetchType.LAZY)
    private List<OrientacaoPosDoutoradoEntity> orientacoesPosDoutorado;
}
package alpc.ufsc.entity.pessoa.atuacaoprofissional;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import org.hibernate.search.annotations.Field;

import alpc.ufsc.entity.pessoa.docente.DocenteEntity;

@Entity
@Getter
@Setter
//@formatter:off
@Table(name = "tb_atuacao_profissional", indexes = {
    @Index(name = "in_docente_tipo_atuacao_profissional",
        ↪ columnList="id_docente,tipo_atuacao_profissional")
})
//@formatter:on
@NoArgsConstructor
public class AtuacaoProfissionalEntity {

    private static final String sequenceName = "sq_atuacao_profissional";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName =
        ↪ AtuacaoProfissionalEntity.sequenceName, name =
        ↪ AtuacaoProfissionalEntity.sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator =
        ↪ AtuacaoProfissionalEntity.sequenceName)
    @Column(name = "id_atuacao_profissional")
    private Long id;

    @Column(name = "mes_inicio")
    protected Integer mesInicio;

    @Column(name = "ano_inicio")
    protected Integer anoInicio;

    @Column(name = "mes_fim")
    protected Integer mesFim;

    @Column(name = "ano_fim")

```

```

protected Integer anoFim;

@Field(name = "tipo_atuacao_profissional")
@Enumerated(EnumType.STRING)
@Column(name = "tipo_atuacao_profissional")
private TipoAtuacaoProfissional tipoAtuacao;

@ManyToOne(fetch = FetchType.LAZY)
@JoinColumn(name = "id_docente", foreignKey = @ForeignKey(name =
    ↪ "fk_orientacao_mestrado_docente"), nullable = false)
private DocenteEntity docente;
}

package alpc.ufsc.entity.pessoa.atuacaoprofissional;

public enum TipoAtuacaoProfissional {

    CONSELHO_COMISSAO_CONSULTORIA,
    DIRECAO_ADMINISTRACAO,
    ENSINO,
    EXTENSAO_UNIVERSITARIA,
    PARTICIPACAO_PROJETO,
    PESQUISA_DESENVOLVIMENTO,
    SERVICIO_TECNICO_ESPECIALIZADO,
    TREINAMENTO_MINISTRADO,
    OUTRA_ATIVIDADE_TECNICO_CIENTIFICA
}

package alpc.ufsc.entity.pessoa.atuacaoprofissional.linhapesquisa;

import java.io.Serializable;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import org.hibernate.search.annotations.Analyzer;
import org.hibernate.search.annotations.Field;
import org.hibernate.search.annotations.Indexed;
import org.hibernate.search.annotations.IndexedEmbedded;
import org.hibernate.search.annotations.Store;

import alpc.ufsc.entity.programa.ProgramaEntity;
import alpc.ufsc.entity.util.QueryUtils;

@Entity
@Getter
@Setter
@Table(name = "tb_linha_pesquisa", indexes = {
    @Index(name = "in_linhas_pesquisa_titulo_filtro", columnList = "titulo_filtro"),
    @Index(name = "in_linha_pesquisa_id_programa", columnList = "id_programa")
})
@Indexed(index = "linhas_pesquisa")
@NoArgsConstructor
public class LinhaPesquisaEntity implements Serializable {

    private static final long serialVersionUID = 2137341388378997219L;

    private static final String sequenceName = "sq_linha_pesquisa";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName =
        ↪ LinhaPesquisaEntity.sequenceName, name = LinhaPesquisaEntity.sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator =
        ↪ LinhaPesquisaEntity.sequenceName)
    @Column(name = "id_linha_pesquisa")
    private Long id;

    @Field(name = "titulo", analyzer = @Analyzer(definition = "ngram-titulo"), store =

```

```

        ↪ Store.YES)
    @Column(name = "titulo", length = 400, nullable = false)
    private String titulo;

    @Column(name = "titulo_filtro", length = 400, nullable = false)
    private String tituloFiltro;

    @IndexedEmbedded
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_programa", foreignKey = @ForeignKey(name =
        ↪ "fk_linhapesquisa_programa"), nullable = false)
    private ProgramaEntity programa;

    public void setTitulo(String titulo) {
        this.titulo = titulo;
        this.tituloFiltro = QueryUtils.lowerCaseStripAccents(titulo);
    }
}
package alpc.ufsc.entity.pessoa.atuacaoprofissional.linhapesquisa;

import java.io.Serializable;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.Setter;

import alpc.ufsc.entity.pessoa.docente.DocenteEntity;

@Entity
@Getter
@Setter
@Table(name = "rl_linha_pesquisa_docente", indexes = {
    @Index(name = "in_linha_pesquisa_docente_id_docente", columnList =
        ↪ "id_docente"),
    @Index(name = "in_linha_pesquisa_docente_id_linha_pesquisa", columnList =
        ↪ "id_linha_pesquisa")
})
public class LinhaPesquisaDocenteEntity implements Serializable {

    private static final long serialVersionUID = 758532304391905120L;

    private static final String sequenceName = "sq_linha_pesquisa_docente";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName = sequenceName,
        ↪ name = sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator = sequenceName)
    @Column(name = "id_linha_pesquisa_docente")
    private Long id;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_docente", foreignKey = @ForeignKey(name =
        ↪ "fk_linhapesquisadocente_docente"), nullable = false)
    private DocenteEntity docente;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_linha_pesquisa", foreignKey = @ForeignKey(name =
        ↪ "fk_linhapesquisadocente_linhapesquisa"), nullable = false)
    private LinhaPesquisaEntity linhaPesquisa;
}
package alpc.ufsc.entity.pessoa.atuacaoprofissional.projetoPesquisa;

import java.io.Serializable;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;

```

```

import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.Setter;

import org.hibernate.search.annotations.Indexed;
import org.hibernate.search.annotations.IndexedEmbedded;

import alpc.ufsc.entity.pessoa.AutorEntity;

@Entity
@Getter
@Setter
@Indexed(index = "autores_projetos_pesquisa")
@Table(name = "rl_autor_projeto_pesquisa", indexes = {
    @Index(name = "in_autor_projeto_pesquisa_id_autor", columnList =
        ↪ "id_autor"),
    @Index(name = "in_autor_projeto_pesquisa_docente_id_projeto_pesquisa",
        ↪ columnList = "id_projeto_pesquisa")
})
public class AutorProjetoPesquisaEntity implements Serializable {

    private static final long serialVersionUID = -807000358386362097L;

    private static final String sequenceName = "sq_autor_projeto_pesquisa";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName = sequenceName,
        ↪ name = sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator = sequenceName)
    @Column(name = "id_autor_projeto_pesquisa")
    private Long id;

    @IndexedEmbedded
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_autor", foreignKey = @ForeignKey(name =
        ↪ "fk_autorprojpesq_autor"), nullable = false)
    private AutorEntity autor;

    @IndexedEmbedded
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_projeto_pesquisa", foreignKey = @ForeignKey(name =
        ↪ "fk_autorprojpesq_projpesq"))
    private ProjetoPesquisaEntity projetoPesquisa;
}
package alpc.ufsc.entity.pessoa.atuacaoprofissional.projtopesquisa;

import java.io.Serializable;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.Setter;

import alpc.ufsc.entity.pessoa.docente.DocenteEntity;

@Entity
@Getter
@Setter
@Table(name = "rl_projeto_pesquisa_docente", indexes = {
    @Index(name = "in_projeto_pesquisa_docente_id_docente", columnList =
        ↪ "id_docente"),
    @Index(name = "in_projeto_pesquisa_docente_id_projeto_pesquisa", columnList
        ↪ = "id_projeto_pesquisa")
})
public class ProjetoPesquisaDocenteEntity implements Serializable {

```

```

private static final long serialVersionUID = 758532304391905120L;

private static final String sequenceName = "sq_projeto_pesquisa_docente";

@Id
@SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName = sequenceName,
    ↪ name = sequenceName)
@GeneratedValue(strategy = GenerationType.AUTO, generator = sequenceName)
@Column(name = "id_projeto_pesquisa_docente")
private Long id;

@ManyToOne(fetch = FetchType.LAZY)
@JoinColumn(name = "id_docente", foreignKey = @ForeignKey(name =
    ↪ "fk_projetopesquisadocente_docente"), nullable = false)
private DocenteEntity docente;

@ManyToOne(fetch = FetchType.LAZY)
@JoinColumn(name = "id_projeto_pesquisa", foreignKey = @ForeignKey(name =
    ↪ "fk_projetopesquisadocente_projetopesquisa"), nullable = false)
private ProjetoPesquisaEntity projetoPesquisa;

@Column(name = "ano_inicio_participacao")
private Integer anoInicioParticipacao;

@Column(name = "ano_fim_participacao")
private Integer anoFimParticipacao;
}
package alpc.ufsc.entity.pessoa.atuacaoprofissional.projetopesquisa;

import java.io.Serializable;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import org.hibernate.search.annotations.Analyze;
import org.hibernate.search.annotations.Analyzer;
import org.hibernate.search.annotations.Field;
import org.hibernate.search.annotations.IndexedEmbedded;
import org.hibernate.search.annotations.Store;

import alpc.ufsc.entity.programa.ProgramaEntity;

@Entity
@Getter
@Setter
@NoArgsConstructor
@Table(name = "tb_projeto_pesquisa", indexes = {
    @Index(name = "in_projeto_pesquisa_nome_filtro", columnList = "nome_filtro"),
    @Index(name = "in_projeto_pesquisa_id_programa", columnList =
        ↪ "id_programa")
})
public class ProjetoPesquisaEntity implements Serializable {

    private static final long serialVersionUID = 2137341388378997219L;

    private static final String sequenceName = "sq_projeto_pesquisa";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName =
        ↪ ProjetoPesquisaEntity.sequenceName, name =
        ↪ ProjetoPesquisaEntity.sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator =
        ↪ ProjetoPesquisaEntity.sequenceName)
    @Column(name = "id_projeto_pesquisa")
    private Long id;

    @Field(name = "nome", analyzer = @Analyzer(definition = "ngram-titulo"), store =

```

```

        ↪ Store.YES)
@Column(name = "nome", length = 400, nullable = false)
private String nome;

@Column(name = "nome_filtro", length = 400, nullable = false)
private String nomeFiltro;

@Field(name = "ano_inicio", analyze = Analyze.NO)
@Column(name = "ano_inicio")
private Integer anoInicio;

@Field(name = "ano_fim", analyze = Analyze.NO)
@Column(name = "ano_fim")
private Integer anoFim;

@IndexedEmbedded
@ManyToOne(fetch = FetchType.LAZY)
@JoinColumn(name = "id_programa", foreignKey = @ForeignKey(name =
    ↪ "fk_projetopesquisa_programa"), nullable = false)
private ProgramaEntity programa;
}
package alpc.ufsc.entity.pessoa;

import java.util.Date;
import java.util.List;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.OneToMany;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import org.apache.solr.analysis.BrazilianStemFilterFactory;
import org.apache.solr.analysis.KeywordTokenizerFactory;
import org.apache.solr.analysis.LowerCaseFilterFactory;
import org.apache.solr.analysis.NGramFilterFactory;
import org.hibernate.search.annotations.Analyze;
import org.hibernate.search.annotations.Analyzer;
import org.hibernate.search.annotations.AnalyzerDef;
import org.hibernate.search.annotations.ContainedIn;
import org.hibernate.search.annotations.Field;
import org.hibernate.search.annotations.Indexed;
import org.hibernate.search.annotations.IndexedEmbedded;
import org.hibernate.search.annotations.Parameter;
import org.hibernate.search.annotations.Store;
import org.hibernate.search.annotations.TokenFilterDef;
import org.hibernate.search.annotations.TokenizerDef;

import alpc.ufsc.entity.pessoa.discente.DiscenteEntity;
import alpc.ufsc.entity.pessoa.docente.DocenteEntity;
import alpc.ufsc.entity.programa.ProgramaEntity;
import alpc.ufsc.entity.util.QueryUtils;

@Entity
@Getter
@Setter
//@formatter:off
@Table(name = "tb_pessoa", indexes = {
    @Index(name = "in_pessoa_data_atualizacao", columnList = "data_atualizacao"),
    @Index(name = "in_pessoa_numero_identificador", columnList =
        ↪ "numero_identificador"),
    @Index(name = "in_pessoa_id_programa", columnList = "id_programa"),
    @Index(name = "in_pessoa_nome_filtro", columnList = "nome_filtro")})
//@formatter:on
@Indexed(index = "pessoas")
@NoArgsConstructor

```



```

@AnalyzerDef(name = "ngram--nome", tokenizer = @TokenizerDef(factory =
↳ KeywordTokenizerFactory.class), filters = { @TokenFilterDef(factory =
↳ BrazilianStemFilterFactory.class),
    @TokenFilterDef(factory = LowerCaseFilterFactory.class),
    @TokenFilterDef(factory = NGramFilterFactory.class, params = {
↳ @Parameter(name = "minGramSize", value = "4"), @Parameter(name =
↳ "maxGramSize", value = "4") }) })
public class PessoaEntity {

    private static final String sequenceName = "sq_pessoa";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName =
↳ PessoaEntity.sequenceName, name = PessoaEntity.sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator =
↳ PessoaEntity.sequenceName)
    @Column(name = "id_pessoa")
    private Long id;

    @Field(name = "nome", analyzer = @Analyzer(definition = "ngram--nome"), store =
↳ Store.YES)
    @Column(name = "nome", length = 400, nullable = false)
    private String nome;

    @Column(name = "nome_filtro", length = 400, nullable = false)
    private String nomeFiltro;

    @Temporal(value = TemporalType.TIMESTAMP)
    @Column(name = "data_atualizacao")
    private Date dataAtualizacao;

    @Field(analyze = Analyze.NO, store = Store.YES)
    @Column(name = "numero_identificador", length = 400)
    private String numerIdentificador;

    @ManyToOne(fetch = FetchType.LAZY)
    @IndexedEmbedded
    @JoinColumn(name = "id_programa", foreignKey = @ForeignKey(name =
↳ "fk_pessoa_programa"), nullable = false)
    private ProgramaEntity programa;

    @Column(name = "lattes_importado")
    private Boolean lattesImportado;

    @Column(name = "force_processamento")
    private Boolean forceProcessamento = false;

    // warning mapeado com @OneToMany pq @ManyToOne o hibernate no faz lazy
    @ContainedIn
    @OneToMany(mappedBy = "pessoa", fetch = FetchType.LAZY)
    private List<DocenteEntity> docente;

    // warning mapeado com @OneToMany pq @ManyToOne o hibernate no faz lazy
    @ContainedIn
    @OneToMany(mappedBy = "pessoa", fetch = FetchType.LAZY)
    private List<DiscenteEntity> discente;

    @IndexedEmbedded
    @OneToMany(mappedBy = "pessoa", fetch = FetchType.LAZY)
    private List<NomeCitacaoEntity> nomeCitacoes;

    public void setNome(String nome) {
        this.nome = nome;
        this.nomeFiltro = QueryUtils.lowerCaseStripAccents(nome);
    }
}

package alpc.ufsc.entity.system;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "tb_system_config")
public class SystemConfig {

    public static final String DATABASE_VERSION = "DATABASE_VERSION";

    @Id
    @Column

```

```

    private String key;

    @Column
    private String value;
}
package alpc.ufsc.entity.util;

import org.apache.commons.lang3.StringUtils;

import com.mysema.query.types.expr.BooleanExpression;
import com.mysema.query.types.path.NumberPath;

public class QueryUtils {

    public static final Integer MIN_YEAR = 0;
    public static final Integer MAX_YEAR = 9999;

    public static String lowerCaseStripAccents(String value) {
        if (StringUtils.isNotBlank(value)) {
            return StringUtils.stripAccents(value).toLowerCase();
        }
        return value;
    }

    public static String like(String value) {
        if (!StringUtils.isBlank(value)) {
            return "%" + lowerCaseStripAccents(value).replace(" ", "%") + "%";
        }
        return value;
    }

    public static Integer toMaxYear(Integer year) {
        return checkNull(year, MAX_YEAR);
    }

    public static Integer toMaxYear(String year) {
        return checkNull(parseInt(year), MAX_YEAR);
    }

    public static Integer toMinYear(Integer year) {
        return checkNull(year, MIN_YEAR);
    }

    public static Integer toMinYear(String year) {
        return checkNull(parseInt(year), MIN_YEAR);
    }

    public static Integer parseInt(String s) {
        if (!StringUtils.isBlank(s)) {
            return Integer.parseInt(s);
        }
        return null;
    }

    private static Integer checkNull(Integer integer, Integer defaultValue) {
        if (integer == null) {
            return defaultValue;
        }
        return integer;
    }

    public static BooleanExpression overlappingInterval(NumberPath<Integer> inicioPath,
        ↪ NumberPath<Integer> fimPath, Integer inicio, Integer fim) {
        return inicioPath.lt(toMaxYear(fim))
            .and(fimPath.goe(toMinYear(inicio)));
    }
}
package alpc.ufsc.entity.login.user;

import java.util.List;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.OneToOne;
import javax.persistence.OneToOne;

```

```

import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.entity.xml.XmlDataEntity;

@Entity
@Getter
@Setter
@Table(name = "tb_usuario", indexes = {
    @Index(name = "in_usuario_nome_filtro", columnList = "nome_filtro"),
    @Index(name = "in_usuario_email", columnList = "email"),
    @Index(name = "in_usuario_id_xml_data", columnList = "id_xml_data")
})
@NoArgsConstructor
public class UserEntity {

    private static final String sequenceName = "sq_user";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName = sequenceName,
        ↵ name = sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator = sequenceName)
    @Column(name = "id_user")
    private Long id;

    @Column(name = "nome", length = 400, nullable = false)
    private String nome;

    @Column(name = "nome_filtro", length = 400, nullable = false)
    private String nomeFiltro;

    @Column(name = "email", length = 400, nullable = false)
    private String email;

    @Column(name = "password", length = 400, nullable = false)
    private String password;

    @Column(name = "adm")
    private Boolean adm = false;

    @Column(name = "criar_programa")
    private Boolean criarPrograma = false;

    @OneToMany(mappedBy = "user")
    private List<ProgramaUserEntity> programaUser;

    @OneToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_xml_data", foreignKey = @ForeignKey(name =
        ↵ "fk_usuario_xml_data"))
    private XmlDataEntity xmlData;

    public void setNome(String nome) {
        this.nome = nome;
        this.nomeFiltro = QueryUtils.lowerCaseStripAccents(nome);
    }
}
package alpc.ufsc.entity.login.user;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import alpc.ufsc.entity.programa.ProgramaEntity;

```

```

@Entity
@Getter
@Setter
@Table(name = "tb_programa_user", indexes = {
    @Index(name = "in_programa_user_id_programa", columnList = "id_programa"),
    @Index(name = "in_programa_user_id_user", columnList = "id_user")
})
@NoArgsConstructor
public class ProgramaUserEntity {

    private static final String sequenceName = "sq_programa_user";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName = sequenceName,
        ↪ name = sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator = sequenceName)
    @Column(name = "id_programa_user")
    private Long id;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_programa", foreignKey = @ForeignKey(name =
        ↪ "fk_programa_user_programa"), nullable = false)
    private ProgramaEntity programa;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_user", foreignKey = @ForeignKey(name =
        ↪ "fk_programa_user_user"), nullable = false)
    private UserEntity user;

    @Column(name = "granted_authority", length = 50)
    private String grantedAuthority;

    @Column(name = "request_authority", length = 50)
    private String requestAuthority;
}

package alpc.ufsc.entity.xml;

import java.util.Date;
import java.util.List;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.OneToMany;
import javax.persistence.OneToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import alpc.ufsc.entity.login.user.UserEntity;

@Entity
@Getter
@Setter
@Table(name = "tb_xml_data", indexes = {
    @Index(name = "in_xml_data_nome_completo_filtro", columnList =
        ↪ "nome_completo_filtro"),
    @Index(name = "in_xml_data_identificador", columnList = "identificador")
})
@NoArgsConstructor
public class XmlDataEntity {

    private static final String sequenceName = "sq_xml_data";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName = sequenceName,
        ↪ name = sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator = sequenceName)
    @Column(name = "id_xml_data")
    private Long id;

    @Column(name = "nome_completo", length = 400)
    private String nomeCompleto;
}

```

```

@Column(name = "nome_completo_filtro", length = 400)
private String nomeCompletoFiltro;

@Column(name = "identificador", length = 50)
private String identificador;

@Column(name = "data_atualizacao")
@Temporal(TemporalType.DATE)
private Date dataAtualizacao;

@Column(name = "data_importacao")
@Temporal(TemporalType.DATE)
private Date dataImportacao;

@OneToMany(mappedBy = "xmlData")
private List<ProgramaXmlDataEntity> programaXml;

// warning @OneToMany mappedBy o hibernate no faz lazy
@OneToOne(mappedBy = "xmlData")
private UserEntity user;

@Column(name = "xml" /* ,columnDefinition = "TEXT" */)
private byte[] xml;
}
package alpc.ufsc.entity.xml;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import alpc.ufsc.entity.programa.ProgramaEntity;

@Entity
@Getter
@Setter
@Table(name = "tb_programa_xml_data", indexes = {
    @Index(name = "in_programa_xml_data_id_xml_data", columnList =
        ↳ "id_xml_data"),
    @Index(name = "in_programa_xml_data_id_programa", columnList =
        ↳ "id_programa")
})
@NoArgsConstructor
public class ProgramaXmlDataEntity {

    private static final String sequenceName = "sq_programa_xml_data";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName = sequenceName,
        ↳ name = sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator = sequenceName)
    @Column(name = "id_programa_xml_data")
    private Long id;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_programa", foreignKey = @ForeignKey(name =
        ↳ "fk_programa_xml_data_programa"), nullable = false)
    private ProgramaEntity programa;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_xml_data", foreignKey = @ForeignKey(name =
        ↳ "fk_programa_xml_data_xml_data"), nullable = false)
    private XmlDataEntity xmlData;
}
package alpc.ufsc.entity.domain.instituicao;

import java.io.Serializable;

import javax.persistence.Column;
import javax.persistence.Convert;

```

```

import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.ForeignKey;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import alpc.ufsc.entity.domain.estado.Estado;
import alpc.ufsc.entity.domain.estado.EstadoJpaConverter;
import alpc.ufsc.entity.domain.pais.PaisEntity;

@Entity
@Getter
@Setter
// @formatter:off
@Table(name = "tb_instituicao", indexes = {
    @Index(name = "in_instituicao_id_pais", columnList = "id_pais"),
    @Index(name = "in_instituicao_nome_filtro", columnList = "nome_filtro"),
    @Index(name = "in_instituicao_nome_sigla_filtro", columnList = "
        ↳ nome_sigla_filtro")})
// @formatter:on
@NoArgsConstructor
public class InstituicaoEntity implements Serializable {

    private static final long serialVersionUID = -4780464916606304808L;
    private static final String sequenceName = "sq_instituicao";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName =
        ↳ InstituicaoEntity.sequenceName, name = InstituicaoEntity.sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator =
        ↳ InstituicaoEntity.sequenceName)
    @Column(name = "id_instituicao")
    private Long id;

    @Column(name = "nome", length = 400, nullable = false)
    private String nome;

    @Column(name = "nome_filtro", length = 400, nullable = false)
    private String nomeFiltro;

    @Column(name = "nome_sigla_filtro", length = 450, nullable = false)
    private String nomeSiglaFiltro;

    @Column(name = "sigla", length = 50)
    private String sigla;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "id_pais", foreignKey = @ForeignKey(name =
        ↳ "fk_instituicao_pais"))
    private PaisEntity pais;

    @Column
    @Convert(converter = EstadoJpaConverter.class)
    private Estado estado;

    @Column
    private Integer referencias;
}
package alpc.ufsc.entity.domain.area;

import javax.persistence.Column;
import javax.persistence.Convert;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;

```

```

import lombok.Setter;

import ufsc.alpc.xml.dto.domain.GrandeAreaDoConhecimento;

@Setter
@Getter
@Entity
@Table(name = "tb_area_conhecimento", indexes = {
    @Index(name = "in_area_conhecimento_area_conhecimento_filtro", columnList =
        ↳ "area_conhecimento_filtro")
})
@NoArgsConstructor
public class AreaConhecimentoEntity {

    private static final String sequenceName = "sq_area_conhecimento";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName = sequenceName,
        ↳ name = sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator = sequenceName)
    @Column(name = "id_area_conhecimento")
    private Long id;

    @Column(name = "id_grande_area_conhecimento")
    @Convert(converter = GrandeAreaConhecimentoJpaConverter.class)
    private GrandeAreaDoConhecimento grandeAreaConhecimento;

    @Column(name = "nome_area_conhecimento", length = 200)
    private String nomeAreaConhecimento;

    @Column(name = "nome_sub_area_conhecimento", length = 200)
    private String nomeSubAreaConhecimento;

    @Column(name = "nome_especialidade", length = 200)
    private String nomeEspecialidade;

    @Column(name = "area_conhecimento_filtro", length = 800)
    private String areaConhecimentoFiltro = "";

}
package alpc.ufsc.entity.domain.area;

import javax.persistence.AttributeConverter;
import javax.persistence.Converter;

import ufsc.alpc.xml.dto.domain.GrandeAreaDoConhecimento;

@Converter
public class GrandeAreaConhecimentoJpaConverter implements
    ↳ AttributeConverter<GrandeAreaDoConhecimento, Long> {

    @Override
    public Long convertToDatabaseColumn(GrandeAreaDoConhecimento attribute) {
        if (attribute != null) {
            return attribute.getId();
        }
        return null;
    }

    @Override
    public GrandeAreaDoConhecimento convertToEntityAttribute(Long dbData) {
        return GrandeAreaDoConhecimento.getById(dbData);
    }

}
package alpc.ufsc.entity.domain.estado;

import javax.persistence.AttributeConverter;
import javax.persistence.Converter;

@Converter
public class EstadoJpaConverter implements AttributeConverter<Estado, String> {

    @Override
    public String convertToDatabaseColumn(Estado attribute) {
        if (attribute != null) {
            return attribute.getSigla();
        }
        return null;
    }

    @Override

```

```

    public Estado convertToEntityAttribute(String dbData) {
        return Estado.getBySigla(dbData);
    }
}

package alpc.ufsc.entity.domain.estado;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;

import lombok.AllArgsConstructor;
import lombok.Getter;

@Entity
@Getter
@Table(name = "tb_estado")
@AllArgsConstructor
public enum Estado {
    ACRE(1, "AC", "Acre"),
    ALAGOAS(2, "AL", "Alagoas"),
    AMAPA(3, "AP", "Amapa"),
    AMAZONAS(4, "AM", "Amazonas"),
    BAHIA(5, "BA", "Bahia"),
    CEARA(6, "CE", "Ceara"),
    DISTRITO_FEDERAL(7, "DF", "Distrito Federal"),
    ESPIRITO_SANTO(8, "ES", "Esprito Santo"),
    GOIAS(9, "GO", "Gois"),
    MARANHAO(10, "MA", "Maranho"),
    MATO_GROSSO(11, "MT", "Mato Grosso"),
    MATO_GROSSO_DO_SUL(12, "MS", "Mato Grosso do Sul"),
    MINAS_GERAIS(13, "MG", "Minas Gerais"),
    PARA(14, "PA", "Par"),
    PARAIBA(15, "PB", "Paraba"),
    PARANA(16, "PR", "Paran"),
    PERNAMBUCO(17, "PE", "Pernambuco"),
    PIAUI(18, "PI", "Piau"),
    RIO_DE_JANEIRO(19, "RJ", "Rio de Janeiro"),
    RIO_GRANDE_DO_NORTE(20, "RN", "Rio Grande do Norte"),
    RIO_GRANDE_DO_SUL(21, "RS", "Rio Grande do Sul"),
    RONDONIA(22, "RO", "Rondnia"),
    RORAIMA(23, "RR", "Roraima"),
    SANTA_CATARINA(24, "SC", "Santa Catarina"),
    SAO_PAULO(25, "SP", "So Paulo"),
    SERGIPE(26, "SE", "Sergipe"),
    TOCANTINS(27, "TO", "Tocantins"), ;

    @Id
    @Column(name = "id_estado")
    private Integer id;

    @Column(name = "sigla", length = 2)
    private String sigla;

    @Column(name = "nome", length = 50)
    private String nome;

    public static Estado getId(Integer id) {
        for (Estado estado : Estado.values()) {
            if (estado.getId().equals(id)) {
                return estado;
            }
        }
        return null;
    }

    public static Estado getSigla(String sigla) {
        for (Estado estado : Estado.values()) {
            if (estado.getSigla().equals(sigla)) {
                return estado;
            }
        }
        return null;
    }
}

package alpc.ufsc.entity.domain;

import lombok.AllArgsConstructor;
import lombok.Getter;

import ufsc.alpc.xml.dto.domain.StatusCurso;

```



```

@Getter
@AllArgsConstructor
public enum StatusOrientacao {

    EM_ANDAMENTO("Em andamento", StatusCurso.EM_ANDAMENTO),
    CONCLUIDO("Concluido", StatusCurso.CONCLUIDO),
    INCOMPLETO("Incompleto", StatusCurso.INCOMPLETO),
    DESISTENCIA("Desistencia", null);

    private String descricao;
    private StatusCurso statusOrientacaoXML;

    public static StatusOrientacao getByStatusOrientacaoXML(StatusCurso
↪ statusOrientacaoXML) {
        if (statusOrientacaoXML != null) {
            for (StatusOrientacao statusOrientacao : StatusOrientacao.values()) {
                if
↪ (statusOrientacaoXML.equals(statusOrientacao.getStatusOrientacaoXML
↪ )) {
                    return statusOrientacao;
                }
            }
        }
        return null;
    }
}

package alpc.ufsc.entity.domain.produtividade;

import javax.persistence.AttributeConverter;
import javax.persistence.Converter;

@Converter
public class ModalidadeJpaConverter implements AttributeConverter<Modalidade, String> {

    @Override
    public String convertToDatabaseColumn(Modalidade attribute) {
        if (attribute != null) {
            return attribute.getSigla();
        }
        return null;
    }

    @Override
    public Modalidade convertToEntityAttribute(String dbData) {
        return Modalidade.getBySigla(dbData);
    }
}

package alpc.ufsc.entity.domain.produtividade;

import javax.persistence.AttributeConverter;
import javax.persistence.Converter;

@Converter
public class CategoriaJpaConverter implements AttributeConverter<Categoria, String> {

    @Override
    public String convertToDatabaseColumn(Categoria attribute) {
        if (attribute != null) {
            return attribute.getSigla();
        }
        return null;
    }

    @Override
    public Categoria convertToEntityAttribute(String dbData) {
        return Categoria.getBySigla(dbData);
    }
}

package alpc.ufsc.entity.domain.produtividade;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;

import lombok.AllArgsConstructor;
import lombok.Getter;

@Entity

```

```

@Entity
@Table(name = "tb_modalidade")
@Getter
@AllArgsConstructor
public enum Modalidade {
    PESQUISA("PQ", "Produtividade em Pesquisa"),
    DESENVOLVIMENTO("DT", "Produtividade em Desenvolvimento Tecnológico e Extensão
↳ Inovadora");

    @Id
    @Column(length = 2)
    private String sigla;

    @Column(length = 100)
    private String descricao;

    public static Modalidade getBySigla(String sigla) {
        for (Modalidade modalidade : Modalidade.values()) {
            if (modalidade.getSigla().equals(sigla)) {
                return modalidade;
            }
        }
        return null;
    }
}
package alpc.ufsc.entity.domain.produtividade;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;

import lombok.AllArgsConstructor;
import lombok.Getter;

@Entity
@Table(name = "tb_categoria")
@Getter
@AllArgsConstructor
public enum Categoria {
    SR("Sr"),
    UM_A("1A"),
    UM_B("1B"),
    UM_C("1C"),
    UM_D("1D"),
    DOIS("2");

    @Id
    @Column
    private String sigla;

    public static Categoria getBySigla(String sigla) {
        for (Categoria categoria : Categoria.values()) {
            if (categoria.getSigla().equals(sigla)) {
                return categoria;
            }
        }
        return null;
    }
}
package alpc.ufsc.entity.domain.pais;

import java.io.Serializable;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Index;
import javax.persistence.SequenceGenerator;
import javax.persistence.Table;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

@Entity
@Getter
@Setter
// @formatter:off
@Table(name = "tb_pais", indexes = {

```

```

        @Index(name = "in_nome_filtro", columnList = "nome_filtro" ))
//@formatter:off
@NoArgsConstructor
public class PaisEntity implements Serializable {

    private static final long serialVersionUID = 6394524209298571246L;
    private static final String sequenceName = "sq_pais";

    @Id
    @SequenceGenerator(initialValue = 1, allocationSize = 1, sequenceName =
        ↳ PaisEntity.sequenceName, name = PaisEntity.sequenceName)
    @GeneratedValue(strategy = GenerationType.AUTO, generator = PaisEntity.sequenceName)
    @Column(name = "id_pais")
    private Long id;

    @Column(length = 200, nullable = false)
    private String nome;

    @Column(name = "nome_filtro", length = 200, nullable = false)
    private String nomeFiltro;

    @Column
    private Integer referencias;
}
package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;
import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QDatabasechangelog is a Querydsl query type for QDatabasechangelog
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QDatabasechangelog extends
    ↳ com.mysema.query.sql.RelationalPathBase<QDatabasechangelog> {

    private static final long serialVersionUID = -1533000796;

    public static final QDatabasechangelog databasechangelog = new
        ↳ QDatabasechangelog("databasechangelog");

    public final StringPath author = createString("author");
    public final StringPath comments = createString("comments");
    public final StringPath contexts = createString("contexts");
    public final DateTimePath<java.sql.Timestamp> dateexecuted =
        ↳ createDate("dateexecuted", java.sql.Timestamp.class);
    public final StringPath deploymentId = createString("deploymentId");
    public final StringPath description = createString("description");
    public final StringPath exectype = createString("exectype");
    public final StringPath filename = createString("filename");
    public final StringPath id = createString("id");
    public final StringPath labels = createString("labels");
    public final StringPath liquibase = createString("liquibase");
    public final StringPath md5sum = createString("md5sum");
    public final NumberPath<Integer> orderexecuted = createNumber("orderexecuted",
        ↳ Integer.class);
    public final StringPath tag = createString("tag");

```

```

public QDatabasechangelog(String variable) {
    super(QDatabasechangelog.class, forVariable(variable), "public", "databasechangelog");
    addMetadata();
}

public QDatabasechangelog(String variable, String schema, String table) {
    super(QDatabasechangelog.class, forVariable(variable), schema, table);
    addMetadata();
}

public QDatabasechangelog(Path<? extends QDatabasechangelog> path) {
    super(path.getType(), path.getMetadata(), "public", "databasechangelog");
    addMetadata();
}

public QDatabasechangelog(PathMetadata<?> metadata) {
    super(QDatabasechangelog.class, metadata, "public", "databasechangelog");
    addMetadata();
}

public void addMetadata() {
    addMetadata(author,
        ↪ ColumnMetadata.named("author").withIndex(2).ofType(Types.VARCHAR).withSize(255).notNull());
    addMetadata(comments,
        ↪ ColumnMetadata.named("comments").withIndex(9).ofType(Types.VARCHAR).withSize(255));
    addMetadata(contexts,
        ↪ ColumnMetadata.named("contexts").withIndex(12).ofType(Types.VARCHAR).withSize(255));
    addMetadata(dateexecuted,
        ↪ ColumnMetadata.named("dateexecuted").withIndex(4).ofType(Types.TIMESTAMP).withSize(29).withDigits(6));
    addMetadata(deploymentId,
        ↪ ColumnMetadata.named("deployment_id").withIndex(14).ofType(Types.VARCHAR).withSize(10));
    addMetadata(description,
        ↪ ColumnMetadata.named("description").withIndex(8).ofType(Types.VARCHAR).withSize(255));
    addMetadata(exectype,
        ↪ ColumnMetadata.named("exectype").withIndex(6).ofType(Types.VARCHAR).withSize(10).notNull());
    addMetadata(filename,
        ↪ ColumnMetadata.named("filename").withIndex(3).ofType(Types.VARCHAR).withSize(255).notNull());
    addMetadata(id,
        ↪ ColumnMetadata.named("id").withIndex(1).ofType(Types.VARCHAR).withSize(255).notNull());
    addMetadata(labels,
        ↪ ColumnMetadata.named("labels").withIndex(13).ofType(Types.VARCHAR).withSize(255));
    addMetadata(liquibase,
        ↪ ColumnMetadata.named("liquibase").withIndex(11).ofType(Types.VARCHAR).withSize(20));
    addMetadata(md5sum,
        ↪ ColumnMetadata.named("md5sum").withIndex(7).ofType(Types.VARCHAR).withSize(35));
    addMetadata(orderexecuted,
        ↪ ColumnMetadata.named("orderexecuted").withIndex(5).ofType(Types.INTEGER).withSize(10).notNull());
    addMetadata(tag,
        ↪ ColumnMetadata.named("tag").withIndex(10).ofType(Types.VARCHAR).withSize(255));
}
}

package alpc.ufsc.querydsl;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbOrientacaoPosDoutorado is a Querydsl query type for QTbOrientacaoPosDoutorado
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbOrientacaoPosDoutorado extends
    ↪ com.mysema.query.sql.RelationalPathBase<QTbOrientacaoPosDoutorado> {

    private static final long serialVersionUID = -909648765;

    public static final QTbOrientacaoPosDoutorado tbOrientacaoPosDoutorado = new
        ↪ QTbOrientacaoPosDoutorado("tb_orientacao_pos_doutorado");

    public final NumberPath<Integer> anoFim = createNumber("anoFim", Integer.class);

```

```

public final NumberPath<Integer> anoInicio = createNumber("anoInicio", Integer.class);
public final BooleanPath bolsa = createBoolean("bolsa");
public final BooleanPath deletado = createBoolean("deletado");
public final BooleanPath discenteCompleto = createBoolean("discenteCompleto");
public final NumberPath<Long> idAreaConhecimentoDois =
    ↪ createNumber("idAreaConhecimentoDois", Long.class);
public final NumberPath<Long> idAreaConhecimentoTres =
    ↪ createNumber("idAreaConhecimentoTres", Long.class);
public final NumberPath<Long> idAreaConhecimentoUm =
    ↪ createNumber("idAreaConhecimentoUm", Long.class);
public final NumberPath<Long> idDiscente = createNumber("idDiscente", Long.class);
public final NumberPath<Long> idInstituicao = createNumber("idInstituicao", Long.class);
public final NumberPath<Long> idOrientacaoPosDoutorado =
    ↪ createNumber("idOrientacaoPosDoutorado", Long.class);
public final NumberPath<Long> idPrograma = createNumber("idPrograma", Long.class);
public final StringPath statusOrientacao = createString("statusOrientacao");
public final StringPath tituloDoTrabalho = createString("tituloDoTrabalho");
public final StringPath tituloDoTrabalhoFiltro = createString("tituloDoTrabalhoFiltro");
public final com.mysema.query.sql.PrimaryKey<QTbOrientacaoPosDoutorado>
    ↪ tbOrientacaoPosDoutoradoPkey = createPrimaryKey(idOrientacaoPosDoutorado);

public QTbOrientacaoPosDoutorado(String variable) {
    super(QTbOrientacaoPosDoutorado.class, forVariable(variable), "public",
        ↪ "tb_orientacao_pos_doutorado");
    addMetadata();
}

public QTbOrientacaoPosDoutorado(String variable, String schema, String table) {
    super(QTbOrientacaoPosDoutorado.class, forVariable(variable), schema, table);
    addMetadata();
}

public QTbOrientacaoPosDoutorado(Path<? extends QTbOrientacaoPosDoutorado> path) {
    super(path.getType(), path.getMetadata(), "public", "tb_orientacao_pos_doutorado");
    addMetadata();
}

public QTbOrientacaoPosDoutorado(PathMetadata<?> metadata) {
    super(QTbOrientacaoPosDoutorado.class, metadata, "public",
        ↪ "tb_orientacao_pos_doutorado");
    addMetadata();
}

public void addMetadata() {
    addMetadata(anoFim,
        ↪ ColumnMetadata.named("ano_fim").withIndex(2).ofType(Types.INTEGER).withSize(10));
    addMetadata(anoInicio,
        ↪ ColumnMetadata.named("ano_inicio").withIndex(3).ofType(Types.INTEGER).withSize(10));
    addMetadata(bolsa,
        ↪ ColumnMetadata.named("bolsa").withIndex(4).ofType(Types.BIT).withSize(1));
    addMetadata(deletado,
        ↪ ColumnMetadata.named("deletado").withIndex(5).ofType(Types.BIT).withSize(1));
    addMetadata(discenteCompleto,
        ↪ ColumnMetadata.named("discente_completo").withIndex(6).ofType(Types.BIT).withSize(1));
    addMetadata(idAreaConhecimentoDois,
        ↪ ColumnMetadata.named("id_area_conhecimento_dois").withIndex(10).ofType(Types.BIGINT).w);
    addMetadata(idAreaConhecimentoTres,
        ↪ ColumnMetadata.named("id_area_conhecimento_tres").withIndex(11).ofType(Types.BIGINT).w);
    addMetadata(idAreaConhecimentoUm,
        ↪ ColumnMetadata.named("id_area_conhecimento_um").withIndex(12).ofType(Types.BIGINT).wi);
    addMetadata(idDiscente,
        ↪ ColumnMetadata.named("id_discente").withIndex(13).ofType(Types.BIGINT).withSize(19).notN);
    addMetadata(idInstituicao,
        ↪ ColumnMetadata.named("id_instituicao").withIndex(14).ofType(Types.BIGINT).withSize(19));
    addMetadata(idOrientacaoPosDoutorado,
        ↪ ColumnMetadata.named("id_orientacao_pos_doutorado").withIndex(1).ofType(Types.BIGINT).);
    addMetadata(idPrograma,

```

```

        ↪ ColumnMetadata.named("id_programa").withIndex(15).ofType(Types.BIGINT).withSize(19).NotNull();
    addMetadata(statusOrientacao,
        ↪ ColumnMetadata.named("status_orientacao").withIndex(7).ofType(Types.VARCHAR).withSize(255));
    addMetadata(tituloDoTrabalho,
        ↪ ColumnMetadata.named("titulo_do_trabalho").withIndex(8).ofType(Types.VARCHAR).withSize(400).NotNull();
    addMetadata(tituloDoTrabalhoFiltro,
        ↪ ColumnMetadata.named("titulo_do_trabalho_filtro").withIndex(9).ofType(Types.VARCHAR).withSize(400).NotNull();
    }
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;
import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbProgramaXmlData is a Querydsl query type for QTbProgramaXmlData
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbProgramaXmlData extends
    ↪ com.mysema.query.sql.RelationalPathBase<QTbProgramaXmlData> {

    private static final long serialVersionUID = -396055519;

    public static final QTbProgramaXmlData tbProgramaXmlData = new
        ↪ QTbProgramaXmlData("tb_programa_xml_data");

    public final NumberPath<Long> idPrograma = createNumber("idPrograma", Long.class);

    public final NumberPath<Long> idProgramaXmlData = createNumber("idProgramaXmlData",
        ↪ Long.class);

    public final NumberPath<Long> idXmlData = createNumber("idXmlData", Long.class);

    public final com.mysema.query.sql.PrimaryKey<QTbProgramaXmlData>
        ↪ tbProgramaXmlDataPkey = createPrimaryKey(idProgramaXmlData);

    public QTbProgramaXmlData(String variable) {
        super(QTbProgramaXmlData.class, forVariable(variable), "public",
            ↪ "tb_programa_xml_data");
        addMetadata();
    }

    public QTbProgramaXmlData(String variable, String schema, String table) {
        super(QTbProgramaXmlData.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbProgramaXmlData(Path<? extends QTbProgramaXmlData> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_programa_xml_data");
        addMetadata();
    }

    public QTbProgramaXmlData(PathMetadata<?> metadata) {
        super(QTbProgramaXmlData.class, metadata, "public", "tb_programa_xml_data");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(idPrograma,
            ↪ ColumnMetadata.named("id_programa").withIndex(2).ofType(Types.BIGINT).withSize(19).NotNull());
        addMetadata(idProgramaXmlData,
            ↪ ColumnMetadata.named("id_programa_xml_data").withIndex(1).ofType(Types.BIGINT).withSize(19).NotNull());
        addMetadata(idXmlData,
            ↪ ColumnMetadata.named("id_xml_data").withIndex(3).ofType(Types.BIGINT).withSize(19).NotNull());
    }
}

package alpc.ufsc.querydsl.sql;

```

```

import static com.mysema.query.types.PathMetadataFactory.*;
import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbNomeCitacao is a Querydsl query type for QTbNomeCitacao
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbNomeCitacao extends
    ↳ com.mysema.query.sql.RelationalPathBase<QTbNomeCitacao> {

    private static final long serialVersionUID = 809487266;

    public static final QTbNomeCitacao tbNomeCitacao = new
        ↳ QTbNomeCitacao("tb_nome_citacao");

    public final StringPath citacao = createString("citacao");

    public final NumberPath<Long> idNomePessoa = createNumber("idNomePessoa", Long.class);

    public final NumberPath<Long> idPessoa = createNumber("idPessoa", Long.class);

    public final com.mysema.query.sql.PrimaryKey<QTbNomeCitacao> tbNomeCitacaoPkey =
        ↳ createPrimaryKey(idNomePessoa);

    public QTbNomeCitacao(String variable) {
        super(QTbNomeCitacao.class, forVariable(variable), "public", "tb_nome_citacao");
        addMetadata();
    }

    public QTbNomeCitacao(String variable, String schema, String table) {
        super(QTbNomeCitacao.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbNomeCitacao(Path<? extends QTbNomeCitacao> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_nome_citacao");
        addMetadata();
    }

    public QTbNomeCitacao(PathMetadata<?> metadata) {
        super(QTbNomeCitacao.class, metadata, "public", "tb_nome_citacao");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(citacao,
            ↳ ColumnMetadata.named("citacao").withIndex(2).ofType(Types.VARCHAR).withSize(400));
        addMetadata(idNomePessoa,
            ↳ ColumnMetadata.named("id_nome_pessoa").withIndex(1).ofType(Types.BIGINT).withSize(19).notNull());
        addMetadata(idPessoa,
            ↳ ColumnMetadata.named("id_pessoa").withIndex(3).ofType(Types.BIGINT).withSize(19).notNull());
    }
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;
import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

```

```

/**
 * QTbPessoa is a Querydsl query type for QTbPessoa
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbPessoa extends com.mysema.query.sql.RelationalPathBase<QTbPessoa> {

    private static final long serialVersionUID = 2032716330;

    public static final QTbPessoa tbPessoa = new QTbPessoa("tb_pessoa");

    public final DateTimePath<java.sql.Timestamp> dataAtualizacao =
        ↪ createDateTime("dataAtualizacao", java.sql.Timestamp.class);

    public final BooleanPath forceProcessamento = createBoolean("forceProcessamento");

    public final NumberPath<Long> idPessoa = createNumber("idPessoa", Long.class);

    public final NumberPath<Long> idPrograma = createNumber("idPrograma", Long.class);

    public final BooleanPath lattesImportado = createBoolean("lattesImportado");

    public final StringPath nome = createString("nome");

    public final StringPath nomeFiltro = createString("nomeFiltro");

    public final StringPath numeroIdentificador = createString("numeroIdentificador");

    public final com.mysema.query.sql.PrimaryKey<QTbPessoa> tbPessoaPkey =
        ↪ createPrimaryKey(idPessoa);

    public QTbPessoa(String variable) {
        super(QTbPessoa.class, forVariable(variable), "public", "tb_pessoa");
        addMetadata();
    }

    public QTbPessoa(String variable, String schema, String table) {
        super(QTbPessoa.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbPessoa(Path<? extends QTbPessoa> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_pessoa");
        addMetadata();
    }

    public QTbPessoa(PathMetadata<?> metadata) {
        super(QTbPessoa.class, metadata, "public", "tb_pessoa");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(dataAtualizacao,
            ↪ ColumnMetadata.named("data_atualizacao").withIndex(2).ofType(Types.TIMESTAMP).withSize(29).withMetadata(forceProcessamento,
            ↪ ColumnMetadata.named("force_processamento").withIndex(3).ofType(Types.BIT).withSize(1));
        addMetadata(idPessoa,
            ↪ ColumnMetadata.named("id_pessoa").withIndex(1).ofType(Types.BIGINT).withSize(19).notNull());
        addMetadata(idPrograma,
            ↪ ColumnMetadata.named("id_programa").withIndex(8).ofType(Types.BIGINT).withSize(19).notNull());
        addMetadata(lattesImportado,
            ↪ ColumnMetadata.named("lattes_importado").withIndex(4).ofType(Types.BIT).withSize(1));
        addMetadata(nome,
            ↪ ColumnMetadata.named("nome").withIndex(5).ofType(Types.VARCHAR).withSize(400).notNull());
        addMetadata(nomeFiltro,
            ↪ ColumnMetadata.named("nome_filtro").withIndex(6).ofType(Types.VARCHAR).withSize(400).notNull());
        addMetadata(numeroIdentificador,
            ↪ ColumnMetadata.named("numero_identificador").withIndex(7).ofType(Types.VARCHAR).withSize(400));
    }
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

```



```

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QDatabasechangelock is a Querydsl query type for QDatabasechangelock
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QDatabasechangelock extends
    ↪ com.mysema.query.sql.RelationalPathBase<QDatabasechangelock> {

    private static final long serialVersionUID = 234919631;

    public static final QDatabasechangelock databasechangelock = new
        ↪ QDatabasechangelock("databasechangelock");

    public final NumberPath<Integer> id = createNumber("id", Integer.class);
    public final BooleanPath locked = createBoolean("locked");
    public final StringPath lockedby = createString("lockedby");
    public final DateTimePath<java.sql.Timestamp> lockgranted = createDateTime("lockgranted",
        ↪ java.sql.Timestamp.class);

    public final com.mysema.query.sql.PrimaryKey<QDatabasechangelock>
        ↪ databasechangelockPk = createPrimaryKey(id);

    public QDatabasechangelock(String variable) {
        super(QDatabasechangelock.class, forVariable(variable), "public",
            ↪ "databasechangelock");
        addMetadata();
    }

    public QDatabasechangelock(String variable, String schema, String table) {
        super(QDatabasechangelock.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QDatabasechangelock(Path<? extends QDatabasechangelock> path) {
        super(path.getType(), path.getMetadata(), "public", "databasechangelock");
        addMetadata();
    }

    public QDatabasechangelock(PathMetadata<?> metadata) {
        super(QDatabasechangelock.class, metadata, "public", "databasechangelock");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(id,
            ↪ ColumnMetadata.named("id").withIndex(1).ofType(Types.INTEGER).withSize(10).notNull());
        addMetadata(locked,
            ↪ ColumnMetadata.named("locked").withIndex(2).ofType(Types.BIT).withSize(1).notNull());
        addMetadata(lockedby,
            ↪ ColumnMetadata.named("lockedby").withIndex(4).ofType(Types.VARCHAR).withSize(255));
        addMetadata(lockgranted,
            ↪ ColumnMetadata.named("lockgranted").withIndex(3).ofType(Types.TIMESTAMP).withSize(29).notNull());
    }
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbUsuario is a Querydsl query type for QTbUsuario

```

```


*/
@Generated("com.mysema.query.sql.codegen.MetadataSerializer")
public class QTbUsuario extends com.mysema.query.sql.RelationalPathBase<QTbUsuario> {

    private static final long serialVersionUID = -865629941;

    public static final QTbUsuario tbUsuario = new QTbUsuario("tb_usuario");

    public final BooleanPath adm = createBoolean("adm");

    public final BooleanPath criarPrograma = createBoolean("criarPrograma");

    public final StringPath email = createString("email");

    public final NumberPath<Long> idUser = createNumber("idUser", Long.class);

    public final NumberPath<Long> idXmlData = createNumber("idXmlData", Long.class);

    public final StringPath nome = createString("nome");

    public final StringPath nomeFiltro = createString("nomeFiltro");

    public final StringPath password = createString("password");

    public final com.mysema.query.sql.PrimaryKey<QTbUsuario> tbUsuarioPkey =
        ↪ createPrimaryKey(idUser);

    public QTbUsuario(String variable) {
        super(QTbUsuario.class, forVariable(variable), "public", "tb_usuario");
        addMetadata();
    }

    public QTbUsuario(String variable, String schema, String table) {
        super(QTbUsuario.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbUsuario(Path<? extends QTbUsuario> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_usuario");
        addMetadata();
    }

    public QTbUsuario(PathMetadata<?> metadata) {
        super(QTbUsuario.class, metadata, "public", "tb_usuario");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(adm,
            ↪ ColumnMetadata.named("adm").withIndex(2).ofType(Types.BIT).withSize(1));
        addMetadata(criarPrograma,
            ↪ ColumnMetadata.named("criar_programa").withIndex(3).ofType(Types.BIT).withSize(1));
        addMetadata(email,
            ↪ ColumnMetadata.named("email").withIndex(4).ofType(Types.VARCHAR).withSize(400).notNull());
        addMetadata(idUser,
            ↪ ColumnMetadata.named("id_user").withIndex(1).ofType(Types.BIGINT).withSize(19).notNull());
        addMetadata(idXmlData,
            ↪ ColumnMetadata.named("id_xml_data").withIndex(8).ofType(Types.BIGINT).withSize(19));
        addMetadata(nome,
            ↪ ColumnMetadata.named("nome").withIndex(5).ofType(Types.VARCHAR).withSize(400).notNull());
        addMetadata(nomeFiltro,
            ↪ ColumnMetadata.named("nome_filtro").withIndex(6).ofType(Types.VARCHAR).withSize(400).notNull());
        addMetadata(password,
            ↪ ColumnMetadata.named("password").withIndex(7).ofType(Types.VARCHAR).withSize(400).notNull());
    }
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;


```

```

/**
 * QTbOrientacaoDoutorado is a Querydsl query type for QTbOrientacaoDoutorado
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbOrientacaoDoutorado extends
    ↪ com.mysema.query.sql.RelationalPathBase<QTbOrientacaoDoutorado> {

    private static final long serialVersionUID = -399294969;

    public static final QTbOrientacaoDoutorado tbOrientacaoDoutorado = new
        ↪ QTbOrientacaoDoutorado("tb_orientacao_doutorado");

    public final NumberPath<Integer> anoFim = createNumber("anoFim", Integer.class);
    public final NumberPath<Integer> anoInicio = createNumber("anoInicio", Integer.class);
    public final BooleanPath bolsa = createBoolean("bolsa");
    public final BooleanPath coorientadorAtivo = createBoolean("coorientadorAtivo");
    public final BooleanPath deletado = createBoolean("deletado");
    public final BooleanPath discenteCompleto = createBoolean("discenteCompleto");
    public final NumberPath<Long> idAreaConhecimentoDois =
        ↪ createNumber("idAreaConhecimentoDois", Long.class);
    public final NumberPath<Long> idAreaConhecimentoTres =
        ↪ createNumber("idAreaConhecimentoTres", Long.class);
    public final NumberPath<Long> idAreaConhecimentoUm =
        ↪ createNumber("idAreaConhecimentoUm", Long.class);
    public final NumberPath<Long> idCoorientador = createNumber("idCoorientador", Long.class);
    public final NumberPath<Long> idDiscente = createNumber("idDiscente", Long.class);
    public final NumberPath<Long> idInstituicao = createNumber("idInstituicao", Long.class);
    public final NumberPath<Long> idInstituicaoSanduiche =
        ↪ createNumber("idInstituicaoSanduiche", Long.class);
    public final NumberPath<Long> idOrientacaoDoutorado =
        ↪ createNumber("idOrientacaoDoutorado", Long.class);
    public final NumberPath<Long> idOrientador = createNumber("idOrientador", Long.class);
    public final NumberPath<Long> idPrograma = createNumber("idPrograma", Long.class);
    public final StringPath nomeOrientadorSanduiche = createString("nomeOrientadorSanduiche");
    public final StringPath nomeOrientadorSanduicheFiltro =
        ↪ createString("nomeOrientadorSanduicheFiltro");
    public final BooleanPath orientadorAtivo = createBoolean("orientadorAtivo");
    public final StringPath statusOrientacao = createString("statusOrientacao");
    public final StringPath tituloDoTrabalho = createString("tituloDoTrabalho");
    public final StringPath tituloDoTrabalhoFiltro = createString("tituloDoTrabalhoFiltro");
    public final com.mysema.query.sql.PrimaryKey<QTbOrientacaoDoutorado>
        ↪ tbOrientacaoDoutoradoPkey = createPrimaryKey(idOrientacaoDoutorado);

    public QTbOrientacaoDoutorado(String variable) {
        super(QTbOrientacaoDoutorado.class, forVariable(variable), "public",
            ↪ "tb_orientacao_doutorado");
        addMetadata();
    }

    public QTbOrientacaoDoutorado(String variable, String schema, String table) {
        super(QTbOrientacaoDoutorado.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbOrientacaoDoutorado(Path<? extends QTbOrientacaoDoutorado> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_orientacao_doutorado");
        addMetadata();
    }
}

```

```

public QTbOrientacaoDoutorado(PathMetadata<?> metadata) {
    super(QTbOrientacaoDoutorado.class, metadata, "public", "tb_orientacao_doutorado");
    addMetadata();
}

public void addMetadata() {
    addMetadata(anoFim,
        ↪ ColumnMetadata.named("ano_fim").withIndex(2).ofType(Types.INTEGER).withSize(10));
    addMetadata(anoInicio,
        ↪ ColumnMetadata.named("ano_inicio").withIndex(3).ofType(Types.INTEGER).withSize(10));
    addMetadata(bolsa,
        ↪ ColumnMetadata.named("bolsa").withIndex(4).ofType(Types.BIT).withSize(1));
    addMetadata(coorientadorAtivo,
        ↪ ColumnMetadata.named("coorientador_ativo").withIndex(5).ofType(Types.BIT).withSize(1));
    addMetadata(deletado,
        ↪ ColumnMetadata.named("deletado").withIndex(6).ofType(Types.BIT).withSize(1));
    addMetadata(discenteCompleto,
        ↪ ColumnMetadata.named("discente_completo").withIndex(7).ofType(Types.BIT).withSize(1));
    addMetadata(idAreaConhecimentoDois,
        ↪ ColumnMetadata.named("id_area_conhecimento_dois").withIndex(14).ofType(Types.BIGINT).withSize(19));
    addMetadata(idAreaConhecimentoTres,
        ↪ ColumnMetadata.named("id_area_conhecimento_tres").withIndex(15).ofType(Types.BIGINT).withSize(19));
    addMetadata(idAreaConhecimentoUm,
        ↪ ColumnMetadata.named("id_area_conhecimento_um").withIndex(16).ofType(Types.BIGINT).withSize(19));
    addMetadata(idCoorientador,
        ↪ ColumnMetadata.named("id_coorientador").withIndex(17).ofType(Types.BIGINT).withSize(19));
    addMetadata(idDiscente,
        ↪ ColumnMetadata.named("id_discente").withIndex(18).ofType(Types.BIGINT).withSize(19).NotNull());
    addMetadata(idInstituicao,
        ↪ ColumnMetadata.named("id_instituicao").withIndex(19).ofType(Types.BIGINT).withSize(19));
    addMetadata(idInstituicaoSanduiche,
        ↪ ColumnMetadata.named("id_instituicao_sanduiche").withIndex(20).ofType(Types.BIGINT).withSize(19));
    addMetadata(idOrientacaoDoutorado,
        ↪ ColumnMetadata.named("id_orientacao_doutorado").withIndex(1).ofType(Types.BIGINT).withSize(19).NotNull());
    addMetadata(idOrientador,
        ↪ ColumnMetadata.named("id_orientador").withIndex(21).ofType(Types.BIGINT).withSize(19));
    addMetadata(idPrograma,
        ↪ ColumnMetadata.named("id_programa").withIndex(22).ofType(Types.BIGINT).withSize(19).NotNull());
    addMetadata(nomeOrientadorSanduiche,
        ↪ ColumnMetadata.named("nome_orientador_sanduiche").withIndex(8).ofType(Types.VARCHAR).withSize(255));
    addMetadata(nomeOrientadorSanduicheFiltro,
        ↪ ColumnMetadata.named("nome_orientador_sanduiche_filtro").withIndex(9).ofType(Types.VARCHAR).withSize(255));
    addMetadata(orientadorAtivo,
        ↪ ColumnMetadata.named("orientador_ativo").withIndex(10).ofType(Types.BIT).withSize(1));
    addMetadata(statusOrientacao,
        ↪ ColumnMetadata.named("status_orientacao").withIndex(11).ofType(Types.VARCHAR).withSize(255));
    addMetadata(tituloDoTrabalho,
        ↪ ColumnMetadata.named("titulo_do_trabalho").withIndex(12).ofType(Types.VARCHAR).withSize(400).NotNull());
    addMetadata(tituloDoTrabalhoFiltro,
        ↪ ColumnMetadata.named("titulo_do_trabalho_filtro").withIndex(13).ofType(Types.VARCHAR).withSize(400));
}

}

package alpc.ufsc.querydsl;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbDocente is a Querydsl query type for QTbDocente
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbDocente extends com.mysema.query.sql.RelationalPathBase<QTbDocente> {

    private static final long serialVersionUID = 1095652335;

    public static final QTbDocente tbDocente = new QTbDocente("tb_docente");

    public final StringPath categoria = createString("categoria");

```

```

public final NumberPath<Long> idDocente = createNumber("idDocente", Long.class);
public final NumberPath<Long> idPessoa = createNumber("idPessoa", Long.class);
public final StringPath modalidade = createString("modalidade");
public final com.mysema.query.sql.PrimaryKey<QTbDocente> tbDocentePkey =
    ↪ createPrimaryKey(idDocente);

public QTbDocente(String variable) {
    super(QTbDocente.class, forVariable(variable), "public", "tb_docente");
    addMetadata();
}

public QTbDocente(String variable, String schema, String table) {
    super(QTbDocente.class, forVariable(variable), schema, table);
    addMetadata();
}

public QTbDocente(Path<? extends QTbDocente> path) {
    super(path.getType(), path.getMetadata(), "public", "tb_docente");
    addMetadata();
}

public QTbDocente(PathMetadata<?> metadata) {
    super(QTbDocente.class, metadata, "public", "tb_docente");
    addMetadata();
}

public void addMetadata() {
    addMetadata(categoria,
        ↪ ColumnMetadata.named("categoria").withIndex(2).ofType(Types.VARCHAR).withSize(255));
    addMetadata(idDocente,
        ↪ ColumnMetadata.named("id_docente").withIndex(1).ofType(Types.BIGINT).withSize(19).notNull());
    addMetadata(idPessoa,
        ↪ ColumnMetadata.named("id_pessoa").withIndex(4).ofType(Types.BIGINT).withSize(19).notNull());
    addMetadata(modalidade,
        ↪ ColumnMetadata.named("modalidade").withIndex(3).ofType(Types.VARCHAR).withSize(255));
}

}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbPais is a Querydsl query type for QTbPais
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbPais extends com.mysema.query.sql.RelationalPathBase<QTbPais> {

    private static final long serialVersionUID = 542892574;

    public static final QTbPais tbPais = new QTbPais("tb_pais");

    public final NumberPath<Long> idPais = createNumber("idPais", Long.class);

    public final StringPath nome = createString("nome");

    public final StringPath nomeFiltro = createString("nomeFiltro");

    public final NumberPath<Integer> referencias = createNumber("referencias", Integer.class);

    public final com.mysema.query.sql.PrimaryKey<QTbPais> tbPaisPkey =
        ↪ createPrimaryKey(idPais);

    public QTbPais(String variable) {
        super(QTbPais.class, forVariable(variable), "public", "tb_pais");
    }
}

```

```

    addMetadata();
}

public QTbPais(String variable, String schema, String table) {
    super(QTbPais.class, forVariable(variable), schema, table);
    addMetadata();
}

public QTbPais(Path<? extends QTbPais> path) {
    super(path.getType(), path.getMetadata(), "public", "tb_pais");
    addMetadata();
}

public QTbPais(PathMetadata<?> metadata) {
    super(QTbPais.class, metadata, "public", "tb_pais");
    addMetadata();
}

public void addMetadata() {
    addMetadata(idPais,
        ↪ ColumnMetadata.named("id_pais").withIndex(1).ofType(Types.BIGINT).withSize(19).NotNull());
    addMetadata(nome,
        ↪ ColumnMetadata.named("nome").withIndex(2).ofType(Types.VARCHAR).withSize(200).NotNull());
    addMetadata(nomeFiltro,
        ↪ ColumnMetadata.named("nome_filtro").withIndex(3).ofType(Types.VARCHAR).withSize(200).NotNull());
    addMetadata(referencias,
        ↪ ColumnMetadata.named("referencias").withIndex(4).ofType(Types.INTEGER).withSize(10));
}
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;
import com.mysema.query.types.path.*;
import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;
import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QRIProjetPesquisaDocente is a Querydsl query type for QRIProjetPesquisaDocente
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QRIProjetPesquisaDocente extends
    ↪ com.mysema.query.sql.RelationalPathBase<QRIProjetPesquisaDocente> {

    private static final long serialVersionUID = 80253577;

    public static final QRIProjetPesquisaDocente rIProjetPesquisaDocente = new
        ↪ QRIProjetPesquisaDocente("rI_projeto_pesquisa_docente");

    public final NumberPath<Integer> anoFimParticipacao = createNumber("anoFimParticipacao",
        ↪ Integer.class);

    public final NumberPath<Integer> anoInicioParticipacao =
        ↪ createNumber("anoInicioParticipacao", Integer.class);

    public final NumberPath<Long> idDocente = createNumber("idDocente", Long.class);

    public final NumberPath<Long> idProjetoPesquisa = createNumber("idProjetoPesquisa",
        ↪ Long.class);

    public final NumberPath<Long> idProjetoPesquisaDocente =
        ↪ createNumber("idProjetoPesquisaDocente", Long.class);

    public final com.mysema.query.sql.PrimaryKey<QRIProjetPesquisaDocente>
        ↪ rIProjetPesquisaDocentePKey = createPrimaryKey(idProjetoPesquisaDocente);

    public QRIProjetPesquisaDocente(String variable) {
        super(QRIProjetPesquisaDocente.class, forVariable(variable), "public",
            ↪ "rI_projeto_pesquisa_docente");
        addMetadata();
    }
}

```

```

public QRIDocentePesquisaDocente(String variable, String schema, String table) {
    super(QRIDocentePesquisaDocente.class, forVariable(variable), schema, table);
    addMetadata();
}

public QRIDocentePesquisaDocente(Path<? extends QRIDocentePesquisaDocente> path) {
    super(path.getType(), path.getMetadata(), "public", "ri_projeto_pesquisa_docente");
    addMetadata();
}

public QRIDocentePesquisaDocente(PathMetadata<?> metadata) {
    super(QRIDocentePesquisaDocente.class, metadata, "public",
        ↪ "ri_projeto_pesquisa_docente");
    addMetadata();
}

public void addMetadata() {
    addMetadata(anoFimParticipacao,
        ↪ ColumnMetadata.named("ano_fim_participacao").withIndex(2).ofType(Types.INTEGER).withSize(10));
    addMetadata(anoInicioParticipacao,
        ↪ ColumnMetadata.named("ano_inicio_participacao").withIndex(3).ofType(Types.INTEGER).withSize(10));
    addMetadata(idDocente,
        ↪ ColumnMetadata.named("id_docente").withIndex(4).ofType(Types.BIGINT).withSize(19).notNullable());
    addMetadata(idProjetoPesquisa,
        ↪ ColumnMetadata.named("id_projeto_pesquisa").withIndex(5).ofType(Types.BIGINT).withSize(19).notNullable());
    addMetadata(idProjetoPesquisaDocente,
        ↪ ColumnMetadata.named("id_projeto_pesquisa_docente").withIndex(1).ofType(Types.BIGINT).withSize(19).notNullable());
}
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QRIDocenteProducao is a Querydsl query type for QRIDocenteProducao
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QRIDocenteProducao extends RelationalPathBase<QRIDocenteProducao> {
    ↪ com.mysema.query.sql.RelationalPathBase<QRIDocenteProducao> {

    private static final long serialVersionUID = -1517021482;

    public static final QRIDocenteProducao riDocenteProducao = new
        ↪ QRIDocenteProducao("ri_docente_producao");

    public final NumberPath<Long> idDocente = createNumber("idDocente", Long.class);

    public final NumberPath<Long> idDocenteProducao = createNumber("idDocenteProducao",
        ↪ Long.class);

    public final NumberPath<Long> idProducao = createNumber("idProducao", Long.class);

    public final com.mysema.query.sql.PrimaryKey<QRIDocenteProducao> riDocenteProducaoPkey
        ↪ = createPrimaryKey(idDocenteProducao);

    public QRIDocenteProducao(String variable) {
        super(QRIDocenteProducao.class, forVariable(variable), "public", "ri_docente_producao");
        addMetadata();
    }

    public QRIDocenteProducao(String variable, String schema, String table) {
        super(QRIDocenteProducao.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QRIDocenteProducao(Path<? extends QRIDocenteProducao> path) {
        super(path.getType(), path.getMetadata(), "public", "ri_docente_producao");
        addMetadata();
    }
}

```

```

    }

    public QRIDocenteProducao(PathMetadata<?> metadata) {
        super(QRIDocenteProducao.class, metadata, "public", "ri_docente_producao");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(idDocente,
            ↪ ColumnMetadata.named("id_docente").withIndex(2).ofType(Types.BIGINT).withSize(19).notNull());
        addMetadata(idDocenteProducao,
            ↪ ColumnMetadata.named("id_docente_producao").withIndex(1).ofType(Types.BIGINT).withSize(19).notNull());
        addMetadata(idProducao,
            ↪ ColumnMetadata.named("id_producao").withIndex(3).ofType(Types.BIGINT).withSize(19).notNull());
    }
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbXmlData is a Querydsl query type for QTbXmlData
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbXmlData extends com.mysema.query.sql.RelationalPathBase<QTbXmlData> {

    private static final long serialVersionUID = 1615914558;

    public static final QTbXmlData tbXmlData = new QTbXmlData("tb_xml_data");

    public final DatePath<java.sql.Date> dataAtualizacao = createDate("dataAtualizacao",
        ↪ java.sql.Date.class);

    public final DatePath<java.sql.Date> dataImportacao = createDate("dataImportacao",
        ↪ java.sql.Date.class);

    public final StringPath identificador = createString("identificador");

    public final NumberPath<Long> idXmlData = createNumber("idXmlData", Long.class);

    public final StringPath nomeCompleto = createString("nomeCompleto");

    public final StringPath nomeCompletoFiltro = createString("nomeCompletoFiltro");

    public final SimplePath<byte[]> xml = createSimple("xml", byte[].class);

    public final com.mysema.query.sql.PrimaryKey<QTbXmlData> tbXmlDataPkey =
        ↪ createPrimaryKey(idXmlData);

    public QTbXmlData(String variable) {
        super(QTbXmlData.class, forVariable(variable), "public", "tb_xml_data");
        addMetadata();
    }

    public QTbXmlData(String variable, String schema, String table) {
        super(QTbXmlData.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbXmlData(Path<? extends QTbXmlData> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_xml_data");
        addMetadata();
    }

    public QTbXmlData(PathMetadata<?> metadata) {
        super(QTbXmlData.class, metadata, "public", "tb_xml_data");
        addMetadata();
    }
}

```



```

public void addMetadata() {
    addMetadata(dataAtualizacao,
        ↪ ColumnMetadata.named("data_atualizacao").withIndex(2).ofType(Types.DATE).withSize(13));
    addMetadata(dataImportacao,
        ↪ ColumnMetadata.named("data_importacao").withIndex(3).ofType(Types.DATE).withSize(13));
    addMetadata(identificador,
        ↪ ColumnMetadata.named("identificador").withIndex(4).ofType(Types.VARCHAR).withSize(50));
    addMetadata(idXmlData,
        ↪ ColumnMetadata.named("id_xml_data").withIndex(1).ofType(Types.BIGINT).withSize(19).notNull());
    addMetadata(nomeCompleto,
        ↪ ColumnMetadata.named("nome_completo").withIndex(5).ofType(Types.VARCHAR).withSize(400));
    addMetadata(nomeCompletoFiltro,
        ↪ ColumnMetadata.named("nome_completo_filtro").withIndex(6).ofType(Types.VARCHAR).withSize(400));
    addMetadata(xml,
        ↪ ColumnMetadata.named("xml").withIndex(7).ofType(Types.BINARY).withSize(2147483647));
}

}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbDiscente is a Querydsl query type for QTbDiscente
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbDiscente extends com.mysema.query.sql.RelationalPathBase<QTbDiscente> {

    private static final long serialVersionUID = -968624718;

    public static final QTbDiscente tbDiscente = new QTbDiscente("tb_discente");

    public final NumberPath<Long> idDiscente = createNumber("idDiscente", Long.class);

    public final NumberPath<Long> idPessoa = createNumber("idPessoa", Long.class);

    public final com.mysema.query.sql.PrimaryKey<QTbDiscente> tbDiscentePkey =
        ↪ createPrimaryKey(idDiscente);

    public QTbDiscente(String variable) {
        super(QTbDiscente.class, forVariable(variable), "public", "tb_discente");
        addMetadata();
    }

    public QTbDiscente(String variable, String schema, String table) {
        super(QTbDiscente.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbDiscente(Path<? extends QTbDiscente> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_discente");
        addMetadata();
    }

    public QTbDiscente(PathMetadata<?> metadata) {
        super(QTbDiscente.class, metadata, "public", "tb_discente");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(idDiscente,
            ↪ ColumnMetadata.named("id_discente").withIndex(1).ofType(Types.BIGINT).withSize(19).notNull());
        addMetadata(idPessoa,
            ↪ ColumnMetadata.named("id_pessoa").withIndex(2).ofType(Types.BIGINT).withSize(19).notNull());
    }

}

```

```

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbPrograma is a Querydsl query type for QTbPrograma
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbPrograma extends com.mysema.query.sql.RelationalPathBase<QTbPrograma> {

    private static final long serialVersionUID = 2052380704;

    public static final QTbPrograma tbPrograma = new QTbPrograma("tb_programa");

    public final DatePath<java.sql.Date> dataInicio = createDate("dataInicio", java.sql.Date.class);

    public final NumberPath<Long> idInstituicao = createNumber("idInstituicao", Long.class);

    public final NumberPath<Long> idPrograma = createNumber("idPrograma", Long.class);

    public final StringPath nome = createString("nome");

    public final StringPath nomeFiltro = createString("nomeFiltro");

    public final com.mysema.query.sql.PrimaryKey<QTbPrograma> tbProgramaPkey =
        ↪ createPrimaryKey(idPrograma);

    public QTbPrograma(String variable) {
        super(QTbPrograma.class, forVariable(variable), "public", "tb_programa");
        addMetadata();
    }

    public QTbPrograma(String variable, String schema, String table) {
        super(QTbPrograma.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbPrograma(Path<? extends QTbPrograma> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_programa");
        addMetadata();
    }

    public QTbPrograma(PathMetadata<?> metadata) {
        super(QTbPrograma.class, metadata, "public", "tb_programa");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(dataInicio,
            ↪ ColumnMetadata.named("data_inicio").withIndex(2).ofType(Types.DATE).withSize(13).notNull());
        addMetadata(idInstituicao,
            ↪ ColumnMetadata.named("id_instituicao").withIndex(5).ofType(Types.BIGINT).withSize(19).notNull());
        addMetadata(idPrograma,
            ↪ ColumnMetadata.named("id_programa").withIndex(1).ofType(Types.BIGINT).withSize(19).notNull());
        addMetadata(nome,
            ↪ ColumnMetadata.named("nome").withIndex(3).ofType(Types.VARCHAR).withSize(400).notNull());
        addMetadata(nomeFiltro,
            ↪ ColumnMetadata.named("nome_filtro").withIndex(4).ofType(Types.VARCHAR).withSize(400).notNull());
    }
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;

```

```

import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbEstado is a Querydsl query type for QTbEstado
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbEstado extends com.mysema.query.sql.RelationalPathBase<QTbEstado> {
    private static final long serialVersionUID = 1730737129;

    public static final QTbEstado tbEstado = new QTbEstado("tb_estado");

    public final NumberPath<Integer> idEstado = createNumber("idEstado", Integer.class);

    public final StringPath nome = createString("nome");

    public final StringPath sigla = createString("sigla");

    public final com.mysema.query.sql.PrimaryKey<QTbEstado> tbEstadoPkey =
        ↪ createPrimaryKey(idEstado);

    public QTbEstado(String variable) {
        super(QTbEstado.class, forVariable(variable), "public", "tb_estado");
        addMetadata();
    }

    public QTbEstado(String variable, String schema, String table) {
        super(QTbEstado.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbEstado(Path<? extends QTbEstado> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_estado");
        addMetadata();
    }

    public QTbEstado(PathMetadata<?> metadata) {
        super(QTbEstado.class, metadata, "public", "tb_estado");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(idEstado,
            ↪ ColumnMetadata.named("id_estado").withIndex(1).ofType(Types.INTEGER).withSize(10).notNull());
        addMetadata(nome,
            ↪ ColumnMetadata.named("nome").withIndex(2).ofType(Types.VARCHAR).withSize(50));
        addMetadata(sigla,
            ↪ ColumnMetadata.named("sigla").withIndex(3).ofType(Types.VARCHAR).withSize(2));
    }
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbProgramaUser is a Querydsl query type for QTbProgramaUser
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbProgramaUser extends
    ↪ com.mysema.query.sql.RelationalPathBase<QTbProgramaUser> {

    private static final long serialVersionUID = 370419723;

```

```

public static final QTbProgramaUser tbProgramaUser = new
    ↳ QTbProgramaUser("tb_programa_user");

public final StringPath grantedAuthority = createString("grantedAuthority");

public final NumberPath<Long> idPrograma = createNumber("idPrograma", Long.class);

public final NumberPath<Long> idProgramaUser = createNumber("idProgramaUser",
    ↳ Long.class);

public final NumberPath<Long> idUser = createNumber("idUser", Long.class);

public final StringPath requestAuthority = createString("requestAuthority");

public final com.mysema.query.sql.PrimaryKey<QTbProgramaUser> tbProgramaUserPkey =
    ↳ createPrimaryKey(idProgramaUser);

public QTbProgramaUser(String variable) {
    super(QTbProgramaUser.class, forVariable(variable), "public", "tb_programa_user");
    addMetadata();
}

public QTbProgramaUser(String variable, String schema, String table) {
    super(QTbProgramaUser.class, forVariable(variable), schema, table);
    addMetadata();
}

public QTbProgramaUser(Path<? extends QTbProgramaUser> path) {
    super(path.getType(), path.getMetadata(), "public", "tb_programa_user");
    addMetadata();
}

public QTbProgramaUser(PathMetadata<?> metadata) {
    super(QTbProgramaUser.class, metadata, "public", "tb_programa_user");
    addMetadata();
}

public void addMetadata() {
    addMetadata(grantedAuthority,
        ↳ ColumnMetadata.named("granted_authority").withIndex(2).ofType(Types.VARCHAR).withSize(50));
    addMetadata(idPrograma,
        ↳ ColumnMetadata.named("id_programa").withIndex(4).ofType(Types.BIGINT).withSize(19).notNull());
    addMetadata(idProgramaUser,
        ↳ ColumnMetadata.named("id_programa_user").withIndex(1).ofType(Types.BIGINT).withSize(19).notNull());
    addMetadata(idUser,
        ↳ ColumnMetadata.named("id_user").withIndex(5).ofType(Types.BIGINT).withSize(19).notNull());
    addMetadata(requestAuthority,
        ↳ ColumnMetadata.named("request_authority").withIndex(3).ofType(Types.VARCHAR).withSize(50));
}

}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbGraduacao is a Querydsl query type for QTbGraduacao
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbGraduacao extends com.mysema.query.sql.RelationalPathBase<QTbGraduacao> {

    private static final long serialVersionUID = -1336762668;

    public static final QTbGraduacao tbGraduacao = new QTbGraduacao("tb_graduacao");

    public final NumberPath<Integer> anoFim = createNumber("anoFim", Integer.class);

    public final NumberPath<Integer> anoInicio = createNumber("anoInicio", Integer.class);

```

```

public final BooleanPath deletado = createBoolean("deletado");
public final NumberPath<Long> idDiscente = createNumber("idDiscente", Long.class);
public final NumberPath<Long> idGraduacao = createNumber("idGraduacao", Long.class);
public final NumberPath<Long> idInstituicao = createNumber("idInstituicao", Long.class);
public final NumberPath<Long> idPrograma = createNumber("idPrograma", Long.class);
public final StringPath titulo = createString("titulo");
public final com.mysema.query.sql.PrimaryKey<QTbGraduacao> tbGraduacaoPkey =
    ↪ createPrimaryKey(idGraduacao);
public QTbGraduacao(String variable) {
    super(QTbGraduacao.class, forVariable(variable), "public", "tb_graduacao");
    addMetadata();
}
public QTbGraduacao(String variable, String schema, String table) {
    super(QTbGraduacao.class, forVariable(variable), schema, table);
    addMetadata();
}
public QTbGraduacao(Path<? extends QTbGraduacao> path) {
    super(path.getType(), path.getMetadata(), "public", "tb_graduacao");
    addMetadata();
}
public QTbGraduacao(PathMetadata<?> metadata) {
    super(QTbGraduacao.class, metadata, "public", "tb_graduacao");
    addMetadata();
}
public void addMetadata() {
    addMetadata(anoFim,
        ↪ ColumnMetadata.named("ano_fim").withIndex(2).ofType(Types.INTEGER).withSize(10));
    addMetadata(anoInicio,
        ↪ ColumnMetadata.named("ano_inicio").withIndex(3).ofType(Types.INTEGER).withSize(10));
    addMetadata(deletado,
        ↪ ColumnMetadata.named("deletado").withIndex(4).ofType(Types.BIT).withSize(1));
    addMetadata(idDiscente,
        ↪ ColumnMetadata.named("id_discente").withIndex(6).ofType(Types.BIGINT).withSize(19).notNull());
    addMetadata(idGraduacao,
        ↪ ColumnMetadata.named("id_graduacao").withIndex(1).ofType(Types.BIGINT).withSize(19).notNull());
    addMetadata(idInstituicao,
        ↪ ColumnMetadata.named("id_instituicao").withIndex(7).ofType(Types.BIGINT).withSize(19));
    addMetadata(idPrograma,
        ↪ ColumnMetadata.named("id_programa").withIndex(8).ofType(Types.BIGINT).withSize(19).notNull());
    addMetadata(titulo,
        ↪ ColumnMetadata.named("titulo").withIndex(5).ofType(Types.VARCHAR).withSize(400));
}
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;
import com.mysema.query.types.path.*;
import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;
import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QRlAutorProjetoPesquisa is a Querydsl query type for QRlAutorProjetoPesquisa
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QRlAutorProjetoPesquisa extends
    ↪ com.mysema.query.sql.RelationalPathBase<QRlAutorProjetoPesquisa> {

    private static final long serialVersionUID = 592787364;

```

```

public static final QRIAutorProjetoPesquisa rIAutorProjetoPesquisa = new
    ↪ QRIAutorProjetoPesquisa("rI_autom_projeto_pesquisa");

public final NumberPath<Long> idAutor = createNumber("idAutor", Long.class);

public final NumberPath<Long> idAutorProjetoPesquisa =
    ↪ createNumber("idAutorProjetoPesquisa", Long.class);

public final NumberPath<Long> idProjetoPesquisa = createNumber("idProjetoPesquisa",
    ↪ Long.class);

public final com.mysema.query.sql.PrimaryKey<QRIAutorProjetoPesquisa>
    ↪ rIAutorProjetoPesquisaPkey = createPrimaryKey(idAutorProjetoPesquisa);

public QRIAutorProjetoPesquisa(String variable) {
    super(QRIAutorProjetoPesquisa.class, forVariable(variable), "public",
        ↪ "rI_autom_projeto_pesquisa");
    addMetadata();
}

public QRIAutorProjetoPesquisa(String variable, String schema, String table) {
    super(QRIAutorProjetoPesquisa.class, forVariable(variable), schema, table);
    addMetadata();
}

public QRIAutorProjetoPesquisa(Path<? extends QRIAutorProjetoPesquisa> path) {
    super(path.getType(), path.getMetadata(), "public", "rI_autom_projeto_pesquisa");
    addMetadata();
}

public QRIAutorProjetoPesquisa(PathMetadata<?> metadata) {
    super(QRIAutorProjetoPesquisa.class, metadata, "public", "rI_autom_projeto_pesquisa");
    addMetadata();
}

public void addMetadata() {
    addMetadata(idAutor,
        ↪ ColumnMetadata.named("id_autom").withIndex(2).ofType(Types.BIGINT).withSize(19).notNull());
    addMetadata(idAutorProjetoPesquisa,
        ↪ ColumnMetadata.named("id_autom_projeto_pesquisa").withIndex(1).ofType(Types.BIGINT).withSize(19));
    addMetadata(idProjetoPesquisa,
        ↪ ColumnMetadata.named("id_projeto_pesquisa").withIndex(3).ofType(Types.BIGINT).withSize(19));
}

}

package alpe.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbAutor is a Querydsl query type for QTbAutor
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbAutor extends com.mysema.query.sql.RelationalPathBase<QTbAutor> {

    private static final long serialVersionUID = -363445824;

    public static final QTbAutor tbAutor = new QTbAutor("tb_autom");

    public final StringPath citacao = createString("citacao");

    public final NumberPath<Long> idAutorProducao = createNumber("idAutorProducao",
        ↪ Long.class);

    public final StringPath nome = createString("nome");

    public final StringPath numeroIdentificador = createString("numeroIdentificador");

    public final com.mysema.query.sql.PrimaryKey<QTbAutor> tbAutorPkey =

```

```

        ↪ createPrimaryKey(idAutorProducao);
    public QTbAutor(String variable) {
        super(QTbAutor.class, forVariable(variable), "public", "tb_autor");
        addMetadata();
    }

    public QTbAutor(String variable, String schema, String table) {
        super(QTbAutor.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbAutor(Path<? extends QTbAutor> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_autor");
        addMetadata();
    }

    public QTbAutor(PathMetadata<?> metadata) {
        super(QTbAutor.class, metadata, "public", "tb_autor");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(citacao,
            ↪ ColumnMetadata.named("citacao").withIndex(2).ofType(Types.VARCHAR).withSize(400));
        addMetadata(idAutorProducao,
            ↪ ColumnMetadata.named("id_autor_producao").withIndex(1).ofType(Types.BIGINT).withSize(19));
        addMetadata(nome,
            ↪ ColumnMetadata.named("nome").withIndex(3).ofType(Types.VARCHAR).withSize(400).notNull());
        addMetadata(numeroIdentificador,
            ↪ ColumnMetadata.named("numero_identificador").withIndex(4).ofType(Types.VARCHAR).withSize(19));
    }
}

package alpe.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbCategoria is a Querydsl query type for QTbCategoria
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbCategoria extends com.mysema.query.sql.RelationalPathBase<QTbCategoria> {

    private static final long serialVersionUID = 1733915600;

    public static final QTbCategoria tbCategoria = new QTbCategoria("tb_categoria");

    public final StringPath sigla = createString("sigla");

    public final com.mysema.query.sql.PrimaryKey<QTbCategoria> tbCategoriaPkey =
        ↪ createPrimaryKey(sigla);

    public QTbCategoria(String variable) {
        super(QTbCategoria.class, forVariable(variable), "public", "tb_categoria");
        addMetadata();
    }

    public QTbCategoria(String variable, String schema, String table) {
        super(QTbCategoria.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbCategoria(Path<? extends QTbCategoria> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_categoria");
        addMetadata();
    }

    public QTbCategoria(PathMetadata<?> metadata) {

```

```

        super(QTbCategoria.class, metadata, "public", "tb_categoria");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(sigla,
            ↪ ColumnMetadata.named("sigla").withIndex(1).ofType(Types.VARCHAR).withSize(255).NotNull());
    }
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;
import com.mysema.query.types.Path.*;
import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbAreaConhecimento is a Querydsl query type for QTbAreaConhecimento
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbAreaConhecimento extends
    ↪ com.mysema.query.sql.RelationalPathBase<QTbAreaConhecimento> {

    private static final long serialVersionUID = 1921753916;

    public static final QTbAreaConhecimento tbAreaConhecimento = new
        ↪ QTbAreaConhecimento("tb_area_conhecimento");

    public final StringPath areaConhecimentoFiltro = createString("areaConhecimentoFiltro");

    public final NumberPath<Long> idAreaConhecimento = createNumber("idAreaConhecimento",
        ↪ Long.class);

    public final NumberPath<Long> idGrandeAreaConhecimento =
        ↪ createNumber("idGrandeAreaConhecimento", Long.class);

    public final StringPath nomeAreaConhecimento = createString("nomeAreaConhecimento");

    public final StringPath nomeEspecialidade = createString("nomeEspecialidade");

    public final StringPath nomeSubAreaConhecimento =
        ↪ createString("nomeSubAreaConhecimento");

    public final com.mysema.query.sql.PrimaryKey<QTbAreaConhecimento>
        ↪ tbAreaConhecimentoPkey = createPrimaryKey(idAreaConhecimento);

    public QTbAreaConhecimento(String variable) {
        super(QTbAreaConhecimento.class, forVariable(variable), "public",
            ↪ "tb_area_conhecimento");
        addMetadata();
    }

    public QTbAreaConhecimento(String variable, String schema, String table) {
        super(QTbAreaConhecimento.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbAreaConhecimento(Path<? extends QTbAreaConhecimento> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_area_conhecimento");
        addMetadata();
    }

    public QTbAreaConhecimento(PathMetadata<?> metadata) {
        super(QTbAreaConhecimento.class, metadata, "public", "tb_area_conhecimento");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(areaConhecimentoFiltro,
            ↪ ColumnMetadata.named("area_conhecimento_filtro").withIndex(2).ofType(Types.VARCHAR).withSize(80)
        addMetadata(idAreaConhecimento,

```



```

        ↪ ColumnMetadata.named("id_area_conhecimento").withIndex(1).ofType(Types.BIGINT).withSize
addMetadata(idGrandeAreaConhecimento,
        ↪ ColumnMetadata.named("id_grande_area_conhecimento").withIndex(3).ofType(Types.BIGINT).
addMetadata(nomeAreaConhecimento,
        ↪ ColumnMetadata.named("nome_area_conhecimento").withIndex(4).ofType(Types.VARCHAR).wi
addMetadata(nomeEspecialidade,
        ↪ ColumnMetadata.named("nome_especialidade").withIndex(5).ofType(Types.VARCHAR).withSize
addMetadata(nomeSubAreaConhecimento,
        ↪ ColumnMetadata.named("nome_sub_area_conhecimento").withIndex(6).ofType(Types.VARCHA
    }
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;
import com.mysema.query.types.path.*;
import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QRILinhaPesquisaDocente is a Querydsl query type for QRILinhaPesquisaDocente
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QRILinhaPesquisaDocente extends
    ↪ com.mysema.query.sql.RelationalPathBase<QRILinhaPesquisaDocente> {

    private static final long serialVersionUID = 1292674370;

    public static final QRILinhaPesquisaDocente rILinhaPesquisaDocente = new
        ↪ QRILinhaPesquisaDocente("r1_linha_pesquisa_docente");

    public final NumberPath<Long> idDocente = createNumber("idDocente", Long.class);

    public final NumberPath<Long> idLinhaPesquisa = createNumber("idLinhaPesquisa",
        ↪ Long.class);

    public final NumberPath<Long> idLinhaPesquisaDocente =
        ↪ createNumber("idLinhaPesquisaDocente", Long.class);

    public final com.mysema.query.sql.PrimaryKey<QRILinhaPesquisaDocente>
        ↪ rLinhaPesquisaDocentePkey = createPrimaryKey(idLinhaPesquisaDocente);

    public QRILinhaPesquisaDocente(String variable) {
        super(QRILinhaPesquisaDocente.class, forVariable(variable), "public",
            ↪ "r1_linha_pesquisa_docente");
        addMetadata();
    }

    public QRILinhaPesquisaDocente(String variable, String schema, String table) {
        super(QRILinhaPesquisaDocente.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QRILinhaPesquisaDocente(Path<? extends QRILinhaPesquisaDocente> path) {
        super(path.getType(), path.getMetadata(), "public", "r1_linha_pesquisa_docente");
        addMetadata();
    }

    public QRILinhaPesquisaDocente(PathMetadata<?> metadata) {
        super(QRILinhaPesquisaDocente.class, metadata, "public", "r1_linha_pesquisa_docente");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(idDocente,
            ↪ ColumnMetadata.named("id_docente").withIndex(2).ofType(Types.BIGINT).withSize(19).notNull
        addMetadata(idLinhaPesquisa,
            ↪ ColumnMetadata.named("id_linha_pesquisa").withIndex(3).ofType(Types.BIGINT).withSize(19)
        addMetadata(idLinhaPesquisaDocente,
            ↪ ColumnMetadata.named("id_linha_pesquisa_docente").withIndex(1).ofType(Types.BIGINT).wit
    }
}

```

```

}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbOrientacaoMestrado is a Querydsl query type for QTbOrientacaoMestrado
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbOrientacaoMestrado extends
    ↳ com.mysema.query.sql.RelationalPathBase<QTbOrientacaoMestrado> {

    private static final long serialVersionUID = 782655275;

    public static final QTbOrientacaoMestrado tbOrientacaoMestrado = new
        ↳ QTbOrientacaoMestrado("tb_orientacao_mestrado");

    public final NumberPath<Integer> anoFim = createNumber("anoFim", Integer.class);
    public final NumberPath<Integer> anoInicio = createNumber("anoInicio", Integer.class);
    public final BooleanPath bolsa = createBoolean("bolsa");
    public final BooleanPath coorientadorAtivo = createBoolean("coorientadorAtivo");
    public final BooleanPath deletado = createBoolean("deletado");
    public final BooleanPath discenteCompleto = createBoolean("discenteCompleto");
    public final NumberPath<Long> idAreaConhecimentoDois =
        ↳ createNumber("idAreaConhecimentoDois", Long.class);
    public final NumberPath<Long> idAreaConhecimentoTres =
        ↳ createNumber("idAreaConhecimentoTres", Long.class);
    public final NumberPath<Long> idAreaConhecimentoUm =
        ↳ createNumber("idAreaConhecimentoUm", Long.class);
    public final NumberPath<Long> idCoorientador = createNumber("idCoorientador", Long.class);
    public final NumberPath<Long> idDiscente = createNumber("idDiscente", Long.class);
    public final NumberPath<Long> idInstituicao = createNumber("idInstituicao", Long.class);
    public final NumberPath<Long> idOrientacaoMestrado =
        ↳ createNumber("idOrientacaoMestrado", Long.class);
    public final NumberPath<Long> idOrientador = createNumber("idOrientador", Long.class);
    public final NumberPath<Long> idPrograma = createNumber("idPrograma", Long.class);
    public final BooleanPath orientadorAtivo = createBoolean("orientadorAtivo");
    public final StringPath statusOrientacao = createString("statusOrientacao");
    public final StringPath tituloDoTrabalho = createString("tituloDoTrabalho");
    public final StringPath tituloDoTrabalhoFiltro = createString("tituloDoTrabalhoFiltro");

    public final com.mysema.query.sql.PrimaryKey<QTbOrientacaoMestrado>
        ↳ tbOrientacaoMestradoPkey = createPrimaryKey(idOrientacaoMestrado);

    public QTbOrientacaoMestrado(String variable) {
        super(QTbOrientacaoMestrado.class, forVariable(variable), "public",
            ↳ "tb_orientacao_mestrado");
        addMetadata();
    }

    public QTbOrientacaoMestrado(String variable, String schema, String table) {

```

```

        super(QTbOrientacaoMestrado.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbOrientacaoMestrado(Path<? extends QTbOrientacaoMestrado> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_orientacao_mestrado");
        addMetadata();
    }

    public QTbOrientacaoMestrado(PathMetadata<?> metadata) {
        super(QTbOrientacaoMestrado.class, metadata, "public", "tb_orientacao_mestrado");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(anoFim,
            ↪ ColumnMetadata.named("ano_fim").withIndex(2).ofType(Types.INTEGER).withSize(10));
        addMetadata(anoInicio,
            ↪ ColumnMetadata.named("ano_inicio").withIndex(3).ofType(Types.INTEGER).withSize(10));
        addMetadata(bolsa,
            ↪ ColumnMetadata.named("bolsa").withIndex(4).ofType(Types.BIT).withSize(1));
        addMetadata(coorientadorAtivo,
            ↪ ColumnMetadata.named("coorientador_ativo").withIndex(5).ofType(Types.BIT).withSize(1));
        addMetadata(deletado,
            ↪ ColumnMetadata.named("deletado").withIndex(6).ofType(Types.BIT).withSize(1));
        addMetadata(discenteCompleto,
            ↪ ColumnMetadata.named("discente_completo").withIndex(7).ofType(Types.BIT).withSize(1));
        addMetadata(idAreaConhecimentoDois,
            ↪ ColumnMetadata.named("id_area_conhecimento_dois").withIndex(12).ofType(Types.BIGINT).withSize(19));
        addMetadata(idAreaConhecimentoTres,
            ↪ ColumnMetadata.named("id_area_conhecimento_tres").withIndex(13).ofType(Types.BIGINT).withSize(19));
        addMetadata(idAreaConhecimentoUm,
            ↪ ColumnMetadata.named("id_area_conhecimento_um").withIndex(14).ofType(Types.BIGINT).withSize(19));
        addMetadata(idCoorientador,
            ↪ ColumnMetadata.named("id_coorientador").withIndex(15).ofType(Types.BIGINT).withSize(19));
        addMetadata(idDiscente,
            ↪ ColumnMetadata.named("id_discente").withIndex(16).ofType(Types.BIGINT).withSize(19).notNull());
        addMetadata(idInstituicao,
            ↪ ColumnMetadata.named("id_instituicao").withIndex(17).ofType(Types.BIGINT).withSize(19));
        addMetadata(idOrientacaoMestrado,
            ↪ ColumnMetadata.named("id_orientacao_mestrado").withIndex(1).ofType(Types.BIGINT).withSize(19));
        addMetadata(idOrientador,
            ↪ ColumnMetadata.named("id_orientador").withIndex(18).ofType(Types.BIGINT).withSize(19));
        addMetadata(idPrograma,
            ↪ ColumnMetadata.named("id_programa").withIndex(19).ofType(Types.BIGINT).withSize(19).notNull());
        addMetadata(orientadorAtivo,
            ↪ ColumnMetadata.named("orientador_ativo").withIndex(8).ofType(Types.BIT).withSize(1));
        addMetadata(statusOrientacao,
            ↪ ColumnMetadata.named("status_orientacao").withIndex(9).ofType(Types.VARCHAR).withSize(2));
        addMetadata(tituloDoTrabalho,
            ↪ ColumnMetadata.named("titulo_do_trabalho").withIndex(10).ofType(Types.VARCHAR).withSize(255));
        addMetadata(tituloDoTrabalhoFiltro,
            ↪ ColumnMetadata.named("titulo_do_trabalho_filtro").withIndex(11).ofType(Types.VARCHAR).withSize(255));
    }
}

package alpe.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.Path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbProjetoPesquisa is a Querydsl query type for QTbProjetoPesquisa
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbProjetoPesquisa extends
    ↪ com.mysema.query.sql.RelationalPathBase<QTbProjetoPesquisa> {

    private static final long serialVersionUID = -1909473131;

```

```

public static final QTbProjetoPesquisa tbProjetoPesquisa = new
    ↪ QTbProjetoPesquisa("tb_projeto_pesquisa");

public final NumberPath<Integer> anoFim = createNumber("anoFim", Integer.class);
public final NumberPath<Integer> anoInicio = createNumber("anoInicio", Integer.class);
public final NumberPath<Long> idPrograma = createNumber("idPrograma", Long.class);
public final NumberPath<Long> idProjetoPesquisa = createNumber("idProjetoPesquisa",
    ↪ Long.class);

public final StringPath nome = createString("nome");
public final StringPath nomeFiltro = createString("nomeFiltro");

public final com.mysema.query.sql.PrimaryKey<QTbProjetoPesquisa> tbProjetoPesquisaPkey =
    ↪ createPrimaryKey(idProjetoPesquisa);

public QTbProjetoPesquisa(String variable) {
    super(QTbProjetoPesquisa.class, forVariable(variable), "public", "tb_projeto_pesquisa");
    addMetadata();
}

public QTbProjetoPesquisa(String variable, String schema, String table) {
    super(QTbProjetoPesquisa.class, forVariable(variable), schema, table);
    addMetadata();
}

public QTbProjetoPesquisa(Path<? extends QTbProjetoPesquisa> path) {
    super(path.getType(), path.getMetadata(), "public", "tb_projeto_pesquisa");
    addMetadata();
}

public QTbProjetoPesquisa(PathMetadata<?> metadata) {
    super(QTbProjetoPesquisa.class, metadata, "public", "tb_projeto_pesquisa");
    addMetadata();
}

public void addMetadata() {
    addMetadata(anoFim,
        ↪ ColumnMetadata.named("ano_fim").withIndex(2).ofType(Types.INTEGER).withSize(10));
    addMetadata(anoInicio,
        ↪ ColumnMetadata.named("ano_inicio").withIndex(3).ofType(Types.INTEGER).withSize(10));
    addMetadata(idPrograma,
        ↪ ColumnMetadata.named("id_programa").withIndex(6).ofType(Types.BIGINT).withSize(19).notNull());
    addMetadata(idProjetoPesquisa,
        ↪ ColumnMetadata.named("id_projeto_pesquisa").withIndex(1).ofType(Types.BIGINT).withSize(19).notNull());
    addMetadata(nome,
        ↪ ColumnMetadata.named("nome").withIndex(4).ofType(Types.VARCHAR).withSize(400).notNull());
    addMetadata(nomeFiltro,
        ↪ ColumnMetadata.named("nome_filtro").withIndex(5).ofType(Types.VARCHAR).withSize(400).notNull());
}
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;
import com.mysema.query.types.path.*;
import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbModalidade is a Querydsl query type for QTbModalidade
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbModalidade extends com.mysema.query.sql.RelationalPathBase<QTbModalidade> {

    private static final long serialVersionUID = -520946659;

    public static final QTbModalidade tbModalidade = new QTbModalidade("tb_modalidade");

```

```

public final StringPath descricao = createString("descricao");
public final StringPath sigla = createString("sigla");
public final com.mysema.query.sql.PrimaryKey<QTbModalidade> tbModalidadePkey =
    ↪ createPrimaryKey(sigla);

public QTbModalidade(String variable) {
    super(QTbModalidade.class, forVariable(variable), "public", "tb_modalidade");
    addMetadata();
}

public QTbModalidade(String variable, String schema, String table) {
    super(QTbModalidade.class, forVariable(variable), schema, table);
    addMetadata();
}

public QTbModalidade(Path<? extends QTbModalidade> path) {
    super(path.getType(), path.getMetadata(), "public", "tb_modalidade");
    addMetadata();
}

public QTbModalidade(PathMetadata<?> metadata) {
    super(QTbModalidade.class, metadata, "public", "tb_modalidade");
    addMetadata();
}

public void addMetadata() {
    addMetadata(descricao,
        ↪ ColumnMetadata.named("descricao").withIndex(2).ofType(Types.VARCHAR).withSize(100));
    addMetadata(sigla,
        ↪ ColumnMetadata.named("sigla").withIndex(1).ofType(Types.VARCHAR).withSize(2).NotNull());
}
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;
import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbAtuacaoProfissional is a Querydsl query type for QTbAtuacaoProfissional
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbAtuacaoProfissional extends
    ↪ com.mysema.query.sql.RelationalPathBase<QTbAtuacaoProfissional> {

    private static final long serialVersionUID = 99921554;

    public static final QTbAtuacaoProfissional tbAtuacaoProfissional = new
        ↪ QTbAtuacaoProfissional("tb_atuacao_profissional");

    public final NumberPath<Integer> anoFim = createNumber("anoFim", Integer.class);
    public final NumberPath<Integer> anoInicio = createNumber("anoInicio", Integer.class);
    public final NumberPath<Long> idAtuacaoProfissional =
        ↪ createNumber("idAtuacaoProfissional", Long.class);
    public final NumberPath<Long> idDocente = createNumber("idDocente", Long.class);
    public final NumberPath<Integer> mesFim = createNumber("mesFim", Integer.class);
    public final NumberPath<Integer> mesInicio = createNumber("mesInicio", Integer.class);
    public final StringPath tipoAtuacaoProfissional = createString("tipoAtuacaoProfissional");

    public final com.mysema.query.sql.PrimaryKey<QTbAtuacaoProfissional>
        ↪ tbAtuacaoProfissionalPkey = createPrimaryKey(idAtuacaoProfissional);

```

```

public QTbAtuacaoProfissional(String variable) {
    super(QTbAtuacaoProfissional.class, forVariable(variable), "public",
        ↪ "tb_atuacao_profissional");
    addMetadata();
}

public QTbAtuacaoProfissional(String variable, String schema, String table) {
    super(QTbAtuacaoProfissional.class, forVariable(variable), schema, table);
    addMetadata();
}

public QTbAtuacaoProfissional(Path<? extends QTbAtuacaoProfissional> path) {
    super(path.getType(), path.getMetadata(), "public", "tb_atuacao_profissional");
    addMetadata();
}

public QTbAtuacaoProfissional(PathMetadata<?> metadata) {
    super(QTbAtuacaoProfissional.class, metadata, "public", "tb_atuacao_profissional");
    addMetadata();
}

public void addMetadata() {
    addMetadata(anoFim,
        ↪ ColumnMetadata.named("ano_fim").withIndex(2).ofType(Types.INTEGER).withSize(10));
    addMetadata(anoInicio,
        ↪ ColumnMetadata.named("ano_inicio").withIndex(3).ofType(Types.INTEGER).withSize(10));
    addMetadata(idAtuacaoProfissional,
        ↪ ColumnMetadata.named("id_atuacao_profissional").withIndex(1).ofType(Types.BIGINT).withSize(19).notNull());
    addMetadata(idDocente,
        ↪ ColumnMetadata.named("id_docente").withIndex(7).ofType(Types.BIGINT).withSize(19).notNull());
    addMetadata(mesFim,
        ↪ ColumnMetadata.named("mes_fim").withIndex(4).ofType(Types.INTEGER).withSize(10));
    addMetadata(mesInicio,
        ↪ ColumnMetadata.named("mes_inicio").withIndex(5).ofType(Types.INTEGER).withSize(10));
    addMetadata(tipoAtuacaoProfissional,
        ↪ ColumnMetadata.named("tipo_atuacao_profissional").withIndex(6).ofType(Types.VARCHAR).withSize(25));
}
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;
import com.mysema.query.types.path.*;
import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QRIAutorProducao is a Querydsl query type for QRIAutorProducao
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QRIAutorProducao extends
    ↪ com.mysema.query.sql.RelationalPathBase<QRIAutorProducao> {

    private static final long serialVersionUID = -1361985369;

    public static final QRIAutorProducao rIAutorProducao = new
        ↪ QRIAutorProducao("ri_autor_producao");

    public final NumberPath<Long> idAutor = createNumber("idAutor", Long.class);

    public final NumberPath<Long> idAutorProducao = createNumber("idAutorProducao",
        ↪ Long.class);

    public final NumberPath<Long> idProducao = createNumber("idProducao", Long.class);

    public final com.mysema.query.sql.PrimaryKey<QRIAutorProducao> rIAutorProducaoPkey =
        ↪ createPrimaryKey(idAutorProducao);

    public QRIAutorProducao(String variable) {
        super(QRIAutorProducao.class, forVariable(variable), "public", "ri_autor_producao");
        addMetadata();
    }
}

```

```

public QRIAutorProducao(String variable, String schema, String table) {
    super(QRIAutorProducao.class, forVariable(variable), schema, table);
    addMetadata();
}

public QRIAutorProducao(Path<? extends QRIAutorProducao> path) {
    super(path.getType(), path.getMetadata(), "public", "ri_autor_producao");
    addMetadata();
}

public QRIAutorProducao(PathMetadata<?> metadata) {
    super(QRIAutorProducao.class, metadata, "public", "ri_autor_producao");
    addMetadata();
}

public void addMetadata() {
    addMetadata(idAutor,
        ↪ ColumnMetadata.named("id_autor").withIndex(2).ofType(Types.BIGINT).withSize(19).notNull(),
    addMetadata(idAutorProducao,
        ↪ ColumnMetadata.named("id_autor_producao").withIndex(1).ofType(Types.BIGINT).withSize(19),
    addMetadata(idProducao,
        ↪ ColumnMetadata.named("id_producao").withIndex(3).ofType(Types.BIGINT).withSize(19).notNull());
}

}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbSystemConfig is a Querydsl query type for QTbSystemConfig
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbSystemConfig extends
    ↪ com.mysema.query.sql.RelationalPathBase<QTbSystemConfig> {

    private static final long serialVersionUID = -2054566252;

    public static final QTbSystemConfig tbSystemConfig = new
        ↪ QTbSystemConfig("tb_system_config");

    public final StringPath key = createString("key");

    public final StringPath value = createString("value");

    public final com.mysema.query.sql.PrimaryKey<QTbSystemConfig> tbSystemConfigPkey =
        ↪ createPrimaryKey(key);

    public QTbSystemConfig(String variable) {
        super(QTbSystemConfig.class, forVariable(variable), "public", "tb_system_config");
        addMetadata();
    }

    public QTbSystemConfig(String variable, String schema, String table) {
        super(QTbSystemConfig.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbSystemConfig(Path<? extends QTbSystemConfig> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_system_config");
        addMetadata();
    }

    public QTbSystemConfig(PathMetadata<?> metadata) {
        super(QTbSystemConfig.class, metadata, "public", "tb_system_config");
        addMetadata();
    }

}

```

```

    public void addMetadata() {
        addMetadata(key,
            ↪ ColumnMetadata.named("key").withIndex(1).ofType(Types.VARCHAR).withSize(255).NotNull());
        addMetadata(value,
            ↪ ColumnMetadata.named("value").withIndex(2).ofType(Types.VARCHAR).withSize(255));
    }
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;
import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbInstituicao is a Querydsl query type for QTbInstituicao
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbInstituicao extends com.mysema.query.sql.RelationalPathBase<QTbInstituicao> {

    private static final long serialVersionUID = -800133943;

    public static final QTbInstituicao tbInstituicao = new QTbInstituicao("tb_instituicao");

    public final StringPath estado = createString("estado");

    public final NumberPath<Long> idInstituicao = createNumber("idInstituicao", Long.class);

    public final NumberPath<Long> idPais = createNumber("idPais", Long.class);

    public final StringPath nome = createString("nome");

    public final StringPath nomeFiltro = createString("nomeFiltro");

    public final StringPath nomeSiglaFiltro = createString("nomeSiglaFiltro");

    public final NumberPath<Integer> referencias = createNumber("referencias", Integer.class);

    public final StringPath sigla = createString("sigla");

    public final com.mysema.query.sql.PrimaryKey<QTbInstituicao> tbInstituicaoPkey =
        ↪ createPrimaryKey(idInstituicao);

    public QTbInstituicao(String variable) {
        super(QTbInstituicao.class, forVariable(variable), "public", "tb_instituicao");
        addMetadata();
    }

    public QTbInstituicao(String variable, String schema, String table) {
        super(QTbInstituicao.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbInstituicao(Path<? extends QTbInstituicao> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_instituicao");
        addMetadata();
    }

    public QTbInstituicao(PathMetadata<?> metadata) {
        super(QTbInstituicao.class, metadata, "public", "tb_instituicao");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(estado,
            ↪ ColumnMetadata.named("estado").withIndex(2).ofType(Types.VARCHAR).withSize(255));
        addMetadata(idInstituicao,
            ↪ ColumnMetadata.named("id_instituicao").withIndex(1).ofType(Types.BIGINT).withSize(19).NotNull());
        addMetadata(idPais,
            ↪ ColumnMetadata.named("id_pais").withIndex(8).ofType(Types.BIGINT).withSize(19));
        addMetadata(nome,

```



```

        ↪ ColumnMetadata.named("nome").withIndex(3).ofType(Types.VARCHAR).withSize(400).notNull()
    addMetadata(nomeFiltro,
        ↪ ColumnMetadata.named("nome_filtro").withIndex(4).ofType(Types.VARCHAR).withSize(400).notNull()
    addMetadata(nomeSiglaFiltro,
        ↪ ColumnMetadata.named("nome_sigla_filtro").withIndex(5).ofType(Types.VARCHAR).withSize(400).notNull()
    addMetadata(referencias,
        ↪ ColumnMetadata.named("referencias").withIndex(6).ofType(Types.INTEGER).withSize(10));
    addMetadata(sigla,
        ↪ ColumnMetadata.named("sigla").withIndex(7).ofType(Types.VARCHAR).withSize(50));
    }
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.forVariable;
import java.sql.Types;
import javax.annotation.Generated;

import com.mysema.query.sql.ColumnMetadata;
import com.mysema.query.types.Path;
import com.mysema.query.types.PathMetadata;
import com.mysema.query.types.path.NumberPath;

/**
 * QtbUsuario is a Querydsl query type for QtbUsuario
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QViewPercencePrograma extends
    ↪ com.mysema.query.sql.RelationalPathBase<QViewPercencePrograma> {

    private static final long serialVersionUID = -865629941;

    public static final QViewPercencePrograma viewPercencePrograma = new
        ↪ QViewPercencePrograma("view_percence_programa");

    public final NumberPath<Long> idDocente = this.createNumber("idDocente", Long.class);

    public final NumberPath<Long> idPrograma = this.createNumber("idPrograma",
        ↪ Long.class);

    public final com.mysema.query.sql.PrimaryKey<QViewPercencePrograma> tbUsuarioPkey =
        ↪ this.createPrimaryKey(this.idDocente);

    public QViewPercencePrograma(String variable) {
        super(QViewPercencePrograma.class, forVariable(variable), "public",
            ↪ "view_percence_programa");
        this.addMetadata();
    }

    public QViewPercencePrograma(String variable, String schema, String table) {
        super(QViewPercencePrograma.class, forVariable(variable), schema, table);
        this.addMetadata();
    }

    public QViewPercencePrograma(Path<? extends QViewPercencePrograma> path) {
        super(path.getType(), path.getMetadata(), "public", "view_percence_programa");
        this.addMetadata();
    }

    public QViewPercencePrograma(PathMetadata<?> metadata) {
        super(QViewPercencePrograma.class, metadata, "public",
            ↪ "view_percence_programa");
        this.addMetadata();
    }

    public void addMetadata() {
        this.addMetadata(this.idDocente,
            ↪ ColumnMetadata.named("id_docente").withIndex(1).ofType(Types.BIGINT).withSize(19));
        this.addMetadata(this.idPrograma,
            ↪ ColumnMetadata.named("id_programa").withIndex(2).ofType(Types.BIGINT).withSize(19));
    }
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;
import com.mysema.query.types.path.*;

```

```

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

/**
 * QTbLinhaPesquisa is a Querydsl query type for QTbLinhaPesquisa
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbLinhaPesquisa extends
    ↳ com.mysema.query.sql.RelationalPathBase<QTbLinhaPesquisa> {

    private static final long serialVersionUID = -954450180;

    public static final QTbLinhaPesquisa tbLinhaPesquisa = new
        ↳ QTbLinhaPesquisa("tb_linha_pesquisa");

    public final NumberPath<Long> idLinhaPesquisa = createNumber("idLinhaPesquisa",
        ↳ Long.class);

    public final NumberPath<Long> idPrograma = createNumber("idPrograma", Long.class);

    public final StringPath titulo = createString("titulo");

    public final StringPath tituloFiltro = createString("tituloFiltro");

    public final com.mysema.query.sql.PrimaryKey<QTbLinhaPesquisa> tbLinhaPesquisaPkey =
        ↳ createPrimaryKey(idLinhaPesquisa);

    public QTbLinhaPesquisa(String variable) {
        super(QTbLinhaPesquisa.class, forVariable(variable), "public", "tb_linha_pesquisa");
        addMetadata();
    }

    public QTbLinhaPesquisa(String variable, String schema, String table) {
        super(QTbLinhaPesquisa.class, forVariable(variable), schema, table);
        addMetadata();
    }

    public QTbLinhaPesquisa(Path<? extends QTbLinhaPesquisa> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_linha_pesquisa");
        addMetadata();
    }

    public QTbLinhaPesquisa(PathMetadata<?> metadata) {
        super(QTbLinhaPesquisa.class, metadata, "public", "tb_linha_pesquisa");
        addMetadata();
    }

    public void addMetadata() {
        addMetadata(idLinhaPesquisa,
            ↳ ColumnMetadata.named("id_linha_pesquisa").withIndex(1).ofType(Types.BIGINT).withSize(19).notNull());
        addMetadata(idPrograma,
            ↳ ColumnMetadata.named("id_programa").withIndex(4).ofType(Types.BIGINT).withSize(19).notNull());
        addMetadata(titulo,
            ↳ ColumnMetadata.named("titulo").withIndex(2).ofType(Types.VARCHAR).withSize(400).notNull());
        addMetadata(tituloFiltro,
            ↳ ColumnMetadata.named("titulo_filtro").withIndex(3).ofType(Types.VARCHAR).withSize(400).notNull());
    }
}

package alpc.ufsc.querydsl.sql;

import static com.mysema.query.types.PathMetadataFactory.*;

import com.mysema.query.types.path.*;

import com.mysema.query.types.PathMetadata;
import javax.annotation.Generated;
import com.mysema.query.types.Path;

import com.mysema.query.sql.ColumnMetadata;
import java.sql.Types;

```

```

/**
 * QTbProducao is a Querydsl query type for QTbProducao
 */
@Generated("com.mysema.query.sql.codegen.MetaDataSerializer")
public class QTbProducao extends com.mysema.query.sql.RelationalPathBase<QTbProducao> {
    private static final long serialVersionUID = 2049701078;
    public static final QTbProducao tbProducao = new QTbProducao("tb_producao");
    public final NumberPath<Integer> ano = createNumber("ano", Integer.class);
    public final NumberPath<Long> idProducao = createNumber("idProducao", Long.class);
    public final NumberPath<Long> idPrograma = createNumber("idPrograma", Long.class);
    public final StringPath tipoProducao = createString("tipoProducao");
    public final StringPath titulo = createString("titulo");
    public final com.mysema.query.sql.PrimaryKey<QTbProducao> tbProducaoPkey =
        ↪ createPrimaryKey(idProducao);
    public QTbProducao(String variable) {
        super(QTbProducao.class, forVariable(variable), "public", "tb_producao");
        addMetadata();
    }
    public QTbProducao(String variable, String schema, String table) {
        super(QTbProducao.class, forVariable(variable), schema, table);
        addMetadata();
    }
    public QTbProducao(Path<? extends QTbProducao> path) {
        super(path.getType(), path.getMetadata(), "public", "tb_producao");
        addMetadata();
    }
    public QTbProducao(PathMetadata<?> metadata) {
        super(QTbProducao.class, metadata, "public", "tb_producao");
        addMetadata();
    }
    public void addMetadata() {
        addMetadata(ano,
            ↪ ColumnMetadata.named("ano").withIndex(2).ofType(Types.INTEGER).withSize(10));
        addMetadata(idProducao,
            ↪ ColumnMetadata.named("id_producao").withIndex(1).ofType(Types.BIGINT).withSize(19).notNull());
        addMetadata(idPrograma,
            ↪ ColumnMetadata.named("id_programa").withIndex(5).ofType(Types.BIGINT).withSize(19).notNull());
        addMetadata(tipoProducao,
            ↪ ColumnMetadata.named("tipo_producao").withIndex(3).ofType(Types.VARCHAR).withSize(255));
        addMetadata(titulo,
            ↪ ColumnMetadata.named("titulo").withIndex(4).ofType(Types.VARCHAR).withSize(400).notNull());
    }
}

package alpc.ufsc;

import java.util.Calendar;
import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.apache.lucene.search.BooleanClause.Occur;
import org.apache.lucene.search.BooleanQuery;
import org.hibernate.search.ProjectionConstants;
import org.hibernate.search.jpa.FullTextEntityManager;
import org.hibernate.search.jpa.FullTextQuery;
import org.hibernate.search.jpa.Search;
import org.hibernate.search.query.dsl.QueryBuilder;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

import alpc.ufsc.common.exception.InvalidDTOException;
import alpc.ufsc.common.exception.InvalidOperationException;

```

```

import alpc.ufsc.domain.instituicao.dto.InstituicaoSaveDTO;
import alpc.ufsc.entity.domain.estado.Estado;
import alpc.ufsc.entity.pessoa.orientacao.mestrado.OrientacaoMestradoEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.login.authority.Authority;
import alpc.ufsc.login.user.UserService;
import alpc.ufsc.login.user.dto.ProgramaUserSaveDTO;
import alpc.ufsc.login.user.dto.UserFormDTO;
import alpc.ufsc.programa.ProgramaService;
import alpc.ufsc.programa.dto.ProgramaFormDTO;

// @RestController
// @RequestMapping("/test")
public class TestRestController {

    @Autowired
    private ProgramaService pService;

    @Autowired
    private UserService userService;

    @PersistenceContext
    private EntityManager em;

    @RequestMapping(value = "/init", method = RequestMethod.GET)
    public ResponseEntity<String> initDefaultUserAndProgram() throws InvalidDTOException,
        ↪ InvalidOperationException {

        InstituicaoSaveDTO insDto = new InstituicaoSaveDTO();
        insDto.setEstado(Estado.SANTA_CATARINA);
        insDto.setNome("Universidade Federal De Santa Catarina");
        insDto.setNomePais("Brasil");
        insDto.setNomePaisFiltro(QueryUtils.lowerCaseStripAccents(insDto.getNomePais()));
        insDto.setSigla("UFSC");

        ProgramaFormDTO pDTO = new ProgramaFormDTO();
        Calendar calendar = Calendar.getInstance();
        calendar.set(Calendar.YEAR, 2005);
        calendar.set(Calendar.DAY_OF_MONTH, 5);
        calendar.set(Calendar.MONTH, 0);
        pDTO.setDataInicio(calendar.getTime());
        pDTO.setNome("Programa de Ps-Graduao em Cincias da Computao");
        pDTO.setInstituicao(insDto);
        Long id = this.pService.save(pDTO);

        UserFormDTO dto = new UserFormDTO();
        dto.setAdm(true);
        dto.setName("Adm");
        dto.setEmail("adm@alpc.com");
        dto.setPassword("123123");
        dto.setConfirmPassword("123123");
        Long userId = this.userService.save(dto);

        dto.setAdm(false);
        dto.setName("discente");
        dto.setEmail("discente@alpc.com");
        userId = this.userService.save(dto);
        this.userService.saveUserPrograma(new ProgramaUserSaveDTO(userId, id,
            ↪ Authority.DISCENTE));

        dto.setName("Docente");
        dto.setEmail("docente@alpc.com");
        userId = this.userService.save(dto);
        this.userService.saveUserPrograma(new ProgramaUserSaveDTO(userId, id,
            ↪ Authority.DOCENTE));

        dto.setName("Coordenador");
        dto.setEmail("coordenador@alpc.com");
        dto.setCriarPrograma(true);
        userId = this.userService.save(dto);
        this.userService.saveUserPrograma(new ProgramaUserSaveDTO(userId, id,
            ↪ Authority.COORDENADOR));

        return new ResponseEntity<>("Sucesso", HttpStatus.OK);
    }

    @RequestMapping(value = "/pages", method = RequestMethod.GET)
    public ResponseEntity<String> pages() {
        for (int i = 0; i < 30; i++) {
            InstituicaoSaveDTO insDto = new InstituicaoSaveDTO();
            insDto.setEstado(Estado.SANTA_CATARINA);
            insDto.setNome("Universidade Federal De Santa Catarina");

```

```

        insDto.setNomePais("Brasil");
        insDto.setNomePaisFiltro(QueryUtils.lowerCaseStripAccents(insDto.getNomePais()));
        insDto.setSigla("UFSC");

        ProgramaFormDTO pDTO = new ProgramaFormDTO();
        Calendar calendar = Calendar.getInstance();
        calendar.set(Calendar.YEAR, 2005);
        calendar.set(Calendar.DAY_OF_MONTH, 5);
        calendar.set(Calendar.MONTH, 0);
        pDTO.setDataInicio(calendar.getTime());
        pDTO.setNome(i + " - Programa de Ps-Graduao em Cincias da
        ↪ Computao");
        pDTO.setInstituicao(insDto);
        this.pService.save(pDTO);
    }

    return new ResponseEntity<>("Sucesso", HttpStatus.OK);
}

@RequestMapping(value = "/lucene", method = RequestMethod.GET)
public void testLucene() throws InterruptedException {
    FullTextEntityManager search = Search.getFullTextEntityManager(this.em);
    // search.createIndexer().startAndWait();
    QueryBuilder qb =
        ↪ search.getSearchFactory().buildQueryBuilder().forEntity(OrientacaoMestradoEntity.class);

    BooleanQuery booleanQuery = new BooleanQuery();
    //
    ↪ booleanQuery.add(qb.keyword().onField("tituloDoTrabalhoFiltro").matching("busca
    ↪ por similaridade em foruns de pergunta/resposta").createQuery(),
    ↪ Occur.MUST);
    booleanQuery.add(qb.keyword().onField("docente.pessoa.nomeFiltro").matching("carina
    ↪ friedrich dorneles").createQuery(), Occur.MUST);

    // BooleanQuery booleanQuery2 = new BooleanQuery();
    //
    ↪ booleanQuery2.add(qb.keyword().onField("tituloDoTrabalhoFiltro").matching("uma
    ↪ estrategia generica para casamento aproximado
    ↪ instancias").createQuery(), Occur.MUST);

    //
    ↪ booleanQuery2.add(qb.keyword().onField("nomeOrientadorSanduicheFiltro").matching("_
    ↪ Occur.MUST);

    org.apache.lucene.search.Query query =
        ↪ qb.bool().should(booleanQuery).createQuery();

    // wrap Lucene query in a org.hibernate.Query
    FullTextQuery fullTextQuery = search.createFullTextQuery(query,
        ↪ OrientacaoMestradoEntity.class);

    fullTextQuery.setProjection(ProjectionConstants.SCORE,
        ↪ ProjectionConstants.THIS);

    // execute search
    List<?> result = fullTextQuery.getResultList();
    for (Object object : result) {
        Object[] e = (Object[]) object;
        float score = (float) e[0];
        OrientacaoMestradoEntity enti = (OrientacaoMestradoEntity) e[1];
        System.out.println("score: " + score + " - titulo: " +
            ↪ enti.getTituloDoTrabalho());
    }
}

package alpc.ufsc.programa;

import static alpc.ufsc.entity.pessoa.QPessoaEntity.pessoaEntity;
import static
    ↪ alpc.ufsc.entity.pessoa.orientacao.doutorado.QOrientacaoDoutoradoEntity.orientacaoDoutoradoEntity;
import static alpc.ufsc.entity.pessoa.orientacao.graduacao.QGraduacaoEntity.graduacaoEntity;
import static
    ↪ alpc.ufsc.entity.pessoa.orientacao.mestrado.QOrientacaoMestradoEntity.orientacaoMestradoEntity;
import static
    ↪ alpc.ufsc.entity.pessoa.orientacao.posdoutorado.QOrientacaoPosDoutoradoEntity.orientacaoPosDoutoradoEntity;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;

```

```

import org.springframework.data.domain.PageImpl;
import org.springframework.data.domain.Pageable;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.jpa.impl.JPAQuery;
import com.mysema.query.jpa.impl.JPAUpdateClause;

import alpc.ufsc.common.exception.InvalidDTOException;
import alpc.ufsc.domain.instituicao.InstituicaoService;
import alpc.ufsc.entity.domain.instituicao.InstituicaoEntity;
import alpc.ufsc.entity.domain.instituicao.QInstituicaoEntity;
import alpc.ufsc.entity.domain.pais.QPaisEntity;
import alpc.ufsc.entity.login.user.QProgramaUserEntity;
import alpc.ufsc.entity.programa.ProgramaEntity;
import alpc.ufsc.entity.programa.QProgramaEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.login.authority.Authority;
import alpc.ufsc.login.user.UserService;
import alpc.ufsc.login.user.dto.ProgramaUserSaveDTO;
import alpc.ufsc.pessoa.PessoaService;
import alpc.ufsc.pessoa.atuacao.profissional.service.AtuacaoProfissionalProcessService;
import alpc.ufsc.pessoa.orientacao.common.service.FormacaoProcessService;
import alpc.ufsc.pessoa.orientacao.common.service.OrientacaoProcessService;
import alpc.ufsc.pessoa.orientacao.graduacao.GraduacaoProcessService;
import alpc.ufsc.pessoa.producao.service.bibliografica.ProducaoBibliograficaProcessService;
import alpc.ufsc.pessoa.producao.service.tecnica.ProducaoTecnicaProcessService;
import alpc.ufsc.programa.dto.ImportProcessDto;
import alpc.ufsc.programa.dto.MProgramaFormDTO;
import alpc.ufsc.programa.dto.MProgramaRowDTO;
import alpc.ufsc.programa.dto.ProgramaFilterDTO;
import alpc.ufsc.programa.dto.ProgramaFormDTO;
import alpc.ufsc.programa.dto.ProgramaRowDTO;
import alpc.ufsc.util.querydsl.Select;
import ufsc.alpc.xml.dto.CurriculoVitaeXmlDto;

```

```

@Service
@Transactional
public class ProgramaService {

    @PersistenceContext
    private EntityManager em;

    @Autowired
    private FormacaoProcessService formacaoService;

    @Autowired
    private OrientacaoProcessService orientacaoService;

    @Autowired
    private InstituicaoService instituicaoService;

    @Autowired
    private PessoaService pessoaProcess;

    @Autowired
    private ProducaoBibliograficaProcessService producaoBibliograficaProcess;

    @Autowired
    private ProducaoTecnicaProcessService producaoTecnicaProcess;

    @Autowired
    private GraduacaoProcessService graduacaoService;

    @Autowired
    private AtuacaoProfissionalProcessService atuacaoService;

    @Autowired
    private UserService userService;

    public Long createPrograma(ProgramaFormDTO dto, Long userId) throws
↳ InvalidDTOException {
        Long programaId = this.save(dto);
        this.userService.saveUserProgramaAndUpdateXml(new
↳ ProgramaUserSaveDTO(userId, programaId,
↳ Authority.COORDENADOR));
        return programaId;
    }

    public Long save(ProgramaFormDTO dto) throws InvalidDTOException {
        dto.validate();
    }

```

```

ProgramaEntity entity = null;
if (dto.getId() != null) {
    entity = this.em.find(ProgramaEntity.class, dto.getId());
} else {
    entity = new ProgramaEntity();
}
entity.setNome(dto.getNome());
entity.setInstituicao(this.em.getReference(InstituicaoEntity.class,
    ↪ this.instituicaoService.save(dto.getInstituicao()));
entity.setDataInicio(dto.getDataInicio());
this.em.persist(entity);
return entity.getId();
}

public ProgramaFormDTO load(Long id) {
    QProgramaEntity qPrograma = QProgramaEntity.programaEntity;
    QInstituicaoEntity qInstituicao = QInstituicaoEntity.instituicaoEntity;
    QPaisEntity qPais = QPaisEntity.paisEntity;
    JPAQuery query = new JPAQuery(this.em).from(qPrograma);
    query.join(qPrograma.instituicao(), qInstituicao);
    query.join(qInstituicao.pais(), qPais);
    query.where(qPrograma.id.eq(id));

    MProgramaFormDTO meta = MProgramaFormDTO.meta;
    Select<ProgramaFormDTO> select = new Select<>(ProgramaFormDTO.class);
    select.as(qPrograma.id, meta.id);
    select.as(qPrograma.nome, meta.nome);
    select.as(qPrograma.dataInicio, meta.dataInicio);
    select.as(qInstituicao.nome, meta.instituicao().nome);
    select.as(qInstituicao.sigla, meta.instituicao().sigla);
    select.as(qPais.nome, meta.instituicao().nomePais);
    select.as(qInstituicao.estado(), meta.instituicao().estado);

    return query.uniqueResult(select);
}

public Page<ProgramaRowDTO> getPage(ProgramaFilterDTO filter, Pageable pageable) {
    QProgramaEntity qPrograma = QProgramaEntity.programaEntity;
    QInstituicaoEntity qInstituicao = QInstituicaoEntity.instituicaoEntity;
    QProgramaUserEntity qProgUser = QProgramaUserEntity.programaUserEntity;

    JPAQuery query = new JPAQuery(this.em).from(qPrograma);
    query.join(qPrograma.instituicao(), qInstituicao);
    query.leftJoin(qPrograma.users,
    ↪ qProgUser).on(qProgUser.user().id.eq(filter.getUserId()));

    if (filter.hasNome()) {
        query.where(qPrograma.nomeFiltro.like(QueryUtils.like(filter.getNome())));
    }
    if (filter.hasInstituicoesId()) {
        query.where(qInstituicao.id.in(filter.getInstituicoesIds()));
    }

    MProgramaRowDTO meta = MProgramaRowDTO.meta;
    Select<ProgramaRowDTO> select = new Select<>(ProgramaRowDTO.class);
    select.as(qPrograma.id, meta.id);
    select.as(qPrograma.nome, meta.nome);
    select.as(qInstituicao.nome, meta.nomeInstituicao);
    select.as(qInstituicao.estado(), meta.estado);
    select.as(qProgUser.grantedAuthority, meta.grantedAuthority);
    select.as(qProgUser.requestAuthority, meta.requestAuthority);

    query.orderBy(qProgUser.grantedAuthority.asc());
    query.orderBy(qProgUser.requestAuthority.asc());
    query.orderBy(qPrograma.nome.asc());

    query.offset(pageable.getOffset());
    query.limit(pageable.getPageSize());

    List<ProgramaRowDTO> list = query.list(select);

    return new PageImpl<>(list, pageable, query.count());
}

@Transactional(propagation = Propagation.NOT_SUPPORTED)
public void importLattes(CurriculoVitaexmlDto curriculo, ProgramaFormDTO
    ↪ programaDto) {
    ImportProcessDto processDto = new ImportProcessDto();
    processDto.setCurriculo(curriculo);
    processDto.setProgramaDto(programaDto);
    if (this.pessoaProcess.processPessoa(processDto)) {

```

```

        this.orientacaoService.processOrientacoes(processDto);
        this.formacaoService.processFormacoes(processDto);
        if (processDto.isSetDocenteId()) {
            this.producaoBibliograficaProcess.processProducoes(processDto);
            this.producaoTecnicaProcess.processProducoes(processDto);
            this.atuacaoService.processAtuacoes(processDto);
        }
        if (processDto.isSetDiscenteId()) {
            this.graduacaoService.processGraduacao(processDto);
        }
        this.pessoaProcess.updateForceProcessamento(processDto.getPessoaId(),
            ↪ false);
    }
}

public void refreshDeleteMestrado(Long programaId) {
    new JPAUpdateClause(this.em, orientacaoMestradoEntity)
        .set(orientacaoMestradoEntity.deletado, true)
        .where(orientacaoMestradoEntity.programa().id.eq(programaId))
        .where(orientacaoMestradoEntity.orientadorAtivo.eq(false))
        .where(orientacaoMestradoEntity.coorientadorAtivo.eq(false))
        .where(orientacaoMestradoEntity.discenteCompleto.eq(false))
        .execute();
}

public void refreshDeleteDoutorado(Long programaId) {
    new JPAUpdateClause(this.em, orientacaoDoutoradoEntity)
        .set(orientacaoDoutoradoEntity.deletado, true)
        .where(orientacaoDoutoradoEntity.programa().id.eq(programaId))
        .where(orientacaoDoutoradoEntity.orientadorAtivo.eq(false))
        .where(orientacaoDoutoradoEntity.coorientadorAtivo.eq(false))
        .where(orientacaoDoutoradoEntity.discenteCompleto.eq(false))
        .execute();
}

public void refreshViewPertencePrograma() {
    this.em.createNativeQuery("REFRESH MATERIALIZED VIEW CONCURRENTLY
        ↪ view_pertence_programa").executeUpdate();
}

public void forceProcessamentoPessoa(Long programaId) {
    new JPAUpdateClause(this.em, pessoaEntity)
        .set(pessoaEntity.forceProcessamento, true)
        .where(pessoaEntity.programa().id.eq(programaId))
        .execute();
}

public void deletePosDoutorado(Long programaId) {
    new JPAUpdateClause(this.em, orientacaoPosDoutoradoEntity)
        .set(orientacaoPosDoutoradoEntity.deletado, true)
        .where(orientacaoPosDoutoradoEntity.programa().id.eq(programaId))
        .execute();
}

public void deleteDoutorado(Long programaId) {
    new JPAUpdateClause(this.em, orientacaoDoutoradoEntity)
        .set(orientacaoDoutoradoEntity.deletado, true)
        .where(orientacaoDoutoradoEntity.programa().id.eq(programaId))
        .execute();
}

public void deleteMestrado(Long programaId) {
    new JPAUpdateClause(this.em, orientacaoMestradoEntity)
        .set(orientacaoMestradoEntity.deletado, true)
        .where(orientacaoMestradoEntity.programa().id.eq(programaId))
        .execute();
}

public void deleteGraduacao(Long programaId) {
    new JPAUpdateClause(this.em, graduacaoEntity)
        .set(graduacaoEntity.deletado, true)
        .where(graduacaoEntity.programa().id.eq(programaId))
        .execute();
}
}
package alpc.ufsc.programa.dto;

import java.util.List;

import lombok.Getter;
import lombok.Setter;

```



```

import org.apache.commons.lang3.StringUtils;

@Getter
@Setter
public class ProgramaFilterDTO {

    private String nome;
    private List<Long> instituicoesIds;
    private Long userId;

    public boolean hasNome() {
        return !StringUtils.isBlank(this.nome);
    }

    public boolean hasInstituicoesId() {
        return this.instituicoesIds != null && !this.instituicoesIds.isEmpty();
    }
}
package alpc.ufsc.programa.dto;

import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
public class ProgramaRequestAuthorityDTO {

    private Long programaId;
    private String requestAuthority;
}
package alpc.ufsc.programa.dto;

import java.util.Date;

import lombok.Getter;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;
import br.ufsc.bridge.metafy.Metafy;

import alpc.ufsc.common.FormDTO;
import alpc.ufsc.common.exception.ErrorMessages;
import alpc.ufsc.common.exception.InvalidDTOException;
import alpc.ufsc.domain.instituicao.dto.InstituicaoSaveDTO;

@Getter
@Setter
@Metafy
public class ProgramaFormDTO implements FormDTO {

    private Long id;

    private String nome;

    private Date dataInicio;

    private InstituicaoSaveDTO instituicao;

    public ProgramaFormDTO() {
        this.instituicao = new InstituicaoSaveDTO();
    }

    public void validate() throws InvalidDTOException {
        InvalidDTOException dtoException = new InvalidDTOException();
        MProgramaFormDTO meta = MProgramaFormDTO.meta;
        if (StringUtils.isBlank(this.nome)) {
            dtoException.putError(meta.nome, ErrorMessages.OBRIGATORIO);
        }
        if (this.dataInicio == null) {
            dtoException.putError(meta.dataInicio, ErrorMessages.OBRIGATORIO);
        }
        if (!dtoException.contains(meta.dataInicio) && this.dataInicio.compareTo(new
            ↪ Date()) > 0) {
            dtoException.putError(meta.dataInicio, "Data inicio deve ser menor ou igual
            ↪ data atual");
        }
        try {
            this.instituicao.validate();
        } catch (InvalidDTOException e) {
            dtoException.putError(meta.instituicao(), e.getErrors());
        }
    }
}

```

```

        dtoException.checkThrow();
    }
}
package alpc.ufsc.programa.dto;

import lombok.Getter;
import lombok.Setter;

import ufsc.alpc.xml.dto.CurriculoVitaeXmlDto;

@Getter
@Setter
public class ImportProcessDto {

    private CurriculoVitaeXmlDto curriculo;

    private ProgramaFormDTO programaDto;

    private Long pessoaId;

    private Long discenteId;

    private Long docenteId;

    public boolean isSetDiscenteId() {
        return this.discenteId != null;
    }

    public boolean isSetDocenteId() {
        return this.docenteId != null;
    }
}
package alpc.ufsc.programa.dto;

import lombok.Getter;
import lombok.Setter;

import br.ufsc.bridge.metafy.Metafy;

import com.fasterxml.jackson.annotation.JsonIgnore;

import alpc.ufsc.entity.domain.estado.Estado;

@Getter
@Setter
@Metafy
public class ProgramaRowDTO {

    private Long id;
    private String nome;
    private String nomeInstituicao;
    private String grantedAuthority;
    private String requestAuthority;

    @JsonIgnore
    private Estado estado;

    public String getNomeEstado() {
        return this.estado.getNome();
    }
}
package alpc.ufsc.programa;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.security.access.annotation.Secured;
import org.springframework.security.access.prepost.PreAuthorize;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;

```

```

import com.mysema.query.jpql.impl.JPAQuery;

import alpc.ufsc.common.PeriodoFilterDTO;
import alpc.ufsc.common.exception.InvalidDTOException;
import alpc.ufsc.entity.login.user.QProgramaUserEntity;
import alpc.ufsc.login.authority.Authority;
import alpc.ufsc.login.user.CurrentUser;
import alpc.ufsc.login.user.UserService;
import alpc.ufsc.login.user.dto.ProgramaUserSaveDTO;
import alpc.ufsc.login.user.dto.UserDetailsImpl;
import alpc.ufsc.pessoa.atuacaoprofissional.linhapesquisa.LinhaPesquisaService;
import alpc.ufsc.pessoa.atuacaoprofissional.linhapesquisa.dto.LinhaPesquisaFilterDTO;
import alpc.ufsc.pessoa.atuacaoprofissional.linhapesquisa.dto.LinhaPesquisaRowDTO;
import alpc.ufsc.pessoa.atuacaoprofissional.projetoPesquisa.dto.ProjetoPesquisaFilterDTO;
import alpc.ufsc.pessoa.atuacaoprofissional.projetoPesquisa.dto.ProjetoPesquisaRowDTO;
import alpc.ufsc.pessoa.atuacaoprofissional.projetoPesquisa.service.ProjetoPesquisaService;
import alpc.ufsc.pessoa.atuacaoprofissional.service.AtuacaoProfissionalService;
import alpc.ufsc.pessoa.discente.DiscenteService;
import alpc.ufsc.pessoa.discente.dto.DiscenteInstituicaoProcedenciaFilterDTO;
import alpc.ufsc.pessoa.discente.dto.DiscenteInstituicaoProcedenciaRowDTO;
import alpc.ufsc.pessoa.producao.dto.QuantidadeRowDTO;
import alpc.ufsc.pessoa.producao.service.ProducaoService;
import alpc.ufsc.programa.dto.ProgramaFilterDTO;
import alpc.ufsc.programa.dto.ProgramaFormDTO;
import alpc.ufsc.programa.dto.ProgramaRequestAuthorityDTO;
import alpc.ufsc.programa.dto.ProgramaRowDTO;

@RestController
@RequestMapping("/services/programa")
public class ProgramaRestController {

    @Autowired
    private ProgramaService service;

    @Autowired
    private UserService userService;

    @PersistenceContext
    private EntityManager em;

    @Autowired
    private LinhaPesquisaService linhaPesquisaService;

    @Autowired
    private ProjetoPesquisaService projetoService;

    @Autowired
    private DiscenteService discenteService;

    @Autowired
    private ProducaoService producaoService;

    @Autowired
    private AtuacaoProfissionalService atuacaoProfissionalService;

    @PreAuthorize("#user.getCriarPrograma()")
    @RequestMapping(method = RequestMethod.POST)
    public ResponseEntity<String> save(@CurrentUser UserDetailsImpl user, @RequestBody
    ↪ ProgramaFormDTO form) {
        try {
            form.getInstituicao().setNomePais("Brasil");
            form.setId(this.service.createPrograma(form, user.getId()));
            return new ResponseEntity<>(form.response(), HttpStatus.CREATED);
        } catch (InvalidDTOException e) {
            return new ResponseEntity<>(e.toJson(), HttpStatus.BAD_REQUEST);
        }
    }

    @Secured({ Authority.COORDENADOR })
    @RequestMapping(value = "{id}", method = RequestMethod.PUT)
    public ResponseEntity<String> update(@PathVariable Long id, @RequestBody
    ↪ ProgramaFormDTO form) {
        try {
            form.setId(id);
            this.service.save(form);
            return new ResponseEntity<>(form.response(), HttpStatus.CREATED);
        } catch (InvalidDTOException e) {
            return new ResponseEntity<>(e.toJson(), HttpStatus.BAD_REQUEST);
        }
    }

    @RequestMapping(method = RequestMethod.GET)

```

```

public @ResponseBody Page<ProgramaRowDTO> getPage(ProgramaFilterDTO filter,
↳ @CurrentUser UserDetailsImpl user, Pageable pageable) {
    filter.setUserId(user.getId());
    return this.service.getPage(filter, pageable);
}

@Secured({ Authority.COORDENADOR })
@RequestMapping(value = "{id}", method = RequestMethod.GET)
public @ResponseBody ProgramaFormDTO load(@PathVariable Long id) {
    return this.service.load(id);
}

@RequestMapping(value = "requestAuthority", method = RequestMethod.POST)
public void requestAuthority(@CurrentUser UserDetailsImpl user, @RequestBody
↳ ProgramaRequestAuthorityDTO requestAuthority) {
    QProgramaUserEntity qProgUser = QProgramaUserEntity.programaUserEntity;

    JPAQuery query = new JPAQuery(this.em).from(qProgUser);
    query.where(qProgUser.programa().id.eq(requestAuthority.getProgramaId()));
    query.where(qProgUser.user().id.eq(user.getId()));

    ProgramaUserSaveDTO dto = new ProgramaUserSaveDTO();

    dto.setUserId(user.getId());
    dto.setProgramaId(requestAuthority.getProgramaId());
    dto.setGrantedAuthority(query.uniqueResult(qProgUser.grantedAuthority));
    dto.setRequestAuthority(requestAuthority.getRequestAuthority());

    this.userService.saveUserPrograma(dto);
}

@RequestMapping(value = "linhaPesquisa", method = RequestMethod.GET)
public @ResponseBody Page<LinhaPesquisaRowDTO> getLinhaPesquisa(@CurrentUser
↳ UserDetailsImpl user, LinhaPesquisaFilterDTO filter, Pageable pageable) {
    filter.setProgramaId(user.getProgramaId());
    return this.linhaPesquisaService.getPage(filter, pageable);
}

@RequestMapping(value = "projetoPesquisa", method = RequestMethod.GET)
public @ResponseBody Page<ProjetoPesquisaRowDTO> getProjetoPesquisa(@CurrentUser
↳ UserDetailsImpl user, ProjetoPesquisaFilterDTO filter, Pageable pageable) {
    filter.setProgramaId(user.getProgramaId());
    return this.projetoService.getPage(filter, pageable);
}

@RequestMapping(value = "instituicaoProcedencia", method = RequestMethod.GET)
public @ResponseBody Page<DiscenteInstituicaoProcedenciaRowDTO>
↳ getInstituicaoProcedencia(@CurrentUser UserDetailsImpl user,
↳ DiscenteInstituicaoProcedenciaFilterDTO filter,
    Pageable pageable) {
    filter.setProgramaId(user.getProgramaId());
    return this.discenteService.getPage(filter, pageable);
}

@RequestMapping(value = "producaoBibliografica", method = RequestMethod.GET)
public @ResponseBody List<QuantidadeRowDTO> getProducaoBibliografica(@CurrentUser
↳ UserDetailsImpl user, PeriodoFilterDTO filter) {
    return this.producaoService.getProducoesBibliograficas(filter, user.getProgramaId());
}

@RequestMapping(value = "producaoTecnica", method = RequestMethod.GET)
public @ResponseBody List<QuantidadeRowDTO> getProducaoTecnica(@CurrentUser
↳ UserDetailsImpl user, PeriodoFilterDTO filter) {
    return this.producaoService.getProducoesTecnicas(filter, user.getProgramaId());
}

@RequestMapping(value = "atividadeAcademica", method = RequestMethod.GET)
public @ResponseBody List<QuantidadeRowDTO> getAtividadeAcademica(@CurrentUser
↳ UserDetailsImpl user, PeriodoFilterDTO filter) {
    return this.atuacaoProfissionalService.getAtuacaoProfissional(filter,
↳ user.getProgramaId());
}
}
package alpc.ufsc.config.web;

import lombok.Getter;
import lombok.Setter;

import org.springframework.boot.context.properties.ConfigurationProperties;
import org.springframework.stereotype.Component;

@Getter

```

```

@Setter
@Component
@ConfigurationProperties(prefix = "process.curriculo")
public class ProcessCurriculoProperties {

    private Integer minThreads;
    private Integer maxThreads;
    private Integer workQueueSize;
}

package alpc.ufsc.config.web.security;

import java.io.IOException;
import java.io.StringWriter;
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.NoSuchAlgorithmException;

import javax.crypto.Cipher;

import lombok.Getter;
import lombok.extern.slf4j.Slf4j;

import org.apache.commons.codec.binary.Base64;
import org.bouncycastle.util.io.pem.PemObject;
import org.bouncycastle.util.io.pem.PemWriter;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;

@Slf4j
public class PasswordEncoderWrapper implements PasswordEncoder {
    private static final String RSA = "RSA";

    private PasswordEncoder passwordEncoder;
    private KeyPair key;
    @Getter
    private String publicKey;

    public PasswordEncoderWrapper() {
        this.passwordEncoder = new BCryptPasswordEncoder();
        try {
            KeyPairGenerator keyGen = KeyPairGenerator.getInstance(RSA);
            keyGen.initialize(1024);
            this.key = keyGen.generateKeyPair();
        } catch (NoSuchAlgorithmException e) {
            throw new RuntimeException("No foi possível criar a RSA key", e);
        }
        try {
            PemObject pemObject = new PemObject("RSA PUBLIC KEY",
                ↪ this.key.getPublic().getEncoded());
            StringWriter stringWriter = new StringWriter();
            PemWriter pemWriter = new PemWriter(stringWriter);
            pemWriter.writeObject(pemObject);
            pemWriter.close();
            this.publicKey = stringWriter.toString();
        } catch (IOException e) {
            throw new RuntimeException("No foi possível gerar a public key em
                ↪ formato pem", e);
        }
    }

    @Override
    public String encode(CharSequence rawPassword) {
        return this.passwordEncoder.encode(rawPassword);
    }

    @Override
    public boolean matches(CharSequence rawPassword, String encodedPassword) {
        return this.passwordEncoder.matches(this.decrypt(rawPassword), encodedPassword);
    }

    public String decrypt(CharSequence text) {
        try {
            final Cipher cipher = Cipher.getInstance(RSA);
            cipher.init(Cipher.DECRYPT_MODE, this.key.getPrivate());
            return new String(cipher.doFinal(Base64.decodeBase64(text.toString()),
                ↪ "UTF-8"));
        } catch (Exception e) {
            log.error("erro ao decrypt senha: " + e.getMessage());
            return null;
        }
    }
}
}

```

```

package alpc.ufsc.config.web.security;

import javax.inject.Inject;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import
    ↳ org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import
    ↳ org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.security.web.csrf.CsrfFilter;

import alpc.ufsc.config.web.util.security.CsrfTokenResponseCookieBindingFilter;
import alpc.ufsc.config.web.util.security.RESTAuthenticationEntryPoint;
import alpc.ufsc.config.web.util.security.RESTAuthenticationFailureHandler;
import alpc.ufsc.config.web.util.security.RESTAuthenticationSuccessHandler;
import alpc.ufsc.config.web.util.security.RESTLogoutSuccessHandler;

@Configuration
@EnableWebSecurity
// @Order(SecurityProperties.ACCESS_OVERRIDE_ORDER)
public class WebSecurityConfig extends WebSecurityConfigurerAdapter {

    @Inject
    private UserDetailsService userService;

    @Autowired
    private RESTAuthenticationEntryPoint authenticationEntryPoint;
    @Autowired
    private RESTAuthenticationFailureHandler authenticationFailureHandler;
    @Autowired
    private RESTAuthenticationSuccessHandler authenticationSuccessHandler;
    @Autowired
    private RESTLogoutSuccessHandler logoutSuccessHandler;

    @Bean
    public PasswordEncoder passwordEncoder() {
        return new PasswordEncoderWrapper();
    }

    @Override
    protected void configure(HttpSecurity http) throws Exception {
        // @formatter:off
        http.authorizeRequests()
            .antMatchers("/public/**").permitAll()
            .antMatchers("/services/**").authenticated();
        // @formatter:on
        http.exceptionHandling().authenticationEntryPoint(this.authenticationEntryPoint);
        http.formLogin().successHandler(this.authenticationSuccessHandler);
        http.formLogin().failureHandler(this.authenticationFailureHandler);
        http.logout().logoutSuccessHandler(this.logoutSuccessHandler);
        http.addFilterAfter(new CsrfTokenResponseCookieBindingFilter(), CsrfFilter.class);
    }

    @Override
    public void configure(AuthenticationManagerBuilder builder) throws Exception {
        builder.userDetailsService(this.userService)
            .passwordEncoder(this.passwordEncoder());
    }
}

package alpc.ufsc.config.web.security;

import org.springframework.context.annotation.Configuration;
import org.springframework.security.access.AccessDecisionManager;
import org.springframework.security.access.AccessDecisionVoter;
import org.springframework.security.access.vote.AffirmativeBased;
import org.springframework.security.access.vote.RoleVoter;
import
    ↳ org.springframework.security.config.annotation.method.configuration.EnableGlobalMethodSecurity;
import
    ↳ org.springframework.security.config.annotation.method.configuration.GlobalMethodSecurityConfiguration;

@Configuration
@EnableGlobalMethodSecurity(prePostEnabled = true, securedEnabled = true)
public class WebGlobalMethodSecurityConfiguration extends GlobalMethodSecurityConfiguration {

    @Override

```

```

protected AccessDecisionManager accessDecisionManager() {
    AffirmativeBased accessDecisionManager = (AffirmativeBased)
        ↪ super.accessDecisionManager();
    for (AccessDecisionVoter<? extends Object> desisor :
        ↪ accessDecisionManager.getDecisionVoters()) {
        if (desisor instanceof RoleVoter) {
            ((RoleVoter) desisor).setRolePrefix("");
        }
    }
    return accessDecisionManager;
}
}
package alpc.ufsc.config.web;

import org.springframework.context.annotation.Configuration;
import org.springframework.messaging.simp.config.MessageBrokerRegistry;
import ↪ org.springframework.web.socket.config.annotation.AbstractWebSocketMessageBrokerConfigurer;
import org.springframework.web.socket.config.annotation.EnableWebSocket;
import org.springframework.web.socket.config.annotation.EnableWebSocketMessageBroker;
import org.springframework.web.socket.config.annotation.StompEndpointRegistry;

@Configuration
@EnableWebSocket
@EnableWebSocketMessageBroker
public class WebSocketConfig extends AbstractWebSocketMessageBrokerConfigurer {

    @Override
    public void configureMessageBroker(MessageBrokerRegistry registry) {
        registry.enableSimpleBroker("/topic");
        registry.setApplicationDestinationPrefixes("/sock");
    }

    @Override
    public void registerStompEndpoints(StompEndpointRegistry registry) {
        registry.addEndpoint("/sock").withSockJS();
    }
}
package alpc.ufsc.config.web;

import javax.servlet.MultipartConfigElement;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.boot.autoconfigure.condition.ConditionalOnMissingBean;
import org.springframework.boot.context.embedded.MultipartConfigFactory;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.multipart.MultipartResolver;
import org.springframework.web.multipart.support.StandardServletMultipartResolver;
import org.springframework.web.servlet.DispatcherServlet;

@Configuration
public class UploadConfig {

    private Logger logger = LoggerFactory.getLogger(UploadConfig.class);

    @Bean
    @ConditionalOnMissingBean
    public MultipartConfigElement multipartConfigElement() {
        MultipartConfigFactory factory = new MultipartConfigFactory();
        factory.setFileSizeThreshold("100MB");
        factory.setMaxRequestSize("1024MB");
        factory.setMaxFileSize("1024MB");
        factory.setLocation(System.getProperty("java.io.tmpdir"));
        this.logger.info("Temp file: " + System.getProperty("java.io.tmpdir"));
        return factory.createMultipartConfig();
    }

    @Bean(name = DispatcherServlet.MULTIPART_RESOLVER_BEAN_NAME)
    @ConditionalOnMissingBean(value = MultipartResolver.class)
    public StandardServletMultipartResolver multipartResolver() {
        return new StandardServletMultipartResolver();
    }
}
package alpc.ufsc.config.web.util.security;

import java.io.IOException;

import javax.servlet.ServletException;

```

```

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.security.core.Authentication;
import org.springframework.security.web.authentication.logout.LogoutSuccessHandler;
import org.springframework.stereotype.Component;

@Component
public class RESTLogoutSuccessHandler implements LogoutSuccessHandler {
    protected static final String REQUEST_ATTRIBUTE_NAME = "csrf";
    protected static final String RESPONSE_PARAM_NAME = "x-csrf-param";
    protected static final String RESPONSE_TOKEN_NAME = "x-csrf-token";

    @Override
    public void onLogoutSuccess(HttpServletRequest request, HttpServletResponse response,
        Authentication authentication) throws IOException, ServletException {
        response.setStatus(HttpStatus.SC_OK);
    }
}

package alpc.ufsc.config.web.util.security;

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.security.core.AuthenticationException;
import org.springframework.security.web.AuthenticationEntryPoint;
import org.springframework.stereotype.Component;

@Component
public class RESTAuthenticationEntryPoint implements AuthenticationEntryPoint {

    @Override
    public void commence(HttpServletRequest request, HttpServletResponse response,
        AuthenticationException authException) throws IOException, ServletException {
        response.sendError(HttpStatus.UNAUTHORIZED);
    }
}

package alpc.ufsc.config.web.util.security;

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.security.core.AuthenticationException;
import org.springframework.security.web.authentication.SimpleUrlAuthenticationFailureHandler;
import org.springframework.stereotype.Component;

@Component
public class RESTAuthenticationFailureHandler extends SimpleUrlAuthenticationFailureHandler {

    @Override
    public void onAuthenticationFailure(HttpServletRequest request, HttpServletResponse response,
        AuthenticationException exception) throws IOException, ServletException {
        super.onAuthenticationFailure(request, response, exception);
    }
}

package alpc.ufsc.config.web.util.security;

import java.io.IOException;

import javax.servlet.FilterChain;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.security.web.csrf.CsrfToken;
import org.springframework.web.filter.OncePerRequestFilter;

public class CsrfTokenResponseCookieBindingFilter extends OncePerRequestFilter {
    protected static final String REQUEST_ATTRIBUTE_NAME = "csrf";
    protected static final String RESPONSE_HEADER_NAME = "csrf-header-name";
    protected static final String RESPONSE_TOKEN_NAME = "csrf-token-name";

    @Override
    protected void doFilterInternal(HttpServletRequest request, HttpServletResponse response,
        FilterChain filterChain) throws ServletException, IOException {
        CsrfToken token = (CsrfToken)
            request.getAttribute(REQUEST_ATTRIBUTE_NAME);

```



```

        if (token != null) {
            response.setHeader(RESPONSE_HEADER_NAME,
                ↵ token.getHeaderName());
            response.setHeader(RESPONSE_TOKEN_NAME, token.getToken());
        }
        filterChain.doFilter(request, response);
    }
}
package alpc.ufsc.config.web.util.security;

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.security.core.Authentication;
import org.springframework.security.web.authentication.SimpleUrlAuthenticationSuccessHandler;
import org.springframework.stereotype.Component;

@Component
public class RESTAuthenticationSuccessHandler extends SimpleUrlAuthenticationSuccessHandler {

    @Override
    public void onAuthenticationSuccess(HttpServletRequest request, HttpServletResponse
        ↵ response, Authentication authentication) throws IOException, ServletException {
        this.clearAuthenticationAttributes(request);
    }
}
package alpc.ufsc.config.web.util.nonblocking;

import java.util.concurrent.ArrayBlockingQueue;
import java.util.concurrent.ThreadPoolExecutor;
import java.util.concurrent.TimeUnit;

import alpc.ufsc.util.NamedThreadFactory;

public class NonBlockingIOThreadPool {

    private ThreadPoolExecutor executor;

    public NonBlockingIOThreadPool(NonBlockingIOProperties properties) {
        ↵ //formatter:off
        this.executor = new ThreadPoolExecutor(
            properties.getMinThreads(),
            properties.getMaxThreads(),
            60, TimeUnit.SECONDS,
            new ArrayBlockingQueue<>(properties.getWorkQueueSize()),
            ↵ new NamedThreadFactory("NonBlockingIO"));
        ↵ //formatter:on
    }

    public void execute(Runnable runnable) {
        this.executor.execute(runnable);
    }

    public void shutdown() {
        this.executor.shutdown();
    }
}
package alpc.ufsc.config.web.util.nonblocking;

import lombok.Getter;
import lombok.Setter;

import org.springframework.boot.context.properties.ConfigurationProperties;
import org.springframework.stereotype.Component;

@Getter
@Setter
@Component
@ConfigurationProperties(prefix = "non.blocking.io")
public class NonBlockingIOProperties {

    private Integer minThreads;
    private Integer maxThreads;
    private Integer workQueueSize;
}
package alpc.ufsc.config.web;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;

```

```

import org.springframework.beans.factory.annotation.Value;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.scheduling.annotation.EnableAsync;
import org.springframework.web.servlet.ViewResolver;
import org.springframework.web.servlet.config.annotation.EnableWebMvc;
import org.springframework.web.servlet.config.annotation.InterceptorRegistry;
import org.springframework.web.servlet.config.annotation.ResourceHandlerRegistry;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurerAdapter;
import org.springframework.web.servlet.mvc.WebContentInterceptor;
import org.springframework.web.servlet.view.InternalResourceViewResolver;

import alpc.ufsc.config.web.util.nonblocking.NonBlockingIOProperties;
import alpc.ufsc.config.web.util.nonblocking.NonBlockingIOThreadPool;

@Configuration
@EnableWebMvc
@EnableAsync
public class WebMvcConfig extends WebMvcConfigurerAdapter {

    private Logger logger = LoggerFactory.getLogger(WebMvcConfig.class);

    @Value("${alpc.ui.resource.location}")
    private String uiResourceLocation;

    @Autowired
    private NonBlockingIOProperties nonBlockingIOProperties;

    @Override
    public void addResourceHandlers(ResourceHandlerRegistry registry) {
        this.logger.info("Folder for resources web: " + this.uiResourceLocation);
        registry.addResourceHandler("/**").addResourceLocations(this.uiResourceLocation);
    }

    @Bean
    public ViewResolver getViewResolver() {
        InternalResourceViewResolver resolver = new InternalResourceViewResolver();
        resolver.setPrefix("/");
        resolver.setSuffix(".html");
        return resolver;
    }

    @Bean(destroyMethod = "shutdown")
    public NonBlockingIOThreadPool threadPool() {
        return new NonBlockingIOThreadPool(this.nonBlockingIOProperties);
    }

    @Override
    public void addInterceptors(InterceptorRegistry registry) {
        registry.addInterceptor(new WebContentInterceptor());
    }
}

package alpc.ufsc.config.database;

import lombok.Getter;
import lombok.Setter;

import org.springframework.boot.context.properties.ConfigurationProperties;
import org.springframework.stereotype.Component;

@Getter
@Setter
@Component
@ConfigurationProperties(prefix = "postgres.jdbc")
public class PostgresProperties {

    private String url;
    private String username;
    private String password;
    private Integer maxConnections;
    private Integer minConnections;
    private Long connectionIdleTimeout;
}

package alpc.ufsc.config.database;

import java.sql.Connection;
import java.util.Properties;

import javax.inject.Provider;
import javax.persistence.EntityManagerFactory;

```

```

import javax.sql.DataSource;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.autoconfigure.liquibase.LiquibaseProperties;
import org.springframework.boot.context.properties.EnableConfigurationProperties;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.DependsOn;
import org.springframework.data.jpa.repository.config.EnableJpaRepositories;
import org.springframework.instrument.classloading.InstrumentationLoadTimeWeaver;
import org.springframework.jdbc.datasource.DataSourceUtils;
import org.springframework.orm.jpa.JpaTransactionManager;
import org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean;
import org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter;
import org.springframework.transaction.PlatformTransactionManager;
import org.springframework.transaction.annotation.EnableTransactionManagement;

import com.mysema.query.sql.PostgresTemplates;
import com.mysema.query.sql.SQLQueryFactory;
import com.zaxxer.hikari.HikariConfig;
import com.zaxxer.hikari.HikariDataSource;

import alpc.ufsc.Application;
import liquibase.integration.spring.SpringLiquibase;

@Configuration
@EnableTransactionManagement
@EnableJpaRepositories(Application.BASE_PACKAGE)
@EnableConfigurationProperties({ LiquibaseProperties.class })
public class DatabaseConfiguration {

    @Autowired
    private PostgresProperties dataBaseProperties;

    @Autowired
    private LiquibaseProperties liquibaseProperties;

    @Bean
    public PlatformTransactionManager transactionManager(EntityManagerFactory emf) {
        return new JpaTransactionManager(emf);
    }

    @Bean
    public LocalContainerEntityManagerFactoryBean entityManagerFactory(DataSource
        ↪ dataSource) {
        LocalContainerEntityManagerFactoryBean factory = new
            ↪ LocalContainerEntityManagerFactoryBean();

        HibernateJpaVendorAdapter vendorAdapter = new HibernateJpaVendorAdapter();
        vendorAdapter.setGenerateDdl(Boolean.FALSE);

        factory.setDataSource(dataSource);
        factory.setJpaVendorAdapter(vendorAdapter);
        factory.setPersistenceUnitName("alpc-persistence");
        factory.setPackagesToScan(Application.BASE_PACKAGE);

        Properties jpaProperties = new Properties();
        jpaProperties.put("hibernate.cache.use_second_level_cache", "false");
        jpaProperties.put("hibernate.cache.use_query_cache", "false");
        jpaProperties.put("hibernate.generate_statistics", "false");
        jpaProperties.put("hibernate.hbm2ddl.auto", "update");
        jpaProperties.put("hibernate.search.default.directory_provider", "filesystem");
        jpaProperties.put("hibernate.search.default.indexBase", "indexes");
        factory.setJpaProperties(jpaProperties);

        factory.afterPropertiesSet();
        factory.setLoadTimeWeaver(new InstrumentationLoadTimeWeaver());
        return factory;
    }

    @Bean
    @DependsOn(value = "entityManagerFactory")
    public SpringLiquibase liquibase(DataSource dataSource) {
        SpringLiquibase liquibase = new SpringLiquibase();
        liquibase.setChangeLog(this.liquibaseProperties.getChangeLog());
        liquibase.setContexts(this.liquibaseProperties.getContexts());
        liquibase.setDataSource(dataSource);
        liquibase.setDefaultSchema(this.liquibaseProperties.getDefaultSchema());
        liquibase.setDropFirst(this.liquibaseProperties.isDropFirst());
        liquibase.setShouldRun(true);
        liquibase.setLabels(this.liquibaseProperties.getLabels());
        liquibase.setChangeLogParameters(this.liquibaseProperties.getParameters());
    }
}

```

```

        return liquibase;
    }

    @Bean
    public SQLQueryFactory queryFactory(final DataSource dataSource) {
        return new SQLQueryFactory(PostgresTemplates.DEFAULT,
            (Provider<Connection>) () -> {
                Connection connection = DataSourceUtils.getConnection(dataSource);
                if (!DataSourceUtils.isConnectionTransactional(connection, dataSource)) {
                    throw new IllegalStateException("Connection is not transactional");
                }
                return connection;
            });
    }

    @Bean(destroyMethod = "close")
    public DataSource dataSourcePostgres() {
        HikariConfig config = new HikariConfig();
        config.setJdbcUrl(this.dataBaseProperties.getUrl());
        config.setUsername(this.dataBaseProperties.getUsername());
        config.setPassword(this.dataBaseProperties.getPassword());
        config.setMaximumPoolSize(this.dataBaseProperties.getMaxConnections());
        config.setMinimumIdle(this.dataBaseProperties.getMinConnections());
        config.setIdleTimeout(this.dataBaseProperties.getConnectionIdleTimeout());
        config.addDataSourceProperty("cachePrepStmts", "true");
        config.addDataSourceProperty("prepStmtCacheSize", "250");
        config.addDataSourceProperty("prepStmtCacheSqlLimit", "2048");
        return new HikariDataSource(config);
    }
}package alpc.ufsc.config.cache;

import org.springframework.cache.CacheManager;
import org.springframework.cache.annotation.EnableCaching;
import org.springframework.cache.ehcache.EhCacheCacheManager;
import org.springframework.cache.ehcache.EhCacheManagerFactoryBean;
import org.springframework.cache.transaction.TransactionAwareCacheManagerProxy;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.core.io.ClassPathResource;

@Configuration
@EnableCaching
public class SystemCacheConfiguration {

    @Bean
    public CacheManager cacheManager(EhCacheManagerFactoryBean bean) {
        return new TransactionAwareCacheManagerProxy(new
            ↪ EhCacheCacheManager(bean.getObject()));
    }

    @Bean
    public EhCacheManagerFactoryBean ehCacheCacheManager() {
        EhCacheManagerFactoryBean cmfb = new EhCacheManagerFactoryBean();
        cmfb.setConfigLocation(new ClassPathResource("config/cache.xml"));
        cmfb.setShared(true);
        return cmfb;
    }
}package alpc.ufsc.pessoa.producao.dto;

import lombok.Getter;
import lombok.Setter;

import br.ufsc.bridge.metafy.Metafy;

@Getter
@Setter
@Metafy
public class QuantidadeRowDTO {

    private String id;
    private Long quantidade;
}package alpc.ufsc.pessoa.producao.dto;

import java.util.List;

import alpc.ufsc.entity.pessoa.producao.TipoProducao;
import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.Setter;
import ufsc.alpc.xml.dto.common.AutorXmlDto;

```

```

@Getter
@Setter
@AllArgsConstructor
public class ProducaoDto {

    private Long producaoId;
    private Long programaId;
    private Long docenteId;
    private String titulo;
    private Integer ano;
    private TipoProducao tipoProducao;
    private List<AutorXmlDto> autores;

}

package alpc.ufsc.pessoa.producao.dto;

import org.apache.lucene.search.spell.NGramDistance;

import alpc.ufsc.entity.pessoa.producao.TipoProducao;
import alpc.ufsc.entity.util.QueryUtils;
import lombok.AllArgsConstructor;
import lombok.Getter;

@AllArgsConstructor
@Getter
public class ProducaoSearchDto {

    private String titulo;
    private Integer ano;
    private TipoProducao tipoProducao;
    private Long programaId;
    private String nomeAutor;
    private String[] nomesCitacao;
    private String numerIdentificador;

    public String getTitulo() {
        return QueryUtils.lowerCaseStripAccents(this.titulo);
    }

    public ProducaoSearchDto(String titulo) {
        this.titulo = titulo;
    }

    public boolean similar(ProducaoSearchDto searchDto) {
        NGramDistance nGram = new NGramDistance(4);
        if (nGram.getDistance(searchDto.getTitulo(), this.getTitulo()) < 0.50) {
            return false;
        }
        return true;
    }

}

package alpc.ufsc.pessoa.producao.service;

import static alpc.ufsc.entity.pessoa.producao.QAutorProducaoEntity.autorProducaoEntity;
import static alpc.ufsc.entity.pessoa.producao.QDocenteProducaoEntity.docenteProducaoEntity;
import static alpc.ufsc.entity.pessoa.producao.QProducaoEntity.producaoEntity;
import static alpc.ufsc.pessoa.producao.dto.MQuantidadeRowDTO.meta;
import static alpc.ufsc.querydsl.sql.QRlDocenteProducao.rlDocenteProducao;
import static alpc.ufsc.querydsl.sql.QTbProducao.tbProducao;
import static alpc.ufsc.querydsl.sql.QViewPertencePrograma.viewPertencePrograma;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.hibernate.search.jpa.FullTextEntityManager;
import org.hibernate.search.jpa.Search;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.jpa.JPASubQuery;
import com.mysema.query.jpa.impl.JPADeleteClause;
import com.mysema.query.jpa.impl.JPAQuery;
import com.mysema.query.sql.SQLQuery;
import com.mysema.query.sql.SQLQueryFactory;
import com.mysema.query.types.query.ListSubQuery;

```

```

import alpc.ufsc.common.PeriodoFilterDTO;
import alpc.ufsc.entity.pessoa.AutorEntity;
import alpc.ufsc.entity.pessoa.docente.DocenteEntity;
import alpc.ufsc.entity.pessoa.producao.AutorProducaoEntity;
import alpc.ufsc.entity.pessoa.producao.DocenteProducaoEntity;
import alpc.ufsc.entity.pessoa.producao.ProducaoEntity;
import alpc.ufsc.entity.pessoa.producao.QDDocenteProducaoEntity;
import alpc.ufsc.entity.pessoa.producao.TipoProducao;
import alpc.ufsc.entity.programa.ProgramaEntity;
import alpc.ufsc.pessoa.AutorService;
import alpc.ufsc.pessoa.docente.query.DocentePertenceProgramaQuery;
import alpc.ufsc.pessoa.producao.dto.ProducaoDto;
import alpc.ufsc.pessoa.producao.dto.QuantidadeRowDTO;
import alpc.ufsc.util.querydsl.Select;
import alpc.ufsc.xml.dto.common.AutorXmlDto;

@Service
@Transactional(propagation = Propagation.SUPPORTS)
public class ProducaoService {

    @PersistenceContext
    private EntityManager em;

    @Autowired
    private AutorService autorService;

    @Autowired
    private SQLQueryFactory queryFactory;

    @Autowired
    private DocentePertenceProgramaQuery docenteQuery;

    public void saveProducao(ProducaoDto dto) {
        Long producaoId = dto.getProducaoId();
        if (producaoId != null) {
            this.updateProducao(dto, producaoId);
        } else {
            ProducaoEntity producao = this.createProducao(dto);
            this.saveAutores(dto, producao.getId());
        }
    }

    private ProducaoEntity updateProducao(ProducaoDto dto, Long producaoId) {
        ProducaoEntity producao = this.em.find(ProducaoEntity.class, producaoId);
        this.save(dto, producao);
        if (!this.hasDocenteProducao(dto, producaoId)) {
            this.saveDocenteProducao(dto);
        }
        return producao;
    }

    private boolean hasDocenteProducao(ProducaoDto dto, Long producaoId) {
        QDocenteProducaoEntity rIDocenteProducao =
            ↪ QDocenteProducaoEntity.docenteProducaoEntity;
        return new JPAQuery(this.em).from(rIDocenteProducao)
            .where(rIDocenteProducao.producao().id.eq(producaoId)
                .and(rIDocenteProducao.docente().id.eq(dto.getDocenteId())))
            .exists();
    }

    private ProducaoEntity createProducao(ProducaoDto dto) {
        ProducaoEntity producao = new ProducaoEntity();
        this.save(dto, producao);
        this.saveDocenteProducao(dto);
        return producao;
    }

    private Long save(ProducaoDto dto, ProducaoEntity producao) {
        Integer ano = dto.getAno();
        producao.setTituloDaProducao(dto.getTitulo());
        producao.setAno(ano);
        producao.setTipoProducao(dto.getTipoProducao());
        producao.setPrograma(this.em.getReference(ProgramaEntity.class,
            ↪ dto.getProgramaId()));
        this.em.persist(producao);
        dto.setProducaoId(producao.getId());
        return producao.getId();
    }

    private void saveDocenteProducao(ProducaoDto dto) {
        DocenteProducaoEntity autorProducao = new DocenteProducaoEntity();

```

```

        autorProducao.setDocente(this.em.getReference(DocenteEntity.class,
            ↪ dto.getDocentelid()));
        autorProducao.setProducao(this.em.getReference(ProducaoEntity.class,
            ↪ dto.getProducaoId()));
        this.em.persist(autorProducao);
    }

    private void saveAutores(ProducaoDto dto, Long producaoId) {
        for (AutorXmlDto autor : dto.getAutores()) {
            Long autorId = this.autorService.saveAutor(autor);
            this.saveAutorProducao(producaoId, autorId);
        }
    }

    private void saveAutorProducao(Long producaoId, Long autorId) {
        if (autorId != null) {
            AutorProducaoEntity autorProducaoEntity = new AutorProducaoEntity();
            autorProducaoEntity.setAutor(this.em.getReference(AutorEntity.class,
                ↪ autorId));
            autorProducaoEntity.setProducao(this.em.getReference(ProducaoEntity.class,
                ↪ producaoId));
            this.em.persist(autorProducaoEntity);
        }
    }

    public List<QuantidadeRowDTO> getProducoesBibliograficas(PeriodoFilterDTO filter,
        ↪ Long programaId) {
        return this.getProducao(filter, programaId,
            ↪ TipoProducao.getProducaoBibliografica());
    }

    public List<QuantidadeRowDTO> getProducoesTecnicas(PeriodoFilterDTO filter, Long
        ↪ programaId) {
        return this.getProducao(filter, programaId, TipoProducao.getProducaoTecnica());
    }

    private List<QuantidadeRowDTO> getProducao(PeriodoFilterDTO filter, Long programaId,
        ↪ List<String> producoes) {
        SQLQuery query = this.queryFactory.query().from(tbProducao);
        query.join(rlDocenteProducao).on(tbProducao.idProducao.eq(rlDocenteProducao.idProducao));
        query.join(this.docenteEntity.getSubPertencePrograma(programaId),
            ↪ viewPertencePrograma)
            .on(viewPertencePrograma.idDocente.eq(rlDocenteProducao.idDocente));

        query.where(tbProducao.tipoProducao.in(producoes));
        query.where(tbProducao.idPrograma.eq(programaId));
        if (filter.hasInicio()) {
            query.where(tbProducao.ano.goe(filter.getInicio()));
        }
        if (filter.hasFim()) {
            query.where(tbProducao.ano.loe(filter.getFim()));
        }

        Select<QuantidadeRowDTO> select = new Select<>(QuantidadeRowDTO.class);
        select.as(tbProducao.tipoProducao, meta.id);
        select.as(tbProducao.count(), meta.quantidade);

        query.groupBy(tbProducao.tipoProducao);

        return query.list(select);
    }

    public List<QuantidadeRowDTO> getProducaoDocente(Long docenteId) {
        SQLQuery query = this.queryFactory.query().from(rlDocenteProducao);
        query.join(tbProducao).on(tbProducao.idProducao.eq(rlDocenteProducao.idProducao));

        query.where(tbProducao.tipoProducao.in(TipoProducao.getProducaoBibliografica()));
        query.where(rlDocenteProducao.idDocente.eq(docenteId));

        Select<QuantidadeRowDTO> select = new Select<>(QuantidadeRowDTO.class);
        select.as(tbProducao.tipoProducao, meta.id);
        select.as(rlDocenteProducao.idDocente.count(), meta.quantidade);

        query.groupBy(tbProducao.tipoProducao);

        return query.list(select);
    }

    @Transactional
    public void deleteProjetoPesquisa(Long programaId) {
        List<Long> ids = new JPAQuery(this.em).from(autorProducaoEntity)
            .join(autorProducaoEntity.producao(), producaoEntity)

```

```

        .where(producaoEntity.programa().id.eq(programaId))
        .list(autorProducaoEntity.id);

    FullTextEntityManager search = Search.getFullTextEntityManager(this.em);
    for (Long id : ids) {
        search.purge(AutorProducaoEntity.class, id);
    }

    ListSubQuery<Long> subQuery = new JPASubQuery().from(producaoEntity)
        .where(producaoEntity.programa().id.eq(programaId))
        .list(producaoEntity.id);

    new JPADeleteClause(this.em, docenteProducaoEntity)
        .where(docenteProducaoEntity.producao().id.in(subQuery))
        .execute();

    new JPADeleteClause(this.em, autorProducaoEntity)
        .where(autorProducaoEntity.producao().id.in(subQuery))
        .execute();

    new JPADeleteClause(this.em, producaoEntity)
        .where(producaoEntity.programa().id.eq(programaId))
        .execute();
}
}package alpc.ufsc.pessoa.producao.service.bibliografica;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import alpc.ufsc.entity.pessoa.producao.TipoProducao;
import alpc.ufsc.pessoa.producao.service.ProducaoProcessService;
import alpc.ufsc.programa.dto.ImportProcessDto;
import ufsc.alpc.xml.dto.producao.bibliografica.ProducaoBibliograficaXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.artigos.aceitos.ArtigosAceitosParaPublicacaoXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.artigos.publicados.ArtigosPublicadosXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.DemaisTiposProducaoBibliograficaXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.outra.OutraProducaoBibliograficaXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.partiturasmusical.PartituraMusicalXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.prefaciosposfacio.PrefacioPosfacioXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.demaistipos.traducao.TraducaoXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.livros capitulos.LivrosCapitulosXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.livros capitulos.capitulos.livros publicados.CapitulosDeLivrosPublicadosXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.livros capitulos.livros publicados organizados.LivrosPublicadosOuOrganizadosXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.textos jornais revistas.TextosJornaisRevistasXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.trabalhos e eventos.TrabalhosEmEventosXmlDto;

@Service
public class ProducaoBibliograficaProcessService {

    @Autowired
    private ProducaoProcessService processService;

    public void processProducoes(ImportProcessDto processDto) {
        ProducaoBibliograficaXmlDto producaoBibliografica =
            processDto.getCurriculo().getProducaoBibliografica();
        this.processTrabalhosEmEventos(processDto,
            producaoBibliografica.getTrabalhosEventos());
        this.processArtigosPublicados(processDto,
            producaoBibliografica.getArtigosPublicados());
        this.processLivrosCapitulos(processDto, producaoBibliografica.getLivrosCapitulos());
        this.processTextosJornaisRevistas(processDto,
            producaoBibliografica.getTextosJornaisRevistas());
        this.processArtigosAceitos(processDto, producaoBibliografica.getArtigosAceitos());
        this.processDemaisProducoes(processDto,
            producaoBibliografica.getDemaisTiposProducao());
    }

    private void processTrabalhosEmEventos(ImportProcessDto processDto,
        TrabalhosEmEventosXmlDto trabalhosEventosXmlDto) {
        if (trabalhosEventosXmlDto != null) {
            this.processService.processProducao(processDto,
                trabalhosEventosXmlDto.getTrabalhosEventos(),
                TipoProducao.TRABALHO_EM_EVENTO);
        }
    }
}

```



```

private void processArtigosPublicados(ImportProcessDto processDto,
    ↪ ArtigosPublicadosXmlDto artigosPublicadosXmlDto) {
    if (artigosPublicadosXmlDto != null) {
        this.processService.processProducao(processDto,
            ↪ artigosPublicadosXmlDto.getArtigoPublicado(),
            ↪ TipoProducao.ARTIGO_PUBLICADO);
    }
}

private void processLivrosCapitulos(ImportProcessDto processDto, LivrosCapitulosXmlDto
    ↪ livrosCapitulosXmlDto) {
    if (livrosCapitulosXmlDto != null) {
        this.processLivros(processDto,
            ↪ livrosCapitulosXmlDto.getLivrosPublicadosOrganizados());
        this.processCapitulos(processDto,
            ↪ livrosCapitulosXmlDto.getCapitulosLivrosPublicados());
    }
}

private void processLivros(ImportProcessDto processDto,
    ↪ LivrosPublicadosOuOrganizadosXmlDto livrosPublicadosOrganizadosXmlDto) {
    if (livrosPublicadosOrganizadosXmlDto != null) {
        this.processService.processProducao(processDto,
            ↪ livrosPublicadosOrganizadosXmlDto.getLivrosPublicadosOrganizados(),
            ↪ TipoProducao.LIVRO_PUBLICADO_OU_ORGANIZADO);
    }
}

private void processCapitulos(ImportProcessDto processDto,
    ↪ CapitulosDeLivrosPublicadosXmlDto capitulosLivrosPublicadosXmlDto) {
    if (capitulosLivrosPublicadosXmlDto != null) {
        this.processService.processProducao(processDto,
            ↪ capitulosLivrosPublicadosXmlDto.getCapituloLivroPublicado(),
            ↪ TipoProducao.CAPITULO_DE_LIVRO);
    }
}

private void processTextosJornaisRevistas(ImportProcessDto processDto,
    ↪ TextosJornaisRevistasXmlDto textosJornaisRevistasXmlDto) {
    if (textosJornaisRevistasXmlDto != null) {
        this.processService.processProducao(processDto,
            ↪ textosJornaisRevistasXmlDto.getTextoJornalRevista(),
            ↪ TipoProducao.TEXTO_JORNAL_OU_REVISTA);
    }
}

private void processArtigosAceitos(ImportProcessDto processDto,
    ↪ ArtigosAceitosParaPublicacaoXmlDto artigosAceitosXmlDto) {
    if (artigosAceitosXmlDto != null) {
        this.processService.processProducao(processDto,
            ↪ artigosAceitosXmlDto.getArtigoAceito(),
            ↪ TipoProducao.ARTIGO_ACEITO);
    }
}

private void processDemaisProducoes(ImportProcessDto processDto,
    ↪ DemaisTiposProducaoBibliograficaXmlDto demaisTiposProducao) {
    if (demaisTiposProducao != null) {
        this.processTraducoes(processDto, demaisTiposProducao);
        this.processPartituras(processDto, demaisTiposProducao);
        this.processPrefaciosPosfacios(processDto, demaisTiposProducao);
        this.processOutrasProducoes(processDto, demaisTiposProducao);
    }
}

private void processTraducoes(ImportProcessDto processDto,
    ↪ DemaisTiposProducaoBibliograficaXmlDto demaisTiposProducao) {
    List<TraducaoXmlDto> traducoes = demaisTiposProducao.getTraducao();
    if (traducoes != null) {
        this.processService.processProducao(processDto, traducoes,
            ↪ TipoProducao.TRADUCAO);
    }
}

private void processPartituras(ImportProcessDto processDto,
    ↪ DemaisTiposProducaoBibliograficaXmlDto demaisTiposProducao) {
    List<PartituraMusicalXmlDto> partiturasMusicais =
        ↪ demaisTiposProducao.getPartituraMusical();
    if (partiturasMusicais != null) {
        this.processService.processProducao(processDto, partiturasMusicais,
            ↪ TipoProducao.PARTITURA);
    }
}

```

```

    }
}

private void processPrefaciosPosfacios(ImportProcessDto processDto,
↳ DemaisTiposProducaoBibliograficaXmlDto demaisTiposProducao) {
    List<PrefacioPosfacioXmlDto> prefaciosPosfacios =
        demaisTiposProducao.getPrefacioPosfacio();
    if (prefaciosPosfacios != null) {
        this.processService.processProducao(processDto, prefaciosPosfacios,
↳ TipoProducao.PREFACIO_POSFACIO);
    }
}

private void processOutrasProducoes(ImportProcessDto processDto,
↳ DemaisTiposProducaoBibliograficaXmlDto demaisTiposProducao) {
    List<OutraProducaoBibliograficaXmlDto> outrasProducoesBibliograficas =
        demaisTiposProducao.getOutraProducaoBibliografica();
    if (outrasProducoesBibliograficas != null) {
        this.processService.processProducao(processDto,
↳ outrasProducoesBibliograficas,
↳ TipoProducao.OUTRA_PRODUCAO_BIBLIOGRAFICA);
    }
}

}
package alpc.ufsc.pessoa.producao.service;

import java.util.List;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import alpc.ufsc.entity.pessoa.producao.TipoProducao;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.producao.dto.ProducaoDto;
import alpc.ufsc.pessoa.producao.dto.ProducaoSearchDto;
import alpc.ufsc.programa.dto.ImportProcessDto;
import ufsc.alpc.xml.dto.CurriculoVitaeXmlDto;
import ufsc.alpc.xml.dto.producao.bibliografica.DadosBasicosProducao;
import ufsc.alpc.xml.dto.producao.bibliografica.Producao;

@Slf4j
@Service
@Transactional
public class ProducaoProcessService {

    @Autowired
    private ProducaoSearchService producaoSearch;

    @Autowired
    private ProducaoService service;

    @Transactional(propagation = Propagation.REQUIRES_NEW)
    public <T extends Producao> void processProducao(ImportProcessDto processDto,
↳ List<T> producoes, TipoProducao tipoProducao) {
        if (producoes == null) {
            return;
        }
        for (Producao producao : producoes) {
            DadosBasicosProducao dadosBasicos = producao.getDadosBasicos();
            int ano = QueryUtils.parseInt(dadosBasicos.getAno());
            String titulo = dadosBasicos.getTitulo();
            Long programaId = processDto.getProgramaDto().getId();
            try {
                Long producaoId = this.search(processDto.getCurriculo(), titulo,
↳ ano, tipoProducao, programaId);
                ProducaoDto producaoDto = new ProducaoDto(producaoId,
↳ programaId, processDto.getDocenteId(), titulo, ano,
↳ tipoProducao, producao.getAutores());
                this.service.saveProducao(producaoDto);
            } catch (Exception e) {
                log.error("Erro ao processar produto", e.getMessage());
            }
        }
    }

    private Long search(CurriculoVitaeXmlDto curriculo, String titulo, int ano, TipoProducao
↳ tipoProducao, Long programaId) throws Exception {

```

```

        String nomeCompleto = curriculo.getDadosGerais().getNomeCompleto();
        String nomeCitacoesBibliograficas =
            ↳ curriculo.getDadosGerais().getNomeCitacoesBibliograficas();
        String numeroIdentificador = curriculo.getNumeroIdentificador();
        return this.producaoSearch.find(new ProducaoSearchDto(titulo, ano, tipoProducao,
            ↳ programaId, nomeCompleto, nomeCitacoesBibliograficas.split(","),
            ↳ numeroIdentificador));
    }
}
package alpc.ufsc.pessoa.producao.service;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.apache.lucene.search.BooleanClause.Occur;
import org.apache.lucene.search.BooleanQuery;
import org.hibernate.search.SearchFactory;
import org.hibernate.search.errors.EmptyQueryException;
import org.hibernate.search.jpa.FullTextEntityManager;
import org.hibernate.search.jpa.FullTextQuery;
import org.hibernate.search.jpa.Search;
import org.hibernate.search.query.dsl.QueryBuilder;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import alpc.ufsc.entity.pessoa.producao.AutorProducaoEntity;
import alpc.ufsc.pessoa.producao.dto.ProducaoSearchDto;

@Service
@Transactional(propagation = Propagation.SUPPORTS, noRollbackFor =
    ↳ EmptyQueryException.class)
public class ProducaoSearchService {

    private static final String TIPO_PRODUCAO = "producao.tipo_producao";
    private static final String ANO = "producao.ano";
    private static final String TITULO_DA_PRODUCAO = "producao.titulo";
    private static final String ID = "producao.id";
    private static final String PROGRAMA_ID = "producao.programa_id";
    private static final String NRO_ID_AUTOR = "autor.numero_identificador";
    private static final String NOME_AUTOR = "autor.nome";
    private static final String NOME_CITACAO = "autor.citacao";

    @PersistenceContext
    private EntityManager em;

    public Long find(ProducaoSearchDto searchDto) throws Exception {
        FullTextEntityManager search = Search.getFullTextEntityManager(this.em);
        search.flushToIndexes();
        SearchFactory searchFactory = search.getSearchFactory();
        QueryBuilder qb =
            ↳ searchFactory.buildQueryBuilder().forEntity(AutorProducaoEntity.class).get();

        BooleanQuery query = new BooleanQuery();

        BooleanQuery queryProducao = this.queryProducao(searchDto, qb);
        query.add(queryProducao, Occur.MUST);

        BooleanQuery queryNome = this.queryAutor(searchDto, qb);
        query.add(queryNome, Occur.MUST);

        FullTextQuery fullTextQuery = search.createFullTextQuery(query,
            ↳ AutorProducaoEntity.class);

        fullTextQuery.setProjection(this.getProjection());
        fullTextQuery.setMaxResults(10);

        List<?> result = fullTextQuery.getResultList();
        for (Object object : result) {
            Object[] fields = (Object[]) object;
            if (searchDto.similar(new ProducaoSearchDto((String) fields[1])) {
                return (Long) fields[0];
            }
        }
        return null;
    }

    private BooleanQuery queryAutor(ProducaoSearchDto searchDto, QueryBuilder qb) {
        BooleanQuery queryNome = new BooleanQuery();

```

```

        queryNome.add(qb.keyword().onField(NRO_ID_AUTOR).matching(searchDto.getNumeroIdentificador()).createQuery(),
            ↳ Occur.SHOULD);
        queryNome.add(qb.keyword().onField(NOME_AUTOR).matching(searchDto.getNomeAutor()).createQuery(),
            ↳ Occur.SHOULD);
        for (String nomeCitacao : searchDto.getNomesCitacao()) {
            queryNome.add(qb.keyword().onField(NOME_CITACAO).matching(nomeCitacao).createQuery(),
                ↳ Occur.SHOULD);
        }
        return queryNome;
    }

    private BooleanQuery queryProducao(ProducaoSearchDto searchDto, QueryBuilder qb) {
        BooleanQuery queryProducao = new BooleanQuery();
        queryProducao.add(qb.keyword().onField(TITULO_DA_PRODUCAO).matching(searchDto.getTitulo()).createQuery(),
            ↳ Occur.MUST);
        queryProducao.add(qb.keyword().onField(ANO).matching(searchDto.getAno()).createQuery(),
            ↳ Occur.MUST);
        queryProducao.add(qb.keyword().onField(TIPO_PRODUCAO).matching(searchDto.getTipoProducao()).createQuery(),
            ↳ Occur.MUST);
        queryProducao.add(qb.keyword().onField(PROGRAMA_ID).matching(searchDto.getProgramaId()).createQuery(),
            ↳ Occur.MUST);
        return queryProducao;
    }

    private String[] getProjection() {
        String[] p = { ID, TITULO_DA_PRODUCAO };
        return p;
    }
}

package alpc.ufsc.pessoa.producao.service.technica;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import alpc.ufsc.entity.pessoa.producao.TipoProducao;
import alpc.ufsc.pessoa.producao.service.ProducaoProcessService;
import alpc.ufsc.programa.dto.ImportProcessDto;
import ufs.alpc.xml.dto.producao.technica.ProducaoTechnicaXmlDto;
import ufs.alpc.xml.dto.producao.technica.demais.producoes.DemaisTipoDeProducaoTechnicaXmlDto;

@Service
public class ProducaoTechnicaProcessService {

    @Autowired
    private ProducaoProcessService processService;

    public void processProducoes(ImportProcessDto processDto) {
        ProducaoTechnicaXmlDto producaoTechnica =
            ↳ processDto.getCurriculo().getProducaoTechnica();
        if (producaoTechnica != null) {
            this.processService.processProducao(processDto,
                ↳ producaoTechnica.getCultivarProtegida(),
                ↳ TipoProducao.CULTIVAR_REGISTRADA);
            this.processService.processProducao(processDto,
                ↳ producaoTechnica.getCultivarRegistrada(),
                ↳ TipoProducao.CULTIVAR_PROTEGIDA);
            this.processService.processProducao(processDto,
                ↳ producaoTechnica.getDesenhoIndustrial(),
                ↳ TipoProducao.DESENHO_INDUSTRIAL);
            this.processService.processProducao(processDto,
                ↳ producaoTechnica.getMarca(), TipoProducao.MARCA);
            this.processService.processProducao(processDto,
                ↳ producaoTechnica.getPatente(), TipoProducao.PATENTE);
            this.processService.processProducao(processDto,
                ↳ producaoTechnica.getProcessosTecnicas(),
                ↳ TipoProducao.PROCESSO_TECNICA);
            this.processService.processProducao(processDto,
                ↳ producaoTechnica.getProdutoTecnologico(),
                ↳ TipoProducao.PRODUTO_TECNOLOGICO);
            this.processService.processProducao(processDto,
                ↳ producaoTechnica.getSoftware(), TipoProducao.SOFTWARE);
            this.processService.processProducao(processDto,
                ↳ producaoTechnica.getTopografiaDeCircuitoIntegrado(),
                ↳ TipoProducao.TOPOGRAFIA_CIRCUITO_INTEGRADO);
            this.processService.processProducao(processDto,
                ↳ producaoTechnica.getTrabalhoTechnico(),
                ↳ TipoProducao.TRABALHO_TECNICO);

            this.processDemaisProducoesTecnicas(processDto, producaoTechnica);
        }
    }
}

```

```

    }
}

private void processDemaisProducoesTecnicas(ImportProcessDto processDto,
↳ ProducaoTecnicaXmlDto producaoTecnica) {
    List<DemaisTipoDeProducaoTecnicaXmlDto> demaisTiposDeProducaoTecnica =
↳ producaoTecnica.getDemaisTiposDeProducaoTecnica();
    if (demaisTiposDeProducaoTecnica != null) {
        for (DemaisTipoDeProducaoTecnicaXmlDto demaisTipoDeProducaoTecnica
↳ : demaisTiposDeProducaoTecnica) {
            this.processService.processProducao(processDto,
↳ demaisTipoDeProducaoTecnica.getApresentacaoTrabalho(),
↳ TipoProducao.APRESENTACAO_TRABALHO);
            this.processService.processProducao(processDto,
↳ demaisTipoDeProducaoTecnica.getCartaMapaSimilar(),
↳ TipoProducao.CARTA_MAPA_SIMILAR);
            this.processService.processProducao(processDto,
↳ demaisTipoDeProducaoTecnica.getCursoCurtaDuracao(),
↳ TipoProducao.CURSO_CURTA_DURACAO);
            this.processService.processProducao(processDto,
↳ demaisTipoDeProducaoTecnica.getDesenvolvimentoMaterialDidaticoInstr
↳ TipoProducao.DESENVOLVIMENTO_MATERIAL_DIDATICO_INSTR);
            this.processService.processProducao(processDto,
↳ demaisTipoDeProducaoTecnica.getEditoracao(),
↳ TipoProducao.EDITORACAO);
            this.processService.processProducao(processDto,
↳ demaisTipoDeProducaoTecnica.getManutencaoObraArtistica(),
↳ TipoProducao.MANUTENCAO_OBRA_ARTISTICA);
            this.processService.processProducao(processDto,
↳ demaisTipoDeProducaoTecnica.getMaquete(),
↳ TipoProducao.MAQUETE);
            this.processService.processProducao(processDto,
↳ demaisTipoDeProducaoTecnica.getMidiaSocialWebsiteBlog(),
↳ TipoProducao.MIDIA_SOCIAL_WEBSITE_BLOG);
            this.processService.processProducao(processDto,
↳ demaisTipoDeProducaoTecnica.getOrganizacaoEvento(),
↳ TipoProducao.ORGANIZACAO_EVENTO);
            this.processService.processProducao(processDto,
↳ demaisTipoDeProducaoTecnica.getOutraProducaoTecnica(),
↳ TipoProducao.OUTRA_PRODUCAO_TECNICA);
            this.processService.processProducao(processDto,
↳ demaisTipoDeProducaoTecnica.getProgramaRadioTv(),
↳ TipoProducao.PROGRAMA_RADIO_TV);
            this.processService.processProducao(processDto,
↳ demaisTipoDeProducaoTecnica.getRelatorioPesquisa(),
↳ TipoProducao.RELATORIO_PESQUISA);
        }
    }
}

}

package alpc.ufsc.pessoa.orientacao.mestrado;

import static
↳ alpc.ufsc.entity.pessoa.orientacao.mestrado.QOrientacaoMestradoEntity.orientacaoMestradoEntity;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.jpaa.impl.JPAUpdateClause;

import alpc.ufsc.entity.pessoa.orientacao.mestrado.OrientacaoMestradoEntity;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoValidadeDto;
import alpc.ufsc.pessoa.orientacao.common.process.BaseOrientacaoGroupingProcessComponent;
import alpc.ufsc.alpc.xml.dto.CurriculoVitaeXmlDto;
import
↳ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.mestrado.OrientacaoEmAndamentoDeMestrado;
import
↳ ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.OrientacoesConcluidasXmlDto;
import
↳ ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.mestrado.OrientacaoConcluidaMestradoXmlDto;

@Service
@Transactional(propagation = Propagation.SUPPORTS)
public class OrientacaoMestradoGroupingProcessComponent
↳ extends BaseOrientacaoGroupingProcessComponent<OrientacaoMestradoEntity,
↳ OrientacaoEmAndamentoDeMestradoXmlDto,

```

```

        ↪ OrientacaoConcluidaMestradoXmlDto> {

    @PersistenceContext
    private EntityManager em;

    @Override
    @Transactional(propagation = Propagation.REQUIRES_NEW)
    public void disableAllOrientacoesDocente(Long docenteId) {
        JPAUpdateClause updateClause = new JPAUpdateClause(this.em,
            ↪ orientacaoMestradoEntity);
        updateClause.set(orientacaoMestradoEntity.orientadorAtivo, false);
        updateClause.where(orientacaoMestradoEntity.orientador().id.eq(docenteId));
        updateClause.execute();

        updateClause = new JPAUpdateClause(this.em, orientacaoMestradoEntity);
        updateClause.set(orientacaoMestradoEntity.coorientadorAtivo, false);
        updateClause.where(orientacaoMestradoEntity.coorientador().id.eq(docenteId));
        updateClause.execute();
    }

    // orientaes em andamento
    @Override
    public boolean hasOrientacoesEmAndamento(CurriculoVitaeXmlDto curriculo) {
        return curriculo.isSetDadosComplementares() &&
            ↪ curriculo.getDadosComplementares().isSetOrientacoesEmAndamento()
                && ↪ curriculo.getDadosComplementares().getOrientacoesEmAndamento().isSetOrientacoesEmAndamento();
    }

    @Override
    public List<OrientacaoEmAndamentoDeMestradoXmlDto>
        ↪ getOrientacoesEmAndamento(CurriculoVitaeXmlDto curriculo) {
        return ↪ curriculo.getDadosComplementares().getOrientacoesEmAndamento().getOrientacoesMestrado();
    }

    @Override
    public OrientacaoValidateDto
        ↪ getOrientacaoValidateDtoEmAndamento(OrientacaoEmAndamentoDeMestradoXmlDto
        ↪ orientacaoEmAndamento) {
        return new OrientacaoValidateDto(orientacaoEmAndamento);
    }

    // orientaes concluidas
    @Override
    public boolean hasOrientacoesConcluidas(CurriculoVitaeXmlDto curriculo) {
        return curriculo.isSetOutraProducao() &&
            ↪ curriculo.getOutraProducao().isSetOrientacoesConcluidas();
    }

    @Override
    public List<OrientacaoConcluidaMestradoXmlDto>
        ↪ getOrientacoesConcluidas(OrientacoesConcluidasXmlDto orientacaoConcluida) {
        return orientacaoConcluida.getOrientacoesConcluidasMestrado();
    }

    @Override
    public OrientacaoValidateDto
        ↪ getOrientacaoValidateDtoConcluida(OrientacaoConcluidaMestradoXmlDto
        ↪ orientacaoConcluida) {
        return new OrientacaoValidateDto(orientacaoConcluida);
    }

    @Override
    public boolean isSetOrientacoesConcluidas(OrientacoesConcluidasXmlDto
        ↪ orientacaoConcluida) {
        return orientacaoConcluida.isSetOrientacoesConcluidasMestrado();
    }
}

package alpc.ufsc.pessoa.orientacao.mestrado;

import org.springframework.stereotype.Component;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import alpc.ufsc.common.exception.EmptyXmlException;
import alpc.ufsc.domain.instituicao.dto.InstituicaoSaveDTO;
import alpc.ufsc.entity.domain.StatusOrientacao;
import alpc.ufsc.entity.pessoa.orientacao.mestrado.OrientacaoMestradoEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.dto.PessoaDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoSearchDto;

```

```

import alpc.ufsc.pessoa.orientacao.common.process.BaseOrientacaoProcessComponent;
import alpc.ufsc.programa.dto.ImportProcessDto;
import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoTipoXmlDto;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoConcluidaXmlDto;
import ufsc.alpc.xml.dto.common.area.conhecimento.AreasDoConhecimentoXmlDto;
import ufsc.alpc.xml.dto.domain.TipoDeOrientacao;
import ↪ ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.mestrado.OrientacaoConcluidaMestradoXmlDto;

@Component
@Transactional(propagation = Propagation.SUPPORTS)
public class OrientacaoMestradoConcluidaProcess extends
↪ BaseOrientacaoProcessComponent<OrientacaoMestradoEntity,
↪ OrientacaoConcluidaMestradoXmlDto> {

    @Override
    protected void setFieldToEntity(OrientacaoMestradoEntity orientacaoEntity,
↪ OrientacaoConcluidaMestradoXmlDto orientacaoXml) {
        DadosBasicosDaParticipacaoTipoXmlDto dadosBasicos =
↪ orientacaoXml.getDadosBasicos();
        DetalhamentoDaOrientacaoConcluidaXmlDto detalhamento =
↪ orientacaoXml.getDetalhamento();

        orientacaoEntity.setAnoInicio(QueryUtils.toMinYear(orientacaoEntity.getAnoInicio()));
        orientacaoEntity.setAnoFim(QueryUtils.toMaxYear(dadosBasicos.getAno()););
        orientacaoEntity.setTituloDoTrabalho(dadosBasicos.getTitulo());
        orientacaoEntity.setBolsa(detalhamento.getFlagBolsa());
        orientacaoEntity.setStatusOrientacao(StatusOrientacao.CONCLUIDO);
    }

    @Override
    public PessoaDto getPessoaDiscente(OrientacaoConcluidaMestradoXmlDto orientacao,
↪ ImportProcessDto processDto) {
        return new PessoaDto(orientacao, processDto.getProgramaDto().getId());
    }

    @Override
    public OrientacaoSearchDto getOrientacaoSearch(OrientacaoConcluidaMestradoXmlDto
↪ orientacao, ImportProcessDto processDto) {
        return new OrientacaoSearchDto(processDto, orientacao);
    }

    @Override
    public InstituicaoSaveDTO getInstituicaoSaveDTO(ImportProcessDto processDto,
↪ OrientacaoConcluidaMestradoXmlDto orientacao) throws EmptyXmlException {
        DetalhamentoDaOrientacaoConcluidaXmlDto detalhamento =
↪ orientacao.getDetalhamento();
        return new InstituicaoSaveDTO(detalhamento.getCodigoCurso(),
↪ detalhamento.getCodigoInstituicao(), detalhamento.getNomeInstituicao(),
↪ processDto.getCurriculo());
    }

    @Override
    public AreasDoConhecimentoXmlDto
↪ getAreasConhecimento(OrientacaoConcluidaMestradoXmlDto formacao) {
        return formacao.getAreasDoConhecimento();
    }

    @Override
    public TipoDeOrientacao getTipoOrientacao(OrientacaoConcluidaMestradoXmlDto
↪ orientacao) {
        return orientacao.getDetalhamento().getTipoOrientacao();
    }
}

package alpc.ufsc.pessoa.orientacao.mestrado;

import static
↪ alpc.ufsc.entity.pessoa.orientacao.mestrado.QOrientacaoMestradoEntity.orientacaoMestradoEntity;

import java.util.List;

import org.springframework.stereotype.Component;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.jpa.impl.JPAUpdateClause;

import alpc.ufsc.common.exception.EmptyXmlException;
import alpc.ufsc.domain.instituicao.dto.InstituicaoSaveDTO;
import alpc.ufsc.entity.domain.StatusOrientacao;
import alpc.ufsc.entity.pessoa.orientacao.mestrado.OrientacaoMestradoEntity;

```

```

import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.dto.PessoaDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoSearchDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoValidateDto;
import alpc.ufsc.pessoa.orientacao.common.process.BaseFormacaoProcessComponent;
import alpc.ufsc.programa.dto.ImportProcessDto;
import ufsc.alpc.xml.dto.CurriculoVitaeXmlDto;
import ufsc.alpc.xml.dto.common.area.conhecimento.AreasDoConhecimentoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.mestrado.MestradoXmlDto;

@Component
@Transactional(propagation = Propagation.SUPPORTS)
public class FormacaoMestradoProcessComponent extends
    BaseFormacaoProcessComponent<OrientacaoMestradoEntity, MestradoXmlDto> {

    @Override
    public OrientacaoSearchDto getOrientacaoSearch(MestradoXmlDto formacao,
        ImportProcessDto processDto) {
        return new OrientacaoSearchDto(processDto, formacao);
    }

    @Override
    public PessoaDto getPessoaDocente(MestradoXmlDto formacao, ImportProcessDto
        processDto, boolean orientadorPrincipal) {
        return new PessoaDto(formacao, processDto.getProgramaDto().getId(),
            orientadorPrincipal);
    }

    @Override
    public boolean hasFormacoes(CurriculoVitaeXmlDto curriculo) {
        return curriculo.getDadosGerais().isSetFormacaoAcademica() &&
            curriculo.getDadosGerais().getFormacaoAcademica().isSetMestrado();
    }

    @Override
    public List<MestradoXmlDto> getFormacoes(CurriculoVitaeXmlDto curriculo) {
        return curriculo.getDadosGerais().getFormacaoAcademica().getMestrado();
    }

    @Override
    protected void setFieldsToEntity(OrientacaoMestradoEntity orientacaoEntity,
        MestradoXmlDto formacaoXml, ImportProcessDto processDto) {
        orientacaoEntity.setAnoInicio(QueryUtils.toMinYear(formacaoXml.getAnoInicio()));
        orientacaoEntity.setAnoFim(QueryUtils.toMaxYear(formacaoXml.getAnoConclusao()));
        orientacaoEntity.setTituloDoTrabalho(formacaoXml.getTituloDissertacaoTese());
        orientacaoEntity.setBolsa(formacaoXml.getFlagBolsa());
        orientacaoEntity.setStatusOrientacao(StatusOrientacao.getByStatusOrientacaoXML(formacaoXml.getStatusC
    )

    @Override
    public OrientacaoValidateDto getOrientacaoValidateDto(MestradoXmlDto formacao) {
        return new OrientacaoValidateDto(formacao);
    }

    @Override
    public InstituicaoSaveDTO getIntuicaoSaveDTO(ImportProcessDto processDto,
        MestradoXmlDto formacaoXml) throws EmptyXmlException {
        return new InstituicaoSaveDTO(formacaoXml.getCodigoCurso(),
            formacaoXml.getCodigoInstituicao(), formacaoXml.getNomeInstituicao(),
            processDto.getCurriculo());
    }

    @Override
    public AreasDoConhecimentoXmlDto getAreasConhecimento(MestradoXmlDto formacao) {
        return formacao.getAreasDoConhecimento();
    }

    @Override
    @Transactional(propagation = Propagation.REQUIRES_NEW)
    public void disableAllFormacoesDiscente(Long discenteId) {
        JPAUpdateClause updateClause = new JPAUpdateClause(this.em,
            orientacaoMestradoEntity);
        updateClause.set(orientacaoMestradoEntity.discenteCompleto, false);
        updateClause.where(orientacaoMestradoEntity.discente().id.eq(discenteId));
        updateClause.execute();
    }
}

package alpc.ufsc.pessoa.orientacao.mestrado;

import org.springframework.stereotype.Component;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

```



```

import alpc.ufsc.common.exception.EmptyXmlException;
import alpc.ufsc.domain.instituicao.dto.InstituicaoSaveDTO;
import alpc.ufsc.entity.domain.StatusOrientacao;
import alpc.ufsc.entity.pessoa.orientacao.mestrado.OrientacaoMestradoEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.dto.PessoaDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoSearchDto;
import alpc.ufsc.pessoa.orientacao.common.process.BaseOrientacaoProcessComponent;
import alpc.ufsc.programa.dto.ImportProcessDto;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoXmlDto;
import ufsc.alpc.xml.dto.common.area.conhecimento.AreasDoConhecimentoXmlDto;
import
↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.DadosBasicosDaOrientacaoEmAndamento;
import
↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.mestrado.OrientacaoEmAndamentoDeMestrado;
import ufsc.alpc.xml.dto.domain.TipoDeOrientacao;

@Component
@Transactional(propagation = Propagation.SUPPORTS)
public class OrientacaoMestradoEmAndamentoProcess extends
↪ BaseOrientacaoProcessComponent<OrientacaoMestradoEntity,
↪ OrientacaoEmAndamentoDeMestradoXmlDto> {

    @Override
    protected void setFieldToEntity(OrientacaoMestradoEntity orientacaoEntity,
↪ OrientacaoEmAndamentoDeMestradoXmlDto orientacaoXml) {
        DadosBasicosDaOrientacaoEmAndamentoXmlDto dadosBasicos =
↪ orientacaoXml.getDadosBasicos();
        DetalhamentoDaOrientacaoXmlDto detalhamento =
↪ orientacaoXml.getDetalhamento();

        orientacaoEntity.setAnoInicio(QueryUtils.toMinYear(dadosBasicos.getAno()));
        orientacaoEntity.setAnoFim(QueryUtils.toMaxYear(orientacaoEntity.getAnoFim()));
        orientacaoEntity.setTituloDoTrabalho(dadosBasicos.getTituloDoTrabalho());
        orientacaoEntity.setBolsa(detalhamento.getFlagBolsa());
        orientacaoEntity.setStatusOrientacao(StatusOrientacao.EM_ANDAMENTO);
    }

    @Override
    public PessoaDto getPessoaDiscente(OrientacaoEmAndamentoDeMestradoXmlDto
↪ orientacao, ImportProcessDto processDto) {
        return new PessoaDto(orientacao, processDto.getProgramaDto().getId());
    }

    @Override
    public OrientacaoSearchDto
↪ getOrientacaoSearch(OrientacaoEmAndamentoDeMestradoXmlDto orientacao,
↪ ImportProcessDto processDto) {
        return new OrientacaoSearchDto(processDto, orientacao);
    }

    @Override
    public InstituicaoSaveDTO getInstituicaoSaveDTO(ImportProcessDto processDto,
↪ OrientacaoEmAndamentoDeMestradoXmlDto orientacao) throws
↪ EmptyXmlException {
        DetalhamentoDaOrientacaoXmlDto detalhamento = orientacao.getDetalhamento();
        return new InstituicaoSaveDTO(detalhamento.getCodigoCurso(),
↪ detalhamento.getCodigoInstituicao(), detalhamento.getNomeInstituicao(),
↪ processDto.getCurriculo());
    }

    @Override
    public AreasDoConhecimentoXmlDto
↪ getAreasConhecimento(OrientacaoEmAndamentoDeMestradoXmlDto formacao) {
        return formacao.getAreasDoConhecimento();
    }

    @Override
    public TipoDeOrientacao getTipoOrientacao(OrientacaoEmAndamentoDeMestradoXmlDto
↪ orientacao) {
        return orientacao.getDetalhamento().getTipoOrientacao();
    }
}

package alpc.ufsc.pessoa.orientacao.graduacao;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

```

```

import org.apache.lucene.search.BooleanClause.Occur;
import org.apache.lucene.search.BooleanQuery;
import org.hibernate.search.SearchFactory;
import org.hibernate.search.errors.EmptyQueryException;
import org.hibernate.search.jpamodeling.FullTextEntityManager;
import org.hibernate.search.jpamodeling.FullTextQuery;
import org.hibernate.search.jpamodeling.Search;
import org.hibernate.search.query.dsl.QueryBuilder;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import alpc.ufsc.entity.pessoa.orientacao.graduacao.GraduacaoEntity;
import alpc.ufsc.entity.pessoa.orientacao.common.dto.OrientacaoSearchDto;

@Service
@Transactional(propagation = Propagation.SUPPORTS, noRollbackFor =
    ↳ EmptyQueryException.class)
public class GraduacaoSearchService {

    private static final String TITULO = "titulo";
    private static final String DISCENTE_PESSOA_NOME = "discente.pessoa.nome";
    private static final String PROGRAMA_ID = "programa.id";
    private static final String ID = "id";

    @PersistenceContext
    private EntityManager em;

    public Long find(OrientacaoSearchDto searchDto) {
        FullTextEntityManager search = Search.getFullTextEntityManager(this.em);
        search.flushToIndexes();
        SearchFactory searchFactory = search.getSearchFactory();
        QueryBuilder qb =
            ↳ searchFactory.buildQueryBuilder().forEntity(GraduacaoEntity.class).get();

        BooleanQuery query = new BooleanQuery();

        query.add(qb.keyword().onField(TITULO).matching(searchDto.getTituloDoTrabalho()).createQuery(),
            ↳ Occur.MUST);
        query.add(qb.keyword().onField(DISCENTE_PESSOA_NOME).matching(searchDto.getNomeDiscente()).createQuery(),
            ↳ Occur.MUST);
        query.add(qb.keyword().onField(PROGRAMA_ID).matching(searchDto.getProgramaId()).createQuery(),
            ↳ Occur.MUST);

        FullTextQuery fullTextQuery = search.createFullTextQuery(query,
            ↳ GraduacaoEntity.class);

        fullTextQuery.setProjection(this.getProjection());
        fullTextQuery.setMaxResults(10);

        List<?> result = fullTextQuery.getResultList();
        for (Object object : result) {
            Object[] fields = (Object[]) object;
            if (searchDto.similar(new OrientacaoSearchDto((String) fields[0], (String)
                ↳ fields[1]))) {
                return (Long) fields[2];
            }
        }
        return null;
    }

    private String[] getProjection() {
        //@formatter:off
        String[] p = new String[] {
            TITULO,
            DISCENTE_PESSOA_NOME,
            ID
        };
        //@formatter:on
        return p;
    }
}

package alpc.ufsc.entity.pessoa.orientacao.graduacao;

import static alpc.ufsc.entity.pessoa.orientacao.graduacao.QGraduacaoEntity.graduacaoEntity;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

```

```

import lombok.extern.slf4j.Slf4j;

import org.hibernate.search.errors.EmptyQueryException;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.jpaa.impl.JPAUpdateClause;

import alpc.ufsc.common.exception.EmptyXmlException;
import alpc.ufsc.domain.instituicao.InstituicaoService;
import alpc.ufsc.domain.instituicao.dto.InstituicaoSaveDTO;
import alpc.ufsc.entity.domain.instituicao.InstituicaoEntity;
import alpc.ufsc.entity.pessoa.discente.DiscenteEntity;
import alpc.ufsc.entity.pessoa.orientacao_graduacao.GraduacaoEntity;
import alpc.ufsc.entity.programa.ProgramaEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoSearchDto;
import alpc.ufsc.programa.dto.ImportProcessDto;
import alpc.xml.dto.CurriculoVitaeXmlDto;
import alpc.xml.dto.dados.gerais.formacao_graduacao.GraduacaoXmlDto;

@Slf4j
@Service
@Transactional
public class GraduacaoProcessService {

    @PersistenceContext
    protected EntityManager em;

    @Autowired
    protected GraduacaoSearchService graduacaoService;

    @Autowired
    protected InstituicaoService instituicaoService;

    public void processGraduacao(ImportProcessDto processDto) {
        CurriculoVitaeXmlDto curriculo = processDto.getCurriculo();

        JPAUpdateClause updateClause = new JPAUpdateClause(this.em, graduacaoEntity);
        updateClause.set(graduacaoEntity.deletedado, true);
        updateClause.where(graduacaoEntity.discente().id.eq(processDto.getDiscenteId()));
        updateClause.execute();

        if (this.hasGraduacao(curriculo)) {
            List<GraduacaoXmlDto> graduacoes =
                ← curriculo.getDadosGerais().getFormacaoAcademica().getGraduacao();
            for (GraduacaoXmlDto graduacao : graduacoes) {
                ← this.save(processDto, graduacao);
            }
        }

        private boolean hasGraduacao(CurriculoVitaeXmlDto curriculo) {
            return curriculo.getDadosGerais().isSetFormacaoAcademica() &&
                ← curriculo.getDadosGerais().getFormacaoAcademica().isSetGraduacao();
        }

        private void save(ImportProcessDto processDto, GraduacaoXmlDto graduacaoXml) {
            Long graduacaoId = null;
            try {
                graduacaoId = this.graduacaoService.find(new
                    ← OrientacaoSearchDto(processDto, graduacaoXml));
            } catch (EmptyQueryException e) {
                log.error(e.getMessage());
                return;
            }
            GraduacaoEntity graduacao = new GraduacaoEntity();
            if (graduacaoId != null) {
                graduacao = this.em.find(GraduacaoEntity.class, graduacaoId);
            }
            graduacao.setDeletado(false);
            graduacao.setDiscente(this.em.getReference(DiscenteEntity.class,
                ← processDto.getDiscenteId()));
            graduacao.setPrograma(this.em.getReference(ProgramaEntity.class,
                ← processDto.getProgramaDto().getId());
            graduacao.setAnoInicio(QueryUtils.toMinYear(graduacaoXml.getAnoInicio()));
            graduacao.setAnoFim(QueryUtils.toMaxYear(graduacaoXml.getAnoConclusao()));
            graduacao.setTitulo(graduacaoXml.getTituloTcc());
            try {
                graduacao.setInstituicao(this.em.getReference(InstituicaoEntity.class,
                    ← this.instituicaoService.save(this.getInstituicaoSaveDTO(processDto,

```

```

        ↪ graduacaoXml));
        this.em.persist(graduacao);
    } catch (EmptyXmlException e) {
        log.error(e.getMessage());
    }
}

private InstituicaoSaveDTO getIntituicaoSaveDTO(ImportProcessDto processDto,
    ↪ GraduacaoXmlDto formacaoXml) throws EmptyXmlException {
    return new InstituicaoSaveDTO(formacaoXml.getCodigoCurso(),
        ↪ formacaoXml.getCodigoInstituicao(), formacaoXml.getNomeInstituicao(),
        ↪ processDto.getCurriculo());
}

}

package alpc.ufsc.pessoa.orientacao.doutorado;

import org.springframework.stereotype.Component;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import alpc.ufsc.common.exception.EmptyXmlException;
import alpc.ufsc.domain.instituicao.dto.InstituicaoSaveDTO;
import alpc.ufsc.entity.domain.StatusOrientacao;
import alpc.ufsc.entity.pessoa.orientacao.doutorado.OrientacaoDoutoradoEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.dto.PessoaDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoSearchDto;
import alpc.ufsc.pessoa.orientacao.common.process.BaseOrientacaoProcessComponent;
import alpc.ufsc.programa.dto.ImportProcessDto;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoXmlDto;
import ufsc.alpc.xml.dto.common.area.conhecimento.AreasDoConhecimentoXmlDto;
import ↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.DadosBasicosDaOrientacaoEmAndamentoXmlDto;
import ↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.doutorado.OrientacaoEmAndamentoDeDoutoradoXmlDto;
import ufsc.alpc.xml.dto.domain.TipoDeOrientacao;

@Component
@Transactional(propagation = Propagation.SUPPORTS)
public class OrientacaoDoutoradoEmAndamentoProcessComponent extends
    ↪ BaseOrientacaoProcessComponent<OrientacaoDoutoradoEntity,
    ↪ OrientacaoEmAndamentoDeDoutoradoXmlDto> {

    @Override
    public OrientacaoSearchDto
        ↪ getOrientacaoSearch(OrientacaoEmAndamentoDeDoutoradoXmlDto orientacao,
        ↪ ImportProcessDto processDto) {
        return new OrientacaoSearchDto(processDto, orientacao);
    }

    @Override
    public PessoaDto getPessoaDiscente(OrientacaoEmAndamentoDeDoutoradoXmlDto
        ↪ orientacao, ImportProcessDto processDto) {
        return new PessoaDto(orientacao, processDto.getProgramaDto().getId());
    }

    @Override
    protected void setFieldToEntity(OrientacaoDoutoradoEntity orientacaoEntity,
        ↪ OrientacaoEmAndamentoDeDoutoradoXmlDto orientacaoXml) {
        DadosBasicosDaOrientacaoEmAndamentoXmlDto dadosBasicos =
            ↪ orientacaoXml.getDadosBasicos();
        DetalhamentoDaOrientacaoXmlDto detalhamento =
            ↪ orientacaoXml.getDetalhamento();

        orientacaoEntity.setAnoInicio(QueryUtils.toMinYear(dadosBasicos.getAno()));
        orientacaoEntity.setAnoFim(QueryUtils.toMaxYear(orientacaoEntity.getAnoFim()));
        orientacaoEntity.setTituloDoTrabalho(dadosBasicos.getTituloDoTrabalho());
        orientacaoEntity.setBolsa(detalhamento.getFlagBolsa());
        orientacaoEntity.setStatusOrientacao(StatusOrientacao.EM_ANDAMENTO);
    }

    @Override
    public InstituicaoSaveDTO getIntituicaoSaveDTO(ImportProcessDto processDto,
        ↪ OrientacaoEmAndamentoDeDoutoradoXmlDto orientacao) throws
        ↪ EmptyXmlException {
        DetalhamentoDaOrientacaoXmlDto detalhamento = orientacao.getDetalhamento();
        return new InstituicaoSaveDTO(detalhamento.getCodigoCurso(),
            ↪ detalhamento.getCodigoInstituicao(), detalhamento.getNomeInstituicao(),
            ↪ processDto.getCurriculo());
    }
}

```

```

@Override
public AreasDoConhecimentoXmlDto
    ↪ getAreasConhecimento(OrientacaoEmAndamentoDeDoutoradoXmlDto formacao) {
    return formacao.getAreasDoConhecimento();
}

@Override
public TipoDeOrientacao getTipoOrientacao(OrientacaoEmAndamentoDeDoutoradoXmlDto
    ↪ orientacao) {
    return orientacao.getDetalhamento().getTipoOrientacao();
}
}

package alpc.ufsc.pessoa.orientacao.doutorado;

import static
    ↪ alpc.ufsc.entity.pessoa.orientacao.doutorado.QOrientacaoDoutoradoEntity.orientacaoDoutoradoEntity;

import java.util.List;

import org.apache.commons.lang3.StringUtils;
import org.springframework.stereotype.Component;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.jpa.impl.JPAUpdateClause;

import alpc.ufsc.common.exception.EmptyXmlException;
import alpc.ufsc.domain.instituicao.dto.InstituicaoSaveDTO;
import alpc.ufsc.entity.domain.StatusOrientacao;
import alpc.ufsc.entity.pessoa.orientacao.doutorado.OrientacaoDoutoradoEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.dto.PessoaDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoSearchDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoValidateDto;
import alpc.ufsc.pessoa.orientacao.common.process.BaseFormacaoProcessComponent;
import alpc.ufsc.programa.dto.ImportProcessDto;
import ufsc.alpc.xml.dto.CurriculoVitaeXmlDto;
import ufsc.alpc.xml.dto.common.area.conhecimento.AreasDoConhecimentoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.doutorado.DoutoradoXmlDto;

@Component
@Transactional(propagation = Propagation.SUPPORTS)
public class FormacaoDoutoradoProcessComponent extends
    ↪ BaseFormacaoProcessComponent<OrientacaoDoutoradoEntity, DoutoradoXmlDto> {

    @Override
    public OrientacaoSearchDto getOrientacaoSearch(DoutoradoXmlDto formacao,
    ↪ ImportProcessDto processDto) {
    return new OrientacaoSearchDto(processDto, formacao);
}

    @Override
    public PessoaDto getPessoaDocente(DoutoradoXmlDto formacao, ImportProcessDto
    ↪ processDto, boolean orientadorPrincipal) {
    return new PessoaDto(formacao, processDto.getProgramaDto().getId(),
    ↪ orientadorPrincipal);
}

    @Override
    public boolean hasFormacoes(CurriculoVitaeXmlDto curriculo) {
    return curriculo.getDadosGerais().isSetFormacaoAcademica() &&
    ↪ curriculo.getDadosGerais().getFormacaoAcademica().isSetDoutorado();
}

    @Override
    public List<DoutoradoXmlDto> getFormacoes(CurriculoVitaeXmlDto curriculo) {
    return curriculo.getDadosGerais().getFormacaoAcademica().getDoutorado();
}

    @Override
    protected void setFieldsToEntity(OrientacaoDoutoradoEntity orientacaoEntity,
    ↪ DoutoradoXmlDto formacaoXml, ImportProcessDto processDto) {
    orientacaoEntity.setAnoInicio(QueryUtils.toMinYear(formacaoXml.getAnoInicio()));
    orientacaoEntity.setAnoFim(QueryUtils.toMaxYear(formacaoXml.getAnoConclusao()));
    orientacaoEntity.setTituloDoTrabalho(formacaoXml.getTituloDissertacaoTese());
    orientacaoEntity.setBolsa(formacaoXml.getFlagBolsa());
    orientacaoEntity.setStatusOrientacao(StatusOrientacao.getByStatusOrientacaoXML(formacaoXml.getNomeOrient
    orientacaoEntity.setNomeOrientadorSanduiche(StringUtils.stripToNull(formacaoXml.getNomeOrient
}

    @Override

```

```

public OrientacaoValidateDto getOrientacaoValidateDto(DoutoradoXmlDto formacao) {
    return new OrientacaoValidateDto(formacao);
}

@Override
public InstituicaoSaveDTO getInstituicaoSaveDTO(ImportProcessDto processDto,
    ↳ DoutoradoXmlDto formacaoXml) throws EmptyXmlException {
    return new InstituicaoSaveDTO(formacaoXml.getCodigoCurso(),
    ↳ formacaoXml.getCodigoInstituicao(), formacaoXml.getNomeInstituicao(),
    ↳ processDto.getCurriculo());
}

@Override
public AreasDoConhecimentoXmlDto getAreasConhecimento(DoutoradoXmlDto formacao) {
    return formacao.getAreasDoConhecimento();
}

@Override
@Transactional(propagation = Propagation.REQUIRES_NEW)
public void disableAllFormacoesDiscente(Long discenteId) {
    JPAUpdateClause updateClause = new JPAUpdateClause(this.em,
    ↳ orientacaoDoutoradoEntity);
    updateClause.set(orientacaoDoutoradoEntity.discenteCompleto, false);
    updateClause.where(orientacaoDoutoradoEntity.discente().id.eq(discenteId));
    updateClause.execute();
}
}

package alpc.ufsc.pessoa.orientacao.doutorado;

import org.springframework.stereotype.Component;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import alpc.ufsc.common.exception.EmptyXmlException;
import alpc.ufsc.domain.instituicao.dto.InstituicaoSaveDTO;
import alpc.ufsc.entity.domain.StatusOrientacao;
import alpc.ufsc.entity.pessoa.orientacao.doutorado.OrientacaoDoutoradoEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.dto.PessoaDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoSearchDto;
import alpc.ufsc.pessoa.orientacao.common.process.BaseOrientacaoProcessComponent;
import alpc.ufsc.programa.dto.ImportProcessDto;
import alpc.ufsc.xml.dto.common.DadosBasicosDaParticipacaoFlagRelevanciaXmlDto;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoConcluidaXmlDto;
import ufsc.alpc.xml.dto.common.area.conhecimento.AreasDoConhecimentoXmlDto;
import ufsc.alpc.xml.dto.domain.TipoDeOrientacao;
import
    ↳ ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.doutorado.OrientacaoConcluidaDoutoradoXmlDto;

@Component
@Transactional(propagation = Propagation.SUPPORTS)
public class OrientacaoDoutoradoConcluidaProcessComponent extends
    ↳ BaseOrientacaoProcessComponent<OrientacaoDoutoradoEntity,
    ↳ OrientacaoConcluidaDoutoradoXmlDto> {

    @Override
    public OrientacaoSearchDto getOrientacaoSearch(OrientacaoConcluidaDoutoradoXmlDto
    ↳ orientacao, ImportProcessDto processDto) {
        return new OrientacaoSearchDto(processDto, orientacao);
    }

    @Override
    public PessoaDto getPessoaDiscente(OrientacaoConcluidaDoutoradoXmlDto orientacao,
    ↳ ImportProcessDto processDto) {
        return new PessoaDto(orientacao, processDto.getProgramaDto().getId());
    }

    @Override
    protected void setFieldToEntity(OrientacaoDoutoradoEntity orientacaoEntity,
    ↳ OrientacaoConcluidaDoutoradoXmlDto orientacaoXml) {
        DadosBasicosDaParticipacaoFlagRelevanciaXmlDto dadosBasicos =
            ↳ orientacaoXml.getDadosBasicos();
        DetalhamentoDaOrientacaoConcluidaXmlDto detalhamento =
            ↳ orientacaoXml.getDetalhamento();

        orientacaoEntity.setAnoInicio(QueryUtils.toMinYear(orientacaoEntity.getAnoInicio()));
        orientacaoEntity.setAnoFim(QueryUtils.toMaxYear(dadosBasicos.getAno()));
        orientacaoEntity.setTituloDoTrabalho(dadosBasicos.getTitulo());
        orientacaoEntity.setBolsa(detalhamento.getFlagBolsa());
        orientacaoEntity.setStatusOrientacao(StatusOrientacao.CONCLUIDO);
    }
}

```

```

@Override
public InstituicaoSaveDTO getInstituicaoSaveDTO(ImportProcessDto processDto,
↳ OrientacaoConcluidaDoutoradoXmlDto orientacao) throws EmptyXmlException {
    DetalhamentoDaOrientacaoConcluidaXmlDto detalhamento =
        ↳ orientacao.getDetalhamento();
    return new InstituicaoSaveDTO(detalhamento.getCodigoCurso(),
        ↳ detalhamento.getCodigoInstituicao(), detalhamento.getNomeInstituicao(),
        ↳ processDto.getCurriculo());
}

@Override
public AreasDoConhecimentoXmlDto
↳ getAreasConhecimento(OrientacaoConcluidaDoutoradoXmlDto formacao) {
    return formacao.getAreasDoConhecimento();
}

@Override
public TipoDeOrientacao getTipoOrientacao(OrientacaoConcluidaDoutoradoXmlDto
↳ orientacao) {
    return orientacao.getDetalhamento().getTipoOrientacao();
}
}

package alpc.ufsc.pessoa.orientacao.doutorado;

import static
↳ alpc.ufsc.entity.pessoa.orientacao.doutorado.QOrientacaoDoutoradoEntity.orientacaoDoutoradoEntity;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.jpa.impl.JPAUpdateClause;

import alpc.ufsc.entity.pessoa.orientacao.doutorado.OrientacaoDoutoradoEntity;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoValidateDto;
import alpc.ufsc.pessoa.orientacao.common.process.BaseOrientacaoGroupingProcessComponent;
import alpc.xml.dto.CurriculoVitaeXmlDto;
import
↳ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.doutorado.OrientacaoEmAndamentoDeDocente;
import
↳ ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.OrientacoesConcluidasXmlDto;
import
↳ ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.doutorado.OrientacaoConcluidaDoutoradoXmlDto;

@Service
@Transactional(propagation = Propagation.SUPPORTS)
public class OrientacaoDoutoradoGroupingProcessComponent
↳ extends BaseOrientacaoGroupingProcessComponent<OrientacaoDoutoradoEntity,
↳ OrientacaoEmAndamentoDeDoutoradoXmlDto,
↳ OrientacaoConcluidaDoutoradoXmlDto> {

    @PersistenceContext
    private EntityManager em;

    @Override
    @Transactional(propagation = Propagation.REQUIRES_NEW)
    public void disableAllOrientacoesDocente(Long docenteId) {
        JPAUpdateClause updateClause = new JPAUpdateClause(this.em,
↳ orientacaoDoutoradoEntity);
        updateClause.set(orientacaoDoutoradoEntity.orientadorAtivo, false);
        updateClause.where(orientacaoDoutoradoEntity.orientador().id.eq(docenteId));
        updateClause.execute();

        updateClause = new JPAUpdateClause(this.em, orientacaoDoutoradoEntity);
        updateClause.set(orientacaoDoutoradoEntity.coorientadorAtivo, false);
        updateClause.where(orientacaoDoutoradoEntity.coorientador().id.eq(docenteId));
        updateClause.execute();
    }

    // orientacoes em andamento
    @Override
    public boolean hasOrientacoesEmAndamento(CurriculoVitaeXmlDto curriculo) {
        return curriculo.isSetDadosComplementares() &&
↳ curriculo.getDadosComplementares().isSetOrientacoesEmAndamento()
↳ &&
↳ curriculo.getDadosComplementares().getOrientacoesEmAndamento().isSe
}

```

```

@Override
public List<OrientacaoEmAndamentoDeDoutoradoXmlDto>
    ↪ getOrientacoesEmAndamento(CurriculoVitaeXmlDto curriculo) {
    ↪ return ↪ curriculo.getDadosComplementares().getOrientacoesEmAndamento().getOrientacoesDoutorado();
}

@Override
public OrientacaoValidateDto
    ↪ getOrientacaoValidateDtoEmAndamento(OrientacaoEmAndamentoDeDoutoradoXmlDto
    ↪ orientacaoEmAndamento) {
    ↪ return new OrientacaoValidateDto(orientacaoEmAndamento);
}

// orientacoes concluidas
@Override
public boolean hasOrientacoesConcluidas(CurriculoVitaeXmlDto curriculo) {
    ↪ return curriculo.isSetOutraProducao() &&
    ↪ curriculo.getOutraProducao().isSetOrientacoesConcluidas();
}

@Override
public List<OrientacaoConcluidaDoutoradoXmlDto>
    ↪ getOrientacoesConcluidas(OrientacoesConcluidasXmlDto orientacaoConcluida) {
    ↪ return orientacaoConcluida.getOrientacoesConcluidasDoutorado();
}

@Override
public OrientacaoValidateDto
    ↪ getOrientacaoValidateDtoConcluida(OrientacaoConcluidaDoutoradoXmlDto
    ↪ orientacaoConcluida) {
    ↪ return new OrientacaoValidateDto(orientacaoConcluida);
}

@Override
public boolean isSetOrientacoesConcluidas(OrientacoesConcluidasXmlDto
    ↪ orientacaoConcluida) {
    ↪ return orientacaoConcluida.isSetOrientacoesConcluidasDoutorado();
}
}

package alpc.ufsc.pessoa.orientacao.common.process;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import alpc.ufsc.entity.pessoa.orientacao.BaseOrientacaoEntity;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoResultDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoValidateDto;
import alpc.ufsc.programa.dto.ImportProcessDto;
import ufsc.alpc.xml.dto.CurriculoVitaeXmlDto;
import ufsc.alpc.xml.dto.domain.TipoDeOrientacao;
import ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.OrientacoesConcluidasXmlDto;

public abstract class BaseOrientacaoGroupingProcessComponent<OrientacaoEntity extends
    ↪ BaseOrientacaoEntity, AndamentoType, ConcluidaType> {

    // andamento
    @Autowired
    ↪ protected BaseOrientacaoProcessComponent<OrientacaoEntity, AndamentoType>
    ↪ orientacaoEmAndamentoProcess;

    @Transactional(propagation = Propagation.REQUIRES_NEW)
    public void processOrientacaoEmAndamento(ImportProcessDto processDto, AndamentoType
    ↪ orientacao) {
    ↪ this.orientacaoEmAndamentoProcess.processOrientacao(processDto, orientacao);
    }

    public OrientacaoResultDto findOrientacaoEmAndamento(ImportProcessDto processDto,
    ↪ AndamentoType orientacao) {
    ↪ return this.orientacaoEmAndamentoProcess.find(processDto, orientacao);
    }

    public abstract void disableAllOrientacoesDocente(Long docenteId);

    public abstract boolean hasOrientacoesEmAndamento(CurriculoVitaeXmlDto curriculo);

    public abstract List<AndamentoType> getOrientacoesEmAndamento(CurriculoVitaeXmlDto
    ↪ curriculo);
}

```



```

public abstract OrientacaoValidadeDto
↳ getOrientacaoValidadeDtoEmAndamento(AndamentoType
↳ orientacaoEmAndamento);

public TipoDeOrientacao getTipoDeOrientacaoEmAndamento(AndamentoType
↳ orientacaoEmAndamento) {
    return ↳ this.orientacaoEmAndamentoProcess.getTipoOrientacao(orientacaoEmAndamento);
}

// concluida
@Autowired
protected BaseOrientacaoProcessComponent<OrientacaoEntity, ConcluidaType>
↳ orientacaoConcluidaProcess;

@Transactional(propagation = Propagation.REQUIRES_NEW)
public void processOrientacaoConcluida(ImportProcessDto processDto, ConcluidaType
↳ orientacao) {
    this.orientacaoConcluidaProcess.processOrientacao(processDto, orientacao);
}

public OrientacaoResultDto findOrientacaoConcluida(ImportProcessDto processDto,
↳ ConcluidaType orientacao) {
    return this.orientacaoConcluidaProcess.find(processDto, orientacao);
}

public TipoDeOrientacao getTipoDeOrientacaoConcluida(ConcluidaType
↳ orientacaoConcluida) {
    return this.orientacaoConcluidaProcess.getTipoOrientacao(orientacaoConcluida);
}

public abstract boolean hasOrientacoesConcluidas(CurriculoVitaXmlDto curriculo);

public abstract List<ConcluidaType>
↳ getOrientacoesConcluidas(OrientacoesConcluidasXmlDto orientacaoConcluida);

public abstract OrientacaoValidadeDto getOrientacaoValidadeDtoConcluida(ConcluidaType
↳ orientacaoConcluida);

public abstract boolean isSetOrientacoesConcluidas(OrientacoesConcluidasXmlDto
↳ orientacaoConcluida);
}
package alpc.ufsc.pessoa.orientacao.common.process;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;

import com.mysema.query.jpaa.impl.JPAQuery;

import alpc.ufsc.common.exception.EmptyXmlException;
import alpc.ufsc.domain.area.AreaConhecimentoService;
import alpc.ufsc.domain.area.dto.AreaConhecimentoSaveDTO;
import alpc.ufsc.domain.instituicao.InstituicaoService;
import alpc.ufsc.domain.instituicao.dto.InstituicaoSaveDTO;
import alpc.ufsc.entity.domain.area.AreaConhecimentoEntity;
import alpc.ufsc.entity.domain.instituicao.InstituicaoEntity;
import alpc.ufsc.entity.pessoa.PessoaEntity;
import alpc.ufsc.entity.pessoa.discente.DiscenteEntity;
import alpc.ufsc.entity.pessoa.discente.QDiscenteEntity;
import alpc.ufsc.entity.pessoa.docente.DocenteEntity;
import alpc.ufsc.entity.pessoa.orientacao.BaseOrientacaoEntity;
import alpc.ufsc.entity.programa.ProgramaEntity;
import alpc.ufsc.pessoa.discente.DiscenteService;
import alpc.ufsc.pessoa.docente.DocenteService;
import alpc.ufsc.pessoa.dto.PessoaDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoResultDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoSearchDto;
import alpc.ufsc.pessoa.orientacao.common.service.OrientacaoSearchService;
import alpc.ufsc.programa.dto.ImportProcessDto;
import net.jodah.typetools.TypeResolver;
import ufsc.alpc.xml.dto.common.area.conhecimento.AreaDoConhecimentoXmlDto;
import ufsc.alpc.xml.dto.common.area.conhecimento.AreasDoConhecimentoXmlDto;
import ufsc.alpc.xml.dto.domain.TipoDeOrientacao;

@Slf4j
public abstract class BaseOrientacaoProcessComponent<OrientacaoEntity extends
↳ BaseOrientacaoEntity, OrientacaoXml> {

```

```

@PersistenceContext
protected EntityManager em;

@Autowired
protected DiscenteService discenteService;

@Autowired
protected DocenteService docenteService;

@Autowired
protected OrientacaoSearchService orientacaoService;

@Autowired
protected InstituicaoService instituicaoService;

@Autowired
protected AreaConhecimentoService areaService;

protected Class<OrientacaoEntity> entityClass;

@SuppressWarnings("unchecked")
public BaseOrientacaoProcessComponent() {
    Class<?>[] resolveRawArguments =
        ↪ TypeResolver.resolveRawArguments(BaseOrientacaoProcessComponent.class,
        ↪ this.getClass());
    this.entityClass = (Class<OrientacaoEntity>) resolveRawArguments[0];
}

public void processOrientacao(ImportProcessDto processDto, OrientacaoXml orientacao) {
    OrientacaoResultDto resultDto;
    try {
        resultDto = this.find(processDto, orientacao);
    } catch (Exception e) {
        log.error("Erro ao buscar orientao", e.getMessage());
        return;
    }

    OrientacaoEntity orientacaoEntity = null;
    if (resultDto.isSetOrientacaoId()) {
        orientacaoEntity = this.em.find(this.entityClass,
            ↪ resultDto.getOrientacaoId());
    } else {
        try {
            orientacaoEntity = this.entityClass.newInstance();
        } catch (InstantiationException | IllegalAccessException e) {
            log.error("Erro ao instanciar entidade orientao", e.getMessage());
        }
    }

    if (!resultDto.isDiscenteCompleto()) {
        this.setFieldToEntity(orientacaoEntity, orientacao);

        PessoaDto pessoaDiscente = this.getPessoaDiscente(orientacao, processDto);
        DiscenteEntity discenteEntity = this.discenteService.save(pessoaDiscente,
            ↪ resultDto, null);
        this.updateIdentificadorDiscente(pessoaDiscente, resultDto);
        orientacaoEntity.setDiscente(discenteEntity);

        AreasDoConhecimentoXmlDto areasConhecimento =
            ↪ this.getAreasConhecimento(orientacao);
        if (areasConhecimento != null) {
            if (areasConhecimento.isSetareaDoConhecimento1()) {
                orientacaoEntity.setAreaConhecimentoUm(this.saveAreaConhecimento(areasConhecimen
            )
            if (areasConhecimento.isSetareaDoConhecimento2()) {
                orientacaoEntity.setAreaConhecimentoDois(this.saveAreaConhecimento(areasConheci
            )
            if (areasConhecimento.isSetareaDoConhecimento3()) {
                orientacaoEntity.setAreaConhecimentoTres(this.saveAreaConhecimento(areasConheci
            )
        }

        try {
            Long idInstituicao =
                ↪ this.instituicaoService.save(this.getIntituicaoSaveDTO(processDto,
                ↪ orientacao));
            orientacaoEntity.setInstituicao(this.em.getReference(InstituicaoEntity.class,
                ↪ idInstituicao));
        } catch (EmptyXmlException e) {
            log.error(e.getMessage());
            this.em.detach(orientacaoEntity);

```

```

    }
}

boolean orientador =
    ↪ TipoDeOrientacao.orientador.PRINCIPAL.equals(this.getTipoOrientacao(orientacao))
this.docenteService.save(new PessoaDto(processDto.getPessoaId()), resultDto,
    ↪ processDto, orientador);
if (orientador) {
    orientacaoEntity.setOrientador(this.em.getReference(DocenteEntity.class,
    ↪ processDto.getDocenteId()));
    orientacaoEntity.setOrientadorAtivo(true);
} else {
    orientacaoEntity.setCoorientador(this.em.getReference(DocenteEntity.class,
    ↪ processDto.getDocenteId()));
    orientacaoEntity.setCoorientadorAtivo(true);
}

orientacaoEntity.setDeletado(false);
orientacaoEntity.setPrograma(this.em.getReference(ProgramaEntity.class,
    ↪ processDto.getProgramaDto().getId()));
this.em.persist(orientacaoEntity);
}

private void updateIdentificadorDiscente(PessoaDto pessoaDiscente, OrientacaoResultDto
    ↪ resultDto) {
    if (resultDto.isSetDiscenteId() && pessoaDiscente.isSetNumeroIdentificador() &&
    ↪ !resultDto.isSetNumeroIdentificadorDiscente()) {
        QDiscenteEntity qDiscente = QDiscenteEntity.discenteEntity;
        Long pessoaId = new
            ↪ JPAQuery(this.em).from(qDiscente).where(qDiscente.id.eq(resultDto.getDiscenteId()));
        PessoaEntity pessoaDiscenteEntity =
            ↪ this.em.getReference(PessoaEntity.class, pessoaId);
        pessoaDiscenteEntity.setNumeroIdentificador(pessoaDiscente.getNumeroIdentificador());
        this.em.persist(pessoaDiscenteEntity);
    }
}

private AreaConhecimentoEntity saveAreaConhecimento(AreaDoConhecimentoXmlDto
    ↪ areaDoConhecimentoXmlDto) {
    try {
        return this.em.getReference(AreaConhecimentoEntity.class,
            ↪ this.areaService.save(new
            ↪ AreaConhecimentoSaveDTO(areaDoConhecimentoXmlDto));
    } catch (EmptyXmlException e) {
        log.info(e.getMessage());
        return null;
    }
}

public OrientacaoResultDto find(ImportProcessDto processDto, OrientacaoXml orientacao) {
    return this.orientacaoService.findOrientacao(this.getOrientacaoSearch(orientacao,
    ↪ processDto));
}

public abstract AreasDoConhecimentoXmlDto getAreasConhecimento(OrientacaoXml
    ↪ formacao);

public abstract InstituicaoSaveDTO getInstituicaoSaveDTO(ImportProcessDto processDto,
    ↪ OrientacaoXml orientacao) throws EmptyXmlException;

public abstract OrientacaoSearchDto getOrientacaoSearch(OrientacaoXml orientacao,
    ↪ ImportProcessDto processDto);

public abstract PessoaDto getPessoaDiscente(OrientacaoXml orientacao, ImportProcessDto
    ↪ processDto);

public abstract TipoDeOrientacao getTipoOrientacao(OrientacaoXml orientacao);

protected abstract void setFieldToEntity(OrientacaoEntity orientacaoEntity, OrientacaoXml
    ↪ orientacaoXml);
}

package alpc.ufsc.pessoa.orientacao.common.process;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import lombok.extern.slf4j.Slf4j;

```

```

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.jpa.impl.JPAQuery;

import alpc.ufsc.common.exception.EmptyXmlException;
import alpc.ufsc.domain.area.AreaConhecimentoService;
import alpc.ufsc.domain.area.dto.AreaConhecimentoSaveDTO;
import alpc.ufsc.domain.instituicao.InstituicaoService;
import alpc.ufsc.domain.instituicao.dto.InstituicaoSaveDTO;
import alpc.ufsc.entity.domain.area.AreaConhecimentoEntity;
import alpc.ufsc.entity.domain.instituicao.InstituicaoEntity;
import alpc.ufsc.entity.pessoa.PessoaEntity;
import alpc.ufsc.entity.pessoa.discente.DiscenteEntity;
import alpc.ufsc.entity.pessoa.docente.DocenteEntity;
import alpc.ufsc.entity.pessoa.docente.QDocenteEntity;
import alpc.ufsc.entity.pessoa.orientacao.BaseOrientacaoEntity;
import alpc.ufsc.entity.programa.ProgramaEntity;
import alpc.ufsc.pessoa.discente.DiscenteService;
import alpc.ufsc.pessoa.docente.DocenteService;
import alpc.ufsc.pessoa.dto.PessoaDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoResultDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoSearchDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoValidateDto;
import alpc.ufsc.pessoa.orientacao.common.service.OrientacaoSearchService;
import alpc.ufsc.programa.dto.ImportProcessDto;
import net.jodah.typeutils.TypeResolver;
import ufsc.alpc.xml.dto.CurriculoVitaeXmlDto;
import ufsc.alpc.xml.dto.common.area.conhecimento.AreaDoConhecimentoXmlDto;
import ufsc.alpc.xml.dto.common.area.conhecimento.AreasDoConhecimentoXmlDto;

@Sif4j
public abstract class BaseFormacaoProcessComponent<OrientacaoEntity extends
↳ BaseOrientacaoEntity, FormacaoType> {

    @PersistenceContext
    protected EntityManager em;

    @Autowired
    protected DiscenteService discenteService;

    @Autowired
    protected DocenteService docenteService;

    @Autowired
    protected OrientacaoSearchService orientacaoService;

    @Autowired
    protected InstituicaoService instituicaoService;

    @Autowired
    protected AreaConhecimentoService areaService;

    protected Class<OrientacaoEntity> entityClass;

    @SuppressWarnings("unchecked")
    public BaseFormacaoProcessComponent() {
        Class<?>[] resolveRawArguments =
↳ TypeResolver.resolveRawArguments(BaseFormacaoProcessComponent.class,
↳ this.getClass());
        this.entityClass = (Class<OrientacaoEntity>) resolveRawArguments[0];
    }

    @Transactional(propagation = Propagation.REQUIRES_NEW)
    public void processFormacao(ImportProcessDto processDto, FormacaoType formacaoXml) {
        OrientacaoResultDto resultDto = null;
        try {
            resultDto = this.find(processDto, formacaoXml);
        } catch (Exception e) {
            log.error("Erro ao buscar orientao", e.getMessage());
            return;
        }

        DiscenteEntity discenteEntity = this.discenteService.save(new
↳ PessoaDto(processDto.getPessoaId()), resultDto, processDto);

        PessoaDto pessoaDocenteOrientador = this.getPessoaOrientadorDocente(processDto,
↳ formacaoXml, resultDto);
        PessoaDto pessoaDocenteCoorientador =
↳ this.getPessoaCoorientadorDocente(processDto, formacaoXml, resultDto);

```

```

DocenteEntity orientador = this.saveDocente(resultDto, pessoaDocenteOrientador,
↪ true);
DocenteEntity coOrientador = null;
if (pessoaDocenteCoorientador.isSetNome()) {
    coOrientador = this.saveDocente(resultDto, pessoaDocenteCoorientador,
↪ false);
}

OrientacaoEntity orientacaoEntity = null;
if (resultDto.isSetOrientacaoId()) {
    orientacaoEntity = this.em.find(this.entityClass,
↪ resultDto.getOrientacaoId());
    this.setFieldsToEntity(orientacaoEntity, formacaoXml, processDto);
} else {
    try {
        orientacaoEntity = this.entityClass.newInstance();
    } catch (InstantiationException | IllegalAccessException e) {
        log.error("Erro ao instanciar entidade formao", e.getMessage());
    }
    this.setFieldsToEntity(orientacaoEntity, formacaoXml, processDto);
}
orientacaoEntity.setDeletado(false);
orientacaoEntity.setDiscenteCompleto(true);
orientacaoEntity.setOrientador(orientador);
orientacaoEntity.setCoorientador(coOrientador);
orientacaoEntity.setDiscente(discenteEntity);
orientacaoEntity.setPrograma(this.em.getReference(ProgramaEntity.class,
↪ processDto.getProgramaDto().getId()));
AreasDoConhecimentoXmlDto areasConhecimento =
↪ this.getAreasConhecimento(formacaoXml);
if (areasConhecimento != null) {
    if (areasConhecimento.isSetareaDoConhecimento1()) {
        orientacaoEntity.setAreaConhecimentoUm(this.saveAreaConhecimento(areasConhecimen
    )
    if (areasConhecimento.isSetareaDoConhecimento2()) {
        orientacaoEntity.setAreaConhecimentoUm(this.saveAreaConhecimento(areasConhecimen
    }
    if (areasConhecimento.isSetareaDoConhecimento3()) {
        orientacaoEntity.setAreaConhecimentoUm(this.saveAreaConhecimento(areasConhecimen
    }
}
try {
    orientacaoEntity.setInstituicao(this.em.getReference(InstituicaoEntity.class,
↪ this.instituicaoService.save(this.getInstituicaoSaveDTO(processDto,
↪ formacaoXml)));
    this.em.persist(orientacaoEntity);
} catch (EmptyXmlException e) {
    log.error(e.getMessage());
    this.em.detach(orientacaoEntity);
    return;
}
}

private PessoaDto getPessoaOrientadorDocente(ImportProcessDto processDto,
↪ FormacaoType formacaoXml, OrientacaoResultDto resultDto) {
    PessoaDto pessoaDocente = this.getPessoaDocente(formacaoXml, processDto, true);
    if (resultDto.isSetOrientadorId() && pessoaDocente.isSetNumeroIdentificador() &&
↪ !resultDto.isSetNumeroIdentificadorOrientador()) {
        this.updatePessoaDocente(resultDto.getOrientadorId(), pessoaDocente);
    }
    return pessoaDocente;
}

private PessoaDto getPessoaCoorientadorDocente(ImportProcessDto processDto,
↪ FormacaoType formacaoXml, OrientacaoResultDto resultDto) {
    PessoaDto pessoaDocente = this.getPessoaDocente(formacaoXml, processDto, false);
    if (resultDto.isSetCoorientadorId() && pessoaDocente.isSetNumeroIdentificador()
↪ && !resultDto.isSetNumeroIdentificadorCoorientador()) {
        this.updatePessoaDocente(resultDto.getCoorientadorId(), pessoaDocente);
    }
    return pessoaDocente;
}

private void updatePessoaDocente(Long docenteId, PessoaDto pessoaDocente) {
    QDocenteEntity qDocente = QDocenteEntity.docenteEntity;
    Long pessoaId = new
↪ JPAQuery(this.em).from(qDocente).where(qDocente.id.eq(docenteId)).uniqueResult(qDocente);

    PessoaEntity pessoaDocenteEntity = this.em.getReference(PessoaEntity.class,
↪ pessoaId);
    pessoaDocenteEntity.setNumeroIdentificador(pessoaDocente.getNumeroIdentificador());
    this.em.persist(pessoaDocenteEntity);
}

```

```

    }

    private AreaConhecimentoEntity saveAreaConhecimento(AreaDoConhecimentoXmlDto
        ↪ areaDoConhecimentoXmlDto) {
        try {
            return this.em.getReference(AreaConhecimentoEntity.class,
                ↪ this.areaService.save(new
                ↪ AreaConhecimentoSaveDTO(areaDoConhecimentoXmlDto)));
        } catch (EmptyXmlException e) {
            log.info(e.getMessage());
            return null;
        }
    }

    public OrientacaoResultDto find(ImportProcessDto processDto, FormacaoType
        ↪ formacaoXml) {
        return this.orientacaoService.findOrientacao(this.getOrientacaoSearch(formacaoXml,
            ↪ processDto));
    }

    protected DocenteEntity saveDocente(OrientacaoResultDto resultDto, PessoaDto
        ↪ pessoaDocente, boolean orientadorPrincipal) {
        return this.docenteService.save(pessoaDocente, resultDto, null, orientadorPrincipal);
    }

    public abstract void disableAllFormacoesDiscente(Long discenteId);

    public abstract AreasDoConhecimentoXmlDto getAreasConhecimento(FormacaoType
        ↪ formacao);

    public abstract InstituicaoSaveDTO getIntituicaoSaveDTO(ImportProcessDto processDto,
        ↪ FormacaoType formacao) throws EmptyXmlException;

    public abstract OrientacaoSearchDto getOrientacaoSearch(FormacaoType formacao,
        ↪ ImportProcessDto processDto);

    public abstract boolean hasFormacoes(CurriculoVitaeXmlDto curriculo);

    public abstract List<FormacaoType> getFormacoes(CurriculoVitaeXmlDto curriculo);

    protected abstract void setFieldsToEntity(OrientacaoEntity orientacaoEntity, FormacaoType
        ↪ formacaoXml, ImportProcessDto processDto);

    public abstract OrientacaoValidateDto getOrientacaoValidateDto(FormacaoType formacao);

    public abstract PessoaDto getPessoaDocente(FormacaoType formacaoXml,
        ↪ ImportProcessDto processDto, boolean orientadorPrincipal);
}

package alpc.ufsc.pessoa.orientacao.common.util;

import java.util.LinkedList;
import java.util.List;

import alpc.ufsc.domain.area.dto.AreaConhecimentoRowDTO;
import ufsc.alpc.xml.dto.domain.GrandeAreaDoConhecimento;

public interface AreasConhecimento {

    Long getGrandeAreaConhecimentoUm();

    String getNomeAreaConhecimentoUm();

    String getNomeSubAreaConhecimentoUm();

    String getNomeEspecialidadeUm();

    Long getGrandeAreaConhecimentoDois();

    String getNomeAreaConhecimentoDois();

    String getNomeSubAreaConhecimentoDois();

    String getNomeEspecialidadeDois();

    Long getGrandeAreaConhecimentoTres();

    String getNomeAreaConhecimentoTres();

    String getNomeSubAreaConhecimentoTres();
}

```

```

String getNomeEspecialidadeTres();

public static List<AreaConhecimentoRowDTO>
↳ extractAreasConhecimento(AreasConhecimento orientacao) {
    List<AreaConhecimentoRowDTO> areas = new LinkedList<>();

    AreaConhecimentoRowDTO area = new AreaConhecimentoRowDTO();
    area.setGrandeAreaConhecimento(GrandeAreaDoConhecimento.getById(orientacao.getGrandeAreaC
    area.setNomeAreaConhecimento(orientacao.getNomeAreaConhecimentoUm());
    area.setNomeSubAreaConhecimento(orientacao.getNomeSubAreaConhecimentoUm());
    area.setNomeEspecialidade(orientacao.getNomeEspecialidadeUm());
    if (!area.isEmpty()) {
        areas.add(area);
    }

    area = new AreaConhecimentoRowDTO();
    area.setGrandeAreaConhecimento(GrandeAreaDoConhecimento.getById(orientacao.getGrandeAreaC
    area.setNomeAreaConhecimento(orientacao.getNomeAreaConhecimentoDois());
    area.setNomeSubAreaConhecimento(orientacao.getNomeSubAreaConhecimentoDois());
    area.setNomeEspecialidade(orientacao.getNomeEspecialidadeDois());
    if (!area.isEmpty()) {
        areas.add(area);
    }

    area = new AreaConhecimentoRowDTO();
    area.setGrandeAreaConhecimento(GrandeAreaDoConhecimento.getById(orientacao.getGrandeAreaC
    area.setNomeAreaConhecimento(orientacao.getNomeAreaConhecimentoTres());
    area.setNomeSubAreaConhecimento(orientacao.getNomeSubAreaConhecimentoTres());
    area.setNomeEspecialidade(orientacao.getNomeEspecialidadeTres());
    if (!area.isEmpty()) {
        areas.add(area);
    }

    return areas;
}
}
package alpc.ufsc.pessoa.orientacao.common.query;

import static alpc.ufsc.querydsl.sql.QTbGraduacao.tbGraduacao;
import static alpc.ufsc.querydsl.sql.QTbOrientacaoDoutorado.tbOrientacaoDoutorado;
import static alpc.ufsc.querydsl.sql.QTbOrientacaoMestrado.tbOrientacaoMestrado;
import static alpc.ufsc.querydsl.sql.QTbOrientacaoPosDoutorado.tbOrientacaoPosDoutorado;
import static alpc.ufsc.util.querydsl.ExpressionsUtils.addAs;

import java.util.LinkedList;
import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;

import com.mysema.query.Tuple;
import com.mysema.query.sql.SQLQuery;
import com.mysema.query.sql.SQLQueryFactory;
import com.mysema.query.sql.SQLSubQuery;
import com.mysema.query.support.Expressions;
import com.mysema.query.types.NullExpression;
import com.mysema.query.types.path.SimplePath;
import com.mysema.query.types.query.ListSubQuery;

import alpc.ufsc.pessoa.orientacao.common.dto.FormacaoRowDTO;
import alpc.ufsc.pessoa.orientacao.common.dto.MFormacaoRowDTO;
import alpc.ufsc.querydsl.sql.QTbAreaConhecimento;
import alpc.ufsc.querydsl.sql.QTbDocente;
import alpc.ufsc.querydsl.sql.QTbInstituicao;
import alpc.ufsc.querydsl.sql.QTbPais;
import alpc.ufsc.querydsl.sql.QTbPessoa;
import alpc.ufsc.util.querydsl.Select;

@Component
public class FormacaoRowQuery {

    @PersistenceContext
    private EntityManager em;

    @Autowired
    private SQLQueryFactory queryFactory;

    public List<FormacaoRowDTO> getFormacoes(Long idDiscente, boolean graduacao, boolean
↳ posdoc) {

```

```

SQLQuery query = this.queryFactory.query();

List<ListSubQuery<Tuple>> subs = new LinkedList<>();
if (graduacao) {
    subs.add(this.getGraduacao(idDiscente));
}
subs.add(this.getMestrado(idDiscente));
subs.add(this.getDoutorado(idDiscente));
if (posdoc) {
    subs.add(this.getPosDoutorado(idDiscente));
}

query.from(this.queryFactory.subQuery().unionAll(subs).as("union_formacoes"));

SimplePath<Void> orientacoes = Expressions.path(Void.class, "union_formacoes");
MFormacaoRowDTO meta = MFormacaoRowDTO.meta;

Select<FormacaoRowDTO> select = new Select<>(FormacaoRowDTO.class);
addAs(select, meta.nomeFormacao, orientacoes);
addAs(select, meta.idFormacao, orientacoes);

addAs(select, meta.tituloTrabalho, orientacoes);

addAs(select, meta.nomeOrientador, orientacoes);
addAs(select, meta.nomeCoorientador, orientacoes);
addAs(select, meta.nomeInstituicao, orientacoes);
addAs(select, meta.nomePais, orientacoes);
addAs(select, meta.nomeDocenteSanduiche, orientacoes);

addAs(select, meta.anoFim, orientacoes);
addAs(select, meta.anoInicio, orientacoes);

addAs(select, meta.statusFormacao, orientacoes);

addAs(select, meta.grandeAreaConhecimentoUm, orientacoes);
addAs(select, meta.nomeAreaConhecimentoUm, orientacoes);
addAs(select, meta.nomeSubAreaConhecimentoUm, orientacoes);
addAs(select, meta.nomeEspecialidadeUm, orientacoes);

addAs(select, meta.grandeAreaConhecimentoDois, orientacoes);
addAs(select, meta.nomeAreaConhecimentoDois, orientacoes);
addAs(select, meta.nomeSubAreaConhecimentoDois, orientacoes);
addAs(select, meta.nomeEspecialidadeDois, orientacoes);

addAs(select, meta.grandeAreaConhecimentoTres, orientacoes);
addAs(select, meta.nomeAreaConhecimentoTres, orientacoes);
addAs(select, meta.nomeSubAreaConhecimentoTres, orientacoes);
addAs(select, meta.nomeEspecialidadeTres, orientacoes);

return query.list(select);
}

private ListSubQuery<Tuple> getGraduacao(Long idDiscente) {
    QTbInstituicao qInstituicao = new QTbInstituicao("ginstituicao");
    QTbPais qPais = new QTbPais("mpais");

    SQLSubQuery query = this.queryFactory.subQuery().from(tbGraduacao);
    query.join(qInstituicao).on(tbGraduacao.idInstituicao.eq(qInstituicao.idInstituicao));
    query.join(qPais).on(qInstituicao.idPais.eq(qPais.idPais));

    query.where(tbGraduacao.deletado.eq(false));
    query.where(tbGraduacao.idDiscente.eq(idDiscente));
    query.orderBy(tbGraduacao.titulo.asc());

    MFormacaoRowDTO meta = MFormacaoRowDTO.meta;
    return query.list(
        Expressions.constantAs("Graduao", Expressions.path(String.class,
            ↪ meta.nomeFormacao.getAlias()),
            tbGraduacao.idGraduacao.as(meta.idFormacao.getAlias()),

            tbGraduacao.titulo.as(meta.tituloTrabalho.getAlias()),

            Expressions.as(NullExpression.DEFAULT,
            ↪ meta.nomeOrientador.getAlias()),
            Expressions.as(NullExpression.DEFAULT,
            ↪ meta.nomeCoorientador.getAlias()),
            qInstituicao.nome.as(meta.nomeInstituicao.getAlias()),
            qPais.nome.as(meta.nomePais.getAlias()),
            Expressions.as(NullExpression.DEFAULT,
            ↪ meta.nomeDocenteSanduiche.getAlias()),

            tbGraduacao.anoFim.as(meta.anoFim.getAlias()),
    );
}

```



```

tbGraduacao.anoInicio.as(meta.anoInicio.getAlias()),
Expressions.as(NullExpression.DEFAULT,
    ↪ meta.statusFormacao.getAlias()),
Expressions.as(NullExpression.DEFAULT,
    ↪ meta.grandeAreaConhecimentoUm.getAlias()),
Expressions.as(NullExpression.DEFAULT,
    ↪ meta.nomeAreaConhecimentoUm.getAlias()),
Expressions.as(NullExpression.DEFAULT,
    ↪ meta.nomeSubAreaConhecimentoUm.getAlias()),
Expressions.as(NullExpression.DEFAULT,
    ↪ meta.nomeEspecialidadeUm.getAlias()),
Expressions.as(NullExpression.DEFAULT,
    ↪ meta.grandeAreaConhecimentoDois.getAlias()),
Expressions.as(NullExpression.DEFAULT,
    ↪ meta.nomeAreaConhecimentoDois.getAlias()),
Expressions.as(NullExpression.DEFAULT,
    ↪ meta.nomeSubAreaConhecimentoDois.getAlias()),
Expressions.as(NullExpression.DEFAULT,
    ↪ meta.nomeEspecialidadeDois.getAlias()),
Expressions.as(NullExpression.DEFAULT,
    ↪ meta.grandeAreaConhecimentoTres.getAlias()),
Expressions.as(NullExpression.DEFAULT,
    ↪ meta.nomeAreaConhecimentoTres.getAlias()),
Expressions.as(NullExpression.DEFAULT,
    ↪ meta.nomeSubAreaConhecimentoTres.getAlias()),
Expressions.as(NullExpression.DEFAULT,
    ↪ meta.nomeEspecialidadeTres.getAlias());
}

private ListSubQuery<Tuple> getMestrado(Long idDiscente) {
    QTbPessoa qPessoaOrientador = new QTbPessoa("mpessoa_orientador");
    QTbDocente qOrientador = new QTbDocente("morientador");
    QTbPessoa qPessoaCoorientador = new QTbPessoa("mpessoa_coorientador");
    QTbDocente qCoorientador = new QTbDocente("mcoorientador");

    QTbInstituicao qInstituicao = new QTbInstituicao("minstituicao");
    QTbPais qPais = new QTbPais("mpais");
    QTbAreaConhecimento qAreaUm = new QTbAreaConhecimento("mareaum");
    QTbAreaConhecimento qAreaDois = new QTbAreaConhecimento("mareadois");
    QTbAreaConhecimento qAreaTres = new QTbAreaConhecimento("mareatres");

    SQLSubQuery query = this.queryFactory.subQuery().from(tbOrientacaoMestrado);

    query.leftJoin(qOrientador).on(tbOrientacaoMestrado.idOrientador.eq(qOrientador.idDocente));
    query.leftJoin(qPessoaOrientador).on(qPessoaOrientador.idPessoa.eq(qOrientador.idPessoa));

    query.leftJoin(qCoorientador).on(tbOrientacaoMestrado.idCoorientador.eq(qCoorientador.idDocente));
    query.leftJoin(qPessoaCoorientador).on(qPessoaCoorientador.idPessoa.eq(qCoorientador.idPessoa));

    query.join(qInstituicao).on(tbOrientacaoMestrado.idInstituicao.eq(qInstituicao.idInstituicao));
    query.join(qPais).on(qInstituicao.idPais.eq(qPais.idPais));
    query.leftJoin(qAreaUm).on(tbOrientacaoMestrado.idAreaConhecimentoUm.eq(qAreaUm.idAreaCon));
    query.leftJoin(qAreaDois).on(tbOrientacaoMestrado.idAreaConhecimentoDois.eq(qAreaDois.idAreaCon));
    query.leftJoin(qAreaTres).on(tbOrientacaoMestrado.idAreaConhecimentoTres.eq(qAreaTres.idAreaCon));

    query.where(tbOrientacaoMestrado.deletado.eq(false));
    query.where(tbOrientacaoMestrado.idDiscente.eq(idDiscente));
    query.orderBy(tbOrientacaoMestrado.tituloDoTrabalho.asc());

    MFormacaoRowDTO meta = MFormacaoRowDTO.meta;
    return query.list(
        Expressions.constantAs("Mestrado", Expressions.path(String.class,
            ↪ meta.nomeFormacao.getAlias())),
        tbOrientacaoMestrado.idOrientacaoMestrado.as(meta.idFormacao.getAlias()),
        tbOrientacaoMestrado.tituloDoTrabalho.as(meta.tituloTrabalho.getAlias()),
        qPessoaOrientador.nome.as(meta.nomeOrientador.getAlias()),
        qPessoaCoorientador.nome.as(meta.nomeCoorientador.getAlias()),
        qInstituicao.nome.as(meta.nomeInstituicao.getAlias()),
        qPais.nome.as(meta.nomePais.getAlias()),
        Expressions.as(NullExpression.DEFAULT,
            ↪ meta.nomeDocenteSanduiche.getAlias()),
        tbOrientacaoMestrado.anoFim.as(meta.anoFim.getAlias()),
        tbOrientacaoMestrado.anoInicio.as(meta.anoInicio.getAlias()),
        tbOrientacaoMestrado.statusOrientacao.as(meta.statusFormacao.getAlias()),

```

```

        qAreaUm.idGrandeAreaConhecimento.as(meta.grandeAreaConhecimentoUm.getAlias()),
        qAreaUm.nomeAreaConhecimento.as(meta.nomeAreaConhecimentoUm.getAlias()),
        qAreaUm.nomeSubAreaConhecimento.as(meta.nomeSubAreaConhecimentoUm.getAlias()),
        qAreaUm.nomeEspecialidade.as(meta.nomeEspecialidadeUm.getAlias()),

        qAreaDois.idGrandeAreaConhecimento.as(meta.grandeAreaConhecimentoDois.getAlias()),
        qAreaDois.nomeAreaConhecimento.as(meta.nomeAreaConhecimentoDois.getAlias()),
        qAreaDois.nomeSubAreaConhecimento.as(meta.nomeSubAreaConhecimentoDois.getAlias()),
        qAreaDois.nomeEspecialidade.as(meta.nomeEspecialidadeDois.getAlias()),

        qAreaTres.idGrandeAreaConhecimento.as(meta.grandeAreaConhecimentoTres.getAlias()),
        qAreaTres.nomeAreaConhecimento.as(meta.nomeAreaConhecimentoTres.getAlias()),
        qAreaTres.nomeSubAreaConhecimento.as(meta.nomeSubAreaConhecimentoTres.getAlias()),
        qAreaTres.nomeEspecialidade.as(meta.nomeEspecialidadeTres.getAlias());
    }

    private ListSubQuery<Tuple> getDoutorado(Long idDiscente) {
        QTbPessoa qPessoaOrientador = new QTbPessoa("dpessoa_orientador");
        QTbDocente qOrientador = new QTbDocente("dorientador");
        QTbPessoa qPessoaCoorientador = new QTbPessoa("dpessoa_coorientador");
        QTbDocente qCoorientador = new QTbDocente("dcoorientador");

        QTbInstituicao qInstituicao = new QTbInstituicao("dinstituicao");
        QTbPais qPais = new QTbPais("dpais");
        QTbAreaConhecimento qAreaUm = new QTbAreaConhecimento("dareaum");
        QTbAreaConhecimento qAreaDois = new QTbAreaConhecimento("dareadois");
        QTbAreaConhecimento qAreaTres = new QTbAreaConhecimento("dareatres");

        SQLSubQuery query = this.queryFactory.subQuery().from(tbOrientacaoDoutorado);

        query.leftJoin(qOrientador).on(tbOrientacaoDoutorado.idOrientador.eq(qOrientador.idDocente));
        query.leftJoin(qPessoaOrientador).on(qPessoaOrientador.idPessoa.eq(qOrientador.idPessoa));

        query.leftJoin(qCoorientador).on(tbOrientacaoDoutorado.idCoorientador.eq(qCoorientador.idDocente));
        query.leftJoin(qPessoaCoorientador).on(qPessoaCoorientador.idPessoa.eq(qCoorientador.idPessoa));

        query.join(qInstituicao).on(tbOrientacaoDoutorado.idInstituicao.eq(qInstituicao.idInstituicao));
        query.join(qPais).on(qInstituicao.idPais.eq(qPais.idPais));
        query.leftJoin(qAreaUm).on(tbOrientacaoDoutorado.idAreaConhecimentoUm.eq(qAreaUm.idAreaConhecimentoUm));
        query.leftJoin(qAreaDois).on(tbOrientacaoDoutorado.idAreaConhecimentoDois.eq(qAreaDois.idAreaConhecimentoDois));
        query.leftJoin(qAreaTres).on(tbOrientacaoDoutorado.idAreaConhecimentoTres.eq(qAreaTres.idAreaConhecimentoTres));

        query.where(tbOrientacaoDoutorado.deletado.eq(false));
        query.where(tbOrientacaoDoutorado.idDiscente.eq(idDiscente));
        query.orderBy(tbOrientacaoDoutorado.tituloDoTrabalho.asc());

        MFormacaoRowDTO meta = MFormacaoRowDTO.meta;
        return query.list(
            Expressions.constantAs("Doutorado", Expressions.path(String.class,
                → meta.nomeFormacao.getAlias()),
            tbOrientacaoDoutorado.idOrientacaoDoutorado.as(meta.idFormacao.getAlias()),
            tbOrientacaoDoutorado.tituloDoTrabalho.as(meta.tituloTrabalho.getAlias()),
            qPessoaOrientador.nome.as(meta.nomeOrientador.getAlias()),
            qPessoaCoorientador.nome.as(meta.nomeCoorientador.getAlias()),
            qInstituicao.nome.as(meta.nomeInstituicao.getAlias()),
            qPais.nome.as(meta.nomePais.getAlias()),
            tbOrientacaoDoutorado.nomeOrientadorSanduiche.as(meta.nomeDocenteSanduiche.getAlias()),
            tbOrientacaoDoutorado.anoFim.as(meta.anoFim.getAlias()),
            tbOrientacaoDoutorado.anoInicio.as(meta.anoInicio.getAlias()),
            tbOrientacaoDoutorado.statusOrientacao.as(meta.statusFormacao.getAlias()),
            qAreaUm.idGrandeAreaConhecimento.as(meta.grandeAreaConhecimentoUm.getAlias()),
            qAreaUm.nomeAreaConhecimento.as(meta.nomeAreaConhecimentoUm.getAlias()),
            qAreaUm.nomeSubAreaConhecimento.as(meta.nomeSubAreaConhecimentoUm.getAlias()),
            qAreaUm.nomeEspecialidade.as(meta.nomeEspecialidadeUm.getAlias()),
            qAreaDois.idGrandeAreaConhecimento.as(meta.grandeAreaConhecimentoDois.getAlias()),
            qAreaDois.nomeAreaConhecimento.as(meta.nomeAreaConhecimentoDois.getAlias()),
            qAreaDois.nomeSubAreaConhecimento.as(meta.nomeSubAreaConhecimentoDois.getAlias()),
            qAreaDois.nomeEspecialidade.as(meta.nomeEspecialidadeDois.getAlias()),
            qAreaTres.idGrandeAreaConhecimento.as(meta.grandeAreaConhecimentoTres.getAlias()),
            qAreaTres.nomeAreaConhecimento.as(meta.nomeAreaConhecimentoTres.getAlias()),
            qAreaTres.nomeSubAreaConhecimento.as(meta.nomeSubAreaConhecimentoTres.getAlias()),
            qAreaTres.nomeEspecialidade.as(meta.nomeEspecialidadeTres.getAlias());
    }
}

```

```

private ListSubQuery<Tuple> getPosDoutorado(Long idDiscente) {
    QTbInstituicao qInstituicao = new QTbInstituicao("pdinstituicao");
    QTbPais qPais = new QTbPais("pdpais");
    QTbAreaConhecimento qAreaUm = new QTbAreaConhecimento("pdareaum");
    QTbAreaConhecimento qAreaDois = new QTbAreaConhecimento("pdareadois");
    QTbAreaConhecimento qAreaTres = new QTbAreaConhecimento("pdareatres");

    SQLSubQuery query =
        ↳ this.queryFactory.subQuery().from(tbOrientacaoPosDoutorado);
    query.join(qInstituicao).on(tbOrientacaoPosDoutorado.idInstituicao.eq(qInstituicao.idInstituicao));
    query.join(qPais).on(qInstituicao.idPais.eq(qPais.idPais));
    query.leftJoin(qAreaUm).on(tbOrientacaoPosDoutorado.idAreaConhecimentoUm.eq(qAreaUm.idAreaUm));
    query.leftJoin(qAreaDois).on(tbOrientacaoPosDoutorado.idAreaConhecimentoDois.eq(qAreaDois.idAreaDois));
    query.leftJoin(qAreaTres).on(tbOrientacaoPosDoutorado.idAreaConhecimentoTres.eq(qAreaTres.idAreaTres));

    query.where(tbOrientacaoPosDoutorado.deletedo.eq(false));
    query.where(tbOrientacaoPosDoutorado.idDiscente.eq(idDiscente));
    query.orderBy(tbOrientacaoPosDoutorado.tituloDoTrabalho.asc());

    MFormacaoRowDTO meta = MFormacaoRowDTO.meta;
    return query.list(
        Expressions.constantAs("Ps-Doutorado",
            ↳ Expressions.path(String.class,
                ↳ meta.nomeFormacao.getAlias()),
            tbOrientacaoPosDoutorado.idOrientacaoPosDoutorado.as(meta.idFormacao.getAlias()),
            tbOrientacaoPosDoutorado.tituloDoTrabalho.as(meta.tituloTrabalho.getAlias()),

            Expressions.as(NullExpression.DEFAULT,
                ↳ meta.nomeOrientador.getAlias()),
            Expressions.as(NullExpression.DEFAULT,
                ↳ meta.nomeCoorientador.getAlias()),
            qInstituicao.nome.as(meta.nomeInstituicao.getAlias()),
            qPais.nome.as(meta.nomePais.getAlias()),
            Expressions.as(NullExpression.DEFAULT,
                ↳ meta.nomeDocenteSanduiche.getAlias()),

            tbOrientacaoPosDoutorado.anoFim.as(meta.anoFim.getAlias()),
            tbOrientacaoPosDoutorado.anoInicio.as(meta.anoInicio.getAlias()),

            tbOrientacaoPosDoutorado.statusOrientacao.as(meta.statusFormacao.getAlias()),

            qAreaUm.idGrandeAreaConhecimento.as(meta.grandeAreaConhecimentoUm.getAlias()),
            qAreaUm.nomeAreaConhecimento.as(meta.nomeAreaConhecimentoUm.getAlias()),
            qAreaUm.nomeSubAreaConhecimento.as(meta.nomeSubAreaConhecimentoUm.getAlias()),
            qAreaUm.nomeEspecialidade.as(meta.nomeEspecialidadeUm.getAlias()),

            qAreaDois.idGrandeAreaConhecimento.as(meta.grandeAreaConhecimentoDois.getAlias()),
            qAreaDois.nomeAreaConhecimento.as(meta.nomeAreaConhecimentoDois.getAlias()),
            qAreaDois.nomeSubAreaConhecimento.as(meta.nomeSubAreaConhecimentoDois.getAlias()),
            qAreaDois.nomeEspecialidade.as(meta.nomeEspecialidadeDois.getAlias()),

            qAreaTres.idGrandeAreaConhecimento.as(meta.grandeAreaConhecimentoTres.getAlias()),
            qAreaTres.nomeAreaConhecimento.as(meta.nomeAreaConhecimentoTres.getAlias()),
            qAreaTres.nomeSubAreaConhecimento.as(meta.nomeSubAreaConhecimentoTres.getAlias()),
            qAreaTres.nomeEspecialidade.as(meta.nomeEspecialidadeTres.getAlias()));
    }
}

package alpc.ufsc.pessoa.orientacao.common.dto;

import org.apache.commons.lang3.StringUtils;

import lombok.Getter;
import lombok.Setter;

import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoConcluidaXmlDto;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoXmlDto;
import ↳ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.doutorado.OrientacaoEmAndamentoDeDoutorado;
import ↳ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.mestrado.OrientacaoEmAndamentoDeMestrado;
import ufsc.alpc.xml.dto.dados.gerais.formacao.CursoXmlDto;
import ↳ ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.doutorado.OrientacaoConcluidaDoutoradoXmlDto;
import ↳ ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.mestrado.OrientacaoConcluidaMestradoXmlDto;

@Getter
@Setter
public class OrientacaoValidateDto {

    private String nomeCurso;

```

```

private String nomeInstituicao;

private String codigoInstituicao;

public boolean isSetCodigoInstituicao() {
    return !StringUtils.isBlank(codigoInstituicao);
}

public OrientacaoValidateDto(OrientacaoConcluidaMestradoXmlDto
    ↪ orientacaoMestradoXmlDto) {
    DetalhamentoDaOrientacaoConcluidaXmlDto detalhamento =
        ↪ orientacaoMestradoXmlDto.getDetalhamento();
    this.nomeCurso = detalhamento.getNomeCurso();
    this.nomeInstituicao = detalhamento.getNomeInstituicao();
    this.codigoInstituicao = detalhamento.getCodigoInstituicao();
}

public OrientacaoValidateDto(OrientacaoEmAndamentoDeMestradoXmlDto
    ↪ mestradoXmlDto) {
    DetalhamentoDaOrientacaoXmlDto detalhamento =
        ↪ mestradoXmlDto.getDetalhamento();
    this.nomeCurso = detalhamento.getNomeCurso();
    this.nomeInstituicao = detalhamento.getNomeInstituicao();
    this.codigoInstituicao = detalhamento.getCodigoInstituicao();
}

public OrientacaoValidateDto(OrientacaoConcluidaDoutoradoXmlDto
    ↪ orientacaoDoutoradoXmlDto) {
    DetalhamentoDaOrientacaoConcluidaXmlDto detalhamento =
        ↪ orientacaoDoutoradoXmlDto.getDetalhamento();
    this.nomeCurso = detalhamento.getNomeCurso();
    this.nomeInstituicao = detalhamento.getNomeInstituicao();
    this.codigoInstituicao = detalhamento.getCodigoInstituicao();
}

public OrientacaoValidateDto(OrientacaoEmAndamentoDeDoutoradoXmlDto
    ↪ doutoradoXmlDto) {
    DetalhamentoDaOrientacaoXmlDto detalhamento =
        ↪ doutoradoXmlDto.getDetalhamento();
    this.nomeCurso = detalhamento.getNomeCurso();
    this.nomeInstituicao = detalhamento.getNomeInstituicao();
    this.codigoInstituicao = detalhamento.getCodigoInstituicao();
}

public OrientacaoValidateDto(CursoXmlDto formacaoDto) {
    this.nomeCurso = formacaoDto.getNomeCurso();
    this.nomeInstituicao = formacaoDto.getNomeInstituicao();
    this.codigoInstituicao = formacaoDto.getCodigoInstituicao();
}
}

package alpc.ufsc.pessoa.orientacao.common.dto;

import lombok.Getter;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;
import org.apache.lucene.search.spell.NGramDistance;

import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.programa.dto.ImportProcessDto;
import info.debatty.java.stringsimilarity.JaroWinkler;
import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoFlagRelevanciaXmlDto;
import ufsc.alpc.xml.dto.common.DadosBasicosDaParticipacaoTipoXmlDto;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoConcluidaXmlDto;
import ufsc.alpc.xml.dto.common.DetalhamentoDaOrientacaoXmlDto;
import ↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.DadosBasicosDaOrientacaoEmAndamentoXmlDto;
import ↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.doutorado.OrientacaoEmAndamentoDeDoutoradoX
import ↪ ufsc.alpc.xml.dto.dados.complementares.orientacoes.andamento.mestrado.OrientacaoEmAndamentoDeMestradoX
import ufsc.alpc.xml.dto.dados.gerais.DadosGeraisXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.doutorado.DoutoradoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.graduacao.GraduacaoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.mestrado.MestradoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.posdoutorado.PosDoutoradoXmlDto;
import ufsc.alpc.xml.dto.domain.TipoDeOrientacao;
import ↪ ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.doutorado.OrientacaoConcluidaDoutoradoXmlDto;
import

```

```

↪ ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.mestrado.OrientacaoConcluidaMestradoXmlDto;

@Getter
@Setter
public class OrientacaoSearchDto {

    private String tituloDoTrabalho;
    private TipoDoTrabalho tipoDoTrabalho;
    private Integer anoInicio;
    private Integer anoFim;
    private String nomeOrientador;
    private String numeroIdentificadorOrientador;

    private String nomeCoorientador;
    private String numeroIdentificadorCoorientador;

    private String nomeDiscente;
    private String numeroIdentificadorDiscente;

    private Long programaId;
    private String instituicao;

    public boolean isSetNumeroIdentificadorDiscente() {
        return !StringUtils.isBlank(this.numeroIdentificadorDiscente);
    }

    public boolean isSetNumeroIdentificadorOrientador() {
        return !StringUtils.isBlank(this.numeroIdentificadorOrientador);
    }

    public boolean isSetNumeroIdentificadorCoorientador() {
        return !StringUtils.isBlank(this.numeroIdentificadorCoorientador);
    }

    public boolean isSetNameOrientador() {
        return !StringUtils.isBlank(this.nomeOrientador);
    }

    public boolean isSetNameCoorientador() {
        return !StringUtils.isBlank(this.nomeCoorientador);
    }

    public String getNomeDiscente() {
        return QueryUtils.lowerCaseStripAccents(this.nomeDiscente);
    }

    public String getNomeOrientador() {
        return QueryUtils.lowerCaseStripAccents(this.nomeOrientador);
    }

    public String getNomeCoorientador() {
        return QueryUtils.lowerCaseStripAccents(this.nomeCoorientador);
    }

    public String getInstituicao() {
        return QueryUtils.lowerCaseStripAccents(this.instituicao);
    }

    public String getTituloDoTrabalho() {
        return QueryUtils.lowerCaseStripAccents(this.tituloDoTrabalho);
    }

    public boolean similar(OrientacaoSearchDto other) {
        JaroWinkler jaroWinkler = new JaroWinkler();
        NGramDistance nGram = new NGramDistance(4);

        if (this.isSetNumeroIdentificadorOrientador() &&
            ↪ other.isSetNumeroIdentificadorOrientador()) {
            if
                ↪ (!this.numeroIdentificadorOrientador.equals(other.numeroIdentificadorOrientador))
                ↪ {
                    return false;
                }
        } else {
            if (this.isSetNameOrientador() && other.isSetNameOrientador()) {
                if (jaroWinkler.similarity(this.getNomeOrientador(),
                    ↪ other.getNomeOrientador()) < 0.98) {

```

```

        }
        }
        return false;
    }
}

if (this.isSetNumeroIdentificadorCoorientador() &&
    ↪ other.isSetNumeroIdentificadorCoorientador()) {
    if
        ↪ (!this.numeroIdentificadorCoorientador.equals(other.numeroIdentificadorCoorientador))
        ↪ {
            return false;
        }
    } else {
        if (this.isSetNomeCoorientador() && other.isSetNomeCoorientador()) {
            if (jaroWinkler.similarity(this.getNomeCoorientador(),
                ↪ other.getNomeCoorientador()) < 0.98) {
                return false;
            }
        }
    }
}

if (this.isSetNumeroIdentificadorDiscente() &&
    ↪ other.isSetNumeroIdentificadorDiscente()) {
    if
        ↪ (!this.numeroIdentificadorDiscente.equals(other.numeroIdentificadorDiscente))
        ↪ {
            return false;
        }
    } else {
        if (jaroWinkler.similarity(this.getNomeDiscente(), other.getNomeDiscente())
            ↪ < 0.98) {
            return false;
        }
    }
}

if (nGram.getDistance(this.getTituloDoTrabalho(), other.getTituloDoTrabalho()) <
    ↪ 0.98) {
    return false;
}

return true;
}

public OrientacaoSearchDto(Object[] fields) {
    this.tituloDoTrabalho = (String) fields[0];
    this.nomeOrientador = (String) fields[1];
    this.numeroIdentificadorOrientador = (String) fields[2];
    this.nomeDiscente = (String) fields[3];
    this.numeroIdentificadorDiscente = (String) fields[4];
    this.nomeDiscente = (String) fields[5];
    this.numeroIdentificadorDiscente = (String) fields[6];
}

public OrientacaoSearchDto(String titulo, String discente) {
    this.tituloDoTrabalho = titulo;
    this.nomeDiscente = discente;
}

public OrientacaoSearchDto(ImportProcessDto processDto,
    ↪ OrientacaoEmAndamentoDeMestradoXmlDto mestradoXmlDto) {
    ↪ DadosBasicosDaOrientacaoEmAndamentoXmlDto dadosBasicos =
        ↪ mestradoXmlDto.getDadosBasicos();
    ↪ DetalhamentoDaOrientacaoXmlDto detalhamento =
        ↪ mestradoXmlDto.getDetalhamento();

    this.tipoDoTrabalho = TipoDoTrabalho.MESTRADO;
    this.numeroIdentificadorDiscente = detalhamento.getNumeroIDOrientado();
    this.setFieldsOrientacaoEmAndamento(processDto, dadosBasicos, detalhamento);
}

public OrientacaoSearchDto(ImportProcessDto processDto,
    ↪ OrientacaoConcluidaMestradoXmlDto orientacaoMestradoXmlDto) {
    ↪ DadosBasicosDaParticipacaoTipoXmlDto dadosBasicos =
        ↪ orientacaoMestradoXmlDto.getDadosBasicos();
    ↪ DetalhamentoDaOrientacaoConcluidaXmlDto detalhamento =
        ↪ orientacaoMestradoXmlDto.getDetalhamento();
    ↪ DadosGeraisXmlDto dadosGerais = processDto.getCurriculo().getDadosGerais();

    this.tipoDoTrabalho = TipoDoTrabalho.MESTRADO;
    this.tituloDoTrabalho = dadosBasicos.getTitulo();
    this.anoInicio = dadosBasicos.getAno();
    this.instituicao = detalhamento.getNomeInstituicao();
}

```

```

        if
            ↪ (detalhamento.getTipoOrientacao().equals(TipoDeOrientacao.ORIENTADOR_PRINCIPAL))
            ↪ {
                this.nomeOrientador = dadosGerais.getNomeCompleto();
                this.numeroIdentificadorOrientador = dadosGerais.getNumeroIdentidade();
            } else {
                this.nomeCoorientador = dadosGerais.getNomeCompleto();
                this.numeroIdentificadorCoorientador = dadosGerais.getNumeroIdentidade();
            }
        this.programaId = processDto.getProgramaDto().getId();
        this.nomeDiscente = detalhamento.getNomeOrientando();
        this.numeroIdentificadorDiscente = detalhamento.getNumeroIDOrientado();
    }

    public OrientacaoSearchDto(ImportProcessDto processDto,
        ↪ OrientacaoEmAndamentoDeDoutoradoXmlDto orientacaoDoutoradoXmlDto) {
        DadosBasicosDaOrientacaoEmAndamentoXmlDto dadosBasicos =
            ↪ orientacaoDoutoradoXmlDto.getDadosBasicos();
        DetalhamentoDaOrientacaoXmlDto detalhamento =
            ↪ orientacaoDoutoradoXmlDto.getDetalhamento();

        this.tipoDoTrabalho = TipoDoTrabalho.DOCTORADO;
        this.numeroIdentificadorDiscente = detalhamento.getNumeroIDOrientando();
        this.setFieldsOrientacaoEmAndamento(processDto, dadosBasicos, detalhamento);
    }

    public OrientacaoSearchDto(ImportProcessDto processDto,
        ↪ OrientacaoConcluidaDoutoradoXmlDto orientacaoDoutoradoXmlDto) {
        DadosBasicosDaParticipacaoFlagRelevanciaXmlDto dadosBasicos =
            ↪ orientacaoDoutoradoXmlDto.getDadosBasicos();
        DetalhamentoDaOrientacaoConcluidaXmlDto detalhamento =
            ↪ orientacaoDoutoradoXmlDto.getDetalhamento();

        this.tipoDoTrabalho = TipoDoTrabalho.DOCTORADO;
        this.tituloDoTrabalho = dadosBasicos.getTitulo();
        this.anoInicio = dadosBasicos.getAno();
        this.instituicao = detalhamento.getNomeInstituicao();
        this.nomeOrientador =
            ↪ processDto.getCurriculo().getDadosGerais().getNomeCompleto();
        this.numeroIdentificadorOrientador =
            ↪ processDto.getCurriculo().getDadosGerais().getNumeroIdentidade();
        this.programaId = processDto.getProgramaDto().getId();
        this.nomeDiscente = detalhamento.getNomeOrientando();
        this.numeroIdentificadorDiscente = detalhamento.getNumeroIDOrientado();
    }

    private void setFieldsOrientacaoEmAndamento(ImportProcessDto processDto,
        ↪ DadosBasicosDaOrientacaoEmAndamentoXmlDto dadosBasicos,
        ↪ DetalhamentoDaOrientacaoXmlDto detalhamento) {
        DadosGeraisXmlDto dadosGerais = processDto.getCurriculo().getDadosGerais();
        this.tituloDoTrabalho = dadosBasicos.getTituloDoTrabalho();
        this.anoInicio = dadosBasicos.getAno();
        this.instituicao = detalhamento.getNomeInstituicao();
        if
            ↪ (detalhamento.getTipoOrientacao().equals(TipoDeOrientacao.ORIENTADOR_PRINCIPAL))
            ↪ {
                this.nomeOrientador = dadosGerais.getNomeCompleto();
                this.numeroIdentificadorOrientador = dadosGerais.getNumeroIdentidade();
            } else {
                this.nomeCoorientador = dadosGerais.getNomeCompleto();
                this.numeroIdentificadorCoorientador = dadosGerais.getNumeroIdentidade();
            }
        this.programaId = processDto.getProgramaDto().getId();
        this.nomeDiscente = detalhamento.getNomeOrientando();
    }

    public OrientacaoSearchDto(ImportProcessDto processDto, GraduacaoXmlDto
        ↪ graduacaoXmlDto) {
        this.tipoDoTrabalho = graduacaoXmlDto.getTituloTcc();
        this.anoInicio = graduacaoXmlDto.getAnoInicio();
        this.anoFim = graduacaoXmlDto.getAnoConclusao();
        this.instituicao = graduacaoXmlDto.getNomeInstituicao();
        this.programaId = processDto.getProgramaDto().getId();
        this.nomeDiscente =
            ↪ processDto.getCurriculo().getDadosGerais().getNomeCompleto();
    }

    public OrientacaoSearchDto(ImportProcessDto processDto, MestradoXmlDto
        ↪ mestradoXmlDto) {
        this.tipoDoTrabalho = TipoDoTrabalho.MESTRADO;
        this.tituloDoTrabalho = mestradoXmlDto.getTituloDissertacaoTese();
        this.anoInicio = mestradoXmlDto.getAnoInicio();
    }

```

```

        this.anoFim = mestradoXmlDto.getAnoConclusao();
        this.instituicao = mestradoXmlDto.getNomeInstituicao();
        this.nomeOrientador = mestradoXmlDto.getNomeCompletoOrientador();
        this.numeroIdentificadorOrientador = mestradoXmlDto.getNumeroIdOrientador();
        this.nomeCoorientador = mestradoXmlDto.getNomeCoOrientador();
        this.programaId = processDto.getProgramaDto().getId();
        this.nomeDiscente =
            ↪ processDto.getCurriculo().getDadosGerais().getNomeCompleto();
        this.numeroIdentificadorDiscente =
            ↪ processDto.getCurriculo().getDadosGerais().getNumeroIdentidade();
    }

    public OrientacaoSearchDto(ImportProcessDto processDto, DoutoradoXmlDto
        ↪ doutoradoXmlDto) {
        this.tipoDoTrabalho = TipoDoTrabalho.DOUTORADO;
        this.tituloDoTrabalho = doutoradoXmlDto.getTituloDissertacaoTese();
        this.anoInicio = doutoradoXmlDto.getAnoInicio();
        this.anoFim = doutoradoXmlDto.getAnoConclusao();
        this.instituicao = doutoradoXmlDto.getNomeInstituicao();
        this.nomeOrientador = doutoradoXmlDto.getNomeCompletoOrientador();
        this.numeroIdentificadorOrientador = doutoradoXmlDto.getNumeroIdOrientador();
        this.nomeCoorientador = doutoradoXmlDto.getNomeCompletoOrientador();
        this.programaId = processDto.getProgramaDto().getId();
        this.nomeDiscente =
            ↪ processDto.getCurriculo().getDadosGerais().getNomeCompleto();
        this.numeroIdentificadorDiscente =
            ↪ processDto.getCurriculo().getDadosGerais().getNumeroIdentidade();
    }

    public OrientacaoSearchDto(ImportProcessDto processDto, PosDoutoradoXmlDto
        ↪ posDoutoradoXmlDto) {
        this.tipoDoTrabalho = TipoDoTrabalho.POS_DOUTORADO;
        this.tituloDoTrabalho = posDoutoradoXmlDto.getTituloTrabalho();
        this.anoInicio = posDoutoradoXmlDto.getAnoInicio();
        this.anoFim = posDoutoradoXmlDto.getAnoConclusao();
        this.instituicao = posDoutoradoXmlDto.getNomeInstituicao();
        this.numeroIdentificadorOrientador =
            ↪ posDoutoradoXmlDto.getNumeroIdOrientador();
        this.programaId = processDto.getProgramaDto().getId();
        this.nomeDiscente =
            ↪ processDto.getCurriculo().getDadosGerais().getNomeCompleto();
        this.numeroIdentificadorDiscente =
            ↪ processDto.getCurriculo().getDadosGerais().getNumeroIdentidade();
    }

    public enum TipoDoTrabalho {
        MESTRADO,
        DOUTORADO,
        POS_DOUTORADO;
    }
}

package alpc.ufsc.pessoa.orientacao.common.dto;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

@Getter
@Setter
@NoArgsConstructor
public class OrientacaoResultDto {

    private Long orientacaoId;

    private Long orientadorId;

    private Long coorientadorId;

    private Long discenteId;

    private boolean discenteCompleto;

    private String numeroIdentificadorOrientador;

    private String numeroIdentificadorCoorientador;

    private String numeroIdentificadorDiscente;

    public OrientacaoResultDto(Object[] fields) {
        this.orientacaoId = (Long) fields[7];
        this.orientadorId = (Long) fields[8];
        this.coorientadorId = (Long) fields[9];
    }
}

```



```

        this.discenteId = (Long) fields[10];
        this.discenteCompleto = (Boolean) fields[11];

        this.numeroIdentificadorOrientador = (String) fields[2];
        this.numeroIdentificadorCoorientador = (String) fields[4];
        this.numeroIdentificadorDiscente = (String) fields[6];
    }

    public boolean isSetOrientadorId() {
        return this.orientadorId != null;
    }

    public boolean isSetCoorientadorId() {
        return this.coorientadorId != null;
    }

    public boolean isSetDiscenteId() {
        return this.discenteId != null;
    }

    public boolean isSetOrientacaoId() {
        return this.orientacaoId != null;
    }

    public boolean isSetNumeroIdentificadorOrientador() {
        return this.numeroIdentificadorOrientador != null;
    }

    public boolean isSetNumeroIdentificadorCoorientador() {
        return this.numeroIdentificadorCoorientador != null;
    }

    public boolean isSetNumeroIdentificadorDiscente() {
        return this.numeroIdentificadorDiscente != null;
    }
}
package alpc.ufsc.pessoa.orientacao.common.dto;

import java.util.List;

import lombok.Getter;
import lombok.Setter;

import br.ufsc.bridge.metafy.Metafy;

import com.fasterxml.jackson.annotation.JsonIgnore;
import com.fasterxml.jackson.annotation.JsonInclude;
import com.fasterxml.jackson.annotation.JsonInclude.Include;
import com.fasterxml.jackson.databind.annotation.JsonSerialize;

import alpc.ufsc.domain.area.dto.AreaConhecimentoRowDTO;
import alpc.ufsc.pessoa.orientacao.common.util.AreasConhecimento;
import alpc.ufsc.util.AnoJsonSerialize;

@Getter
@Setter
@Metafy
@JsonInclude(Include.NON_EMPTY)
public class FormacaoRowDTO implements AreasConhecimento {

    private String nomeFormacao;
    private Long idFormacao;

    private String tituloTrabalho;

    private String nomeOrientador;
    private String nomeCoorientador;
    private String nomeDocenteSanduiche;

    @JsonSerialize(using = AnoJsonSerialize.class)
    private Integer anoInicio;
    @JsonSerialize(using = AnoJsonSerialize.class)
    private Integer anoFim;

    private String nomeInstituicao;
    private String nomePais;

    private String statusFormacao;

    @JsonIgnore
    private Long grandeAreaConhecimentoUm;
    @JsonIgnore

```

```

        private String nomeAreaConhecimentoUm;
        @JsonIgnore
        private String nomeSubAreaConhecimentoUm;
        @JsonIgnore
        private String nomeEspecialidadeUm;

        @JsonIgnore
        private Long grandeAreaConhecimentoDois;
        @JsonIgnore
        private String nomeAreaConhecimentoDois;
        @JsonIgnore
        private String nomeSubAreaConhecimentoDois;
        @JsonIgnore
        private String nomeEspecialidadeDois;

        @JsonIgnore
        private Long grandeAreaConhecimentoTres;
        @JsonIgnore
        private String nomeAreaConhecimentoTres;
        @JsonIgnore
        private String nomeSubAreaConhecimentoTres;
        @JsonIgnore
        private String nomeEspecialidadeTres;

        public List<AreaConhecimentoRowDTO> getAreasConhecimento() {
            return AreasConhecimento.extractAreasConhecimento(this);
        }
    }
}

package alpc.ufsc.pessoa.orientacao.common.dto;

import org.apache.commons.lang3.StringUtils;

import alpc.ufsc.entity.util.QueryUtils;
import info.debatty.java.stringsimilarity.JaroWinkler;
import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
@AllArgsConstructor
public class PessoaSearchDto {

    private Long programaId;
    private String nome;

    public boolean isSetName() {
        return !StringUtils.isBlank(this.nome);
    }

    public PessoaSearchDto(String nome) {
        this.nome = nome;
    }

    public String getNome() {
        return QueryUtils.lowerCaseStripAccents(this.nome);
    }

    public boolean similar(PessoaSearchDto other) {
        JaroWinkler jaroWinkler = new JaroWinkler();
        if (other.isSetName()) {
            if (jaroWinkler.similarity(this.getNome(), other.getNome()) > 0.98) {
                return true;
            }
        }
        return false;
    }
}

package alpc.ufsc.pessoa.orientacao.common.service;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.apache.commons.lang3.StringUtils;
import org.apache.lucene.search.BooleanClause.Occur;
import org.apache.lucene.search.BooleanQuery;
import org.apache.lucene.search.spell.LevenshteinDistance;
import org.hibernate.search.SearchFactory;
import org.hibernate.search.errors.EmptyQueryException;

```

```

import org.hibernate.search.jpa.FullTextEntityManager;
import org.hibernate.search.jpa.FullTextQuery;
import org.hibernate.search.jpa.Search;
import org.hibernate.search.query.dsl.QueryBuilder;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import alpc.ufsc.domain.instituicao.dto.InstituicaoSaveDTO;
import alpc.ufsc.entity.pessoa.orientacao.doutorado.OrientacaoDoutoradoEntity;
import alpc.ufsc.entity.pessoa.orientacao.mestrado.OrientacaoMestradoEntity;
import alpc.ufsc.entity.pessoa.orientacao.posdoutorado.OrientacaoPosDoutoradoEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoResultDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoSearchDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoSearchDto.TipoDoTrabalho;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoValidateDto;
import alpc.ufsc.programa.dto.ProgramaFormDTO;
import info.debatty.java.stringsimilarity.JaroWinkler;

@Service
@Transactional(propagation = Propagation.SUPPORTS, noRollbackFor =
    ↳ EmptyQueryException.class)
public class OrientacaoSearchService {

    private static final String TITULO = "titulo";
    private static final String ORIENTADOR_PESSOA_NOME = "orientador.pessoa.nome";
    private static final String ORIENTADOR_PESSOA_NUMERO_IDENTIFICADOR =
        ↳ "orientador.pessoa.numeroIdentificador";
    private static final String COORIENTADOR_PESSOA_NOME =
        ↳ "coorientador.pessoa.nome";
    private static final String COORIENTADOR_PESSOA_NUMERO_IDENTIFICADOR =
        ↳ "coorientador.pessoa.numeroIdentificador";
    private static final String DISCENTE_PESSOA_NOME = "discente.pessoa.nome";
    private static final String DISCENTE_PESSOA_NUMERO_IDENTIFICADOR =
        ↳ "discente.pessoa.numeroIdentificador";

    public static final String ORIENTACAO_ID = "id";
    public static final String ORIENTADOR_ID = "orientador.id";
    public static final String COORIENTADOR_ID = "coorientador.id";
    public static final String DISCENTE_ID = "discente.id";
    public static final String DISCENTE_COMPLETO = "discenteCompleto";

    private static final String[] PREFIXES = { "programa ", "de ", "pos-", "pos ", "graduacao
        ↳ ", "em " };

    @PersistenceContext
    private EntityManager em;

    public OrientacaoResultDto findOrientacao(OrientacaoSearchDto searchDto) {
        OrientacaoResultDto resultadoDto = null;
        switch (searchDto.getTipoDoTrabalho()) {
            case DOUTORADO:
                resultadoDto = this.find(searchDto, OrientacaoDoutoradoEntity.class);
                break;
            case MESTRADO:
                resultadoDto = this.find(searchDto, OrientacaoMestradoEntity.class);
                break;
            case POS_DOUTORADO:
                resultadoDto = this.find(searchDto, OrientacaoPosDoutoradoEntity.class);
                break;
        }
        return resultadoDto;
    }

    public boolean pertencePrograma(OrientacaoValidateDto orientacao, ProgramaFormDTO
        ↳ programaDto) {
        LevenshteinDistance levenshtein = new LevenshteinDistance();
        if (StringUtil.isNotBlank(orientacao.getNomeCurso()) &&
            ↳ StringUtil.isNotBlank(programaDto.getInstituicao().getNome())) {
            return
                ↳ levenshtein.getDistance(this.getNomePrograma(orientacao.getNomeCurso()),
                    ↳ this.getNomePrograma(programaDto.getNome())) > 0.75
                && this.equalsInstituicao(orientacao,
                    ↳ programaDto.getInstituicao());
        }
        return false;
    }

    private boolean equalsInstituicao(OrientacaoValidateDto orientacao, InstituicaoSaveDTO
        ↳ instituicao) {
        JaroWinkler jaroWinkler = new JaroWinkler();
    }

```

```

return
    ↪ jaroWinkler.similarity(QueryUtils.lowerCaseStripAccents(orientacao.getNomeInstituicao()),
    ↪ QueryUtils.lowerCaseStripAccents(instituicao.getNome())) > 0.98
        ||
        ↪ jaroWinkler.similarity(QueryUtils.lowerCaseStripAccents(orientacao.getNomeInsti
        ↪ QueryUtils.lowerCaseStripAccents(instituicao.getSigla()))
        ↪ >= 0.99;
}

private OrientacaoResultDto find(OrientacaoSearchDto searchDto, Class<?> entity) {
    FullTextEntityManager search = Search.getFullTextEntityManager(this.em);
    search.flushToIndexes();
    SearchFactory searchFactory = search.getSearchFactory();
    QueryBuilder qb = searchFactory.buildQueryBuilder().forEntity(entity).get();

    BooleanQuery query = new BooleanQuery();
    query.add(qb.keyword().onField(DISCENTE_PESSOA_NOME).matching(searchDto.getNomeDiscente()).createQuery(),
        ↪ Occur.MUST);
    query.add(qb.keyword().onField(TITULO).matching(searchDto.getTituloDoTrabalho()).createQuery(),
        ↪ Occur.MUST);

    if (searchDto.isSetNumeroIdentificadorDiscente()) {
        query.add(qb.keyword().onField(DISCENTE_PESSOA_NUMERO_IDENTIFICADOR).matching(sea
            ↪ Occur.SHOULD);
    }

    FullTextQuery fullTextQuery = search.createFullTextQuery(query, entity);

    fullTextQuery.setProjection(this.getProjection(!searchDto.getTipoDoTrabalho().equals(TipoDoTrabalho.POS
    fullTextQuery.setMaxResults(10);

    List<?> result = fullTextQuery.getResultList();
    for (Object object : result) {
        Object[] fields = (Object[]) object;
        if (searchDto.similar(new OrientacaoSearchDto(fields))) {
            return new OrientacaoResultDto(fields);
        }
    }
    return new OrientacaoResultDto();
}

private String[] getProjection(boolean notIsPosDoutorado) {
    if (notIsPosDoutorado) {
        //@formatter:off
        String[] p = {
            TITULO,
            ORIENTADOR_PESSOA_NOME,
            ORIENTADOR_PESSOA_NUMERO_IDENTIFICADOR,
            COORIENTADOR_PESSOA_NOME,
            COORIENTADOR_PESSOA_NUMERO_IDENTIFICADOR,
            DISCENTE_PESSOA_NOME,
            DISCENTE_PESSOA_NUMERO_IDENTIFICADOR,
            ORIENTACAO_ID,
            ORIENTADOR_ID,
            COORIENTADOR_ID,
            DISCENTE_ID,
            DISCENTE_COMPLETO
        };
        //@formatter:on
        return p;
    } else {
        //@formatter:off
        String[] p = {
            TITULO,
            DISCENTE_PESSOA_NOME,
            DISCENTE_PESSOA_NUMERO_IDENTIFICADOR,
            ORIENTACAO_ID,
            DISCENTE_ID,
            DISCENTE_COMPLETO
        };
        //@formatter:on
        return p;
    }
}

public String getNomePrograma(String nome) {
    if (StringUtils.isNotBlank(nome)) {
        String nomePrograma = QueryUtils.lowerCaseStripAccents(nome);
        for (String prefix : PREFIXES) {
            if (nomePrograma.startsWith(prefix)) {
                nomePrograma = nomePrograma.replaceFirst(prefix, "");
            }
        }
    }
}

```

```

    }
    return nomePrograma;
}
return null;
}
}
package alpc.ufsc.pessoa.orientacao.common.service;

import static alpc.ufsc.entity.pessoa.QPessoaEntity.pessoaEntity;
import static alpc.ufsc.entity.pessoa.discente.QDiscenteEntity.discenteEntity;
import static alpc.ufsc.entity.pessoa.docente.QDocenteEntity.docenteEntity;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.Tuple;
import com.mysema.query.jpa.impl.JPAQuery;

import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoResultDto;
import alpc.ufsc.pessoa.orientacao.common.process.BaseOrientacaoGroupingProcessComponent;
import alpc.ufsc.programa.dto.ImportProcessDto;
import ufsc.alpc.xml.dto.CurriculoVitaeXmlDto;
import ufsc.alpc.xml.dto.domain.TipoDeOrientacao;
import ufsc.alpc.xml.dto.outra.producao.orientacoes.concluidas.OrientacoesConcluidasXmlDto;

@Slf4j
@Service
@Transactional
@SuppressWarnings({ "rawtypes", "unchecked" })
public class OrientacaoProcessService {

    @Autowired
    private OrientacaoSearchService orientacaoService;

    @PersistenceContext
    private EntityManager em;

    @Autowired
    protected List<BaseOrientacaoGroupingProcessComponent> orientacoesProcess;

    @Transactional(propagation = Propagation.SUPPORTS)
    public void processOrientacoes(ImportProcessDto processDto) {
        CurriculoVitaeXmlDto curriculo = processDto.getCurriculo();

        for (BaseOrientacaoGroupingProcessComponent process : this.orientacoesProcess) {
            if (processDto.isSetDocentId()) {
                process.disableAllOrientacoesDocente(processDto.getDocenteId());
            }
            if (process.hasOrientacoesEmAndamento(curriculo)) {
                this.processOrientacoesEmAndamento(process, processDto);
            }

            if (process.hasOrientacoesConcluidas(curriculo)) {
                this.processOrientacoesConcluidas(process, processDto);
            }
        }
    }

    private void processOrientacoesEmAndamento(BaseOrientacaoGroupingProcessComponent
        ↪ process, ImportProcessDto processDto) {
        for (Object orientacaoEmAndamento :
            ↪ process.getOrientacoesEmAndamento(processDto.getCurriculo())) {
            if
                ↪ (this.orientacaoService.pertencePrograma(process.getOrientacaoValidateDtoEmAndamento()
                ↪ processDto.getProgramaDto())) {
                    process.processOrientacaoEmAndamento(processDto,
                        ↪ orientacaoEmAndamento);
                }
            }
        }
    }

    private void processOrientacoesConcluidas(BaseOrientacaoGroupingProcessComponent
        ↪ process, ImportProcessDto processDto) {

```

```

for (OrientacoesConcluidasXmlDto orientacaoConcluida :
    ↪ processDto.getCurriculo().getOutraProducao().getOrientacoesConcluidas())
    ↪ {
        if (process.isSetOrientacoesConcluidas(orientacaoConcluida)) {
            for (Object orientacaoConcluidaXml :
                ↪ process.getOrientacoesConcluidas(orientacaoConcluida)) {
                    if
                        ↪ (this.orientacaoService.pertencePrograma(process.getOrientacaoValidatedtoEmAndamento(
                            ↪ processDto.getProgramaDto())) {
                                process.processOrientacaoConcluida(processDto,
                                    ↪ orientacaoConcluidaXml);
                            }
                    }
            }
        }
    }
}

@Transactional(propagation = Propagation.SUPPORTS)
public void searchInOrientacoes(ImportProcessDto processDto) {
    CurriculoVitaeXmlDto curriculo = processDto.getCurriculo();

    for (BaseOrientacaoGroupingProcessComponent process : this.orientacoesProcess) {
        if (process.hasOrientacoesEmAndamento(curriculo)) {
            if (this.processOrientacoesEmAndamento(processDto, process)) {
                return;
            }
        }

        if (process.hasOrientacoesConcluidas(curriculo)) {
            this.processOrientacoesConcluidas(processDto, process);
        }
    }
}

private boolean processOrientacoesEmAndamento(ImportProcessDto processDto,
    ↪ BaseOrientacaoGroupingProcessComponent process) {
    for (Object orientacaoEmAndamento :
        ↪ process.getOrientacoesEmAndamento(processDto.getCurriculo())) {
            if
                ↪ (this.orientacaoService.pertencePrograma(process.getOrientacaoValidatedtoEmAndamento(
                    ↪ processDto.getProgramaDto())) {
                    OrientacaoResultDto resultDto = null;
                    try {
                        resultDto =
                            ↪ process.findOrientacaoEmAndamento(processDto,
                                ↪ orientacaoEmAndamento);
                    } catch (Exception e) {
                        log.error("Erro ao buscar orientao em andamento",
                            ↪ e.getMessage());
                        continue;
                    }
                    if (this.checkResult(processDto, resultDto,
                        ↪ process.getTipoDeOrientacaoEmAndamento(orientacaoEmAndamento)))
                        ↪ {
                            return true;
                        }
                }
        }
    }
    return false;
}

private void processOrientacoesConcluidas(ImportProcessDto processDto,
    ↪ BaseOrientacaoGroupingProcessComponent process) {
    for (OrientacoesConcluidasXmlDto orientacaoConcluida :
        ↪ processDto.getCurriculo().getOutraProducao().getOrientacoesConcluidas())
        ↪ {
            if (process.isSetOrientacoesConcluidas(orientacaoConcluida)) {
                for (Object orientacaoConcluidaXml :
                    ↪ process.getOrientacoesConcluidas(orientacaoConcluida)) {
                        if
                            ↪ (this.orientacaoService.pertencePrograma(process.getOrientacaoValidatedtoEmAndamento(
                                ↪ processDto.getProgramaDto())) {
                                    OrientacaoResultDto resultDto = null;
                                    try {
                                        resultDto =
                                            ↪ process.findOrientacaoConcluida(processDto,
                                                ↪ orientacaoConcluidaXml);
                                    } catch (Exception e) {
                                        log.error("Erro ao buscar orientao
                                            ↪ concluida", e.getMessage());
                                        continue;
                                    }
                                }
                            }
                }
            }
        }
    }
}

```

```

        if (this.checkResult(processDto, resultDto,
            ↪ process.getTipoDeOrientacaoConcluida(orientacaoConclu
            ↪ {
            ↪     return;
        }
    }
}
}
}
}
}

private boolean checkResult(ImportProcessDto processDto, OrientacaoResultDto resultDto,
    ↪ TipoDeOrientacao tipoDeOrientacao) {
    Long docenteId = null;
    if (resultDto.isSetOrientadorId() &&
        ↪ TipoDeOrientacao.orientador.equals(tipoDeOrientacao))
        ↪ {
        docenteId = resultDto.getOrientadorId();
    } else if (resultDto.isSetCoorientadorId() &&
        ↪ TipoDeOrientacao.coorientador.equals(tipoDeOrientacao)) {
        docenteId = resultDto.getCoorientadorId();
    }
    if (docenteId != null) {
        processDto.setDocenteId(docenteId);

        Tuple result = new JPAQuery(this.em).from(docenteEntity)
            .join(docenteEntity.pessoa(), pessoaEntity)
            .leftJoin(pessoaEntity.discente, discenteEntity)
            .where(docenteEntity.id.eq(docenteId))
            .uniqueResult(pessoaEntity.id, discenteEntity.id);
        processDto.setPessoaId(result.get(pessoaEntity.id));
        processDto.setDiscenteId(result.get(discenteEntity.id));
        return true;
    }
    return false;
}
}
package alpc.ufsc.pessoa.orientacao.common.service;

import static alpc.ufsc.entity.pessoa.QPessoaEntity.pessoaEntity;
import static alpc.ufsc.entity.pessoa.discente.QDiscenteEntity.discenteEntity;
import static alpc.ufsc.entity.pessoa.docente.QDocenteEntity.docenteEntity;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.Tuple;
import com.mysema.query.jpql.impl.JPAQuery;

import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoResultDto;
import alpc.ufsc.pessoa.orientacao.common.process.BaseFormacaoProcessComponent;
import alpc.ufsc.pessoa.orientacao.graduacao.GraduacaoProcessService;
import alpc.ufsc.programa.dto.ImportProcessDto;
import alpc.ufsc.xml.dto.CurriculoVitaeXmlDto;

@Slf4j
@Service
@Transactional(propagation = Propagation.SUPPORTS)
@SuppressWarnings("rawtypes")
public class FormacaoProcessService {

    @Autowired
    protected List<BaseFormacaoProcessComponent> formacaoProcess;

    @Autowired
    protected GraduacaoProcessService graduacaoProcess;

    @PersistenceContext
    protected EntityManager em;

    @SuppressWarnings("unchecked")
    public void processFormacoes(ImportProcessDto processDto) {
        CurriculoVitaeXmlDto curriculo = processDto.getCurriculo();

```

```

    for (BaseFormacaoProcessComponent processComponent : this.formacaoProcess) {
        if (processDto.isSetDiscenteId()) {
            processComponent.disableAllFormacoesDiscente(processDto.getDiscenteId());
        }

        if (processComponent.hasFormacoes(curriculo)) {
            for (Object formacao : processComponent.getFormacoes(curriculo)) {
                processComponent.processFormacao(processDto, formacao);
            }
        }
    }
}

@SuppressWarnings("unchecked")
public void searchInFormacoes(ImportProcessDto processDto) {
    CurriculoVitaeXmlDto curriculo = processDto.getCurriculo();

    for (BaseFormacaoProcessComponent processComponent : this.formacaoProcess) {
        if (processComponent.hasFormacoes(curriculo)) {
            for (Object formacao : processComponent.getFormacoes(curriculo)) {
                OrientacaoResultDto resultDto = null;
                try {
                    resultDto = processComponent.find(processDto,
                                                       ↪ formacao);
                } catch (Exception e) {
                    log.error("Erro ao buscar orientao", e.getMessage());
                    continue;
                }
                if (resultDto.isSetDiscenteId()) {
                    processDto.setDiscenteId(resultDto.getDiscenteId());

                    Tuple result = new
                        ↪ JPAQuery(this.em).from(discenteEntity)
                            .join(discenteEntity.pessoa(),
                                   ↪ pessoaEntity)
                            .leftJoin(pessoaEntity.docente,
                                       ↪ docenteEntity)
                            .where(discenteEntity.id.eq(resultDto.getDiscenteId()))
                            .uniqueResult(pessoaEntity.id,
                                           ↪ docenteEntity.id);
                    processDto.setPessoaId(result.get(pessoaEntity.id));
                    processDto.setDocenteId(result.get(docenteEntity.id));
                    return;
                }
            }
        }
    }
}

package alpc.ufsc.pessoa.orientacao.posdoutorado;

import static
    ↪ alpc.ufsc.entity.pessoa.orientacao.posdoutorado.QOrientacaoPosDoutoradoEntity.orientacaoPosDoutoradoEntity;

import java.util.List;

import org.springframework.stereotype.Component;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.jpa.impl.JPAUpdateClause;

import alpc.ufsc.common.exception.EmptyXmlException;
import alpc.ufsc.domain.instituicao.dto.InstituicaoSaveDTO;
import alpc.ufsc.entity.domain.StatusOrientacao;
import alpc.ufsc.entity.pessoa.docente.DocenteEntity;
import alpc.ufsc.entity.pessoa.orientacao.posdoutorado.OrientacaoPosDoutoradoEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.dto.PessoaDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoResultDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoSearchDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoValidateDto;
import alpc.ufsc.pessoa.orientacao.common.process.BaseFormacaoProcessComponent;
import alpc.ufsc.programa.dto.ImportProcessDto;
import ufsc.alpc.xml.dto.CurriculoVitaeXmlDto;
import ufsc.alpc.xml.dto.common.area.conhecimento.AreasDoConhecimentoXmlDto;
import ufsc.alpc.xml.dto.dados.gerais.formacao.posdoutorado.PosDoutoradoXmlDto;

@Component
@Transactional(propagation = Propagation.SUPPORTS)

```



```

public class FormacaoPosDoutoradoProcessComponent extends
↳ BaseFormacaoProcessComponent<OrientacaoPosDoutoradoEntity, PosDoutoradoXmlDto>
↳ {

    @Override
    public OrientacaoSearchDto getOrientacaoSearch(PosDoutoradoXmlDto formacao,
↳ ImportProcessDto processDto) {
        return new OrientacaoSearchDto(processDto, formacao);
    }

    @Override
    public PessoaDto getPessoaDocente(PosDoutoradoXmlDto formacao, ImportProcessDto
↳ processDto, boolean orientadorPrincipal) {
        return new PessoaDto(formacao, processDto.getProgramaDto().getId());
    }

    @Override
    protected DocenteEntity saveDocente(OrientacaoResultDto resultDto, PessoaDto
↳ pessoaDocente, boolean orientadorPrincipal) {
        // ps doutorado no tem docente
        return null;
    }

    @Override
    public boolean hasFormacoes(CurriculoVitaeXmlDto curriculo) {
        return curriculo.getDadosGerais().isSetFormacaoAcademica() &&
↳ curriculo.getDadosGerais().getFormacaoAcademica().isSetPosDoutorado();
    }

    @Override
    public List<PosDoutoradoXmlDto> getFormacoes(CurriculoVitaeXmlDto curriculo) {
        return curriculo.getDadosGerais().getFormacaoAcademica().getPosDoutorado();
    }

    @Override
    protected void setFieldsToEntity(OrientacaoPosDoutoradoEntity orientacaoEntity,
↳ PosDoutoradoXmlDto formacaoXml, ImportProcessDto processDto) {
        orientacaoEntity.setAnoInicio(QueryUtils.toMinYear(formacaoXml.getAnoInicio()));
        orientacaoEntity.setAnoFim(QueryUtils.toMaxYear(formacaoXml.getAnoConclusao()));
        orientacaoEntity.setTituloDoTrabalho(formacaoXml.getTituloTrabalho());
        orientacaoEntity.setBolsa(formacaoXml.getFlagBolsa());
        orientacaoEntity.setStatusOrientacao(StatusOrientacao.getByStatusOrientacaoXML(formacaoXml.get
    )

    @Override
    public OrientacaoValidateDto getOrientacaoValidateDto(PosDoutoradoXmlDto formacao) {
        return new OrientacaoValidateDto(formacao);
    }

    @Override
    public InstituicaoSaveDTO getInstituicaoSaveDTO(ImportProcessDto processDto,
↳ PosDoutoradoXmlDto formacaoXml) throws EmptyXmlException {
        return new InstituicaoSaveDTO(formacaoXml.getCodigoCurso(),
↳ formacaoXml.getCodigoInstituicao(), formacaoXml.getNomeInstituicao(),
↳ processDto.getCurriculo());
    }

    @Override
    public AreasDoConhecimentoXmlDto getAreasConhecimento(PosDoutoradoXmlDto
↳ formacao) {
        return formacao.getAreasDoConhecimento();
    }

    @Override
    @Transactional(propagation = Propagation.REQUIRES_NEW)
    public void disableAllFormacoesDiscente(Long discenteId) {
        JPAUpdateClause updateClause = new JPAUpdateClause(this.em,
↳ orientacaoPosDoutoradoEntity);
        updateClause.set(orientacaoPosDoutoradoEntity.discenteCompleto, false);
        updateClause.where(orientacaoPosDoutoradoEntity.discente().id.eq(discenteId));
        updateClause.execute();
    }
}

package alpc.ufsc.pessoa.docente;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;

```

```

import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.jpa.impl.JPAQuery;
import com.mysema.query.jpa.impl.JPAUpdateClause;

import alpc.ufsc.entity.pessoa.PessoaEntity;
import alpc.ufsc.entity.pessoa.docente.DocenteEntity;
import alpc.ufsc.entity.pessoa.docente.QDocenteEntity;
import alpc.ufsc.pessoa.PessoaService;
import alpc.ufsc.pessoa.docente.dto.DocenteFilterDTO;
import alpc.ufsc.pessoa.docente.dto.DocenteHeaderDTO;
import alpc.ufsc.pessoa.docente.dto.DocenteProdutividadeDTO;
import alpc.ufsc.pessoa.docente.dto.DocenteRowBodyDTO;
import alpc.ufsc.pessoa.docente.dto.DocenteRowHeaderDTO;
import alpc.ufsc.pessoa.docente.query.DocenteHeaderQuery;
import alpc.ufsc.pessoa.docente.query.DocenteRowHeaderQuery;
import alpc.ufsc.pessoa.dto.PessoaDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoResultDto;
import alpc.ufsc.pessoa.orientacao.common.query.FormacaoRowQuery;
import alpc.ufsc.pessoa.producao.service.ProducaoService;
import alpc.ufsc.programa.dto.ImportProcessDto;

@Service
@Transactional(propagation = Propagation.REQUIRED)
public class DocenteService {

    @PersistenceContext
    private EntityManager em;

    @Autowired
    private PessoaService pessoaService;

    @Autowired
    private DocenteHeaderQuery docenteHeaderQuery;

    @Autowired
    private DocenteRowHeaderQuery docenteRowHeaderQuery;

    @Autowired
    private FormacaoRowQuery formacaoQuery;

    @Autowired
    private ProducaoService producaoService;

    public DocenteEntity save(PessoaDto pessoaDto, OrientacaoResultDto orientacaoResult,
        ↪ ImportProcessDto processDto, boolean orientadorPrincipal) {
        DocenteEntity docente = null;
        if (orientadorPrincipal && orientacaoResult.isSetOrientadorId()) {
            docente = this.em.getReference(DocenteEntity.class,
                ↪ orientacaoResult.getOrientadorId());
        } else if (!orientadorPrincipal && orientacaoResult.isSetCoorientadorId()) {
            docente = this.em.getReference(DocenteEntity.class,
                ↪ orientacaoResult.getCoorientadorId());
        } else if (processDto != null && processDto.isSetDocenteId()) {
            docente = this.em.getReference(DocenteEntity.class,
                ↪ processDto.getDocenteId());
        } else {
            docente = this.saveByPessoa(pessoaDto);
        }
        if (processDto != null) {
            processDto.setDocenteId(docente.getId());
        }
        return docente;
    }

    private DocenteEntity saveByPessoa(PessoaDto pessoaDto) {
        Long pessoaId = this.pessoaService.findOtherwiseSave(pessoaDto);

        QDocenteEntity qDocente = QDocenteEntity.docenteEntity;
        DocenteEntity docente = new
            ↪ JPAQuery(this.em).from(qDocente).where(qDocente.pessoa().id.eq(pessoaId)).uniqueResult(qDocente);

        if (docente == null) {
            docente = new DocenteEntity();
            docente.setPessoa(this.em.getReference(PessoaEntity.class, pessoaId));
            this.em.persist(docente);
        }
        return docente;
    }
}

```

```

public DocenteHeaderDTO getHeader(DocenteFilterDTO filter) {
    return this.docenteHeaderQuery.getHeader(filter);
}

public Page<DocenteRowHeaderDTO> getPage(DocenteFilterDTO filter, Pageable
    ↪ pageable) {
    return this.docenteRowHeaderQuery.getList(filter, pageable);
}

public DocenteRowBodyDTO getDocenteRowBody(Long docenteId, Long discenteId) {
    DocenteRowBodyDTO bodyDTO = new DocenteRowBodyDTO();
    bodyDTO.setProdcoes(this.producaoService.getProducaoDocente(docenteId));
    bodyDTO.setFormacoes(this.formacaoQuery.getFormacoes(discenteId, false, true));
    return bodyDTO;
}

public void updateProdutividade(DocenteProdutividadeDTO dto) {
    QDocenteEntity qDocente = QDocenteEntity.docenteEntity;
    JPAUpdateClause update = new JPAUpdateClause(this.em, qDocente);
    update.set(qDocente.modalidade(), dto.getModalidade());
    update.set(qDocente.categoria(), dto.getCategoria());
    update.where(qDocente.id.eq(dto.getId()));
    update.execute();
}
}

package alpc.ufsc.pessoa.docente.orientacao;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import org.springframework.security.access.annotation.Secured;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;

import alpc.ufsc.login.authority.Authority;
import alpc.ufsc.login.user.CurrentUser;
import alpc.ufsc.login.user.dto.UserDetailsImpl;
import alpc.ufsc.pessoa.docente.orientacao.dto.DocenteOrientacaoFilterDTO;
import alpc.ufsc.pessoa.docente.orientacao.dto.DocenteOrientacaoRowDTO;
import alpc.ufsc.pessoa.docente.orientacao.dto.DocenteOrientacaoStatusCursoDTO;

@RestController
@RequestMapping("/services/docente")
public class DocenteOrientacaoRestController {

    @Autowired
    private DocenteOrientacaoService service;

    @Secured({ Authority.COORDENADOR, Authority.DOCENTE })
    @RequestMapping(value = "{id}/orientacoes", method = RequestMethod.GET)
    public @ResponseBody Page<DocenteOrientacaoRowDTO> getPage(@PathVariable Long id,
        ↪ DocenteOrientacaoFilterDTO filter, Pageable pageable) {
        filter.setIdDocente(id);
        return this.service.getPage(filter, pageable);
    }

    @Secured(Authority.COORDENADOR)
    @RequestMapping(value = "/orientacao-mestrado/status", method =
        ↪ RequestMethod.POST)
    public void updateStatusOrientacaoMestrado(@CurrentUser UserDetailsImpl user,
        ↪ @RequestBody DocenteOrientacaoStatusCursoDTO dto) {
        dto.setProgramaId(user.getProgramaId());
        this.service.updateStatusOrientacaoMestrado(dto);
    }

    @Secured(Authority.COORDENADOR)
    @RequestMapping(value = "/orientacao-doutorado/status", method =
        ↪ RequestMethod.POST)
    public void updateStatusOrientacaoDoutorado(@CurrentUser UserDetailsImpl user,
        ↪ @RequestBody DocenteOrientacaoStatusCursoDTO dto) {
        dto.setProgramaId(user.getProgramaId());
        this.service.updateStatusOrientacaoDoutorado(dto);
    }
}

package alpc.ufsc.pessoa.docente.orientacao;

import static alpc.ufsc.pessoa.docente.orientacao.dto.MDocenteOrientacaoRowDTO.meta;

```

```

import static alpc.ufsc.util.querydsl.ExpressionsUtils.addAs;

import java.util.ArrayList;
import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageImpl;
import org.springframework.data.domain.Pageable;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.Tuple;
import com.mysema.query.jpa.impl.JPAUpdateClause;
import com.mysema.query.sql.SQLQuery;
import com.mysema.query.sql.SQLQueryFactory;
import com.mysema.query.sql.SQLSubQuery;
import com.mysema.query.support.Expressions;
import com.mysema.query.types.NullExpression;
import com.mysema.query.types.expr.BooleanExpression;
import com.mysema.query.types.expr.CaseBuilder;
import com.mysema.query.types.expr.StringExpression;
import com.mysema.query.types.path.NumberPath;
import com.mysema.query.types.path.NumberPath;
import com.mysema.query.types.path.SimplePath;
import com.mysema.query.types.query.ListSubQuery;

import alpc.ufsc.entity.pessoa.orientacao.doutorado.QOrientacaoDoutoradoEntity;
import alpc.ufsc.entity.pessoa.orientacao.mestrado.QOrientacaoMestradoEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.docente.orientacao.dto.DocenteOrientacaoFilterDTO;
import alpc.ufsc.pessoa.docente.orientacao.dto.DocenteOrientacaoRowDTO;
import alpc.ufsc.pessoa.docente.orientacao.dto.DocenteOrientacaoStatusCursoDTO;
import alpc.ufsc.pessoa.docente.orientacao.dto.MDocenteOrientacaoRowDTO;
import alpc.ufsc.querydsl.sql.QTbAreaConhecimento;
import alpc.ufsc.querydsl.sql.QTbDiscente;
import alpc.ufsc.querydsl.sql.QTbInstituicao;
import alpc.ufsc.querydsl.sql.QTbOrientacaoDoutorado;
import alpc.ufsc.querydsl.sql.QTbOrientacaoMestrado;
import alpc.ufsc.querydsl.sql.QTbPais;
import alpc.ufsc.querydsl.sql.QTbPessoa;
import alpc.ufsc.util.querydsl.Select;
import ufsc.alpc.xml.dto.domain.TipoDeOrientacao;

@Service
@Transactional(propagation = Propagation.REQUIRED)
public class DocenteOrientacaoService {

    @PersistenceContext
    private EntityManager em;

    @Autowired
    private SQLQueryFactory queryFactory;

    private QTbOrientacaoMestrado qMestrado =
        ↳ QTbOrientacaoMestrado.tbOrientacaoMestrado;
    private QTbOrientacaoDoutorado qDoutorado =
        ↳ QTbOrientacaoDoutorado.tbOrientacaoDoutorado;

    private QOrientacaoMestradoEntity qJPAMestrado =
        ↳ QOrientacaoMestradoEntity.orientacaoMestradoEntity;
    private QOrientacaoDoutoradoEntity qJPADoutorado =
        ↳ QOrientacaoDoutoradoEntity.orientacaoDoutoradoEntity;

    public void updateStatusOrientacaoMestrado(DocenteOrientacaoStatusCursoDTO dto) {
        JPAUpdateClause updateClause = new JPAUpdateClause(this.em,
            ↳ this.qJPAMestrado);
        updateClause.set(this.qJPAMestrado.statusOrientacao, dto.getStatusOrientacao());
        updateClause.where(this.qJPAMestrado.id.eq(dto.getOrientacaoId()));
        updateClause.where(this.qJPAMestrado.programa().id.eq(dto.getProgramaId())).execute();
    }

    public void updateStatusOrientacaoDoutorado(DocenteOrientacaoStatusCursoDTO dto) {
        JPAUpdateClause updateClause = new JPAUpdateClause(this.em,
            ↳ this.qJPADoutorado);
        updateClause.set(this.qJPADoutorado.statusOrientacao, dto.getStatusOrientacao());
        updateClause.where(this.qJPADoutorado.id.eq(dto.getOrientacaoId()));
        updateClause.where(this.qJPADoutorado.programa().id.eq(dto.getProgramaId())).execute();
    }

```

```

}

public Page<DocenteOrientacaoRowDTO> getPage(DocenteOrientacaoFilterDTO filter,
↳ Pageable pageable) {
    SQLQuery query = this.queryFactory.query();

    List<ListSubQuery<Tuple>> subs = new ArrayList<>();
    subs.add(this.getOrientacaoMestrado(filter));
    subs.add(this.getOrientacaoDoutorado(filter));

    query.from(this.queryFactory.subQuery().unionAll(subs).as("union_orientacoes"));

    SimplePath<Void> orientacoes = Expressions.path(Void.class, "union_orientacoes");
    MDocenteOrientacaoRowDTO meta = MDocenteOrientacaoRowDTO.meta;

    Select<DocenteOrientacaoRowDTO> select = new
↳ Select<>(DocenteOrientacaoRowDTO.class);
    addAs(select, meta.nomeOrientacao, orientacoes);
    addAs(select, meta.orientacaoId, orientacoes);
    addAs(select, meta.tituloTrabalho, orientacoes);
    addAs(select, meta.nomeDiscente, orientacoes);
    addAs(select, meta.nomeInstituicao, orientacoes);
    addAs(select, meta.nomePais, orientacoes);
    addAs(select, meta.nomeDocenteSanduiche, orientacoes);

    addAs(select, meta.anoFim, orientacoes);
    addAs(select, meta.anoInicio, orientacoes);
    addAs(select, meta.tipoOrientacao, orientacoes);
    addAs(select, meta.statusOrientacao, orientacoes);
    addAs(select, meta.discenteCompleto, orientacoes);
    addAs(select, meta.pertencentePrograma, orientacoes);

    addAs(select, meta.grandeAreaConhecimentoUm, orientacoes);
    addAs(select, meta.nomeAreaConhecimentoUm, orientacoes);
    addAs(select, meta.nomeSubAreaConhecimentoUm, orientacoes);
    addAs(select, meta.nomeEspecialidadeUm, orientacoes);

    addAs(select, meta.grandeAreaConhecimentoDois, orientacoes);
    addAs(select, meta.nomeAreaConhecimentoDois, orientacoes);
    addAs(select, meta.nomeSubAreaConhecimentoDois, orientacoes);
    addAs(select, meta.nomeEspecialidadeDois, orientacoes);

    addAs(select, meta.grandeAreaConhecimentoTres, orientacoes);
    addAs(select, meta.nomeAreaConhecimentoTres, orientacoes);
    addAs(select, meta.nomeSubAreaConhecimentoTres, orientacoes);
    addAs(select, meta.nomeEspecialidadeTres, orientacoes);

    query.offset(pageable.getOffset());
    query.limit(pageable.getPageSize());

    query.orderBy(Expressions.stringPath(orientacoes,
↳ "titulo_do_trabalho_filtro").asc());

    return new PageImpl<>(query.list(select), pageable, query.count());
}

private ListSubQuery<Tuple> getOrientacaoMestrado(DocenteOrientacaoFilterDTO filter) {
    QTbPessoa qPessoa = new QTbPessoa("mpessoa");
    QTbDiscente qDiscente = new QTbDiscente("mdiscente");
    QTbInstituicao qInstituicao = new QTbInstituicao("minstituicao");
    QTbPais qPais = new QTbPais("mpais");
    QTbAreaConhecimento qAreaUm = new QTbAreaConhecimento("mareaum");
    QTbAreaConhecimento qAreaDois = new QTbAreaConhecimento("mareadois");
    QTbAreaConhecimento qAreaTres = new QTbAreaConhecimento("mareatres");

    SQLSubQuery query = this.queryFactory.subQuery().from(this.qMestrado);
    query.join(qDiscente).on(this.qMestrado.idDiscente.eq(qDiscente.idDiscente));
    query.join(qPessoa).on(qPessoa.idPessoa.eq(qDiscente.idPessoa));
    query.join(qInstituicao).on(this.qMestrado.idInstituicao.eq(qInstituicao.idInstituicao));
    query.join(qPais).on(qInstituicao.idPais.eq(qPais.idPais));
    query.leftJoin(qAreaUm).on(this.qMestrado.idAreaConhecimentoUm.eq(qAreaUm.idAreaConhecimen
    query.leftJoin(qAreaDois).on(this.qMestrado.idAreaConhecimentoDois.eq(qAreaDois.idAreaConhecim
    query.leftJoin(qAreaTres).on(this.qMestrado.idAreaConhecimentoTres.eq(qAreaTres.idAreaConhecim

    query.where(this.qMestrado.idOrientador.eq(filter.getIdDocente()))
        .or(this.qMestrado.idCoorientador.eq(filter.getIdDocente()));

    query.where(this.qMestrado.deletado.eq(false));
    if (filter.hasNomeDiscente()) {
        query.where(qPessoa.nomeFiltro.like(QueryUtils.like(filter.getNomeDiscente())));
    }
    if (filter.hasTitulo()) {

```

```

        query.where(this.qMestrado.tituloDoTrabalhoFiltro.like(QueryUtils.like(filter.getTitulo())));
    }
    if (filter.hasAnoInicio() || filter.hasAnoFim()) {
        query.where(QueryUtils.overlappingInterval(this.qMestrado.anoInicio,
            ↪ this.qMestrado.anoFim, filter.getAnoInicio(), filter.getAnoFim()));
    }

    return query.list(
        Expressions.constantAs("Orientao de Mestrado",
            ↪ Expressions.path(String.class,
            ↪ meta.nomeOrientacao.getAlias()),
            this.qMestrado.idOrientacaoMestrado.as(meta.orientacaoId.getAlias()),
            this.qMestrado.tituloDoTrabalho.as(meta.tituloTrabalho.getAlias()),
            this.qMestrado.tituloDoTrabalhoFiltro,
            qPessoa.nome.as(meta.nomeDiscente.getAlias()),
            qInstituicao.nome.as(meta.nomeInstituicao.getAlias()),
            qPais.nome.as(meta.nomePais.getAlias()),
            Expressions.as(NullExpression.DEFAULT,
            ↪ meta.nomeDocenteSauduche.getAlias()),
            this.qMestrado.anoFim.as(meta.anoFim.getAlias()),
            this.qMestrado.anoInicio.as(meta.anoInicio.getAlias()),
            this.getCasoTipoOrientacao(filter.getIdDocente(),
            ↪ this.qMestrado.idOrientador),
            this.qMestrado.statusOrientacao.as(meta.statusOrientacao.getAlias()),
            this.qMestrado.discenteCompleto.as(meta.discenteCompleto.getAlias()),
            this.getCasoPertencePrograma(this.qMestrado.orientadorAtivo,
            ↪ this.qMestrado.coorientadorAtivo),

            qAreaUm.idGrandeAreaConhecimento.as(meta.grandeAreaConhecimentoUm.getAlias()),
            qAreaUm.nomeAreaConhecimento.as(meta.nomeAreaConhecimentoUm.getAlias()),
            qAreaUm.nomeSubAreaConhecimento.as(meta.nomeSubAreaConhecimentoUm.getAlias()),
            qAreaUm.nomeEspecialidade.as(meta.nomeEspecialidadeUm.getAlias()),

            qAreaDois.idGrandeAreaConhecimento.as(meta.grandeAreaConhecimentoDois.getAlias()),
            qAreaDois.nomeAreaConhecimento.as(meta.nomeAreaConhecimentoDois.getAlias()),
            qAreaDois.nomeSubAreaConhecimento.as(meta.nomeSubAreaConhecimentoDois.getAlias()),
            qAreaDois.nomeEspecialidade.as(meta.nomeEspecialidadeDois.getAlias()),

            qAreaTres.idGrandeAreaConhecimento.as(meta.grandeAreaConhecimentoTres.getAlias()),
            qAreaTres.nomeAreaConhecimento.as(meta.nomeAreaConhecimentoTres.getAlias()),
            qAreaTres.nomeSubAreaConhecimento.as(meta.nomeSubAreaConhecimentoTres.getAlias()),
            qAreaTres.nomeEspecialidade.as(meta.nomeEspecialidadeTres.getAlias()));
    }

    private BooleanExpression getCasePertencePrograma(BooleanPath orientadorAtivo,
        ↪ BooleanPath coorientadorAtivo) {
        return new
            ↪ CaseBuilder().when(orientadorAtivo.eq(true).or(coorientadorAtivo.eq(true)))
            .then(true)
            .otherwise(false)
            .as(meta.pertencentePrograma.getAlias());
    }

    private StringExpression getCaseTipoOrientacao(Long idDocente, NumberPath<Long>
        ↪ idOrientador) {
        return new CaseBuilder().when(idOrientador.eq(idDocente))
            .then(TipoDeOrientacao.ORIENTADOR_PRINCIPAL.getDescricao())
            .otherwise(TipoDeOrientacao.CO_ORIENTADOR.getDescricao())
            .as(meta.tipoOrientacao.getAlias());
    }

    private ListSubQuery<Tuple> getOrientacaoDoutorado(DocenteOrientacaoFilterDTO filter)
        ↪ {
        QTbPessoa qPessoa = new QTbPessoa("dpessoa");
        QTbDiscente qDiscente = new QTbDiscente("ddiscente");
        QTbInstituicao qInstituicao = new QTbInstituicao("dinstituicao");
        QTbPais qPais = new QTbPais("dpais");
        QTbAreaConhecimento qAreaUm = new QTbAreaConhecimento("dareaum");
        QTbAreaConhecimento qAreaDois = new QTbAreaConhecimento("dareadois");
        QTbAreaConhecimento qAreaTres = new QTbAreaConhecimento("dareatres");

        SQLSubQuery query = this.queryFactory.subQuery().from(this.qDoutorado);
        query.join(qDiscente).on(this.qDoutorado.idDiscente.eq(qDiscente.idDiscente));
        query.join(qPessoa).on(qPessoa.idPessoa.eq(qDiscente.idPessoa));
        query.join(qInstituicao).on(this.qDoutorado.idInstituicao.eq(qInstituicao.idInstituicao));
        query.join(qPais).on(qInstituicao.idPais.eq(qPais.idPais));
        query.leftJoin(qAreaUm).on(this.qDoutorado.idAreaConhecimentoUm.eq(qAreaUm.idAreaConhecimento));
        query.leftJoin(qAreaDois).on(this.qDoutorado.idAreaConhecimentoDois.eq(qAreaDois.idAreaConhecimento));
        query.leftJoin(qAreaTres).on(this.qDoutorado.idAreaConhecimentoTres.eq(qAreaTres.idAreaConhecimento));

        query.where(this.qDoutorado.idOrientador.eq(filter.getIdDocente())
            .or(this.qDoutorado.idCoorientador.eq(filter.getIdDocente()));
    }

```

```

query.where(this.qDoutorado.deletado.eq(false));
if (filter.hasNomeDiscente()) {
    query.where(Q Pessoa.nomeFiltro.like(QueryUtils.like(filter.getNomeDiscente())));
}
if (filter.hasTitulo()) {
    query.where(this.qDoutorado.tituloDoTrabalhoFiltro.like(QueryUtils.like(filter.getTitulo())));
}
if (filter.hasAnoInicio() || filter.hasAnoFim()) {
    query.where(QueryUtils.overlappingInterval(this.qDoutorado.anoInicio,
        ↪ this.qDoutorado.anoFim, filter.getAnoInicio(),
        ↪ filter.getAnoFim()));
}

MDocenteOrientacaoRowDTO meta = MDocenteOrientacaoRowDTO.meta;

return query.list(
    Expressions.constantAs("Orientao de Doutorado",
        ↪ Expressions.path(String.class,
        ↪ meta.nomeOrientacao.getAlias()),
    this.qDoutorado.idOrientacaoDoutorado.as(meta.orientacaoId.getAlias()),
    this.qDoutorado.tituloDoTrabalho.as(meta.tituloTrabalho.getAlias()),
    this.qDoutorado.tituloDoTrabalhoFiltro,
    qPessoa.nome.as(meta.nomeDiscente.getAlias()),
    qInstituicao.nome.as(meta.nomeInstituicao.getAlias()),
    qPais.nome.as(meta.nomePais.getAlias()),
    this.qDoutorado.nomeOrientadorSanduiche.as(meta.nomeDocenteSanduiche.getAlias()),
    this.qDoutorado.anoFim.as(meta.anoFim.getAlias()),
    this.qDoutorado.anoInicio.as(meta.anoInicio.getAlias()),
    this.getCasosTipoOrientacao(filter.getIdDocente(),
        ↪ this.qDoutorado.idOrientador),
    this.qDoutorado.statusOrientacao.as(meta.statusOrientacao.getAlias()),
    this.qDoutorado.discenteCompleto.as(meta.discenteCompleto.getAlias()),
    this.getCasosPertencePrograma(this.qDoutorado.orientadorAtivo,
        ↪ this.qDoutorado.coorientadorAtivo),

    qAreaUm.idGrandeAreaConhecimento.as(meta.grandeAreaConhecimentoUm.getAlias()),
    qAreaUm.nomeAreaConhecimento.as(meta.nomeAreaConhecimentoUm.getAlias()),
    qAreaUm.nomeSubAreaConhecimento.as(meta.nomeSubAreaConhecimentoUm.getAlias()),
    qAreaUm.nomeEspecialidade.as(meta.nomeEspecialidadeUm.getAlias()),

    qAreaDois.idGrandeAreaConhecimento.as(meta.grandeAreaConhecimentoDois.getAlias()),
    qAreaDois.nomeAreaConhecimento.as(meta.nomeAreaConhecimentoDois.getAlias()),
    qAreaDois.nomeSubAreaConhecimento.as(meta.nomeSubAreaConhecimentoDois.getAlias()),
    qAreaDois.nomeEspecialidade.as(meta.nomeEspecialidadeDois.getAlias()),

    qAreaTres.idGrandeAreaConhecimento.as(meta.grandeAreaConhecimentoTres.getAlias()),
    qAreaTres.nomeAreaConhecimento.as(meta.nomeAreaConhecimentoTres.getAlias()),
    qAreaTres.nomeSubAreaConhecimento.as(meta.nomeSubAreaConhecimentoTres.getAlias()),
    qAreaTres.nomeEspecialidade.as(meta.nomeEspecialidadeTres.getAlias()));
}
}

package alpc.ufsc.pessoa.docente.orientacao.dto;

import lombok.Getter;
import lombok.Setter;

import alpc.ufsc.entity.domain.StatusOrientacao;

@Getter
@Setter
public class DocenteOrientacaoStatusCursoDTO {

    private Long orientacaoId;
    private Long programId;
    private StatusOrientacao statusOrientacao;
}

package alpc.ufsc.pessoa.docente.orientacao.dto;

import lombok.Getter;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;

@Getter
@Setter
public class DocenteOrientacaoFilterDTO {

    private Long idDocente;
    private Integer anoInicio;
    private Integer anoFim;
    private String nomeDiscente;

```

```

    private String titulo;

    public boolean hasAnoInicio() {
        return this.anoInicio != null;
    }

    public boolean hasAnoFim() {
        return this.anoFim != null;
    }

    public boolean hasNomeDiscente() {
        return !StringUtils.isBlank(this.nomeDiscente);
    }

    public boolean hasTitulo() {
        return !StringUtils.isBlank(this.titulo);
    }
}
package alpc.ufsc.pessoa.docente.orientacao.dto;

import java.util.List;

import lombok.Getter;
import lombok.Setter;

import br.ufsc.bridge.metafy.Metafy;

import com.fasterxml.jackson.annotation.JsonIgnore;
import com.fasterxml.jackson.annotation.JsonInclude;
import com.fasterxml.jackson.annotation.JsonInclude.Include;
import com.fasterxml.jackson.databind.annotation.JsonSerialize;

import alpc.ufsc.domain.area.dto.AreaConhecimentoRowDTO;
import alpc.ufsc.pessoa.orientacao.common.util.AreasConhecimento;
import alpc.ufsc.util.AnoJsonSerialize;

@Getter
@Setter
@Metafy
@JsonInclude(Include.NON_EMPTY)
public class DocenteOrientacaoRowDTO implements AreasConhecimento {

    private Long orientacaoId;

    private String nomeOrientacao;

    private String tituloTrabalho;
    private String nomeDiscente;
    private String nomeInstituicao;
    private String nomePais;

    private String nomeDocenteSanduiche;

    @JsonSerialize(using = AnoJsonSerialize.class)
    private Integer anoInicio;
    @JsonSerialize(using = AnoJsonSerialize.class)
    private Integer anoFim;

    private String tipoOrientacao;
    private String statusOrientacao;

    private Boolean discenteCompleto;
    private Boolean pertencentePrograma;

    @JsonIgnore
    private Long grandeAreaConhecimentoUm;
    @JsonIgnore
    private String nomeAreaConhecimentoUm;
    @JsonIgnore
    private String nomeSubAreaConhecimentoUm;
    @JsonIgnore
    private String nomeEspecialidadeUm;

    @JsonIgnore
    private Long grandeAreaConhecimentoDois;
    @JsonIgnore
    private String nomeAreaConhecimentoDois;
    @JsonIgnore
    private String nomeSubAreaConhecimentoDois;
    @JsonIgnore
    private String nomeEspecialidadeDois;
}

```



```

    @JsonIgnore
    private Long grandeAreaConhecimentoTres;
    @JsonIgnore
    private String nomeAreaConhecimentoTres;
    @JsonIgnore
    private String nomeSubAreaConhecimentoTres;
    @JsonIgnore
    private String nomeEspecialidadeTres;

    public List<AreaConhecimentoRowDTO> getAreasConhecimento() {
        return AreasConhecimento.extractAreasConhecimento(this);
    }
}
package alpc.ufsc.pessoa.docente;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import org.springframework.security.access.annotation.Secured;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;

import alpc.ufsc.login.authority.Authority;
import alpc.ufsc.login.user.CurrentUser;
import alpc.ufsc.login.user.dto.UserDetailsImpl;
import alpc.ufsc.pessoa.docente.dto.DocenteFilterDTO;
import alpc.ufsc.pessoa.docente.dto.DocenteHeaderDTO;
import alpc.ufsc.pessoa.docente.dto.DocenteProdutividadeDTO;
import alpc.ufsc.pessoa.docente.dto.DocenteRowBodyDTO;
import alpc.ufsc.pessoa.docente.dto.DocenteRowHeaderDTO;

@RestController
@RequestMapping("/services/docente")
public class DocenteRestController {

    @Autowired
    private DocenteService service;

    @Secured({ Authority.COORDENADOR, Authority.DOCENTE })
    @RequestMapping(value = "header", method = RequestMethod.GET)
    public @ResponseBody DocenteHeaderDTO getHeader(@CurrentUser UserDetailsImpl user,
        ↳ DocenteFilterDTO docenteFilter) {
        docenteFilter.setProgramaId(user.getProgramaId());
        return this.service.getHeader(docenteFilter);
    }

    @Secured({ Authority.COORDENADOR, Authority.DOCENTE })
    @RequestMapping(method = RequestMethod.GET)
    public @ResponseBody Page<DocenteRowHeaderDTO> getPage(@CurrentUser
        ↳ UserDetailsImpl user, DocenteFilterDTO docenteFilter, Pageable pageable) {
        docenteFilter.setProgramaId(user.getProgramaId());
        return this.service.getPage(docenteFilter, pageable);
    }

    @Secured({ Authority.COORDENADOR, Authority.DOCENTE })
    @RequestMapping(value = "docenteDetalhe", method = RequestMethod.GET)
    public @ResponseBody DocenteRowBodyDTO getFormacoes(Long docenteId, Long
        ↳ discenteId) {
        return this.service.getDocenteRowBody(docenteId, discenteId);
    }

    @Secured(Authority.COORDENADOR)
    @RequestMapping(value = "produtividade", method = RequestMethod.POST)
    public void updateProdutividade(@RequestBody DocenteProdutividadeDTO dto) {
        this.service.updateProdutividade(dto);
    }
}
package alpc.ufsc.pessoa.docente.query;

import static alpc.ufsc.querydsl.sql.QTbDiscente.tbDiscente;
import static alpc.ufsc.querydsl.sql.QTbDocente.tbDocente;
import static alpc.ufsc.querydsl.sql.QTbOrientacaoDoutorado.tbOrientacaoDoutorado;
import static alpc.ufsc.querydsl.sql.QTbOrientacaoMestrado.tbOrientacaoMestrado;
import static alpc.ufsc.querydsl.sql.QTbPessoa.tbPessoa;
import static alpc.ufsc.querydsl.sql.QViewPertencePrograma.viewPertencePrograma;

import java.util.ArrayList;
import java.util.List;

```

```

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageImpl;
import org.springframework.data.domain.Pageable;
import org.springframework.stereotype.Component;

import com.mysema.query.Tuple;
import com.mysema.query.sql.SQLQuery;
import com.mysema.query.sql.SQLQueryFactory;
import com.mysema.query.sql.SQLSubQuery;
import com.mysema.query.support.Expressions;
import com.mysema.query.types.ExpressionUtils;
import com.mysema.query.types.expr.CaseBuilder;
import com.mysema.query.types.expr.CollectionExpressionBase;
import com.mysema.query.types.expr.NumberExpression;
import com.mysema.query.types.path.NumberPath;
import com.mysema.query.types.path.SimplePath;
import com.mysema.query.types.path.StringPath;
import com.mysema.query.types.query.ListSubQuery;

import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.docente.dto.DocenteFilterDTO;
import alpc.ufsc.pessoa.docente.dto.DocenteRowHeaderDTO;
import alpc.ufsc.pessoa.docente.dto.MDocenteRowHeaderDTO;
import alpc.ufsc.querydsl.sql.QTbAreaConhecimento;
import alpc.ufsc.querydsl.sql.QTbInstituicao;
import alpc.ufsc.querydsl.sql.QTbOrientacaoDoutorado;
import alpc.ufsc.querydsl.sql.QTbOrientacaoMestrado;
import alpc.ufsc.querydsl.sql.QTbOrientacaoPosDoutorado;
import alpc.ufsc.util.querydsl.Select;
import ufsc.alpc.xml.dto.domain.StatusCurso;

@Component
public class DocenteRowHeaderQuery {

    private static final String ID_DOCENTE = "id_docente";

    private static final String ID_DISCENTE = "id_discente";
    private static final String MESTRADO_ANDAMENTO = "mestrado_andamento";
    private static final String MESTRADO_CONCLUIDO = "mestrado_concluido";

    private static final String DOUTORADO_ANDAMENTO = "doutorado_andamento";
    private static final String DOUTORADO_CONCLUIDO = "doutorado_concluido";

    @Autowired
    private SQLQueryFactory queryFactory;

    @Autowired
    private DocentePertenceProgramaQuery docenteQuery;

    public Page<DocenteRowHeaderDTO> getList(DocenteFilterDTO filter, Pageable pageable)
    ↪ {
        SQLQuery query = this.queryFactory.query().from(tbDocente);
        query.join(tbPessoa).on(tbDocente.idPessoa.eq(tbPessoa.idPessoa));
        query.join(tbDiscente).on(tbPessoa.idPessoa.eq(tbDiscente.idPessoa));

        SimplePath<Void> mestrado = Expressions.path(Void.class, "mestrado");
        query.leftJoin(this.getSubUnionMestrado(filter),
            ↪ mestrado).on(Expressions.numberPath(Long.class, mestrado,
            ↪ ID_DOCENTE).eq(tbDocente.idDocente));

        SimplePath<Void> doutorado = Expressions.path(Void.class, "doutorado");
        query.leftJoin(this.getSubUnionDoutorado(filter),
            ↪ doutorado).on(Expressions.numberPath(Long.class, doutorado,
            ↪ ID_DOCENTE).eq(tbDocente.idDocente));

        this.applyFilter(filter, query);

        MDocenteRowHeaderDTO meta = MDocenteRowHeaderDTO.meta;
        Select<DocenteRowHeaderDTO> select = new
            ↪ Select<>(DocenteRowHeaderDTO.class);
        select.as(tbDocente.idDocente, meta.id);
        select.as(tbDiscente.idDiscente, meta.discenteId);
        select.as(tbPessoa.nome, meta.nome);
        select.as(tbDocente.modalidade, meta.tipoBolsa().modalidade);
        select.as(tbDocente.categoria, meta.tipoBolsa().categoria);

        select.as(Expressions.numberPath(Long.class, mestrado,
            ↪ MESTRADO_ANDAMENTO),
            ↪ meta.numeroDiscentesMestradoAndamento);
        select.as(Expressions.numberPath(Long.class, mestrado,
            ↪ MESTRADO_CONCLUIDO), meta.numeroDiscentesMestradoConcluido);
    }

```

```

select.as(Expressions.numberPath(Long.class, doutorado,
    ↳ DOUTORADO_ANDAMENTO),
    ↳ meta.numeroDiscentesDoutoradoAndamento);
select.as(Expressions.numberPath(Long.class, doutorado,
    ↳ DOUTORADO_CONCLUIDO),
    ↳ meta.numeroDiscentesDoutoradoConcluido);

query.orderBy(tbPessoa.nomeFiltro.asc());
query.offset(pageable.getOffset());
query.limit(pageable.getPageSize());
return new PageImpl<>(query.list(select), pageable, query.count());
}

private ListSubQuery<Tuple> getSubUnionMestrado(DocenteFilterDTO filter) {
    List<ListSubQuery<Tuple>> subs = new ArrayList<>();
    subs.add(this.getSubQueryOrientacaoMestrado(filter, true));
    subs.add(this.getSubQueryOrientacaoMestrado(filter, false));

    CollectionExpressionBase<?, List<Tuple>> union =
        ↳ this.queryFactory.subQuery().unionAll(subs);

    SimplePath<Void> unionPath = Expressions.path(Void.class, "union_mestrado");
    NumberPath<Long> idDocente = Expressions.numberPath(Long.class, unionPath,
        ↳ ID_DOCENTE);
    NumberPath<Long> andamento = Expressions.numberPath(Long.class, unionPath,
        ↳ MESTRADO_ANDAMENTO);
    NumberPath<Long> concluido = Expressions.numberPath(Long.class, unionPath,
        ↳ MESTRADO_CONCLUIDO);
    SQLSubQuery query =
        ↳ this.queryFactory.subQuery().from(union.as(unionPath.getMetadata().getName()));
    query.groupBy(idDocente);

    return query.list(idDocente.as(ID_DOCENTE),
        andamento.sum().as(MESTRADO_ANDAMENTO),
        concluido.sum().as(MESTRADO_CONCLUIDO));
}

private ListSubQuery<Tuple> getSubQueryOrientacaoMestrado(DocenteFilterDTO filter,
    ↳ boolean orientador) {
    SQLSubQuery subQuery =
        ↳ this.queryFactory.subQuery().from(tbOrientacaoMestrado);
    subQuery.where(tbOrientacaoMestrado.idPrograma.eq(filter.getProgramaId()));
    if (filter.hasAnoInicioOrientacao() || filter.hasAnoFimOrientacao()) {
        subQuery.where(QueryUtils.overlappingInterval(
            tbOrientacaoMestrado.anoInicio,
            tbOrientacaoMestrado.anoFim,
            filter.getAnoInicioOrientacao(),
            filter.getAnoFimOrientacao()));
    }

    subQuery.where(tbOrientacaoMestrado.deletado.eq(false));
    if (orientador) {
        subQuery.where(tbOrientacaoMestrado.orientadorAtivo.eq(true));
        return subQuery.list(tbOrientacaoMestrado.idOrientador.as(ID_DOCENTE),
            this.getCasestatus(tbOrientacaoMestrado.statusOrientacao,
                ↳ StatusCurso.EM_ANDAMENTO).as(MESTRADO_ANDAMENTO),
            this.getCasestatus(tbOrientacaoMestrado.statusOrientacao,
                ↳ StatusCurso.CONCLUIDO).as(MESTRADO_CONCLUIDO));
    } else {
        subQuery.where(tbOrientacaoMestrado.coorientadorAtivo.eq(true));
        return
            ↳ subQuery.list(tbOrientacaoMestrado.idCoorientador.as(ID_DOCENTE),
                this.getCasestatus(tbOrientacaoMestrado.statusOrientacao,
                    ↳ StatusCurso.EM_ANDAMENTO).as(MESTRADO_ANDAMENTO),
                this.getCasestatus(tbOrientacaoMestrado.statusOrientacao,
                    ↳ StatusCurso.CONCLUIDO).as(MESTRADO_CONCLUIDO));
    }
}

private ListSubQuery<Tuple> getSubUnionDoutorado(DocenteFilterDTO filter) {
    List<ListSubQuery<Tuple>> subs = new ArrayList<>();
    subs.add(this.getSubQueryOrientacaoDoutorado(filter, true));
    subs.add(this.getSubQueryOrientacaoDoutorado(filter, false));

    CollectionExpressionBase<?, List<Tuple>> union =
        ↳ this.queryFactory.subQuery().unionAll(subs);

    SimplePath<Void> unionPath = Expressions.path(Void.class, "union_doutorado");
    NumberPath<Long> idDocente = Expressions.numberPath(Long.class, unionPath,
        ↳ ID_DOCENTE);
    NumberPath<Long> andamento = Expressions.numberPath(Long.class, unionPath,

```

```

        ↪ DOUTORADO_ANDAMENTO);
    NumberPath<Long> concluido = Expressions.numberPath(Long.class, unionPath,
        ↪ DOUTORADO_CONCLUIDO);
    SQLSubQuery query =
        ↪ this.queryFactory.subQuery().from(union.as(unionPath.getMetadata().getName()));
    query.groupBy(idDocente);

    return query.list(idDocente.as(ID_DOCENTE),
        andamento.sum(),as(DOUTORADO_ANDAMENTO),
        concluido.sum(),as(DOUTORADO_CONCLUIDO));
}

private ListSubQuery<Tuple> getSubQueryOrientacaoDoutorado(DocenteFilterDTO filter,
    ↪ boolean orientador) {
    SQLSubQuery subQuery =
        ↪ this.queryFactory.subQuery().from(tbOrientacaoDoutorado);
    subQuery.where(tbOrientacaoDoutorado.idPrograma.eq(filter.getProgramaId()));
    if (filter.hasAnoInicioOrientacao() || filter.hasAnoFimOrientacao()) {
        subQuery.where(QueryUtils.overlappingInterval(
            tbOrientacaoDoutorado.anoInicio,
            tbOrientacaoDoutorado.anoFim,
            filter.getAnoInicioOrientacao(),
            filter.getAnoFimOrientacao()));
    }

    subQuery.where(tbOrientacaoDoutorado.deletado.eq(false));
    if (orientador) {
        subQuery.where(tbOrientacaoDoutorado.orientadorAtivo.eq(true));
        return
            ↪ subQuery.list(tbOrientacaoDoutorado.idOrientador.as(ID_DOCENTE),
                this.getCasestatus(tbOrientacaoDoutorado.statusOrientacao,
                    ↪ StatusCurso.EM_ANDAMENTO).as(DOUTORADO_ANDAMENTO),
                this.getCasestatus(tbOrientacaoDoutorado.statusOrientacao,
                    ↪ StatusCurso.CONCLUIDO).as(DOUTORADO_CONCLUIDO));
    } else {
        subQuery.where(tbOrientacaoDoutorado.coorientadorAtivo.eq(true));
        return
            ↪ subQuery.list(tbOrientacaoDoutorado.idCoorientador.as(ID_DOCENTE),
                this.getCasestatus(tbOrientacaoDoutorado.statusOrientacao,
                    ↪ StatusCurso.EM_ANDAMENTO).as(DOUTORADO_ANDAMENTO),
                this.getCasestatus(tbOrientacaoDoutorado.statusOrientacao,
                    ↪ StatusCurso.CONCLUIDO).as(DOUTORADO_CONCLUIDO));
    }
}

private NumberExpression<Long> getCaseStatus(StringPath statusPath, StatusCurso
    ↪ statusCurso) {
    return new
        ↪ CaseBuilder().when(statusPath.eq(statusCurso.name())).then(1L).otherwise(0L);
}

void applyFilter(DocenteFilterDTO filter, SQLQuery query) {
    query.where(tbPessoa.idPrograma.eq(filter.getProgramaId()));

    if (filter.hasNome()) {
        query.where(tbPessoa.nomeFiltro.like(QueryUtils.like(filter.getNome())));
    }

    List<ListSubQuery<Long>> subs = new ArrayList<>();
    if (filter.hasArea()) {
        subs.add(this.getSubQueryFormacaoMestrado(filter));
    }
    if (filter.hasPaisesDoutoradoIds() || filter.hasInstsDoutoradoIds() || filter.hasArea())
        ↪ {
        subs.add(this.getSubQueryFormacaoDoutorado(filter));
    }
    if (filter.hasPaisesPosDoutoradoIds() || filter.hasInstsPosDoutoradoIds() ||
        ↪ filter.hasArea()) {
        subs.add(this.getSubQueryFormacaoPosDoutorado(filter));
    }
    if (!subs.isEmpty()) {
        NumberPath<Long> idUnionFormacoesDiscente =
            ↪ Expressions.numberPath(Long.class, Expressions.path(Void.class,
            ↪ "union_formacoes"), ID_DISCENTE);
        CollectionExpressionBase<?, List<Long>> union =
            ↪ this.queryFactory.subQuery().union(subs);
        ListSubQuery<Long> list =
            ↪ this.queryFactory.subQuery().from(union.as("union_formacoes")).list(idUnionFormacoesDiscente);

        SimplePath<Void> formacoes = Expressions.path(Void.class, "formacoes");
        NumberPath<Long> idFormacoesDiscente =
            ↪ Expressions.numberPath(Long.class, formacoes, ID_DISCENTE);
    }
}

```

```

        query.join(list,
            ↪ formacoes).on(tbDiscente.idDiscente.eq(idFormacoesDiscente));
    }

    query.join(this.docenteQuery.getSubPertencePrograma(filter.getProgramaId()),
        ↪ viewPertencePrograma)
        .on(viewPertencePrograma.idDocente.eq(tbDocente.idDocente));
}

private ListSubQuery<Long> getSubQueryFormacaoMestrado(DocenteFilterDTO filter) {
    QTbOrientacaoMestrado tbOrientacaoMestrado = new
        ↪ QTbOrientacaoMestrado("subMestrado");
    SQLSubQuery subQuery =
        ↪ this.queryFactory.subQuery().from(tbOrientacaoMestrado);

    subQuery.where(tbOrientacaoMestrado.idPrograma.eq(filter.getProgramaId()));
    if (filter.hasArea()) {
        QTbAreaConhecimento qAreaUm = new
            ↪ QTbAreaConhecimento("subMestradoArea1");
        subQuery.leftJoin(qAreaUm)
            .on(tbOrientacaoMestrado.idAreaConhecimentoUm.eq(qAreaUm.idAreaConhecimentoUm));

        QTbAreaConhecimento qAreaDois = new
            ↪ QTbAreaConhecimento("subMestradoArea2");
        subQuery.leftJoin(qAreaDois)
            .on(tbOrientacaoMestrado.idAreaConhecimentoDois.eq(qAreaDois.idAreaConhecimentoDois));

        QTbAreaConhecimento qAreaTres = new
            ↪ QTbAreaConhecimento("subMestradoArea3");
        subQuery.leftJoin(qAreaTres)
            .on(tbOrientacaoMestrado.idAreaConhecimentoTres.eq(qAreaTres.idAreaConhecimentoTres));

        subQuery.where(ExpressionUtils.anyOf(qAreaUm.idAreaConhecimento.isNotNull(),
            ↪ qAreaDois.idAreaConhecimento.isNotNull(),
            ↪ qAreaTres.idAreaConhecimento.isNotNull()));
    }
    return subQuery.list(tbOrientacaoMestrado.idDiscente.as(ID_DISCENTE));
}

private ListSubQuery<Long> getSubQueryFormacaoDoutorado(DocenteFilterDTO filter) {
    QTbOrientacaoDoutorado tbOrientacaoDoutorado = new
        ↪ QTbOrientacaoDoutorado("subDoutorado");
    SQLSubQuery subQuery =
        ↪ this.queryFactory.subQuery().from(tbOrientacaoDoutorado);

    subQuery.where(tbOrientacaoDoutorado.idPrograma.eq(filter.getProgramaId()));
    if (filter.hasInstsDoutoradoIds()) {
        subQuery.where(tbOrientacaoDoutorado.idInstituicao.in(filter.getInstsDoutoradoIds()));
    }
    if (filter.hasPaisesDoutoradoIds()) {
        QTbInstituicao qInstituicao = new QTbInstituicao("subDoutoradoInst");
        subQuery.join(qInstituicao).on(qInstituicao.idInstituicao.eq(tbOrientacaoDoutorado.idInstituicao));
        subQuery.where(qInstituicao.idPais.in(filter.getPaisesDoutoradoIds()));
    }
    if (filter.hasArea()) {
        QTbAreaConhecimento qAreaUm = new
            ↪ QTbAreaConhecimento("subDoutoradoArea1");
        subQuery.leftJoin(qAreaUm)
            .on(tbOrientacaoDoutorado.idAreaConhecimentoUm.eq(qAreaUm.idAreaConhecimentoUm));

        QTbAreaConhecimento qAreaDois = new
            ↪ QTbAreaConhecimento("subDoutoradoArea2");
        subQuery.leftJoin(qAreaDois)
            .on(tbOrientacaoDoutorado.idAreaConhecimentoDois.eq(qAreaDois.idAreaConhecimentoDois)
                .and(qAreaDois.areaConhecimentoFiltro.like(QueryUtils.likeAreaConhecimento)));

        QTbAreaConhecimento qAreaTres = new
            ↪ QTbAreaConhecimento("subDoutoradoArea3");
        subQuery.leftJoin(qAreaTres)
            .on(tbOrientacaoDoutorado.idAreaConhecimentoTres.eq(qAreaTres.idAreaConhecimentoTres)
                .and(qAreaTres.areaConhecimentoFiltro.like(QueryUtils.likeAreaConhecimento)));

        subQuery.where(ExpressionUtils.anyOf(qAreaUm.idAreaConhecimento.isNotNull(),
            ↪ qAreaDois.idAreaConhecimento.isNotNull(),
            ↪ qAreaTres.idAreaConhecimento.isNotNull()));
    }
    return subQuery.list(tbOrientacaoDoutorado.idDiscente.as(ID_DISCENTE));
}

private ListSubQuery<Long> getSubQueryFormacaoPosDoutorado(DocenteFilterDTO filter)
    ↪ {
    QTbOrientacaoPosDoutorado qPosDoutorado = new

```

```

        ↪ QTbOrientacaoPosDoutorado("subPosDoutorado");
        SQLSubQuery subQuery = this.queryFactory.subQuery().from(qPosDoutorado);

        subQuery.where(qPosDoutorado.idPrograma.eq(filter.getProgramaId()));
        if (filter.hasInstsPosDoutoradoIds()) {
            subQuery.where(qPosDoutorado.idInstituicao.in(filter.getInstsPosDoutoradoIds()));
        }
        if (filter.hasPaisesPosDoutoradoIds()) {
            QTbInstituicao qInstituicao = new QTbInstituicao("subPosDoutoradoInst");
            subQuery.join(qInstituicao).on(qInstituicao.idInstituicao.eq(qPosDoutorado.idInstituicao));
            subQuery.where(qInstituicao.idPais.in(filter.getPaisesPosDoutoradoIds()));
        }
        if (filter.hasArea()) {
            QTbAreaConhecimento qAreaUm = new
                ↪ QTbAreaConhecimento("subPosDoutoradoArea1");
            subQuery.leftJoin(qAreaUm)
                .on(qPosDoutorado.idAreaConhecimentoUm.eq(qAreaUm.idAreaConhecimento).and(q

            QTbAreaConhecimento qAreaDois = new
                ↪ QTbAreaConhecimento("subPosDoutoradoArea2");
            subQuery.leftJoin(qAreaDois)
                .on(qPosDoutorado.idAreaConhecimentoDois.eq(qAreaDois.idAreaConhecimento).and(q

            QTbAreaConhecimento qAreaTres = new
                ↪ QTbAreaConhecimento("subPosDoutoradoArea3");
            subQuery.leftJoin(qAreaTres)
                .on(qPosDoutorado.idAreaConhecimentoTres.eq(qAreaTres.idAreaConhecimento).and(q

            subQuery.where(ExpressionUtils.anyOf(qAreaUm.idAreaConhecimento.isNotNull(),
                ↪ qAreaDois.idAreaConhecimento.isNotNull(),
                ↪ qAreaTres.idAreaConhecimento.isNotNull()));
        }
        }
        return subQuery.list(qPosDoutorado.idDiscente.as(ID_DISCENTE));
    }
}
package alpc.ufsc.pessoa.docente.query;

import static alpc.ufsc.querydsl.sql.QTbOrientacaoDoutorado.tbOrientacaoDoutorado;
import static alpc.ufsc.querydsl.sql.QTbOrientacaoMestrado.tbOrientacaoMestrado;

import java.util.ArrayList;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;

import com.mysema.query.sql.SQLQuery;
import com.mysema.query.sql.SQLQueryFactory;
import com.mysema.query.sql.SQLSubQuery;
import com.mysema.query.support.Expressions;
import com.mysema.query.types.expr.CaseBuilder;
import com.mysema.query.types.expr.CollectionExpressionBase;
import com.mysema.query.types.expr.NumberExpression;
import com.mysema.query.types.path.NumberPath;
import com.mysema.query.types.path.SimplePath;
import com.mysema.query.types.query.ListSubQuery;

import alpc.ufsc.entity.domain.produtividade.Categoria;
import alpc.ufsc.entity.domain.produtividade.Modalidade;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.docente.dto.DocenteFilterDTO;
import alpc.ufsc.pessoa.docente.dto.DocenteHeaderDTO;
import alpc.ufsc.pessoa.docente.dto.MDocenteHeaderDTO;
import alpc.ufsc.querydsl.sql.QTbDiscente;
import alpc.ufsc.querydsl.sql.QTbDocente;
import alpc.ufsc.querydsl.sql.QTbPessoa;
import alpc.ufsc.util.querydsl.Select;

@Component
public class DocenteHeaderQuery {

    private static final String ID_DOCENTE = "id_docente";
    private QTbDocente qDocente = QTbDocente.tbDocente;
    private QTbPessoa qPessoa = QTbPessoa.tbPessoa;
    private QTbDiscente qDiscente = QTbDiscente.tbDiscente;

    @Autowired
    private SQLQueryFactory queryFactory;

    @Autowired
    private DocenteRowHeaderQuery rowHeaderQuery;

```

```

public DocenteHeaderDTO getHeader(DocenteFilterDTO filter) {
    SQLQuery query = this.queryFactory.query().from(this.qDocente);
    query.join(this.qPessoa).on(this.qDocente.idPessoa.eq(this.qPessoa.idPessoa));
    query.join(this.qDiscente).on(this.qPessoa.idPessoa.eq(this.qDiscente.idPessoa));

    SimplePath<Void> mestrado = Expressions.path(Void.class, "mestrado");
    NumberPath<Long> mestradoIdDocente = Expressions.numberPath(Long.class,
        ↪ mestrado, ID_DOCENTE);
    query.leftJoin(this.getSubUnionMestrado(filter),
        ↪ mestrado).on(mestradoIdDocente.eq(this.qDocente.idDocente));

    SimplePath<Void> doutorado = Expressions.path(Void.class, "doutorado");
    NumberPath<Long> doutoradoIdDocente = Expressions.numberPath(Long.class,
        ↪ doutorado, ID_DOCENTE);
    query.leftJoin(this.getSubUnionDoutorado(filter),
        ↪ doutorado).on(doutoradoIdDocente.eq(this.qDocente.idDocente));

    this.rowHeaderQuery.applyFilter(filter, query);

    MDocenteHeaderDTO meta = MDocenteHeaderDTO.meta;
    Select<DocenteHeaderDTO> select = new Select<>(DocenteHeaderDTO.class);

    select.as(this.getCaseProdutividade(Modalidade.PESQUISA, Categoria.SR),
        ↪ meta.numeroBolsistasPQsr);
    select.as(this.getCaseProdutividade(Modalidade.PESQUISA, Categoria.UM_A),
        ↪ meta.numeroBolsistasPQ1A);
    select.as(this.getCaseProdutividade(Modalidade.PESQUISA, Categoria.UM_B),
        ↪ meta.numeroBolsistasPQ1B);
    select.as(this.getCaseProdutividade(Modalidade.PESQUISA, Categoria.UM_C),
        ↪ meta.numeroBolsistasPQ1C);
    select.as(this.getCaseProdutividade(Modalidade.PESQUISA, Categoria.UM_D),
        ↪ meta.numeroBolsistasPQ1D);
    select.as(this.getCaseProdutividade(Modalidade.PESQUISA, Categoria.DOIS),
        ↪ meta.numeroBolsistasPQ2);

    select.as(this.getCaseProdutividade(Modalidade.DESENVOLVIMENTO,
        ↪ Categoria.UM_A), meta.numeroBolsistasDT1A);
    select.as(this.getCaseProdutividade(Modalidade.DESENVOLVIMENTO,
        ↪ Categoria.UM_B), meta.numeroBolsistasDT1B);
    select.as(this.getCaseProdutividade(Modalidade.DESENVOLVIMENTO,
        ↪ Categoria.UM_C), meta.numeroBolsistasDT1C);
    select.as(this.getCaseProdutividade(Modalidade.DESENVOLVIMENTO,
        ↪ Categoria.UM_D), meta.numeroBolsistasDT1D);
    select.as(this.getCaseProdutividade(Modalidade.DESENVOLVIMENTO,
        ↪ Categoria.DOIS), meta.numeroBolsistasDT2);

    select.as(new
        ↪ CaseBuilder().when(mestradoIdDocente.isNotNull()).then(1L).otherwise(0L).sum(),
        ↪ meta.numeroOrientadoresMestrado);
    select.as(new
        ↪ CaseBuilder().when(doutoradoIdDocente.isNotNull()).then(1L).otherwise(0L).sum(),
        ↪ meta.numeroOrientadoresDoutorado);

    return query.uniqueResult(select);
}

private NumberExpression<Long> getCaseProdutividade(Modalidade modalidade, Categoria
    ↪ categoria) {
    return new
        ↪ CaseBuilder().when(this.qDocente.modalidade.eq(modalidade.getSigla()).and(this.qDocente
}

private ListSubQuery<Long> getSubUnionMestrado(DocenteFilterDTO filter) {
    List<ListSubQuery<Long>> subs = new ArrayList<>();
    subs.add(this.getSubQueryOrientacaoMestrado(filter, true));
    subs.add(this.getSubQueryOrientacaoMestrado(filter, false));

    CollectionExpressionBase<?, List<Long>> union =
        ↪ this.queryFactory.subQuery().union(subs);

    SimplePath<Void> unionM = Expressions.path(Void.class, "union_mestrado");
    NumberPath<Long> idDocenteM = Expressions.numberPath(Long.class, unionM,
        ↪ ID_DOCENTE);
    return
        ↪ this.queryFactory.subQuery().from(union.as(unionM.getMetadata().getName()).list(idDoc
}

private ListSubQuery<Long> getSubQueryOrientacaoMestrado(DocenteFilterDTO filter,
    ↪ boolean orientador) {
    SQLSubQuery subQuery =
        ↪ this.queryFactory.subQuery().from(tbOrientacaoMestrado);
    subQuery.where(tbOrientacaoMestrado.idPrograma.eq(filter.getProgramaId()));
}

```

```

        if (filter.hasAnoInicioOrientacao() || filter.hasAnoFimOrientacao()) {
            subQuery.where(QueryUtils.overlappingInterval(
                tbOrientacaoMestrado.anoInicio,
                tbOrientacaoMestrado.anoFim,
                filter.getAnoInicioOrientacao(),
                filter.getAnoFimOrientacao()));
        }
        subQuery.where(tbOrientacaoMestrado.deletado.eq(false));
        if (orientador) {
            subQuery.where(tbOrientacaoMestrado.orientadorAtivo.eq(true));
            return ↪ subQuery.list(tbOrientacaoMestrado.idOrientador.as(ID_DOCENTE));
        } else {
            subQuery.where(tbOrientacaoMestrado.coorientadorAtivo.eq(true));
            return ↪ subQuery.list(tbOrientacaoMestrado.idCoorientador.as(ID_DOCENTE));
        }
    }

    private ListSubQuery<Long> getSubUnionDoutorado(DocenteFilterDTO filter) {
        List<ListSubQuery<Long>> subs = new ArrayList<>();
        subs.add(this.getSubQueryOrientacaoDoutorado(filter, true));
        subs.add(this.getSubQueryOrientacaoDoutorado(filter, false));

        CollectionExpressionBase<?, List<Long>> union =
            ↪ this.queryFactory.subQuery().union(subs);

        SimplePath<Void> unionD = Expressions.path(Void.class, "union_doutorado");
        NumberPath<Long> idDocenteD = Expressions.numberPath(Long.class, unionD,
            ↪ ID_DOCENTE);
        return ↪ this.queryFactory.subQuery().from(union.as(unionD.getMetadata().getName())).list(idDocenteD);
    }

    private ListSubQuery<Long> getSubQueryOrientacaoDoutorado(DocenteFilterDTO filter,
        ↪ boolean orientador) {
        SQLSubQuery subQuery =
            ↪ this.queryFactory.subQuery().from(tbOrientacaoDoutorado);
        subQuery.where(tbOrientacaoDoutorado.idPrograma.eq(filter.getProgramaId()));
        if (filter.hasAnoInicioOrientacao() || filter.hasAnoFimOrientacao()) {
            subQuery.where(QueryUtils.overlappingInterval(
                tbOrientacaoDoutorado.anoInicio,
                tbOrientacaoDoutorado.anoFim,
                filter.getAnoInicioOrientacao(),
                filter.getAnoFimOrientacao()));
        }
        subQuery.where(tbOrientacaoDoutorado.deletado.eq(false));
        if (orientador) {
            subQuery.where(tbOrientacaoDoutorado.orientadorAtivo.eq(true));
            return ↪ subQuery.list(tbOrientacaoDoutorado.idOrientador.as(ID_DOCENTE));
        } else {
            subQuery.where(tbOrientacaoDoutorado.coorientadorAtivo.eq(true));
            return ↪ subQuery.list(tbOrientacaoDoutorado.idCoorientador.as(ID_DOCENTE));
        }
    }
}
package alpc.ufsc.pessoa.docente.query;

import static alpc.ufsc.querydsl.sql.QViewPertencePrograma.viewPertencePrograma;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;

import com.mysema.query.sql.SQLQueryFactory;
import com.mysema.query.types.query.ListSubQuery;

@Component
public class DocentePertenceProgramaQuery {

    @Autowired
    private SQLQueryFactory queryFactory;

    public ListSubQuery<Long> getSubPertencePrograma(Long idPrograma) {
        return this.queryFactory.subQuery().from(viewPertencePrograma)
            .where(viewPertencePrograma.idPrograma.eq(idPrograma))
            .list(viewPertencePrograma.idDocente);
    }
}
package alpc.ufsc.pessoa.docente.query;

```



```

import static alpc.ufsc.querydsl.sql.QRIAutorProducao.rIAutorProducao;
import static alpc.ufsc.querydsl.sql.QRIDocenteProducao.rIDocenteProducao;
import static alpc.ufsc.querydsl.sql.QTbPessoa.tbPessoa;
import static alpc.ufsc.querydsl.sql.QTbProducao.tbProducao;

import org.springframework.beans.factory.annotation.Autowired;

import com.mysema.query.sql.SQLQuery;
import com.mysema.query.sql.SQLQueryFactory;

public class DocenteProducaoQuery {

    @Autowired
    private SQLQueryFactory queryFactory;

    public void getProducao() {
        SQLQuery query = this.queryFactory.query().from(tbProducao);
        query.join(rIDocenteProducao).on(tbProducao.idProducao.eq(rIDocenteProducao.idProducao));
        query.join(rIAutorProducao).on(tbProducao.idProducao.eq(rIAutorProducao.idProducao));
        query.join(tbPessoa).on(rIAutorProducao.idAutor.eq(tbPessoa.idPessoa));
    }
}

package alpc.ufsc.pessoa.docente.dto;

import lombok.Getter;
import lombok.Setter;
import br.ufsc.bridge.metafy.Metafy;

@Getter
@Setter
@Metafy
public class TipoBolsaDTO {
    private String modalidade;
    private String categoria;
}

package alpc.ufsc.pessoa.docente.dto;

import lombok.Getter;
import lombok.Setter;

import alpc.ufsc.entity.domain.produtividade.Categoria;
import alpc.ufsc.entity.domain.produtividade.Modalidade;

@Getter
@Setter
public class DocenteProdutividadeDTO {
    private Long id;
    private Modalidade modalidade;
    private Categoria categoria;
}

package alpc.ufsc.pessoa.docente.dto;

import lombok.Getter;
import lombok.Setter;
import br.ufsc.bridge.metafy.Metafy;

@Setter
@Getter
@Metafy
public class DocenteRowHeaderDTO {

    private Long id;
    private Long discenteId;

    private String nome;
    private TipoBolsaDTO tipoBolsa;

    private Long numeroDiscentesMestradoAndamento;
    private Long numeroDiscentesDoutoradoAndamento;
    private Long totalDiscentesAndamento = 0L;

    private Long numeroDiscentesMestradoConcluido;
    private Long numeroDiscentesDoutoradoConcluido;
    private Long totalDiscentesConcluido = 0L;

    public void setNumeroDiscentesMestradoAndamento(Long
        ↪ numeroDiscentesMestradoAndamento) {
        this.numeroDiscentesMestradoAndamento = numeroDiscentesMestradoAndamento;
        this.totalDiscentesAndamento += numeroDiscentesMestradoAndamento;
    }
}

```

```

public void setNumeroDiscentesDoutoradoAndamento(Long
    ↪ numeroDiscentesDoutoradoAndamento) {
    this.numeroDiscentesDoutoradoAndamento = numeroDiscentesDoutoradoAndamento;
    this.totalDiscentesAndamento += numeroDiscentesDoutoradoAndamento;
}

public void setNumeroDiscentesMestradoConcluido(Long
    ↪ numeroDiscentesMestradoConcluido) {
    this.numeroDiscentesMestradoConcluido = numeroDiscentesMestradoConcluido;
    this.totalDiscentesConcluido += numeroDiscentesMestradoConcluido;
}

public void setNumeroDiscentesDoutoradoConcluido(Long
    ↪ numeroDiscentesDoutoradoConcluido) {
    this.numeroDiscentesDoutoradoConcluido = numeroDiscentesDoutoradoConcluido;
    this.totalDiscentesConcluido += numeroDiscentesDoutoradoConcluido;
}

public boolean hasDiscentelId() {
    return this.discentelId != null;
}
}package alpc.ufsc.pessoa.docente.dto;

import java.util.List;

import lombok.Getter;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;
import org.springframework.util.CollectionUtils;

@Getter
@Setter
public class DocenteFilterDTO {

    private String nome;
    private String area;

    private Long programaId;

    private Integer anoInicioOrientacao;
    private Integer anoFimOrientacao;

    private List<Long> paisesDoutoradoIds;
    private List<Long> instsDoutoradoIds;

    private List<Long> paisesPosDoutoradoIds;
    private List<Long> instsPosDoutoradoIds;

    public boolean hasNome() {
        return !StringUtils.isBlank(this.nome);
    }

    public boolean hasArea() {
        return !StringUtils.isBlank(this.area);
    }

    public boolean hasAnoInicioOrientacao() {
        return this.anoInicioOrientacao != null;
    }

    public boolean hasAnoFimOrientacao() {
        return this.anoFimOrientacao != null;
    }

    public boolean hasPaisesDoutoradoIds() {
        return !CollectionUtils.isEmpty(this.paisesDoutoradoIds);
    }

    public boolean hasInstsDoutoradoIds() {
        return !CollectionUtils.isEmpty(this.instsDoutoradoIds);
    }

    public boolean hasPaisesPosDoutoradoIds() {
        return !CollectionUtils.isEmpty(this.paisesPosDoutoradoIds);
    }

    public boolean hasInstsPosDoutoradoIds() {
        return !CollectionUtils.isEmpty(this.instsPosDoutoradoIds);
    }
}
package alpc.ufsc.pessoa.docente.dto;

```

```

import java.util.List;

import lombok.Getter;
import lombok.Setter;

import com.fasterxml.jackson.annotation.JsonInclude;
import com.fasterxml.jackson.annotation.JsonInclude.Include;

import alpc.ufsc.pessoa.orientacao.common.dto.FormacaoRowDTO;
import alpc.ufsc.pessoa.producao.dto.QuantidadeRowDTO;

@Setter
@Getter
@JsonInclude(Include.NON_EMPTY)
public class DocenteRowBodyDTO {
    private List<QuantidadeRowDTO> producoes;
    private List<FormacaoRowDTO> formacoes;
}
package alpc.ufsc.pessoa.docente.dto;

import lombok.Getter;
import lombok.Setter;
import br.ufsc.bridge.metafy.Metafy;

@Setter
@Getter
@Metafy
public class DocenteHeaderDTO {
    private Long numeroBolsistasPQ5r;
    private Long numeroBolsistasPQ1A;
    private Long numeroBolsistasPQ1B;
    private Long numeroBolsistasPQ1C;
    private Long numeroBolsistasPQ1D;
    private Long numeroBolsistasPQ2;

    private Long numeroBolsistasDT1A;
    private Long numeroBolsistasDT1B;
    private Long numeroBolsistasDT1C;
    private Long numeroBolsistasDT1D;
    private Long numeroBolsistasDT2;

    private Long numeroOrientadoresMestrado;
    private Long numeroOrientadoresDoutorado;
}
package alpc.ufsc.pessoa;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.apache.lucene.search.BooleanClause.Occur;
import org.apache.lucene.search.BooleanQuery;
import org.hibernate.search.SearchFactory;
import org.hibernate.search.errors.EmptyQueryException;
import org.hibernate.search.jpa.FullTextEntityManager;
import org.hibernate.search.jpa.FullTextQuery;
import org.hibernate.search.jpa.Search;
import org.hibernate.search.query.dsl.QueryBuilder;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import alpc.ufsc.entity.pessoa.PessoaEntity;
import alpc.ufsc.pessoa.orientacao.common.dto.PessoaSearchDto;

@Service
@Transactional(propagation = Propagation.SUPPORTS, noRollbackFor =
    ↳ EmptyQueryException.class)
public class PessoaSearchService {

    private static final String ID = "id";
    private static final String NOME = "nome";
    private static final String PROGRAMA_ID = "programa.id";

    @PersistenceContext
    private EntityManager em;

    public Long searchPessoa(PessoaSearchDto pessoaSearch) {
        FullTextEntityManager search = Search.getFullTextEntityManager(this.em);
        search.flushToIndexes();
    }
}

```

```

SearchFactory searchFactory = search.getSearchFactory();
QueryBuilder qb =
    ↪ searchFactory.buildQueryBuilder().forEntity(PessoaEntity.class).get();

BooleanQuery query = new BooleanQuery();
query.add(qb.keyword().onField(PROGRAMA_ID).matching(pessoaSearch.getProgramaId()).createQuery()
    ↪ Occur.MUST);
query.add(qb.keyword().onField(NOME).matching(pessoaSearch.getNome()).createQuery(),
    ↪ Occur.MUST);

FullTextQuery fullTextQuery = search.createFullTextQuery(query,
    ↪ PessoaEntity.class);

fullTextQuery.setProjection(this.getProjection());
fullTextQuery.setMaxResults(5);

List<?> result = fullTextQuery.getResultList();
for (Object object : result) {
    Object[] fields = (Object[]) object;
    if (pessoaSearch.similar(new PessoaSearchDto((String) fields[1]))) {
        ↪ return (Long) fields[0];
    }
}
return null;
}

private String[] getProjection() {
    String[] p = { ID, NOME };
    return p;
}
}

package alpc.ufsc.pessoa.discente;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.jpa.impl.JPAQuery;

import alpc.ufsc.entity.pessoa.PessoaEntity;
import alpc.ufsc.entity.pessoa.discente.DiscenteEntity;
import alpc.ufsc.entity.pessoa.discente.QDiscenteEntity;
import alpc.ufsc.pessoa.PessoaService;
import alpc.ufsc.pessoa.discente.dto.DiscenteFilterDTO;
import alpc.ufsc.pessoa.discente.dto.DiscenteHeaderDTO;
import alpc.ufsc.pessoa.discente.dto.DiscenteInstituicaoProcedenciaFilterDTO;
import alpc.ufsc.pessoa.discente.dto.DiscenteInstituicaoProcedenciaRowDTO;
import alpc.ufsc.pessoa.discente.dto.DiscenteRowBodyDTO;
import alpc.ufsc.pessoa.discente.dto.DiscenteRowHeaderDTO;
import alpc.ufsc.pessoa.discente.query.DiscenteHeaderQuery;
import alpc.ufsc.pessoa.discente.query.DiscenteInstituicaoProcedenciaQuery;
import alpc.ufsc.pessoa.discente.query.DiscenteRowHeaderQuery;
import alpc.ufsc.pessoa.dto.PessoaDto;
import alpc.ufsc.pessoa.orientacao.common.dto.OrientacaoResultDto;
import alpc.ufsc.pessoa.orientacao.common.query.FormacaoRowQuery;
import alpc.ufsc.programa.dto.ImportProcessDto;

@Service
@Transactional
public class DiscenteService {

    @PersistenceContext
    private EntityManager em;

    @Autowired
    private PessoaService pessoaService;

    @Autowired
    private FormacaoRowQuery formacaoQuery;

    @Autowired
    private DiscenteRowHeaderQuery rowHeaderQuery;

    @Autowired
    private DiscenteHeaderQuery headerQuery;
}

```

```

@Autowired
private DiscenteInstituicaoProcedenciaQuery procedenciaQuery;

public DiscenteEntity save(PessoaDto pessoaDto, OrientacaoResultDto orientacaoResult,
    ↪ ImportProcessDto processDto) {
    DiscenteEntity discenteEntity = null;
    if (orientacaoResult.isSetDiscentelId()) {
        discenteEntity = this.em.getReference(DiscenteEntity.class,
            ↪ orientacaoResult.getDiscentelId());
    } else if (processDto != null && processDto.isSetDiscentelId()) {
        discenteEntity = this.em.getReference(DiscenteEntity.class,
            ↪ processDto.getDiscentelId());
    } else {
        discenteEntity = this.saveByPessoa(pessoaDto);
    }
    if (processDto != null) {
        processDto.setDiscentelId(discenteEntity.getId());
    }
    return discenteEntity;
}

private DiscenteEntity saveByPessoa(PessoaDto pessoaDto) {
    Long pessoaId = this.pessoaService.findOtherwiseSave(pessoaDto);

    QDiscenteEntity qDiscente = QDiscenteEntity.discenteEntity;
    DiscenteEntity discenteEntity = new
        ↪ JPAQuery(this.em).from(qDiscente).where(qDiscente.pessoa().id.eq(pessoaId)).uniqueResult();

    if (discenteEntity == null) {
        discenteEntity = new DiscenteEntity();
        discenteEntity.setPessoa(this.em.getReference(PessoaEntity.class, pessoaId));
        this.em.persist(discenteEntity);
    }
    return discenteEntity;
}

public DiscenteHeaderDTO getHeader(DiscenteFilterDTO filter) {
    return this.headerQuery.getHeader(filter);
}

public Page<DiscenteRowHeaderDTO> getPage(DiscenteFilterDTO filter, Pageable
    ↪ pageable) {
    return this.rowHeaderQuery.getPage(filter, pageable);
}

public DiscenteRowBodyDTO getDiscenteRowBody(Long idDiscente) {
    return new DiscenteRowBodyDTO(this.formacaoQuery.getFormacoes(idDiscente,
        ↪ true, false));
}

public Page<DiscenteInstituicaoProcedenciaRowDTO>
    ↪ getPage(DiscenteInstituicaoProcedenciaFilterDTO filter, Pageable pageable) {
    return this.procedenciaQuery.getPage(filter, pageable);
}
}
package alpc.ufsc.pessoa.discente;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import org.springframework.security.access.annotation.Secured;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;

import alpc.ufsc.login.authority.Authority;
import alpc.ufsc.login.user.CurrentUser;
import alpc.ufsc.login.user.dto.UserDetailsImpl;
import alpc.ufsc.pessoa.discente.dto.DiscenteFilterDTO;
import alpc.ufsc.pessoa.discente.dto.DiscenteHeaderDTO;
import alpc.ufsc.pessoa.discente.dto.DiscenteRowBodyDTO;
import alpc.ufsc.pessoa.discente.dto.DiscenteRowHeaderDTO;

@RestController
@RequestMapping("/services/discente")
public class DiscenteRestController {

    @Autowired
    private DiscenteService service;

```

```

@Secured({ Authority.COORDENADOR, Authority.DOCENTE })
@RequestMapping(value = "header", method = RequestMethod.GET)
public @ResponseBody DiscenteHeaderDTO getHeader(@CurrentUser UserDetailsImpl user,
    ↪ DiscenteFilterDTO filter) {
    filter.setIdPrograma(user.getProgramaId());
    return this.service.getHeader(filter);
}

@Secured({ Authority.COORDENADOR, Authority.DOCENTE })
@RequestMapping(method = RequestMethod.GET)
public @ResponseBody Page<DiscenteRowHeaderDTO> getPage(@CurrentUser
    ↪ UserDetailsImpl user, DiscenteFilterDTO filter, Pageable pageable) {
    filter.setIdPrograma(user.getProgramaId());
    return this.service.getPage(filter, pageable);
}

@Secured({ Authority.COORDENADOR, Authority.DOCENTE })
@RequestMapping(value = "formacoes/{id}", method = RequestMethod.GET)
public @ResponseBody DiscenteRowBodyDTO getFormacoes(@PathVariable Long id) {
    return this.service.getDiscenteRowBody(id);
}
}
package alpc.ufsc.pessoa.discente.query;

import static alpc.ufsc.pessoa.discente.dto.MDiscenteInstituicaoProcedenciaRowDTO.meta;
import static alpc.ufsc.querydsl.sql.QTbGraduacao.tbGraduacao;
import static alpc.ufsc.querydsl.sql.QTbInstituicao.tbInstituicao;
import static alpc.ufsc.querydsl.sql.QTbOrientacaoDoutorado.tbOrientacaoDoutorado;
import static alpc.ufsc.querydsl.sql.QTbOrientacaoMestrado.tbOrientacaoMestrado;

import java.util.ArrayList;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageImpl;
import org.springframework.data.domain.Pageable;
import org.springframework.stereotype.Component;

import com.mysema.query.Tuple;
import com.mysema.query.sql.SQLQuery;
import com.mysema.query.sql.SQLQueryFactory;
import com.mysema.query.sql.SQLSubQuery;
import com.mysema.query.support.Expressions;
import com.mysema.query.types.expr.CollectionExpressionBase;
import com.mysema.query.types.path.NumberPath;
import com.mysema.query.types.path.SimplePath;
import com.mysema.query.types.query.ListSubQuery;

import alpc.ufsc.entity.domain.StatusOrientacao;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.discente.dto.DiscenteInstituicaoProcedenciaFilterDTO;
import alpc.ufsc.pessoa.discente.dto.DiscenteInstituicaoProcedenciaRowDTO;
import alpc.ufsc.util.querydsl.Select;

@Component
public class DiscenteInstituicaoProcedenciaQuery {

    private static final String ID_INST = "id_inst";
    private static final String ID_DISCENTE = "id_discente";

    @Autowired
    private SQLQueryFactory queryFactory;

    private static final SimplePath<Void> unionPro = Expressions.path(Void.class,
    ↪ "union_procedencia");
    private static final NumberPath<Long> idDiscenteP = Expressions.numberPath(Long.class,
    ↪ unionPro, ID_DISCENTE);
    private static final NumberPath<Long> idInstiP = Expressions.numberPath(Long.class,
    ↪ unionPro, ID_INST);

    public Page<DiscenteInstituicaoProcedenciaRowDTO>
    ↪ getPage(DiscenteInstituicaoProcedenciaFilterDTO filter, Pageable pageable) {
        SQLQuery query = this.queryFactory.query().from(tbInstituicao);
        ListSubQuery<Tuple> subProcedencia =
    ↪ this.getSubProcedencia(filter).list(idDiscenteP, idInstiP);
        query.join(subProcedencia, unionPro).on(tbInstituicao.idInstituicao.eq(idInstiP));
        if (filter.hasNome()) {
            query.where(tbInstituicao.nomeFiltro.like(QueryUtils.like(filter.getNome())));
        }

        query.groupBy(tbInstituicao.idInstituicao, tbInstituicao.nome);
    }
}

```

```

query.offset(pageable.getOffset());
query.limit(pageable.getPageSize());

query.orderBy(tbInstituicao.nomeFiltro.asc());

Select<DiscenteInstituicaoProcedenciaRowDTO> select = new
    Select<>>(DiscenteInstituicaoProcedenciaRowDTO.class);
select.as(tbInstituicao.idInstituicao, meta.id);
select.as(tbInstituicao.nome, meta.nome);
select.as(idDiscenteP.count(), meta.numeroDiscentes);

SQLQuery queryCount = this.queryFactory.query().from(tbInstituicao);
ListSubQuery<Long> subProcedenciaCount =
    this.getSubProcedencia(filter).distinct().list(idInstiP);
queryCount.join(subProcedenciaCount,
    unionPro.on(tbInstituicao.idInstituicao.eq(idInstiP));
if (filter.hasNome()) {
    queryCount.where(tbInstituicao.nomeFiltro.like(QueryUtils.like(filter.getNome())));
}

return new PageImpl<>(query.list(select), pageable, queryCount.count());
}

private SQLSubQuery getSubProcedencia(DiscenteInstituicaoProcedenciaFilterDTO filter) {
    List<ListSubQuery<Tuple>> subs = new ArrayList<>();
    subs.add(this.getSubProcedenciaMestrado(filter));
    subs.add(this.getSubProcedenciaDoutorado(filter));

    CollectionExpressionBase<?, List<Tuple>> union =
        this.queryFactory.subQuery().union(subs);

    return
        this.queryFactory.subQuery().from(union.as(unionPro.getMetadata()).getName());
}

private ListSubQuery<Tuple>
    getSubProcedenciaMestrado(DiscenteInstituicaoProcedenciaFilterDTO filter) {
        SQLSubQuery query = this.queryFactory.subQuery().from(tbOrientacaoMestrado);
        query.where(tbOrientacaoMestrado.idPrograma.eq(filter.getProgramaId()));
        query.where(tbOrientacaoMestrado.deletado.eq(false));
        query.where(tbOrientacaoMestrado.orientadorAtivo.eq(true).or(tbOrientacaoMestrado.coorientadorAtivo.eq(true)));
        if (filter.hasInicio() || filter.hasFim()) {
            query.where(QueryUtils.overlappingInterval(tbOrientacaoMestrado.anoInicio,
                tbOrientacaoMestrado.anoFim, filter.getInicio(), filter.getFim()));
        }

        ListSubQuery<Tuple> listGraduacao = this.getSubGraduacao(filter);
        query.join(listGraduacao,
            subGraduacao.on(tbOrientacaoMestrado.idDiscente.eq(idDiscenteSubG));
        return query.list(idDiscenteSubG, idInstiSubG);
    }

private ListSubQuery<Tuple>
    getSubProcedenciaDoutorado(DiscenteInstituicaoProcedenciaFilterDTO filter) {
        List<ListSubQuery<Tuple>> subs = new ArrayList<>();
        subs.add(this.getSubGraduacao(filter));
        subs.add(this.getSubMestrado(filter));
        CollectionExpressionBase<?, List<Tuple>> union =
            this.queryFactory.subQuery().union(subs);

        SimplePath<Void> gradMest = Expressions.path(Void.class, "union_grad_mest");
        NumberPath<Long> idDiscenteGM = Expressions.numberPath(Long.class,
            gradMest, ID_DISCENTE);
        NumberPath<Long> idInstiGM = Expressions.numberPath(Long.class, gradMest,
            ID_INSTI);
        SQLSubQuery subQuery =
            this.queryFactory.subQuery().from(union.as(gradMest.getMetadata()).getName());
        ListSubQuery<Tuple> listSubQuery = subQuery.list(idDiscenteGM, idInstiGM);

        SQLSubQuery query = this.queryFactory.subQuery().from(tbOrientacaoDoutorado);
        query.where(tbOrientacaoDoutorado.idPrograma.eq(filter.getProgramaId()));
        query.where(tbOrientacaoDoutorado.deletado.eq(false));
        query.where(tbOrientacaoDoutorado.orientadorAtivo.eq(true).or(tbOrientacaoDoutorado.coorientadorAtivo.eq(true)));
        if (filter.hasInicio() || filter.hasFim()) {
            query.where(QueryUtils.overlappingInterval(tbOrientacaoDoutorado.anoInicio,
                tbOrientacaoDoutorado.anoFim, filter.getInicio(), filter.getFim()));
        }

        SimplePath<Void> joinGradMest = Expressions.path(Void.class,
            "join_grad_mest");
        NumberPath<Long> idDiscenteJGM = Expressions.numberPath(Long.class,

```

```

        ↪ joinGradMest, ID_DISCENTE);
    NumberPath<Long> idInstiJGM = Expressions.numberPath(Long.class,
        ↪ joinGradMest, ID_INST);
    query.join(listSubQuery,
        ↪ joinGradMest).on(tbOrientacaoDoutorado.idDiscente.eq(idDiscenteJGM));

    return query.list(idDiscenteJGM, idInstiJGM);
}

private static final SimplePath<Void> subMestrado = Expressions.path(Void.class,
    ↪ "sub_mestrado");
private static final NumberPath<Long> idDiscenteSubM =
    ↪ Expressions.numberPath(Long.class, subMestrado, ID_DISCENTE);
private static final NumberPath<Long> idInstiSubM = Expressions.numberPath(Long.class,
    ↪ subMestrado, ID_INST);

private ListSubQuery<Tuple> getSubMestrado(DiscenteInstituicaoProcedenciaFilterDTO
    ↪ filter) {
    SQLSubQuery query = this.queryFactory.subQuery().from(tbOrientacaoMestrado);
    query.where(tbOrientacaoMestrado.idPrograma.eq(filter.getProgramaId()));
    query.where(tbOrientacaoMestrado.deletado.eq(false));
    query.where(tbOrientacaoMestrado.statusOrientacao.eq(StatusOrientacao.CONCLUIDO.name()));
    query.groupBy(tbOrientacaoMestrado.idDiscente,
        ↪ tbOrientacaoMestrado.idInstituicao);

    return query.list(
        tbOrientacaoMestrado.idDiscente.as(idDiscenteSubM.getMetadata().getName()),
        tbOrientacaoMestrado.idInstituicao.as(idInstiSubM.getMetadata().getName()),
        tbOrientacaoMestrado.anoFim.max());
}

private static final SimplePath<Void> subGraduacao = Expressions.path(Void.class,
    ↪ "sub_graduacao");
private static final NumberPath<Long> idDiscenteSubG =
    ↪ Expressions.numberPath(Long.class, subGraduacao, ID_DISCENTE);
private static final NumberPath<Long> idInstiSubG = Expressions.numberPath(Long.class,
    ↪ subGraduacao, ID_INST);

private ListSubQuery<Tuple> getSubGraduacao(DiscenteInstituicaoProcedenciaFilterDTO
    ↪ filter) {
    SQLSubQuery query = this.queryFactory.subQuery().from(tbGraduacao);
    query.where(tbGraduacao.idPrograma.eq(filter.getProgramaId()));
    query.where(tbGraduacao.deletado.eq(false));
    query.groupBy(tbGraduacao.idDiscente, tbGraduacao.idInstituicao);

    return query.list(
        tbGraduacao.idDiscente.as(idDiscenteSubG.getMetadata().getName()),
        tbGraduacao.idInstituicao.as(idInstiSubG.getMetadata().getName()),
        tbGraduacao.anoFim.max());
}
}
package alpc.ufsc.pessoa.discente.query;

import static alpc.ufsc.pessoa.discente.dto.MDiscenteRowHeaderDTO.meta;
import static alpc.ufsc.querydsl.sql.QTbDiscente.tbDiscente;
import static alpc.ufsc.querydsl.sql.QTbGraduacao.tbGraduacao;
import static alpc.ufsc.querydsl.sql.QTbOrientacaoDoutorado.tbOrientacaoDoutorado;
import static alpc.ufsc.querydsl.sql.QTbOrientacaoMestrado.tbOrientacaoMestrado;
import static alpc.ufsc.querydsl.sql.QTbPessoa.tbPessoa;
import static alpc.ufsc.util.NumberUtilsInternal.compareTo;
import static alpc.ufsc.util.querydsl.ExpressionsUtils.integerPath;
import static alpc.ufsc.util.querydsl.ExpressionsUtils.longPath;
import static alpc.ufsc.util.querydsl.ExpressionsUtils.stringPath;

import java.util.LinkedList;
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageImpl;
import org.springframework.data.domain.Pageable;
import org.springframework.stereotype.Component;

import com.mysema.query.Tuple;
import com.mysema.query.sql.SQLQuery;
import com.mysema.query.sql.SQLQueryFactory;
import com.mysema.query.sql.SQLSubQuery;
import com.mysema.query.support.Expressions;
import com.mysema.query.types.path.NumberPath;
import com.mysema.query.types.path.SimplePath;
import com.mysema.query.types.path.StringPath;
import com.mysema.query.types.query.ListSubQuery;

```



```

import alpc.ufsc.entity.domain.StatusOrientacao;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.discente.dto.DiscenteFilterDTO;
import alpc.ufsc.pessoa.discente.dto.DiscenteRowHeaderDTO;
import alpc.ufsc.pessoa.discente.dto.MDiscenteRowHeaderDTO;
import alpc.ufsc.querydsl.sql.QTbInstituicao;
import alpc.ufsc.util.querydsl.Select;

@Component
public class DiscenteRowHeaderQuery {

    private static final SimplePath<Void> filterPeriodo = Expressions.path(Void.class,
        ↪ "filter_periodo");
    private static final NumberPath<Long> idFilterPeriodo = longPath(meta.id, filterPeriodo);

    private static final SimplePath<Void> filterPertencePrograma =
        ↪ Expressions.path(Void.class, "filter_pertence_programa");
    private static final NumberPath<Long> idFilterPertencePrograma = longPath(meta.id,
        ↪ filterPertencePrograma);

    @Autowired
    private SQLQueryFactory queryFactory;

    public Page<DiscenteRowHeaderDTO> getPage(DiscenteFilterDTO filter, Pageable
        ↪ pageable) {
        SQLQuery query = this.queryFactory.query().from(tbDiscente);
        query.join(tbPessoa).on(tbDiscente.idPessoa.eq(tbPessoa.idPessoa));

        this.applyFilter(filter, query);

        query.orderBy(tbPessoa.nomeFiltro.asc());

        query.offset(pageable.getOffset());
        query.limit(pageable.getPageSize());

        Select<DiscenteRowHeaderDTO> select = new
            ↪ Select<>(DiscenteRowHeaderDTO.class);
        MDiscenteRowHeaderDTO meta = MDiscenteRowHeaderDTO.meta;

        select.as(tbDiscente.idDiscente, meta.id);
        select.as(tbPessoa.nome, meta.nome);

        List<DiscenteRowHeaderDTO> content = query.list(select);
        this.postRetrievePageItem(content);
        return new PageImpl<>(content, pageable, query.count());
    }

    void applyFilter(DiscenteFilterDTO filter, SQLQuery query) {
        query.join(this.subQueryPertencePrograma(filter),
            ↪ filterPertencePrograma).on(tbDiscente.idDiscente.eq(idFilterPertencePrograma));

        if (filter.hasAnoInicio() || filter.hasAnoFim()) {
            query.join(this.subQueryPeriodo(filter),
                ↪ filterPeriodo).on(tbDiscente.idDiscente.eq(idFilterPeriodo));
        }
        if (filter.hasNome()) {
            query.where(tbPessoa.nomeFiltro.like(QueryUtils.like(filter.getNome())));
        }
    }

    private static final SimplePath<Void> unionPertencePrograma =
        ↪ Expressions.path(Void.class, "union_pertence_programa");
    private static final NumberPath<Long> idUnionPertence = longPath(meta.id,
        ↪ unionPertencePrograma);

    private ListSubQuery<Long> subQueryPertencePrograma(DiscenteFilterDTO filter) {
        List<ListSubQuery<Long>> subs = new LinkedList<>();

        SQLSubQuery query = this.queryFactory.subQuery().from(tbOrientacaoMestrado);
        query.where(tbOrientacaoMestrado.idPrograma.eq(filter.getIdPrograma()));
        query.where(tbOrientacaoMestrado.deletado.eq(false));
        query.where(tbOrientacaoMestrado.orientadorAtivo.eq(true).or(tbOrientacaoMestrado.coorientadorA
        subs.add(query.list(tbOrientacaoMestrado.idDiscente.as(meta.id, getAlias())));

        query = this.queryFactory.subQuery().from(tbOrientacaoDoutorado);
        query.where(tbOrientacaoDoutorado.idPrograma.eq(filter.getIdPrograma()));
        query.where(tbOrientacaoDoutorado.deletado.eq(false));
        query.where(tbOrientacaoDoutorado.orientadorAtivo.eq(true).or(tbOrientacaoDoutorado.coorientador
        subs.add(query.list(tbOrientacaoDoutorado.idDiscente.as(meta.id, getAlias())));
        SQLSubQuery subQueryUnion =
            ↪ this.queryFactory.subQuery().from(this.queryFactory.subQuery().union(subs).as(unionPer

```



```

        ordemAndamento = tuple.get(ordem);
        anoInicioAndamento = tuple.get(anoInicio);
        discente.setAnoInicio(tuple.get(anoInicio));
        if (ordemAndamento.equals(1)) {
            discente.setNomeFormacaoAndamento("Discente
                ↳ de mestrado");
        } else {
            discente.setNomeFormacaoAndamento("Discente
                ↳ de doutorado");
        }
    }
}
}
}
}
}

private static final String STATUS = "status";
private static final String ANO_FIM = "anoFim";
private static final String ORDEM = "ordem";

private static final SimplePath<Void> unionPostRetrieve = Expressions.path(Void.class,
    ↳ "union_post_retrieve");

private static final StringPath status = Expressions.stringPath(unionPostRetrieve, STATUS);
private static final NumberPath<Integer> anoInicio = integerPath(meta.anoInicio,
    ↳ unionPostRetrieve);
private static final NumberPath<Integer> anoFim = Expressions.numberPath(Integer.class,
    ↳ unionPostRetrieve, ANO_FIM);
private static final StringPath instProc = stringPath(meta.instituicaoProcedencia,
    ↳ unionPostRetrieve);
private static final NumberPath<Integer> ordem = Expressions.numberPath(Integer.class,
    ↳ unionPostRetrieve, ORDEM);

private List<Tuple> queryPostRetrieve(Long idDiscente) {
    List<ListSubQuery<Tuple>> subs = new LinkedList<>();
    subs.add(this.getGraduacaoPostRetrieve(idDiscente));
    subs.add(this.getMestradoPostRetrieve(idDiscente));
    subs.add(this.getDoutoradoPostRetrieve(idDiscente));
    SQLQuery query =
        ↳ this.queryFactory.query().from(this.queryFactory.subQuery().union(subs)).as(unionPostRetrieve);
    return query.list(
        status,
        anoInicio,
        anoFim,
        instProc,
        ordem);
}

private ListSubQuery<Tuple> getGraduacaoPostRetrieve(Long idDiscente) {
    QTbInstituicao qInstituicao = new QTbInstituicao("ginstituicao");
    SQLSubQuery query = this.queryFactory.subQuery().from(tbGraduacao);
    query.join(qInstituicao).on(tbGraduacao.idInstituicao.eq(qInstituicao.idInstituicao));
    query.where(tbGraduacao.idDiscente.eq(idDiscente));
    query.where(tbGraduacao.deletado.eq(false));
    return query.list(
        Expressions.constantAs(StatusOrientacao.CONCLUIDO.name(),
            ↳ Expressions.path(String.class, STATUS)),
        tbGraduacao.anoInicio.as(meta.anoInicio.getAlias()),
        tbGraduacao.anoFim.as(ANO_FIM),
        qInstituicao.nome.as(meta.instituicaoProcedencia.getAlias()),
        Expressions.constantAs(0, Expressions.path(Integer.class,
            ↳ ORDEM)));
}

private ListSubQuery<Tuple> getMestradoPostRetrieve(Long idDiscente) {
    QTbInstituicao qInstituicao = new QTbInstituicao("minstituicao");
    SQLSubQuery query = this.queryFactory.subQuery().from(tbOrientacaoMestrado);
    query.join(qInstituicao).on(tbOrientacaoMestrado.idInstituicao.eq(qInstituicao.idInstituicao));
    query.where(tbOrientacaoMestrado.idDiscente.eq(idDiscente));
    query.where(tbOrientacaoMestrado.deletado.eq(false));
    return query.list(
        tbOrientacaoMestrado.statusOrientacao.as(STATUS),
        tbOrientacaoMestrado.anoInicio.as(meta.anoInicio.getAlias()),
        tbOrientacaoMestrado.anoFim.as(ANO_FIM),
        qInstituicao.nome.as(meta.instituicaoProcedencia.getAlias()),
        Expressions.constantAs(1, Expressions.path(Integer.class,
            ↳ ORDEM)));
}

private ListSubQuery<Tuple> getDoutoradoPostRetrieve(Long idDiscente) {
    QTbInstituicao qInstituicao = new QTbInstituicao("dinstituicao");

```

```

SQLSubQuery query = this.queryFactory.subQuery().from(tbOrientacaoDoutorado);
query.join(qInstituicao).on(tbOrientacaoDoutorado.idInstituicao.eq(qInstituicao.idInstituicao));
query.where(tbOrientacaoDoutorado.idDiscente.eq(idDiscente));
query.where(tbOrientacaoDoutorado.deletado.eq(false));
return query.list(
    tbOrientacaoDoutorado.statusOrientacao.as(STATUS),
    tbOrientacaoDoutorado.anoInicio.as(meta.anoInicio.getAlias()),
    tbOrientacaoDoutorado.anoFim.as(ANO_FIM),
    qInstituicao.nome.as(meta.instituicaoProcedencia.getAlias()),
    Expressions.constantAs(2, Expressions.path(Integer.class,
        ↳ ORDEM)));
}
}
package alpc.ufsc.pessoa.discente.query;

import static alpc.ufsc.pessoa.discente.dto.MDiscenteHeaderDTO.meta;
import static alpc.ufsc.querydsl.sql.QTbDiscente.tbDiscente;
import static alpc.ufsc.querydsl.sql.QTbOrientacaoDoutorado.tbOrientacaoDoutorado;
import static alpc.ufsc.querydsl.sql.QTbOrientacaoMestrado.tbOrientacaoMestrado;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;

import com.mysema.query.Tuple;
import com.mysema.query.sql.SQLQuery;
import com.mysema.query.sql.SQLQueryFactory;
import com.mysema.query.sql.SQLSubQuery;
import com.mysema.query.support.Expressions;
import com.mysema.query.types.expr.CaseBuilder;
import com.mysema.query.types.expr.NumberExpression;
import com.mysema.query.types.path.BooleanPath;
import com.mysema.query.types.path.NumberPath;
import com.mysema.query.types.path.SimplePath;
import com.mysema.query.types.path.StringPath;
import com.mysema.query.types.query.ListSubQuery;

import alpc.ufsc.entity.domain.StatusOrientacao;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.discente.dto.DiscenteFilterDTO;
import alpc.ufsc.pessoa.discente.dto.DiscenteHeaderDTO;
import alpc.ufsc.util.querydsl.Select;

@Component
public class DiscenteHeaderQuery {

    @Autowired
    private SQLQueryFactory queryFactory;

    @Autowired
    private DiscenteRowHeaderQuery rowHeaderQuery;

    private static final SimplePath<Void> groupDoutorado = Expressions.path(Void.class,
        ↳ "group_doutorado");
    private static final NumberPath<Long> idDiscenteGroupDoutorado =
        ↳ Expressions.numberPath(Long.class, groupDoutorado, "id_discente");
    private static final NumberPath<Long> andamentoGroupDoutorado =
        ↳ Expressions.numberPath(Long.class, groupDoutorado, "andamento");
    private static final NumberPath<Long> andamentoBolsaGroupDoutorado =
        ↳ Expressions.numberPath(Long.class, groupDoutorado, "andamento_bolsa");
    private static final NumberPath<Long> concluidoGroupDoutorado =
        ↳ Expressions.numberPath(Long.class, groupDoutorado, "concluido");
    private static final NumberPath<Long> sanduicheGroupDoutorado =
        ↳ Expressions.numberPath(Long.class, groupDoutorado, "sanduiche");

    public DiscenteHeaderDTO getHeader(DiscenteFilterDTO filter) {
        SQLQuery query = this.queryFactory.query().from(tbDiscente);
        query.leftJoin(this.getSubQueryOrientacaoMestrado(filter),
            ↳ mestrado).on(tbDiscente.idDiscente.eq(idDiscenteMestrado));
        query.leftJoin(this.getSubQueryOrientacaoDoutorado(filter),
            ↳ groupDoutorado).on(tbDiscente.idDiscente.eq(idDiscenteGroupDoutorado));

        this.rowHeaderQuery.applyFilter(filter, query);

        Select<DiscenteHeaderDTO> select = new Select<>(DiscenteHeaderDTO.class);
        select.as(this.caseStatus(statusMestrado, StatusOrientacao.EM_ANDAMENTO),
            ↳ meta.mestradoEmAndamento);
        select.as(this.caseStatusBolsa(statusMestrado, bolsaMestrado,
            ↳ StatusOrientacao.EM_ANDAMENTO),
            ↳ meta.mestradoEmAndamentoBolsa);
        select.as(this.caseStatus(statusMestrado, StatusOrientacao.CONCLUIDO),
            ↳ meta.mestradoConcluido);
    }
}

```

```

select.as(andamentoGroupDoutorado.sum(), meta.doutoradoEmAndamento);
select.as(andamentoBolsaGroupDoutorado.sum(),
    ↪ meta.doutoradoEmAndamentoBolsa);
select.as(concluidoGroupDoutorado.sum(), meta.doutoradoConcluido);
select.as(sanduiचेGroupDoutorado.sum(), meta.doutoradoSanduiचे);
DiscenteHeaderDTO dto = query.uniqueResult(select);
this.postRetriveHeaderDTO(dto, filter);
return dto;
}

private NumberExpression<Long> caseStatus(StringPath status, StatusOrientacao
    ↪ statusEnum) {
    return new
        ↪ CaseBuilder().when(status.eq(statusEnum.name())).then(1L).otherwise(0L).sum();
}

private NumberExpression<Long> caseStatusBolsa(StringPath status,
    ↪ NumberPath<Integer> bolsa, StatusOrientacao statusEnum) {
    return new
        ↪ CaseBuilder().when(status.eq(statusEnum.name()).and(bolsa.eq(1))).then(1L).otherwise(0L).sum();
}

private static final SimplePath<Void> mestrado = Expressions.path(Void.class, "mestrado");
private static final NumberPath<Long> idDiscenteMestrado =
    ↪ Expressions.numberPath(Long.class, mestrado, "id_discente");
private static final StringPath statusMestrado = Expressions.stringPath(mestrado, "status");
private static final NumberPath<Integer> bolsaMestrado =
    ↪ Expressions.numberPath(Integer.class, mestrado, "bolsa");

private ListSubQuery<Tuple> getSubQueryOrientacaoMestrado(DiscenteFilterDTO filter) {
    SQLSubQuery subQuery =
        ↪ this.queryFactory.subQuery().from(tbOrientacaoMestrado);
    subQuery.where(tbOrientacaoMestrado.orientadorAtivo.eq(true).or(tbOrientacaoMestrado.coorientadorAtivo.eq(true)));
    subQuery.where(tbOrientacaoMestrado.deletado.eq(false));
    subQuery.where(tbOrientacaoMestrado.idPrograma.eq(filter.getIdPrograma()));
    if (filter.hasAnoInicio() || filter.hasAnoFim()) {
        subQuery.where(QueryUtils.overlappingInterval(tbOrientacaoMestrado.anoInicio,
            ↪ tbOrientacaoMestrado.anoFim, filter.getAnoInicio(),
            ↪ filter.getAnoFim()));
    }
    subQuery.where(
        ↪ tbOrientacaoMestrado.statusOrientacao.eq(StatusOrientacao.EM_ANDAMENTO));
    subQuery.groupBy(tbOrientacaoMestrado.idDiscente,
        ↪ tbOrientacaoMestrado.statusOrientacao);
    return subQuery.list(
        ↪ tbOrientacaoMestrado.idDiscente.as(idDiscenteMestrado.getMetadata().getName()),
        ↪ tbOrientacaoMestrado.statusOrientacao.as(statusMestrado.getMetadata().getName()),
        ↪ this.maxBolsa(tbOrientacaoMestrado.bolsa).as(bolsaMestrado.getMetadata().getName()));
}

private static final SimplePath<Void> doutorado = Expressions.path(Void.class,
    ↪ "doutorado");
private static final NumberPath<Long> idDiscenteDoutorado =
    ↪ Expressions.numberPath(Long.class, doutorado, "id_discente");
private static final StringPath statusDoutorado = Expressions.stringPath(doutorado,
    ↪ "status");
private static final NumberPath<Integer> bolsaDoutorado =
    ↪ Expressions.numberPath(Integer.class, doutorado, "bolsa");
private static final NumberPath<Long> sanduiчеDoutorado =
    ↪ Expressions.numberPath(Long.class, doutorado, "sanduiче");

private ListSubQuery<Tuple> getSubQueryOrientacaoDoutorado(DiscenteFilterDTO filter) {
    SQLSubQuery subQuery =
        ↪ this.queryFactory.subQuery().from(tbOrientacaoDoutorado);
    subQuery.where(tbOrientacaoDoutorado.orientadorAtivo.eq(true).or(tbOrientacaoDoutorado.coorientadorAtivo.eq(true)));
    subQuery.where(tbOrientacaoDoutorado.deletado.eq(false));
    subQuery.where(tbOrientacaoDoutorado.idPrograma.eq(filter.getIdPrograma()));
    if (filter.hasAnoInicio() || filter.hasAnoFim()) {
        subQuery.where(QueryUtils.overlappingInterval(tbOrientacaoDoutorado.anoInicio,
            ↪ tbOrientacaoDoutorado.anoFim, filter.getAnoInicio(),
            ↪ filter.getAnoFim()));
    }
    subQuery.where(
        ↪ tbOrientacaoDoutorado.statusOrientacao.eq(StatusOrientacao.EM_ANDAMENTO));
    subQuery.groupBy(tbOrientacaoDoutorado.idDiscente,
        ↪ tbOrientacaoDoutorado.statusOrientacao);
    ListSubQuery<Tuple> doutoradoList = subQuery.list(
        ↪ tbOrientacaoDoutorado.idDiscente.as(idDiscenteDoutorado.getMetadata().getName()),
        ↪ tbOrientacaoDoutorado.statusOrientacao.as(statusDoutorado.getMetadata().getName()),
        ↪ this.maxBolsa(tbOrientacaoDoutorado.bolsa).as(bolsaDoutorado.getMetadata().getName()),
        ↪ new
            ↪ CaseBuilder().when(tbOrientacaoDoutorado.nomeOrientadorSanduiчеFil

```

```

SQLSubQuery subQueryGroup =
    ↪ this.queryFactory.subQuery().from(doutoradoList.as(doutorado.getMetadata().getName()));
subQueryGroup.groupBy(idDiscenteDoutorado);
return subQueryGroup.list(
    idDiscenteDoutorado.as(idDiscenteGroupDoutorado.getMetadata().getName()),
    this.caseStatus(statusDoutorado,
        ↪ StatusOrientacao.EM_ANDAMENTO).as(andamentoGroupDoutorado.getMetadata().getName()),
    this.caseStatus(Bolsa(statusDoutorado, bolsaDoutorado),
        ↪ StatusOrientacao.EM_ANDAMENTO).as(andamentoBolsaGroupDoutorado.getMetadata().getName()),
    this.caseStatus(statusDoutorado,
        ↪ StatusOrientacao.CONCLUIDO).as(concluidoGroupDoutorado.getMetadata().getName()),
    sanduicheDoutorado.max().as(sanduicheGroupDoutorado.getMetadata().getName()));
}

private NumberExpression<Integer> maxBolsa(BooleanPath bolsa) {
    return new CaseBuilder().when(bolsa.eq(true)).then(1).otherwise(0).max();
}

private void postRetrieveHeaderDTO(DiscenteHeaderDTO dto, DiscenteFilterDTO filter) {
    SQLSubQuery query = this.queryFactory.subQuery().from(tbOrientacaoMestrado);
    query.where(tbOrientacaoMestrado.idPrograma.eq(filter.getIdPrograma()));
    query.where(tbOrientacaoMestrado.deletado.eq(false));
    query.where(tbOrientacaoMestrado.orientadorAtivo.eq(true).or(tbOrientacaoMestrado.coorientadorAtivo.eq(true)));
    if (filter.hasAnoInicio() || filter.hasAnoFim()) {
        query.where(QueryUtils.overlappingInterval(tbOrientacaoMestrado.anoInicio,
            ↪ tbOrientacaoMestrado.anoFim, filter.getAnoInicio(),
            ↪ filter.getAnoFim()));
    }
    ListSubQuery<Tuple> subMestrado = query.list(
        this.caseStatus(tbOrientacaoMestrado.statusOrientacao,
            ↪ StatusOrientacao.DESISTENCIA).as("desistentes"),
        this.caseTime(tbOrientacaoMestrado.statusOrientacao,
            ↪ tbOrientacaoMestrado.anoInicio,
            ↪ tbOrientacaoMestrado.anoFim, 2).as("time"),
        this.caseStatus(tbOrientacaoMestrado.statusOrientacao,
            ↪ StatusOrientacao.CONCLUIDO).as("concluidos"));

    query = this.queryFactory.subQuery().from(tbOrientacaoDoutorado);
    query.where(tbOrientacaoDoutorado.idPrograma.eq(filter.getIdPrograma()));
    query.where(tbOrientacaoDoutorado.deletado.eq(false));
    query.where(tbOrientacaoDoutorado.orientadorAtivo.eq(true).or(tbOrientacaoDoutorado.coorientadorAtivo.eq(true)));
    if (filter.hasAnoInicio() || filter.hasAnoFim()) {
        query.where(QueryUtils.overlappingInterval(tbOrientacaoDoutorado.anoInicio,
            ↪ tbOrientacaoDoutorado.anoFim, filter.getAnoInicio(),
            ↪ filter.getAnoFim()));
    }
    ListSubQuery<Tuple> subDoutorado = query.list(
        this.caseStatus(tbOrientacaoDoutorado.statusOrientacao,
            ↪ StatusOrientacao.DESISTENCIA).as("desistentes"),
        this.caseTime(tbOrientacaoDoutorado.statusOrientacao,
            ↪ tbOrientacaoDoutorado.anoInicio,
            ↪ tbOrientacaoDoutorado.anoFim, 4).as("time"),
        this.caseStatus(tbOrientacaoDoutorado.statusOrientacao,
            ↪ StatusOrientacao.CONCLUIDO).as("concluidos"));

    SimplePath<Void> doutorado = Expressions.path(Void.class, "doutorado");
    NumberPath<Long> dDesistentes = Expressions.numberPath(Long.class, doutorado,
        ↪ "desistentes");
    NumberPath<Long> dTime = Expressions.numberPath(Long.class, doutorado,
        ↪ "time");
    NumberPath<Long> dConcluidos = Expressions.numberPath(Long.class, doutorado,
        ↪ "concluidos");

    SimplePath<Void> mestrado = Expressions.path(Void.class, "mestrado");
    NumberPath<Long> mDesistentes = Expressions.numberPath(Long.class, mestrado,
        ↪ "desistentes");
    NumberPath<Long> mTime = Expressions.numberPath(Long.class, mestrado,
        ↪ "time");
    NumberPath<Long> mConcluidos = Expressions.numberPath(Long.class, mestrado,
        ↪ "concluidos");

    Tuple result = this.queryFactory.query().from(subMestrado.as("mestrado"),
        ↪ subDoutorado.as("doutorado")).uniqueResult(
        dDesistentes, dTime, dConcluidos,
        mDesistentes, mTime, mConcluidos);

    dto.setTotalDesistentes((result.get(mDesistentes) != null ? result.get(mDesistentes) :
        ↪ 0) + (result.get(dDesistentes) != null ? result.get(dDesistentes) : 0));
    Long totalMConcluidos = result.get(mConcluidos);
    dto.setTempoMedioCursoMestrado(totalMConcluidos != null && totalMConcluidos

```

```

        ↪ > 0 ? result.get(mTime) / totalMConcluidos.doubleValue() : 0);
    Long totalDConcluidos = result.get(dConcluidos);
    dto.setTempoMedioCursoDoutorado(totalDConcluidos != null && totalDConcluidos
        ↪ > 0 ? result.get(dTime) / totalDConcluidos.doubleValue() : 0);
    }

    private NumberExpression<Integer> caseTime(StringPath status, NumberPath<Integer>
        ↪ anoInicio, NumberPath<Integer> anoFim, Integer time) {
        return new CaseBuilder()
            .when(status.eq(StatusOrientacao.CONCLUIDO.name()))
            .then(new CaseBuilder()
                .when(anoInicio.ne(QueryUtils.MIN_YEAR).and(anoFim.ne(Query
                    .otherwise(time))
                .then(anoFim.subtract(anoInicio))
                .otherwise(0).sum();
            }
        }
    }
}
package alpc.ufsc.pessoa.discente.dto;

import lombok.Getter;
import lombok.Setter;
import br.ufsc.bridge.metafy.Metafy;

@Getter
@Setter
@Metafy
public class DiscenteRowHeaderDTO {

    private Long id;
    private String nome;

    private String nomeFormacaoAndamento;
    private Integer anoInicio;

    private String instituicaoProcedencia;
}
package alpc.ufsc.pessoa.discente.dto;

import java.util.List;

import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import com.fasterxml.jackson.annotation.JsonInclude;
import com.fasterxml.jackson.annotation.JsonInclude.Include;

import alpc.ufsc.pessoa.orientacao.common.dto.FormacaoRowDTO;

@Setter
@Getter
@AllArgsConstructor
@NoArgsConstructor
@JsonInclude(Include.NON_EMPTY)
public class DiscenteRowBodyDTO {
    private List<FormacaoRowDTO> formacoes;
}
package alpc.ufsc.pessoa.discente.dto;

import lombok.Getter;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;

@Getter
@Setter
public class DiscenteInstituicaoProcedenciaFilterDTO {

    private Long programaId;
    private String nome;
    private Integer inicio;
    private Integer fim;

    public boolean hasInicio() {
        return this.inicio != null;
    }

    public boolean hasFim() {
        return this.fim != null;
    }

    public boolean hasNome() {

```

```

        return StringUtils.isNotBlank(this.nome);
    }
}
package alpc.ufsc.pessoa.discente.dto;

import lombok.Getter;
import lombok.Setter;
import br.ufsc.bridge.metafy.Metafy;

@Getter
@Setter
@Metafy
public class DiscenteInstituicaoProcedenciaRowDTO {

    private Long id;
    private String nome;
    private Long numeroDiscentes;
}
package alpc.ufsc.pessoa.discente.dto;

import lombok.Getter;
import lombok.Setter;

import br.ufsc.bridge.metafy.Metafy;

@Getter
@Setter
@Metafy
public class DiscenteHeaderDTO {

    private Double tempoMedioCursoMestrado;
    private Double tempoMedioCursoDoutorado;
    private Long totalDesistentes;
    private Long total;

    private Long mestradoEmAndamento;
    private Long mestradoEmAndamentoBolsa;
    private Long mestradoConcluido;

    private Long doutoradoEmAndamento;
    private Long doutoradoEmAndamentoBolsa;
    private Long doutoradoConcluido;
    private Long doutoradoSanduiche;
}
package alpc.ufsc.pessoa.discente.dto;

import lombok.Getter;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;

@Getter
@Setter
public class DiscenteFilterDTO {

    private Long idPrograma;
    private Integer anoInicio;
    private Integer anoFim;
    private String nome;

    public boolean hasAnoInicio() {
        return this.anoInicio != null;
    }

    public boolean hasAnoFim() {
        return this.anoFim != null;
    }

    public boolean hasNome() {
        return !StringUtils.isBlank(this.nome);
    }
}
package alpc.ufsc.pessoa;

import static alpc.ufsc.entity.pessoa.QPessoaEntity.pessoaEntity;
import static alpc.ufsc.entity.pessoa.discente.QDiscenteEntity.discenteEntity;
import static alpc.ufsc.entity.pessoa.docente.QDocenteEntity.docenteEntity;
import static alpc.ufsc.pessoa.dto.MPessoaDto.meta;

import java.util.Date;

import javax.persistence.EntityManager;

```



```

import javax.persistence.PersistenceContext;

import lombok.extern.slf4j.Slf4j;

import org.apache.commons.lang3.StringUtils;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.Tuple;
import com.mysema.query.jpaa.impl.JPAQuery;
import com.mysema.query.jpaa.impl.JPAUpdateClause;

import alpc.ufsc.entity.pessoa.NomeCitacaoEntity;
import alpc.ufsc.entity.pessoa.PessoaEntity;
import alpc.ufsc.entity.programa.ProgramaEntity;
import alpc.ufsc.pessoa.dto.PessoaDto;
import alpc.ufsc.pessoa.orientacao.common.dto.PessoaSearchDto;
import alpc.ufsc.pessoa.orientacao.common.service.FormacaoProcessService;
import alpc.ufsc.pessoa.orientacao.common.service.OrientacaoProcessService;
import alpc.ufsc.programa.dto.ImportProcessDto;
import alpc.ufsc.util.DateUtilsInternal;
import alpc.ufsc.util.querydsl.Select;
import alpc.ufsc.xml.dto.CurriculoVitaeXmlDto;

@Slf4j
@Service
@Transactional
public class PessoaService {

    @PersistenceContext
    private EntityManager em;

    @Autowired
    private FormacaoProcessService formacaoService;

    @Autowired
    private OrientacaoProcessService orientacaoService;

    @Autowired
    private PessoaSearchService searchService;

    @Transactional(propagation = Propagation.REQUIRES_NEW)
    public boolean processPessoa(ImportProcessDto processDto) {
        CurriculoVitaeXmlDto curriculo = processDto.getCurriculo();

        PessoaDto pessoaDto =
            ↪ this.loadByNumeroIdentificador(curriculo.getNumeroIdentificador(),
            ↪ processDto);
        if (pessoaDto != null) {
            return this.updatePessoaOutdated(processDto,
                ↪ curriculo.getDataAtualizacao(), curriculo.getHoraAtualizacao(),
                ↪ pessoaDto);
        }
        pessoaDto = new PessoaDto(curriculo, processDto.getProgramaDto().getId());
        if (this.fingGeral(processDto)) {
            pessoaDto.setId(processDto.getPessoaId());
            this.save(pessoaDto);
        } else {
            Long pessoaId = this.findOtherwiseSave(pessoaDto);
            processDto.setPessoaId(pessoaId);
        }
        return true;
    }

    private boolean updatePessoaOutdated(ImportProcessDto processDto, Date dataAtualizacao,
        ↪ String horaAtualizacao, PessoaDto pessoaDto) {
        processDto.setPessoaId(pessoaDto.getId());
        if (pessoaDto.isSetDataAtualizacao()) {
            Date data = DateUtilsInternal.applyTimeToDate(dataAtualizacao,
                ↪ horaAtualizacao);
            if (pessoaDto.getDataAtualizacao().compareTo(data) >= 0) {
                return pessoaDto.getForceProcessamento();
            }
            pessoaDto.setDataAtualizacao(dataAtualizacao);
        }
        this.save(pessoaDto);
        return true;
    }

    @Transactional(propagation = Propagation.REQUIRES_NEW)

```

```

public void updateForceProcessamento(Long id, Boolean forceProcessamento) {
    new JPAUpdateClause(this.em, pessoaEntity)
        .set(pessoaEntity.forceProcessamento, forceProcessamento)
        .where(pessoaEntity.id.eq(id))
        .execute();
}

private boolean fingGeral(ImportProcessDto processDto) {
    this.orientacaoService.searchInOrientacoes(processDto);
    if (processDto.isSetDocentId()) {
        return true;
    }
    this.formacaoService.searchInFormacoes(processDto);
    if (processDto.isSetDiscentId()) {
        return true;
    }
    return false;
}

public Long findOtherwiseSave(PessoaDto pessoaDto) {
    Long pessoaId = pessoaDto.getId();
    if (pessoaId == null) {
        try {
            pessoaId = this.searchService.searchPessoa(new
                PessoaSearchDto(pessoaDto.getId(),
                pessoaDto.getNome());
        } catch (Exception e) {
            log.error("Erro ao buscar pessoa", e);
        }
        if (pessoaId == null) {
            PessoaEntity pessoaEntity = new PessoaEntity();
            pessoaId = this.persist(pessoaDto, pessoaEntity);
        }
    }
    return pessoaId;
}

private Long save(PessoaDto pessoaDto) {
    PessoaEntity entity = null;
    if (pessoaDto.isSetId()) {
        entity = this.em.find(PessoaEntity.class, pessoaDto.getId());
    } else {
        entity = new PessoaEntity();
    }
    return this.persist(pessoaDto, entity);
}

private Long persist(PessoaDto pessoaDto, PessoaEntity pessoaEntity) {
    pessoaEntity.setPrograma(this.em.getReference(ProgramaEntity.class,
        pessoaDto.getProgramaId());
    pessoaEntity.setNome(pessoaDto.getNome());
    pessoaEntity.setDataAtualizacao(pessoaDto.getDataAtualizacao());
    pessoaEntity.setNumeroIdentificador(StringUtils.trim(pessoaDto.getNumeroIdentificador()));
    this.em.persist(pessoaEntity);
    this.saveCitacoes(pessoaDto, pessoaEntity);
    return pessoaEntity.getId();
}

private void saveCitacoes(PessoaDto pessoaDto, PessoaEntity entity) {
    for (String citacao : pessoaDto.citacoes()) {
        NomeCitacaoEntity nomeCitacao = new NomeCitacaoEntity();
        nomeCitacao.setPessoa(entity);
        nomeCitacao.setCitacao(citacao);
        this.em.persist(nomeCitacao);
    }
}

private PessoaDto loadByNumeroIdentificador(String numeroIdentificador, ImportProcessDto
    processDto) {
    if (StringUtils.isNotBlank(numeroIdentificador) &&
        processDto.getProgramaDto().getId() != null) {
        JPAQuery query = new JPAQuery(this.em).from(pessoaEntity);
        query.leftJoin(pessoaEntity.docente, docenteEntity);
        query.leftJoin(pessoaEntity.discente, discenteEntity);

        query.where(pessoaEntity.numeroIdentificador.eq(StringUtils.trim(numeroIdentificador)));
        query.where(pessoaEntity.programa().id.eq(processDto.getProgramaDto().getId()));

        Select<PessoaDto> select = new Select<>(PessoaDto.class);
        select.as(pessoaEntity.id, meta.id);
        select.as(pessoaEntity.nome, meta.nome);
        select.as(pessoaEntity.dataAtualizacao, meta.dataAtualizacao);
    }
}

```

```

        select.as(pessoaEntity.programa().id, meta.programaId);
        select.as(pessoaEntity.numeroIdentificador, meta.numeroIdentificador);
        select.as(pessoaEntity.forceProcessamento, meta.forceProcessamento);
        Tuple result = query.uniqueResult(select, discenteEntity.id,
            ↪ discenteEntity.id);
        if (result != null) {
            PessoaDto dto = result.get(select);
            processDto.setDocenteId(result.get(discenteEntity.id));
            processDto.setDiscenteId(result.get(discenteEntity.id));
            return dto;
        }
        return null;
    }
}
package alpc.ufsc.pessoa.dto;

import java.util.Date;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;

import br.ufsc.bridge.metafy.Metafy;

import alpc.ufsc.util.DateUtilsInternal;
import ufs.alpc.xml.dto.CurriculoVitaeXmlDto;
import ufs.alpc.xml.dto.common.DetalhamentoDaOrientacaoConcluidaXmlDto;
import ufs.alpc.xml.dto.common.DetalhamentoDaOrientacaoXmlDto;
import ↪ ufs.alpc.xml.dto.dados.complementares.orientacoes.andamento.doutorado.OrientacaoEmAndamentoDeDoutoradoXmlDto;
import ↪ ufs.alpc.xml.dto.dados.complementares.orientacoes.andamento.mestrado.OrientacaoEmAndamentoDeMestradoXmlDto;
import ufs.alpc.xml.dto.dados.gerais.formacao.doutorado.DoutoradoXmlDto;
import ufs.alpc.xml.dto.dados.gerais.formacao.mestrado.MestradoXmlDto;
import ufs.alpc.xml.dto.dados.gerais.formacao.posdoutorado.PosDoutoradoXmlDto;
import ↪ ufs.alpc.xml.dto.outra.producao.orientacoes.concluidas.doutorado.OrientacaoConcluidaDoutoradoXmlDto;
import ↪ ufs.alpc.xml.dto.outra.producao.orientacoes.concluidas.mestrado.OrientacaoConcluidaMestradoXmlDto;

@Getter
@Setter
@Metafy
@NoArgsConstructor
public class PessoaDto {

    private Long id;

    private String nome;

    private Date dataAtualizacao;

    private String numeroIdentificador;

    private Long programaId;

    private String nomeCitacoes;

    private Boolean forceProcessamento;

    public PessoaDto(Long id) {
        this.id = id;
    }

    public PessoaDto(CurriculoVitaeXmlDto curriculo, Long programaId) {
        this.nome = curriculo.getDadosGerais().getNomeCompleto();
        this.nomeCitacoes = curriculo.getDadosGerais().getNomeCitacoesBibliograficas();
        this.dataAtualizacao =
            ↪ DateUtilsInternal.applyTimeToDate(curriculo.getDataAtualizacao(),
            ↪ curriculo.getHoraAtualizacao());
        this.setNumeroIdentificador(curriculo.getNumeroIdentificador());
        this.programaId = programaId;
    }

    public PessoaDto(OrientacaoEmAndamentoDeMestradoXmlDto mestradoXmlDto, Long
        ↪ programaId) {
        DetalhamentoDaOrientacaoXmlDto detalhamento =
            ↪ mestradoXmlDto.getDetalhamento();
        this.nome = detalhamento.getNomeOrientando();
    }
}

```

```

        this.setNumeroIdentificador(detalhamento.getNumeroIDOrientado());
        this.programaId = programaId;
    }

    public PessoaDto(OrientacaoConcluidaMestradoXmlDto orientacaoMestradoXmlDto, Long
    ↪ programaId) {
        DetalhamentoDaOrientacaoConcluidaXmlDto detalhamento =
            ↪ orientacaoMestradoXmlDto.getDetalhamento();
        this.nome = detalhamento.getNomeOrientando();
        this.setNumeroIdentificador(detalhamento.getNumeroIDOrientado());
        this.programaId = programaId;
    }

    public PessoaDto(MestradoXmlDto mestradoXml, Long programaId, boolean
    ↪ orientadorPrincipal) {
        if (orientadorPrincipal) {
            this.nome = mestradoXml.getNomeCompletoOrientador();
            this.setNumeroIdentificador(mestradoXml.getNumeroIdOrientador());
        } else {
            this.nome = mestradoXml.getNomeCoOrientador();
        }
        this.programaId = programaId;
    }

    public PessoaDto(OrientacaoEmAndamentoDeDoutoradoXmlDto doutoradoXmlDto, Long
    ↪ programaId) {
        DetalhamentoDaOrientacaoXmlDto detalhamento =
            ↪ doutoradoXmlDto.getDetalhamento();
        this.nome = detalhamento.getNomeOrientando();
        this.setNumeroIdentificador(detalhamento.getNumeroIDOrientado());
        this.programaId = programaId;
    }

    public PessoaDto(OrientacaoConcluidaDoutoradoXmlDto orientacaoDoutoradoXmlDto, Long
    ↪ programaId) {
        DetalhamentoDaOrientacaoConcluidaXmlDto detalhamento =
            ↪ orientacaoDoutoradoXmlDto.getDetalhamento();
        this.nome = detalhamento.getNomeOrientando();
        this.setNumeroIdentificador(detalhamento.getNumeroIDOrientado());
        this.programaId = programaId;
    }

    public PessoaDto(DoutoradoXmlDto doutoradoXmlDto, Long programaId, boolean
    ↪ orientadorPrincipal) {
        if (orientadorPrincipal) {
            this.nome = doutoradoXmlDto.getNomeCompletoOrientador();
            this.setNumeroIdentificador(doutoradoXmlDto.getNumeroIdOrientador());
        } else {
            this.nome = doutoradoXmlDto.getNomeCoOrientador();
        }
        this.programaId = programaId;
    }

    public PessoaDto(PosDoutoradoXmlDto posDoutoradoXmlDto, Long programaId) {
        this.setNumeroIdentificador(posDoutoradoXmlDto.getNumeroIdOrientador());
        this.programaId = programaId;
    }

    public void setNumeroIdentificador(String numeroIdentificador) {
        this.numeroIdentificador = StringUtils.trimToNull(numeroIdentificador);
    }

    public boolean isSetId() {
        return this.id != null;
    }

    public boolean isSetNome() {
        return StringUtils.isNotBlank(this.nome);
    }

    public boolean isSetNumeroIdentificador() {
        return StringUtils.isNotBlank(this.numeroIdentificador);
    }

    public boolean isSetDataAtualizacao() {
        return this.dataAtualizacao != null;
    }

    public String[] citacoes() {
        if (StringUtils.isNotBlank(this.nomeCitacoes)) {
            return this.nomeCitacoes.split(";");
        }
    }

```

```

    }
    return new String[0];
}
}

package alpc.ufsc.pessoa.atuacaoprofissional.linhapesquisa;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.apache.lucene.search.BooleanClause. Occur;
import org.apache.lucene.search. BooleanQuery;
import org.hibernate.search. SearchFactory;
import org.hibernate.search.errors. EmptyQueryException;
import org.hibernate.search.jpamodeling. FullTextEntityManager;
import org.hibernate.search.jpamodeling. FullTextQuery;
import org.hibernate.search.jpamodeling. Search;
import org.hibernate.search.query.dsl. QueryBuilder;
import org.springframework.stereotype. Service;
import org.springframework.transaction.annotation. Propagation;
import org.springframework.transaction.annotation. Transactional;

import alpc.ufsc.entity.pessoa.atuacaoprofissional.linhapesquisa. LinhaPesquisaEntity;

@Service
@Transactional(propagation = Propagation.SUPPORTS, noRollbackFor =
    ↳ EmptyQueryException.class)
public class LinhaPesquisaSearchService {

    private static final String TITULO = "titulo";
    private static final String PROGRAMA_ID = "programa.id";
    private static final String ID = "id";

    @PersistenceContext
    private EntityManager em;

    public Long find(LinhaPesquisaSearchDto searchDto) {
        FullTextEntityManager search = Search.getFullTextEntityManager(this.em);
        search.flushToIndexes();
        SearchFactory searchFactory = search.getSearchFactory();
        QueryBuilder qb =
            ↳ searchFactory.buildQueryBuilder().forEntity(LinhaPesquisaEntity.class).get();

        BooleanQuery query = new BooleanQuery();
        query.add(qb.keyword().onField(TITULO).matching(searchDto.getTitulo()).createQuery(),
            ↳ Occur.MUST);
        query.add(qb.keyword().onField(PROGRAMA_ID).matching(searchDto.getProgramaId()).createQuery(),
            ↳ Occur.MUST);

        FullTextQuery fullTextQuery = search.createFullTextQuery(query,
            ↳ LinhaPesquisaEntity.class);

        fullTextQuery.setProjection(this.getProjection());
        fullTextQuery.setMaxResults(5);

        List<?> result = fullTextQuery.getResultList();
        for (Object object : result) {
            Object[] fields = (Object[]) object;
            if (searchDto.similar(fields[0])) {
                return (Long) fields[1];
            }
        }
        return null;
    }

    private String[] getProjection() {
        String[] p = { TITULO, ID };
        return p;
    }
}

package alpc.ufsc.pessoa.atuacaoprofissional.linhapesquisa;

import java.util.List;

import lombok.extern.slf4j..Slf4j;

import org.springframework.beans.factory.annotation. Autowired;
import org.springframework.stereotype. Service;

```

```

import ↪ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.pesquisadesenvolvimento.PesquisaDesenvolvimentoXmlDto;
import ↪ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.pesquisadesenvolvimento.linhapesquisa.LinhaPesquisaXmlDto;

@Sif4j
@Service
public class LinhaPesquisaProcessService {

    @Autowired
    private LinhaPesquisaSearchService linhaPesquisaSearch;

    @Autowired
    private LinhaPesquisaService linhaPesquisaService;

    public void processLinhaPesquisa(Long docenteId, Long programaId,
        ↪ PesquisaDesenvolvimentoXmlDto pesquisaDesenvolvimento) {
        List<LinhaPesquisaXmlDto> linhasPesquisa =
            ↪ pesquisaDesenvolvimento.getLinhaPesquisa();
        if (linhasPesquisa == null) {
            return;
        }
        for (LinhaPesquisaXmlDto linhaPesquisa : linhasPesquisa) {
            Long linhaPesquisaId = null;
            try {
                linhaPesquisaId = this.linhaPesquisaSearch.find(new
                    ↪ LinhaPesquisaSearchDto(linhaPesquisa.getTituloLinhaPesquisa(),
                    ↪ programaId));
                this.linhaPesquisaService.save(linhaPesquisaId, programaId,
                    ↪ linhaPesquisa.getTituloLinhaPesquisa(), docenteId);
            } catch (Exception e) {
                log.error("Erro ao processar linha de pesquisa", e.getMessage());
            }
        }
    }
}

package alpc.ufsc.pessoa.atuacaoprofissional.linhapesquisa;

import static
    ↪ alpc.ufsc.entity.pessoa.atuacaoprofissional.linhapesquisa.QLinhaPesquisaDocenteEntity.linhaPesquisaDocenteEntity;
import static
    ↪ alpc.ufsc.entity.pessoa.atuacaoprofissional.linhapesquisa.QLinhaPesquisaEntity.linhaPesquisaEntity;
import static alpc.ufsc.pessoa.atuacaoprofissional.linhapesquisa.dto.MLinhaPesquisaRowDTO.meta;
import static alpc.ufsc.querydsl.sql.QRILinhaPesquisaDocente.rILinhaPesquisaDocente;
import static alpc.ufsc.querydsl.sql.QTbLinhaPesquisa.tbLinhaPesquisa;
import static alpc.ufsc.querydsl.sql.QViewPertencePrograma.viewPertencePrograma;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.hibernate.search.jpa.FullTextEntityManager;
import org.hibernate.search.jpa.Search;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageImpl;
import org.springframework.data.domain.Pageable;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.jpa.JPASubQuery;
import com.mysema.query.jpa.impl.JPADeleteClause;
import com.mysema.query.jpa.impl.JPAQuery;
import com.mysema.query.sql.SQLQuery;
import com.mysema.query.sql.SQLQueryFactory;
import com.mysema.query.sql.SQLSubQuery;
import com.mysema.query.support.Expressions;
import com.mysema.query.types.query.ListSubQuery;

import alpc.ufsc.entity.pessoa.atuacaoprofissional.linhapesquisa.LinhaPesquisaDocenteEntity;
import alpc.ufsc.entity.pessoa.atuacaoprofissional.linhapesquisa.LinhaPesquisaEntity;
import alpc.ufsc.entity.pessoa.atuacaoprofissional.linhapesquisa.QLinhaPesquisaDocenteEntity;
import alpc.ufsc.entity.pessoa.docente.DocenteEntity;
import alpc.ufsc.entity.programa.ProgramaEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.atuacaoprofissional.linhapesquisa.dto.LinhaPesquisaFilterDTO;
import alpc.ufsc.pessoa.atuacaoprofissional.linhapesquisa.dto.LinhaPesquisaRowDTO;
import alpc.ufsc.pessoa.docente.query.DocentePertenceProgramaQuery;
import alpc.ufsc.util.querydsl.Select;

```

```

@Service
@Transactional(propagation = Propagation.SUPPORTS)
public class LinhaPesquisaService {

    @PersistenceContext
    private EntityManager em;

    @Autowired
    private SQLQueryFactory queryFactory;

    @Autowired
    private DocentePertenceProgramaQuery docenteQuery;

    public void save(Long linhaPesquisaId, Long programaId, String titulo, Long docenteId) {
        if (linhaPesquisaId == null) {
            LinhaPesquisaEntity linhaPesquisa = new LinhaPesquisaEntity();
            linhaPesquisa.setTitulo(titulo);
            linhaPesquisa.setPrograma(this.em.getReference(ProgramaEntity.class,
                programaId));
            this.em.persist(linhaPesquisa);
            linhaPesquisaId = linhaPesquisa.getId();
            this.saveLinhaPesquisaDocente(linhaPesquisaId, docenteId);
        } else {
            QLinhaPesquisaDocenteEntity rLinhaPesquisaDocente =
                QLinhaPesquisaDocenteEntity.linhaPesquisaDocenteEntity;
            boolean docenteLinhaPesquisa = new
                JPASQuery(this.em).from(rLinhaPesquisaDocente)
                .where(rLinhaPesquisaDocente.linhaPesquisa().id.eq(linhaPesquisaId)
                    .and(rLinhaPesquisaDocente.docente().id.eq(docenteId)))
                .exists();
            if (!docenteLinhaPesquisa) {
                this.saveLinhaPesquisaDocente(linhaPesquisaId, docenteId);
            }
        }
    }

    private void saveLinhaPesquisaDocente(Long linhaPesquisaId, Long docenteId) {
        LinhaPesquisaDocenteEntity linhaPesquisaDocente = new
            LinhaPesquisaDocenteEntity();
        linhaPesquisaDocente.setDocente(this.em.getReference(DocenteEntity.class,
            docenteId));
        linhaPesquisaDocente.setLinhaPesquisa(this.em.getReference(LinhaPesquisaEntity.class,
            linhaPesquisaId));
        this.em.persist(linhaPesquisaDocente);
    }

    public Page<LinhaPesquisaRowDTO> getPage(LinhaPesquisaFilterDTO filter, Pageable
        pageable) {
        SQLQuery query = this.queryFactory.query().from(tbLinhaPesquisa);
        query.join(rLinhaPesquisaDocente).on(tbLinhaPesquisa.idLinhaPesquisa.eq(rLinhaPesquisaDocente
            query.join(this.docenteQuery.getSubPertencePrograma(filter.getProgramaId()),
                viewPertencePrograma)
                .on(rLinhaPesquisaDocente.idDocente.eq(viewPertencePrograma.idDocente)));

        if (filter.hasTitulo()) {
            query.where(tbLinhaPesquisa.tituloFiltro.like(QueryUtils.like(filter.getTitulo()));
        }

        query.where(tbLinhaPesquisa.idPrograma.eq(filter.getProgramaId()));
        query.groupBy(tbLinhaPesquisa.idLinhaPesquisa);
        query.orderBy(tbLinhaPesquisa.tituloFiltro.asc());

        query.offset(pageable.getOffset());
        query.limit(pageable.getPageSize());

        Select<LinhaPesquisaRowDTO> select = new
            Select<>(LinhaPesquisaRowDTO.class);
        select.as(tbLinhaPesquisa.idLinhaPesquisa, meta.id);
        select.as(tbLinhaPesquisa.titulo.max(), meta.titulo);
        select.as(tbLinhaPesquisa.count(), meta.numeroDocentes);

        return new PageImpl<>(query.list(select), pageable, this.getCount(filter));
    }

    private Long getCount(LinhaPesquisaFilterDTO filter) {
        SQLSubQuery subQueryCount =
            this.queryFactory.subQuery().from(tbLinhaPesquisa);
        subQueryCount.join(rLinhaPesquisaDocente).on(tbLinhaPesquisa.idLinhaPesquisa.eq(rLinhaPesqui
            subQueryCount.join(this.docenteQuery.getSubPertencePrograma(filter.getProgramaId()),
                viewPertencePrograma)
                .on(rLinhaPesquisaDocente.idDocente.eq(viewPertencePrograma.idDocente)));
    }
}

```

```

        if (filter.hasTitulo()) {
            subQueryCount.where(tbLinhaPesquisa.tituloFiltro.like(QueryUtils.like(filter.getTitulo())));
        }
        subQueryCount.where(tbLinhaPesquisa.idPrograma.eq(filter.getProgramaId()));

        ListSubQuery<Long> list =
            ↪ subQueryCount.distinct().list(tbLinhaPesquisa.idLinhaPesquisa);

        return this.queryFactory.query().from(list, Expressions.path(Void.class,
            ↪ "linha_pesquisa")).count();
    }

    @Transactional
    public void deleteLinhaPesquisa(Long programaId) {
        List<Long> ids = new JPAQuery(this.em).from(linhaPesquisaEntity)
            .where(linhaPesquisaEntity.programa().id.eq(programaId))
            .list(linhaPesquisaEntity.id);

        FullTextEntityManager search = Search.getFullTextEntityManager(this.em);
        for (Long id : ids) {
            search.purge(LinhaPesquisaEntity.class, id);
        }

        new JPADeleteClause(this.em, linhaPesquisaDocenteEntity)
            .where(linhaPesquisaDocenteEntity.linhaPesquisa().id.in(
                new JPASubQuery().from(linhaPesquisaEntity)
                    .where(linhaPesquisaEntity.programa().id.eq(programaId))
                    .list(linhaPesquisaEntity.id)))
            .execute();

        new JPADeleteClause(this.em, linhaPesquisaEntity)
            .where(linhaPesquisaEntity.programa().id.eq(programaId))
            .execute();
    }
}
package alpc.ufsc.pessoa.atuacaoprofissional.linhapesquisa.dto;

import lombok.Getter;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;

@Getter
@Setter
public class LinhaPesquisaFilterDTO {

    private String titulo;
    private Long programaId;

    public boolean hasTitulo() {
        return StringUtils.isNotBlank(this.titulo);
    }
}
package alpc.ufsc.pessoa.atuacaoprofissional.linhapesquisa.dto;

import lombok.Getter;
import lombok.Setter;
import br.ufsc.bridge.metafy.Metafy;

@Getter
@Setter
@Metafy
public class LinhaPesquisaRowDTO {

    private Long id;
    private String titulo;
    private Long numeroDocentes;
}
package alpc.ufsc.pessoa.atuacaoprofissional.linhapesquisa;

import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.Setter;

import org.apache.lucene.search.spell.NGramDistance;

@Getter
@Setter
@AllArgsConstructor
public class LinhaPesquisaSearchDto {

```



```

        private String titulo;
        private Long programaId;

        public boolean similar(Object titulo) {
            NGramDistance nGram = new NGramDistance(4);
            if (nGram.getDistance((String) titulo, this.titulo) < 0.90) {
                return false;
            }
            return true;
        }
    }

}
package alpc.ufsc.pessoa.atuacaoprofissional.projetopesquisa.dto;

import java.util.List;

import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.Setter;

import
↪ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetopesquisa.ProjetoPesquisaXm
import
↪ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetopesquisa.equipe.IntegrantesP

@Setter
@Getter
@AllArgsConstructor
public class ProjetoPesquisaDto {

    private Long projetoPesquisaId;
    private Long docenteId;
    private Long programaId;
    private String nome;
    private String projetoAnoInicio;
    private String projetoAnoFim;
    private String participacaoAnoInicio;
    private String participacaoAnoFim;
    private List<IntegrantesProjetoXmlDto> integrantesProjeto;

    public ProjetoPesquisaDto(
        Long projetoPesquisaId,
        Long docenteId,
        Long programaId,
        ProjetoPesquisaXmlDto projetoPesquisa,
        String participacaoAnoInicio,
        String participacaoAnoFim) {
        this.projetoPesquisaId = projetoPesquisaId;
        this.docenteId = docenteId;
        this.programaId = programaId;
        this.participacaoAnoInicio = participacaoAnoInicio;
        this.participacaoAnoFim = participacaoAnoFim;
        this.nome = projetoPesquisa.getNomeProjeto();
        this.projetoAnoInicio = projetoPesquisa.getAnoInicio();
        this.projetoAnoFim = projetoPesquisa.getAnoFim();
        this.integrantesProjeto =
            ↪ projetoPesquisa.getEquipeProjeto().getIntegrantesProjeto();
    }
}
package alpc.ufsc.pessoa.atuacaoprofissional.projetopesquisa.dto;

import lombok.Getter;
import lombok.Setter;
import br.ufsc.bridge.metafy.Metafy;

@Getter
@Setter
@Metafy
public class ProjetoPesquisaRowDTO {

    private Long id;
    private String nome;
    private Long numeroDocentes;
}
package alpc.ufsc.pessoa.atuacaoprofissional.projetopesquisa.dto;

import lombok.Getter;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;

```

```

@Getter
@Setter
public class ProjetoPesquisaFilterDTO {

    private String nome;
    private Long programaId;
    private Integer inicio;
    private Integer fim;

    public boolean hasInicio() {
        return this.inicio != null;
    }

    public boolean hasFim() {
        return this.fim != null;
    }

    public boolean hasNome() {
        return StringUtils.isNotBlank(this.nome);
    }
}

package alpc.ufsc.pessoa.atuacaoprofissional.projetoPesquisa.service;

import java.util.List;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.atuacaoprofissional.projetoPesquisa.dto.ProjetoPesquisaDto;
import alpc.ufsc.programa.dto.ImportProcessDto;
import
↪ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.ParticipacaoProjetoXmlDto;
import
↪ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.ProjetoPesquisaXmlDto;

@Slf4j
@Service
public class ProjetoPesquisaProcessService {

    @Autowired
    private ProjetoPesquisaSearchService projetoPesquisaSearch;

    @Autowired
    private ProjetoPesquisaService projetoPesquisaService;

    public void processProjetoPesquisa(ImportProcessDto processDto,
↪ ParticipacaoProjetoXmlDto participacaoProjetoXmlDto) {
        List<ProjetoPesquisaXmlDto> projetosPesquisas =
↪ participacaoProjetoXmlDto.getProjetoPesquisa();
        if (projetosPesquisas == null) {
            return;
        }
        for (ProjetoPesquisaXmlDto projetoPesquisa : projetosPesquisas) {
            try {
                Long projetoPesquisaId = this.search(projetoPesquisa, processDto);

                ProjetoPesquisaDto projetoPesquisaDto = new ProjetoPesquisaDto(
                    projetoPesquisaId,
                    processDto.getDocentelId(),
                    processDto.getProgramaDto().getId(),
                    projetoPesquisa,
                    participacaoProjetoXmlDto.getAnoInicio(),
                    participacaoProjetoXmlDto.getAnoFim());

                this.projetoPesquisaService.save(projetoPesquisaDto);
            } catch (Exception e) {
                log.error("Erro ao processar projeto de pesquisa", e.getMessage());
            }
        }
    }

    private Long search(ProjetoPesquisaXmlDto projetoPesquisa, ImportProcessDto processDto)
↪ {
        ProjetoPesquisaSearchDto searchDto = new ProjetoPesquisaSearchDto(
            projetoPesquisa.getNomeProjeto(),
            processDto.getProgramaDto().getId(),
            processDto.getCurriculo().getDadosGerais().getNomeCompleto(),
            processDto.getCurriculo().getDadosGerais().getNomeCitacoesBibliograficas().split(";"),
            processDto.getCurriculo().getNumeroIdentificador(),

```

```

        QueryUtils.toMinYear(projetoPesquisa.getAnoInicio()),
        QueryUtils.toMaxYear(projetoPesquisa.getAnoFim()));
    }
    return this.projetoPesquisaSearch.find(searchDto);
}
}
package alpc.ufsc.pessoa.atuacaoprofissional.projetoPesquisa.service;

import static
↳ alpc.ufsc.entity.pessoa.atuacaoprofissional.projetoPesquisa.QAutorProjetoPesquisaEntity.autorProjetoPesquisaEntity;
import static
↳ alpc.ufsc.entity.pessoa.atuacaoprofissional.projetoPesquisa.QProjetoPesquisaDocenteEntity.projetoPesquisaDocenteEntity;
import static
↳ alpc.ufsc.entity.pessoa.atuacaoprofissional.projetoPesquisa.QProjetoPesquisaEntity.projetoPesquisaEntity;
import static
↳ alpc.ufsc.pessoa.atuacaoprofissional.projetoPesquisa.dto.MProjetoPesquisaRowDTO.meta;
import static alpc.ufsc.querydsl.sql.QRProjetoPesquisaDocente.rProjetoPesquisaDocente;
import static alpc.ufsc.querydsl.sql.QTbProjetoPesquisa.tbProjetoPesquisa;
import static alpc.ufsc.querydsl.sql.QViewPertencePrograma.viewPertencePrograma;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.hibernate.search.jpa.FullTextEntityManager;
import org.hibernate.search.jpa.Search;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageImpl;
import org.springframework.data.domain.Pageable;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.jpa.JPASubQuery;
import com.mysema.query.jpa.impl.JPADeleteClause;
import com.mysema.query.jpa.impl.JPAQuery;
import com.mysema.query.sql.SQLQuery;
import com.mysema.query.sql.SQLQueryFactory;
import com.mysema.query.sql.SQLSubQuery;
import com.mysema.query.support.Expressions;
import com.mysema.query.types.query.ListSubQuery;

import alpc.ufsc.entity.pessoa.AutorEntity;
import alpc.ufsc.entity.pessoa.atuacaoprofissional.projetoPesquisa.AutorProjetoPesquisaEntity;
import alpc.ufsc.entity.pessoa.atuacaoprofissional.projetoPesquisa.ProjetoPesquisaDocenteEntity;
import alpc.ufsc.entity.pessoa.atuacaoprofissional.projetoPesquisa.ProjetoPesquisaEntity;
import alpc.ufsc.entity.pessoa.atuacaoprofissional.projetoPesquisa.QProjetoPesquisaDocenteEntity;
import alpc.ufsc.entity.pessoa.docente.DocenteEntity;
import alpc.ufsc.entity.programa.ProgramaEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.AutorService;
import alpc.ufsc.pessoa.atuacaoprofissional.projetoPesquisa.dto.ProjetoPesquisaDto;
import alpc.ufsc.pessoa.atuacaoprofissional.projetoPesquisa.dto.ProjetoPesquisaFilterDTO;
import alpc.ufsc.pessoa.atuacaoprofissional.projetoPesquisa.dto.ProjetoPesquisaRowDTO;
import alpc.ufsc.pessoa.docente.query.DocentePertenceProgramaQuery;
import alpc.ufsc.util.querydsl.Select;
import
↳ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.equipe.IntegrantesF

@Service
@Transactional(propagation = Propagation.SUPPORTS)
public class ProjetoPesquisaService {

    @PersistenceContext
    private EntityManager em;

    @Autowired
    private AutorService autorService;

    @Autowired
    private SQLQueryFactory queryFactory;

    @Autowired
    private DocentePertenceProgramaQuery docenteQuery;

    public void save(ProjetoPesquisaDto dto) {
        Long projetoPesquisaId = dto.getProjetoPesquisaId();
        Long docenteId = dto.getDocenteId();
        if (projetoPesquisaId != null) {

```

```

    } else {
        this.updateProjetoPesquisa(dto, projetoPesquisaId, docenteId);
    }
    ProjetoPesquisaEntity projetoPesquisaEntity =
        this.createProjetoPesquisa(dto);
    this.saveAutores(dto.getIntegrantesProjeto(), projetoPesquisaEntity.getId());
}

private void updateProjetoPesquisa(ProjetoPesquisaDto dto, Long projetoPesquisaId, Long
    docenteId) {
    ProjetoPesquisaEntity projetoPesquisa = this.em.find(ProjetoPesquisaEntity.class,
        projetoPesquisaId);
    this.save(dto, projetoPesquisa);
    if (!this.hasProjetoPesquisaDocente(projetoPesquisaId, docenteId)) {
        this.saveProjetoPesquisaDocente(dto);
    }
}

private boolean hasProjetoPesquisaDocente(Long projetoPesquisaId, Long docenteId) {
    QProjetoPesquisaDocenteEntity r1ProjetoPesquisaDocente =
        QProjetoPesquisaDocenteEntity.projetoPesquisaDocenteEntity;
    return new JPAQuery(this.em).from(r1ProjetoPesquisaDocente)
        .where(r1ProjetoPesquisaDocente.projetoPesquisa().id.eq(projetoPesquisaId)
            .and(r1ProjetoPesquisaDocente.docente().id.eq(docenteId)))
        .exists();
}

private ProjetoPesquisaEntity createProjetoPesquisa(ProjetoPesquisaDto dto) {
    ProjetoPesquisaEntity projetoPesquisa = new ProjetoPesquisaEntity();
    this.save(dto, projetoPesquisa);
    this.saveProjetoPesquisaDocente(dto);
    return projetoPesquisa;
}

private Long save(ProjetoPesquisaDto dto, ProjetoPesquisaEntity projetoPesquisa) {
    projetoPesquisa.setNome(dto.getNome());
    projetoPesquisa.setNomeFiltro(QueryUtils.lowerCaseStripAccents(dto.getNome()));
    projetoPesquisa.setAnoInicio(QueryUtils.toMinYear(dto.getProjetoAnoInicio()));
    projetoPesquisa.setAnoFim(QueryUtils.toMaxYear(dto.getProjetoAnoFim()));
    projetoPesquisa.setPrograma(this.em.getReference(ProgramaEntity.class,
        dto.getProgramaId()));
    this.em.persist(projetoPesquisa);
    dto.setProjetoPesquisaId(projetoPesquisa.getId());
    return projetoPesquisa.getId();
}

private void saveProjetoPesquisaDocente(ProjetoPesquisaDto dto) {
    ProjetoPesquisaDocenteEntity projetoPesquisaDocente = new
        ProjetoPesquisaDocenteEntity();
    projetoPesquisaDocente.setDocente(this.em.getReference(DocenteEntity.class,
        dto.getDocenteId()));
    projetoPesquisaDocente.setProjetoPesquisa(this.em.getReference(ProjetoPesquisaEntity.class,
        dto.getProjetoPesquisaId()));
    projetoPesquisaDocente.setAnoInicioParticipacao(QueryUtils.toMinYear(dto.getParticipacaoAnoInicio()));
    projetoPesquisaDocente.setAnoFimParticipacao(QueryUtils.toMaxYear(dto.getParticipacaoAnoFim()));
    this.em.persist(projetoPesquisaDocente);
}

private void saveAutores(List<IntegrantesProjetoXmlDto> integrantes, Long
    idProjetoPesquisa) {
    for (IntegrantesProjetoXmlDto integrante : integrantes) {
        Long autorId = this.autorService.saveAutor(integrante);
        this.saveAutorProducao(idProjetoPesquisa, autorId);
    }
}

private void saveAutorProducao(Long idProjetoPesquisa, Long autorId) {
    if (autorId != null) {
        AutorProjetoPesquisaEntity autorProducaoEntity = new
            AutorProjetoPesquisaEntity();
        autorProducaoEntity.setAutor(this.em.getReference(AutorEntity.class,
            autorId));
        autorProducaoEntity.setProjetoPesquisa(this.em.getReference(ProjetoPesquisaEntity.class,
            idProjetoPesquisa));
        this.em.persist(autorProducaoEntity);
    }
}

public Page<ProjetoPesquisaRowDTO> getPage(ProjetoPesquisaFilterDTO filter, Pageable
    pageable) {
    SQLQuery query = this.queryFactory.query().from(tbProjetoPesquisa);
    query.join(r1ProjetoPesquisaDocente).on(r1ProjetoPesquisaDocente.idProjetoPesquisa.eq(tbProjetoPesquisa.id

```

```

query.join(this.docenteQuery.getSubPertencePrograma(filter.getProgramaId()),
    ↪ viewPertencePrograma)
    .on(viewPertencePrograma.idDocente.eq(rlProjetoPesquisaDocente.idDocente));

if (filter.hasNome()) {
    query.where(tbProjetoPesquisa.nomeFiltro.like(QueryUtils.like(filter.getNome())));
}
if (filter.hasInicio() || filter.hasFim()) {
    query.where(QueryUtils.overlappingInterval(tbProjetoPesquisa.anoInicio,
    ↪ tbProjetoPesquisa.anoFim, filter.getInicio(), filter.getFim()));
}

query.where(tbProjetoPesquisa.idPrograma.eq(filter.getProgramaId()));
query.groupBy(tbProjetoPesquisa.idProjetoPesquisa);
query.orderBy(tbProjetoPesquisa.nomeFiltro.asc());

query.offset(pageable.getOffset());
query.limit(pageable.getPageSize());

Select<ProjetoPesquisaRowDTO> select = new
    ↪ Select<>(ProjetoPesquisaRowDTO.class);
select.as(tbProjetoPesquisa.idProjetoPesquisa, meta.id);
select.as(tbProjetoPesquisa.nome.max(), meta.nome);
select.as(rlProjetoPesquisaDocente.idDocente.count(), meta.numeroDocentes);

return new PageImpl<>(query.list(select), pageable, this.getCount(filter));
}

private Long getCount(ProjetoPesquisaFilterDTO filter) {
    SQLSubQuery SubQueryCount =
    ↪ this.queryFactory.subQuery().from(tbProjetoPesquisa);
    SubQueryCount.join(rlProjetoPesquisaDocente).on(rlProjetoPesquisaDocente.idProjetoPesquisa.eq(
    SubQueryCount.join(this.docenteQuery.getSubPertencePrograma(filter.getProgramaId()),
    ↪ viewPertencePrograma)
    .on(viewPertencePrograma.idDocente.eq(rlProjetoPesquisaDocente.idDocente));

    if (filter.hasNome()) {
        SubQueryCount.where(tbProjetoPesquisa.nome.like(QueryUtils.like(filter.getNome())));
    }
    if (filter.hasInicio() || filter.hasFim()) {
        SubQueryCount.where(QueryUtils.overlappingInterval(tbProjetoPesquisa.anoInicio,
    ↪ tbProjetoPesquisa.anoFim, filter.getInicio(), filter.getFim()));
    }
    SubQueryCount.where(tbProjetoPesquisa.idPrograma.eq(filter.getProgramaId()));

    ListSubQuery<Long> list =
    ↪ SubQueryCount.distinct().list(tbProjetoPesquisa.idProjetoPesquisa);
    return this.queryFactory.query().from(list, Expressions.path(Void.class,
    ↪ "projeto").count());
}

@Transactional
public void deleteProjetoPesquisa(Long programaId) {
    List<Long> ids = new JPAQuery(this.em).from(autorProjetoPesquisaEntity)
        .join(autorProjetoPesquisaEntity.projetoPesquisa(),
    ↪ projetoPesquisaEntity)
        .where(projetoPesquisaEntity.programa().id.eq(programaId))
        .list(autorProjetoPesquisaEntity.id);

    FullTextEntityManager search = Search.getFullTextEntityManager(this.em);
    for (Long id : ids) {
        search.purge(AutorProjetoPesquisaEntity.class, id);
    }

    ListSubQuery<Long> subQuery = new JPASubQuery().from(projetoPesquisaEntity)
        .where(projetoPesquisaEntity.programa().id.eq(programaId))
        .list(projetoPesquisaEntity.id);

    new JPADeleteClause(this.em, projetoPesquisaDocenteEntity)
        .where(projetoPesquisaDocenteEntity.projetoPesquisa().id.in(subQuery))
        .execute();

    new JPADeleteClause(this.em, autorProjetoPesquisaEntity)
        .where(autorProjetoPesquisaEntity.projetoPesquisa().id.in(subQuery))
        .execute();

    new JPADeleteClause(this.em, projetoPesquisaEntity)
        .where(projetoPesquisaEntity.programa().id.eq(programaId))
        .execute();
}
}
package alpc.ufsc.pessoa.atuacaoprofissional.projetoPesquisa.service;

```

```

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.apache.lucene.search.BooleanClause.Occur;
import org.apache.lucene.search.BooleanQuery;
import org.hibernate.search.SearchFactory;
import org.hibernate.search.errors.EmptyQueryException;
import org.hibernate.search.jpamodels.FullTextEntityManager;
import org.hibernate.search.jpamodels.FullTextQuery;
import org.hibernate.search.jpamodels.Search;
import org.hibernate.search.query.dsl.QueryBuilder;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import alpc.ufsc.entity.pessoa.atuacaoprofissional.projetoPesquisa.AutorProjetoPesquisaEntity;

@Service
@Transactional(propagation = Propagation.SUPPORTS, noRollbackFor =
    ↳ EmptyQueryException.class)
public class ProjetoPesquisaSearchService {

    private static final String NOME = "projetoPesquisa.nome";
    private static final String ANO_INICIO = "projetoPesquisa.ano_inicio";
    private static final String ANO_FIM = "projetoPesquisa.ano_fim";
    private static final String PROGRAMA_ID = "projetoPesquisa.programa.id";
    private static final String NRO_ID_AUTOR = "autor.numero_identificador";
    private static final String NOME_AUTOR = "autor.nome";
    private static final String NOME_CITACAO = "autor.citacao";

    private static final String ID = "projetoPesquisa.id";

    @PersistenceContext
    private EntityManager em;

    public Long find(ProjetoPesquisaSearchDto searchDto) {
        FullTextEntityManager search = Search.getFullTextEntityManager(this.em);
        search.flushToIndexes();
        SearchFactory searchFactory = search.getSearchFactory();
        QueryBuilder qb =
            ↳ searchFactory.buildQueryBuilder().forEntity(AutorProjetoPesquisaEntity.class).get();

        BooleanQuery query = new BooleanQuery();

        BooleanQuery queryProjetoPesquisa = this.queryProjetoPesquisa(searchDto, qb);
        query.add(queryProjetoPesquisa, Occur.MUST);

        BooleanQuery queryNome = this.queryAutor(searchDto, qb);
        query.add(queryNome, Occur.MUST);

        FullTextQuery fullTextQuery = search.createFullTextQuery(query,
            ↳ AutorProjetoPesquisaEntity.class);

        fullTextQuery.setProjection(this.getProjection());
        fullTextQuery.setMaxResults(5);

        List<?> result = fullTextQuery.getResultList();
        for (Object object : result) {
            Object[] fields = (Object[]) object;
            if (searchDto.similar(fields[0])) {
                return (Long) fields[1];
            }
        }
        return null;
    }

    private BooleanQuery queryAutor(ProjetoPesquisaSearchDto searchDto, QueryBuilder qb) {
        BooleanQuery queryNome = new BooleanQuery();
        queryNome.add(qb.keyword().onField(NRO_ID_AUTOR).matching(searchDto.getNumeroIdentificador()).createQuery(),
            ↳ Occur.SHOULD);
        queryNome.add(qb.keyword().onField(NOME_AUTOR).matching(searchDto.getNomeAutor()).createQuery(),
            ↳ Occur.SHOULD);
        for (String nomeCitacao : searchDto.getNomesCitacao()) {
            queryNome.add(qb.keyword().onField(NOME_CITACAO).matching(nomeCitacao).createQuery(),
                ↳ Occur.SHOULD);
        }
        return queryNome;
    }
}

```

```

private BooleanQuery queryProjetoPesquisa(ProjetoPesquisaSearchDto searchDto,
↳ QueryBuilder qb) {
    BooleanQuery queryProjetoPesquisa = new BooleanQuery();
    queryProjetoPesquisa.add(qb.keyword().onField(NOME).matching(searchDto.getNomeProjeto()).createQuery()
↳ Occur.MUST);
    Integer anoInicio = searchDto.getAnoInicio();
    if (anoInicio != null) {
        queryProjetoPesquisa.add(qb.keyword().onField(ANO_INICIO).matching(anoInicio).createQuery()
↳ Occur.MUST);
    }
    Integer anoFim = searchDto.getAnoFim();
    if (anoFim != null) {
        queryProjetoPesquisa.add(qb.keyword().onField(ANO_FIM).matching(anoFim).createQuery()
↳ Occur.MUST);
    }
    queryProjetoPesquisa.add(qb.keyword().onField(PROGRAMA_ID).matching(searchDto.getProgramaId()).createQuery()
↳ Occur.MUST);
    return queryProjetoPesquisa;
}

private String[] getProjection() {
    String[] p = { NOME, ID };
    return p;
}
}

package alpc.ufsc.pessoa.atuacaoprofissional.projetoPesquisa.service;

import org.apache.lucene.search.spell.NGramDistance;

import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
@AllArgsConstructor
public class ProjetoPesquisaSearchDto {

    private String nomeProjeto;
    private Long programaId;
    private String nomeAutoc;
    private String[] nomesCitacao;
    private String numerIdentificador;
    private Integer anoInicio;
    private Integer anoFim;

    public boolean similar(Object titulo) {
        NGramDistance nGram = new NGramDistance(4);
        if (nGram.getDistance((String) titulo, this.nomeProjeto) < 0.90) {
            return false;
        }
        return true;
    }
}

package alpc.ufsc.pessoa.atuacaoprofissional.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import alpc.ufsc.domain.instituicao.dto.InstituicaoSaveDTO;
import alpc.ufsc.entity.pessoa.atuacaoprofissional.TipoAtuacaoProfissional;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.atuacaoprofissional.linhapesquisa.LinhaPesquisaProcessService;
import alpc.ufsc.pessoa.atuacaoprofissional.projetoPesquisa.service.ProjetoPesquisaProcessService;
import alpc.ufsc.programa.dto.ImportProcessDto;
import info.debatty.java.stringsimilarity.JaroWinkler;
import ufc.alpc.xml.dto.dados.gerais.atuacoes.AtuacaoProfissionalXmlDto;
import ufc.alpc.xml.dto.dados.gerais.atuacoes.AtuacoesProfissionaisXmlDto;
import ufc.alpc.xml.dto.dados.gerais.atuacoes.atividades.AtividadeXmlDto;
import ufc.alpc.xml.dto.dados.gerais.atuacoes.atividades.conselhocomissao.AtividadesConselhoComissaoConsultoriaXmlDto;
import ufc.alpc.xml.dto.dados.gerais.atuacoes.atividades.conselhocomissao.ConselhoComissaoConsultoriaXmlDto;
import ufc.alpc.xml.dto.dados.gerais.atuacoes.atividades.direcaoadministracao.AtividadesDirecaoAdministracaoXmlDto;
import ufc.alpc.xml.dto.dados.gerais.atuacoes.atividades.direcaoadministracao.DirecaoAdministracaoXmlDto;

```

```

↳ ufdc.alpc.xml.dto.dados.gerais.atuacoes.atividades.direcaoadministracao.DirecaoAdministracaoXmlDto;
import ufdc.alpc.xml.dto.dados.gerais.atuacoes.atividades.ensino.AtividadesEnsinoXmlDto;
import ufdc.alpc.xml.dto.dados.gerais.atuacoes.atividades.extensaouniversitaria.AtividadesExtensaoUniversitariaXmlDto;
import ufdc.alpc.xml.dto.dados.gerais.atuacoes.atividades.extensaouniversitaria.ExtensaoUniversitariaXmlDto;
import ufdc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.AtividadesParticipacaoProjetoXmlDto;
import ufdc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.ParticipacaoProjetoXmlDto;
import ufdc.alpc.xml.dto.dados.gerais.atuacoes.atividades.pesquisadesenvolvimento.AtividadesPesquisaDesenvolvimentoXmlDto;
import ufdc.alpc.xml.dto.dados.gerais.atuacoes.atividades.pesquisadesenvolvimento.PesquisaDesenvolvimentoXmlDto;
import ufdc.alpc.xml.dto.dados.gerais.atuacoes.atividades.pesquisadesenvolvimento.ensino.EnsinoXmlDto;
import ufdc.alpc.xml.dto.dados.gerais.atuacoes.atividades.servicotecnico.AtividadesServicoTecnicoEspecializadoXmlDto;
import ufdc.alpc.xml.dto.dados.gerais.atuacoes.atividades.servicotecnico.ServicoTecnicoEspecializadoXmlDto;
import ufdc.alpc.xml.dto.dados.gerais.atuacoes.atividades.treinamentoministrado.AtividadesTreinamentoMinistradoXmlDto;
import ufdc.alpc.xml.dto.dados.gerais.atuacoes.atividades.treinamentoministrado.TreinamentoMinistradoXmlDto;
import ufdc.alpc.xml.dto.dados.gerais.atuacoes.outras.atividadestecnicocientifica.OutraAtividadeTecnicoCientificaXmlDto;
import ufdc.alpc.xml.dto.dados.gerais.atuacoes.outras.atividadestecnicocientifica.OutrasAtividadesTecnicoCientificaXmlDto;

```

```

@Service
@Transactional
public class AtuacaoProfissionalProcessService {

    @Autowired
    private AtuacaoProfissionalService atuacaoService;

    @Autowired
    private LinhaPesquisaProcessService linhaPesquisaService;

    @Autowired
    private ProjetoPesquisaProcessService projetoPesquisaService;

    public void processAtuacoes(ImportProcessDto processDto) {
        this.atuacaoService.clearAtuacoes(processDto.getId());
        AtuacoesProfissionaisXmlDto atuacoesProfissionais =
            ↳ processDto.getCurriculo().getDadosGerais().getAtuacoesProfissionais();
        List<AtuacaoProfissionalXmlDto> atuacoes = atuacoesProfissionais.getAtuacoes();
        for (AtuacaoProfissionalXmlDto atuacao : atuacoes) {
            InstituicaoSaveDTO instituicao =
                ↳ processDto.getProgramaDto().getInstituicao();
            if (this.isInstituicaoFromPrograma(instituicao, atuacao)) {
                this.processAtuacoes(processDto, atuacao);
                return;
            }
        }
    }

    private boolean isInstituicaoFromPrograma(InstituicaoSaveDTO instituicao,
        ↳ AtuacaoProfissionalXmlDto atuacao) {
        JaroWinkler jaroWinkler = new JaroWinkler();
        ↳ jaroWinkler.similarity(QueryUtils.lowerCaseStripAccents(atuacao.getNomeInstituicao()),
        ↳ QueryUtils.lowerCaseStripAccents(instituicao.getNome())) > 0.98
        ||
        ↳ jaroWinkler.similarity(QueryUtils.lowerCaseStripAccents(atuacao.getNomeInstituicao()),
        ↳ QueryUtils.lowerCaseStripAccents(instituicao.getSigla()))
        ↳ >= 0.99;
    }

    private void processAtuacoes(ImportProcessDto processDto, AtuacaoProfissionalXmlDto
        ↳ atuacao) {
        this.processConselhoComissao(atuacao, processDto);
        this.processDirecaoAdministrativa(atuacao, processDto);
        this.processEnsino(atuacao, processDto);
        this.processExtensaoUniversitaria(atuacao, processDto);
        this.processParticipacaoProjeto(atuacao, processDto);
        this.processPesquisaDesenvolvimento(atuacao, processDto);
        this.processServicoTecnicoEspecializado(atuacao, processDto);
        this.processTreinamentoMinistrado(atuacao, processDto);
        this.processOutraAtividade(atuacao, processDto);
    }
}

```



```

private void processConselhoComissao(AtuacaoProfissionalXmlDto atuacao,
↪ ImportProcessDto processDto) {
    AtividadesConselhoComissaoConsultoriaXmlDto
    ↪ atividadesConselhoComissaoConsultoria =
    ↪ atuacao.getAtividadesConselhoComissaoConsultoria();
    if (atividadesConselhoComissaoConsultoria != null) {
        List<ConselhoComissaoConsultoriaXmlDto> conselhoComissaoConsultoria
        ↪ =
        ↪ atividadesConselhoComissaoConsultoria.getConselhoComissaoConsultoria();
        for (ConselhoComissaoConsultoriaXmlDto conselhoComissaoConsultoriaDto
        ↪ : conselhoComissaoConsultoria) {
            this.processAtuacao(processDto, conselhoComissaoConsultoriaDto,
            ↪ TipoAtuacaoProfissional.CONSELHO_COMISSAO_CONSULTORIA);
        }
    }
}

private void processDirecaoAdministrativa(AtuacaoProfissionalXmlDto atuacao,
↪ ImportProcessDto processDto) {
    AtividadesDirecaoAdministracaoXmlDto atividadesDirecaoAdministracao =
    ↪ atuacao.getAtividadesDirecaoAdministracao();
    if (atividadesDirecaoAdministracao != null) {
        List<DirecaoAdministracaoXmlDto> direcoesAdministracoes =
        ↪ atividadesDirecaoAdministracao.getDirecaoAdministracao();
        for (DirecaoAdministracaoXmlDto direcaoAdministracaoXmlDto :
        ↪ direcoesAdministracoes) {
            this.processAtuacao(processDto, direcaoAdministracaoXmlDto,
            ↪ TipoAtuacaoProfissional.DIRECAO_ADMINISTRACAO);
        }
    }
}

private void processEnsino(AtuacaoProfissionalXmlDto atuacao, ImportProcessDto
↪ processDto) {
    AtividadesEnsinoXmlDto atividadesEnsino = atuacao.getAtividadesEnsino();
    if (atividadesEnsino != null) {
        List<EnsinoXmlDto> ensinos = atividadesEnsino.getEnsino();
        for (EnsinoXmlDto ensinoXmlDto : ensinos) {
            this.processAtuacao(processDto, ensinoXmlDto,
            ↪ TipoAtuacaoProfissional.ENSINO);
        }
    }
}

private void processExtensaoUniversitaria(AtuacaoProfissionalXmlDto atuacao,
↪ ImportProcessDto processDto) {
    AtividadesExtensaoUniversitariaXmlDto atividadesExtensaoUniversitaria =
    ↪ atuacao.getAtividadesExtensaoUniversitaria();
    if (atividadesExtensaoUniversitaria != null) {
        List<ExtensaoUniversitariaXmlDto> extensoesUniversitarias =
        ↪ atividadesExtensaoUniversitaria.getExtensoesUniversitarias();
        for (ExtensaoUniversitariaXmlDto extensaoUniversitariaXmlDto :
        ↪ extensoesUniversitarias) {
            this.processAtuacao(processDto, extensaoUniversitariaXmlDto,
            ↪ TipoAtuacaoProfissional.EXTENSAO_UNIVERSITARIA);
        }
    }
}

private void processParticipacaoProjeto(AtuacaoProfissionalXmlDto atuacao,
↪ ImportProcessDto processDto) {
    AtividadesParticipacaoProjetoXmlDto atividadesParticipacaoProjeto =
    ↪ atuacao.getAtividadesParticipacaoProjeto();
    if (atividadesParticipacaoProjeto != null) {
        List<ParticipacaoProjetoXmlDto> participacaoProjeto =
        ↪ atividadesParticipacaoProjeto.getParticipacaoProjeto();
        for (ParticipacaoProjetoXmlDto participacaoProjetoXmlDto :
        ↪ participacaoProjeto) {
            this.processAtuacao(processDto, participacaoProjetoXmlDto,
            ↪ TipoAtuacaoProfissional.PARTICIPACAO_PROJETO);
            this.projetoPesquisaService.processProjetoPesquisa(processDto,
            ↪ participacaoProjetoXmlDto);
        }
    }
}

private void processPesquisaDesenvolvimento(AtuacaoProfissionalXmlDto atuacao,
↪ ImportProcessDto processDto) {
    AtividadesPesquisaDesenvolvimentoXmlDto atividadesPesquisaDesenvolvimento =
    ↪ atuacao.getAtividadesPesquisaDesenvolvimento();
    if (atividadesPesquisaDesenvolvimento != null) {
        List<PesquisaDesenvolvimentoXmlDto> pesquisasDesenvolvimentos =

```

```

        ↪ atividadesPesquisaDesenvolvimento.getPesquisasDesenvolvimentos();
    for (PesquisaDesenvolvimentoXmlDto pesquisaDesenvolvimentoXmlDto :
        ↪ pesquisasDesenvolvimentos) {
        this.processAtuacao(processDto, pesquisaDesenvolvimentoXmlDto,
        ↪ TipoAtuacaoProfissional.PESQUISA_DESENVOLVIMENTO);
        this.linhaPesquisaService.processLinhaPesquisa(processDto.getDocenteId(),
        ↪ processDto.getProgramaDto().getId(),
        ↪ pesquisaDesenvolvimentoXmlDto);
    }
}

private void processServicoTecnicoEspecializado(AtuacaoProfissionalXmlDto atuacao,
    ↪ ImportProcessDto processDto) {
    AtividadesServicoTecnicoEspecializadoXmlDto
    ↪ atividadesServicoTecnicoEspecializado =
    ↪ atuacao.getAtividadesServicoTecnicoEspecializado();
    if (atividadesServicoTecnicoEspecializado != null) {
        List<ServicoTecnicoEspecializadoXmlDto> servicosTecnicoEspecializado =
        ↪ atividadesServicoTecnicoEspecializado.getServicosTecnicoEspecializado();
        for (ServicoTecnicoEspecializadoXmlDto servicoTecnicoEspecializadoXmlDto
        ↪ : servicosTecnicoEspecializado) {
            this.processAtuacao(processDto,
            ↪ servicoTecnicoEspecializadoXmlDto,
            ↪ TipoAtuacaoProfissional.SERVICO_TECNICO_ESPECIALIZADO);
        }
    }
}

private void processTreinamentoMinistrado(AtuacaoProfissionalXmlDto atuacao,
    ↪ ImportProcessDto processDto) {
    AtividadesTreinamentoMinistradoXmlDto atividadesTreinamentoMinistrado =
    ↪ atuacao.getAtividadesTreinamentoMinistrado();
    if (atividadesTreinamentoMinistrado != null) {
        List<TreinamentoMinistradoXmlDto> treinamentosMinistrados =
        ↪ atividadesTreinamentoMinistrado.getTreinamentosMinistrados();
        for (TreinamentoMinistradoXmlDto treinamentoMinistradoXmlDto :
        ↪ treinamentosMinistrados) {
            this.processAtuacao(processDto, treinamentoMinistradoXmlDto,
            ↪ TipoAtuacaoProfissional.TREINAMENTO_MINISTRADO);
        }
    }
}

private void processOutraAtividade(AtuacaoProfissionalXmlDto atuacao, ImportProcessDto
    ↪ processDto) {
    OutrasAtividadesTecnicoCientificaXmlDto outrasAtividadesTecnicoCientifica =
    ↪ atuacao.getOutrasAtividadesTecnicoCientifica();
    if (outrasAtividadesTecnicoCientifica != null) {
        List<OutraAtividadeTecnicoCientificaXmlDto>
        ↪ outrasAtividadesTecnicoCientificas =
        ↪ outrasAtividadesTecnicoCientifica.getOutrasAtividadesTecnicoCientifica();
        for (OutraAtividadeTecnicoCientificaXmlDto
        ↪ outraAtividadeTecnicoCientificaXmlDto :
        ↪ outrasAtividadesTecnicoCientificas) {
            this.processAtuacao(processDto,
            ↪ outraAtividadeTecnicoCientificaXmlDto,
            ↪ TipoAtuacaoProfissional.OUTRA_ATIVIDADE_TECNICO_CIENTIFICA);
        }
    }
}

private void processAtuacao(ImportProcessDto processDto, AtividadeXmlDto atividade,
    ↪ TipoAtuacaoProfissional tipoAtuacao) {
    this.atuacaoService.save(processDto.getDocenteId(), atividade, tipoAtuacao);
}

package alpc.ufsc.pessoa.atuacaoprofissional.service;

import static alpc.ufsc.pessoa.producao.dto.MQuantidadeRowDTO.meta;
import static alpc.ufsc.querydsl.sql.QTbAtuacaoProfissional.tbAtuacaoProfissional;
import static alpc.ufsc.querydsl.sql.QViewPertencePrograma.viewPertencePrograma;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

```

```

import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.jpaa.impl.JPADeleteClause;
import com.mysema.query.sql.SQLQuery;
import com.mysema.query.sql.SQLQueryFactory;

import alpc.ufsc.common.PeriodoFilterDTO;
import alpc.ufsc.entity.pessoa.atuacaoprofissional.AtuacaoProfissionalEntity;
import alpc.ufsc.entity.pessoa.atuacaoprofissional.QAtuacaoProfissionalEntity;
import alpc.ufsc.entity.pessoa.atuacaoprofissional.TipoAtuacaoProfissional;
import alpc.ufsc.entity.pessoa.docente.DocenteEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.pessoa.docente.query.DocentePertenceProgramaQuery;
import alpc.ufsc.pessoa.producao.dto.QuantidadeRowDTO;
import alpc.ufsc.util.querydsl.Select;
import ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.AtividadeXmlDto;

@Slf4j
@Service
@Transactional
public class AtuacaoProfissionalService {

    @PersistenceContext
    private EntityManager em;

    @Autowired
    private SQLQueryFactory queryFactory;

    @Autowired
    private DocentePertenceProgramaQuery docenteQuery;

    public void save(Long docenteId, AtividadeXmlDto atividade, TipoAtuacaoProfissional
        tipoAtuacao) {
        try {
            AtuacaoProfissionalEntity atuacao = new AtuacaoProfissionalEntity();
            atuacao.setDocente(this.em.getReference(DocenteEntity.class, docenteId));
            atuacao.setTipoAtuacao(tipoAtuacao);
            atuacao.setAnoFim(QueryUtils.toMaxYear(atividade.getAnoFim()));
            atuacao.setAnoInicio(QueryUtils.toMinYear(atividade.getAnoInicio()));
            atuacao.setMesFim(QueryUtils.parseInt(atividade.getMesFim()));
            atuacao.setMesInicio(QueryUtils.parseInt(atividade.getMesInicio()));
            this.em.persist(atuacao);
        } catch (NumberFormatException e) {
            log.error("Erro ao converter ms/ano", e.getMessage());
        }
    }

    public void clearAtuacoes(Long docenteId) {
        QAtuacaoProfissionalEntity atuacaoProfissionalEntity =
            new JPADeleteClause(this.em,
                atuacaoProfissionalEntity.where(atuacaoProfissionalEntity.docente().id.eq(docenteId)).ex
            );
    }

    public List<QuantidadeRowDTO> getAtuacaoProfissional(PeriodoFilterDTO filter, Long
        programaId) {
        SQLQuery query = this.queryFactory.query().from(tbAtuacaoProfissional);
        query.join(this.docenteQuery.getSubPertencePrograma(programaId),
            viewPertencePrograma
                .on(viewPertencePrograma.idDocente.eq(tbAtuacaoProfissional.idDocente));

        if (filter.hasInicio() || filter.hasFim()) {
            query.where(QueryUtils.overlappingInterval(tbAtuacaoProfissional.anoInicio,
                tbAtuacaoProfissional.anoFim, filter.getInicio(), filter.getFim()));
        }
        query.groupBy(tbAtuacaoProfissional.tipoAtuacaoProfissional);

        Select<QuantidadeRowDTO> select = new Select<>(QuantidadeRowDTO.class);
        select.as(tbAtuacaoProfissional.tipoAtuacaoProfissional, meta.id);
        select.as(tbAtuacaoProfissional.idDocente.count(), meta.quantidade);

        return query.list(select);
    }
}

}package alpc.ufsc.pessoa;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

```

```

import org.springframework.stereotype.Service;

import alpc.ufsc.entity.pessoa.AutorEntity;
import ufsc.alpc.xml.dto.common.AutorXmlDto;
import
↪ ufsc.alpc.xml.dto.dados.gerais.atuacoes.atividades.participacaoprojeto.projetoPesquisa.equipe.IntegrantesProjetoXm

@Service
public class AutorService {

    @PersistenceContext
    private EntityManager em;

    public Long saveAutor(AutorXmlDto autorXmlDto) {
        AutorEntity autor = new AutorEntity();
        autor.setNome(this.getNomeAutor(autorXmlDto.getNomeCompleto()));
        autor.setCitacao(autorXmlDto.getNomeCitacao());
        autor.setNumeroIdentificador(autorXmlDto.getNroIdCnpq());
        this.em.persist(autor);
        return autor.getId();
    }

    public Long saveAutor(IntegrantesProjetoXmlDto integrante) {
        AutorEntity autor = new AutorEntity();
        autor.setNome(this.getNomeAutor(integrante.getNomeCompleto()));
        autor.setCitacao(integrante.getNomeCitacao());
        autor.setNumeroIdentificador(integrante.getNroIdCnpq());
        this.em.persist(autor);
        return autor.getId();
    }

    private String getNomeAutor(String nomeAutor) {
        String pattern = "(.*)\\.(.*)";
        Pattern r = Pattern.compile(pattern);
        Matcher m = r.matcher(nomeAutor);
        if (m.find()) {
            return m.group(3).trim() + " " + m.group(1).trim();
        }
        return nomeAutor;
    }
}

package alpc.ufsc;

import org.springframework.boot.autoconfigure.EnableAutoConfiguration;
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.Import;

import alpc.ufsc.config.cache.SystemCacheConfiguration;
import alpc.ufsc.config.database.DatabaseConfiguration;
import alpc.ufsc.config.web.UploadConfig;
import alpc.ufsc.config.web.WebMvcConfig;

@Configuration
@EnableAutoConfiguration
@ComponentScan(Application.BASE_PACKAGE)
@Import({ DatabaseConfiguration.class, SystemCacheConfiguration.class, WebMvcConfig.class,
↪ UploadConfig.class })
public class Application {

    public static final String BASE_PACKAGE = "alpc.ufsc";
}

package alpc.ufsc.util;

import org.apache.commons.lang3.StringUtils;

import com.fasterxml.jackson.databind.ObjectMapper;

public class JsonObjectMapper {

    public static <T> T toObject(String jsonObject, Class<T> clazz) {
        if (StringUtils.isBlank(jsonObject)) {
            return getInstance(clazz);
        } else {
            try {
                return new ObjectMapper().readValue(jsonObject, clazz);
            } catch (Exception e) {
                return getInstance(clazz);
            }
        }
    }
}

```

```

    }

    private static <T> T getInstance(Class<T> clazz) {
        try {
            return clazz.newInstance();
        } catch (InstantiationException | IllegalAccessException e) {
            throw new RuntimeException(e);
        }
    }
}

package alpc.ufsc.util;

import java.io.Serializable;
import java.util.Iterator;

import lombok.Getter;

import alpc.ufsc.util.MinMaxPagination.FromTo;

@Getter
public class MinMaxPagination implements Serializable, Iterable<FromTo> {
    private static final long serialVersionUID = -7009694609672804274L;

    private Long min;
    private Integer batch;
    private Integer pages;
    private Long total;

    public MinMaxPagination(Long min, Long max, Integer batch) {
        if (min != null) {
            this.total = max - min + 1;
            this.pages = (int) Math.ceil(this.total / (double) batch);
        } else {
            this.total = 0L;
            this.pages = 0;
        }
        this.min = min;
        this.batch = batch;
    }

    public boolean hasPages() {
        return this.pages > 0;
    }

    public void setPages(Integer pages) {
        this.pages = pages;
        this.total = (long) pages * this.batch;
    }

    public class FromTo {
        public Long from;
        public Long to;

        public FromTo(Integer page, Long min, Integer batch) {
            this.from = page * batch + min;
            this.to = this.from + batch - 1;
        }
    }

    @Override
    public Iterator<FromTo> iterator() {
        return new Itr();
    }

    private class Itr implements Iterator<FromTo> {
        private Integer currentPage = 0;

        @Override
        public boolean hasNext() {
            return this.currentPage < MinMaxPagination.this.pages;
        }

        @Override
        public FromTo next() {
            if (this.currentPage < MinMaxPagination.this.pages) {
                return new FromTo(this.currentPage++,
                    ↪ MinMaxPagination.this.min,
                    ↪ MinMaxPagination.this.batch);
            }
            return null;
        }
    }
}

```

```

        @Override
        public void remove() {
        }
    }
}package alpc.ufsc.util;

public class NumberUtilsInternal {
    public static int compareTo(Integer one, Integer other) {
        if (one == null && other == null) {
            return 0;
        } else if (one == null) {
            return -1;
        } else if (other == null) {
            return +1;
        }
        return one.compareTo(other);
    }
}
package alpc.ufsc.util.querydsl;

import com.mysema.query.types.Expression;

public interface SelectExpression<T> {
    public Expression<T> createSelection();
}package alpc.ufsc.util.querydsl;

import java.util.Map;

import com.mysema.query.types.Expression;
import com.mysema.query.types.Path;
import com.mysema.query.types.QBean;

public class QChildBean<T> extends QBean<T> {
    private static final long serialVersionUID = -3180340538904753892L;

    public QChildBean(Class<T> type, boolean fieldAccess, Expression<?>... args) {
        super(type, fieldAccess, args);
    }

    public QChildBean(Class<T> type, boolean fieldAccess, Map<String, ? extends
    ↪ Expression<?>> bindings) {
        super(type, fieldAccess, bindings);
    }

    public QChildBean(Class<T> type, Expression<?>... args) {
        super(type, args);
    }

    public QChildBean(Class<T> type, Map<String, ? extends Expression<?>> bindings) {
        super(type, bindings);
    }

    public QChildBean(Path<T> type, boolean fieldAccess, Expression<?>... args) {
        super(type, fieldAccess, args);
    }

    public QChildBean(Path<T> type, boolean fieldAccess, Map<String, ? extends
    ↪ Expression<?>> bindings) {
        super(type, fieldAccess, bindings);
    }

    public QChildBean(Path<T> type, Expression<?>... args) {
        super(type, args);
    }

    public QChildBean(Path<T> type, Map<String, ? extends Expression<?>> bindings) {
        super(type, bindings);
    }
}

@Override
public T newInstance(Object... a) {
    boolean hasValues = false;
    for (int i = 0; i < a.length; i++) {
        if (a[i] != null) {
            hasValues = true;
        }
    }
}

```

```

        }
        }
        } else {
            return super.newInstance(a);
        } else {
            return null;
        }
    }
}

}package alpc.ufsc.util.querydsl;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import br.ufsc.bridge.metafy.MetaField;

import com.mysema.query.types.Expression;
import com.mysema.query.types.FactoryExpression;
import com.mysema.query.types.Projections;
import com.mysema.query.types.QBean;
import com.mysema.query.types.Visitor;
import com.mysema.query.types.expr.SimpleExpression;

public class Select<T> implements SelectExpression<T>, FactoryExpression<T> {

    private static final long serialVersionUID = 1286449954664399314L;
    private String alias;
    private Class<T> resultType;
    private Map<MetaField<?>, SelectExpression<?>> selectionMap;

    public Select(Class<T> resultType) {
        this.resultType = resultType;
        this.selectionMap = new HashMap<>();
    }

    public Select(Class<T> resultType, String alias) {
        this(resultType);
        this.alias = alias;
    }

    public <E> void as(SimpleExpression<E> path, MetaField<E> metaField) {
        this.select(path, metaField);
    }

    /**
     * This method is no longer needed.
     * Pass the Select object directly to the query.
     *
     * @return
     */
    @Deprecated
    @SuppressWarnings({ "unchecked", "rawtypes" })
    @Override
    public Expression<T> createSelection() {
        List<Expression<?>> properties = new ArrayList<>();
        for (SelectExpression<?> expression : this.selectionMap.values()) {
            properties.add(expression.createSelection());
        }

        Expression<?>[] array = properties.toArray(new Expression<?>[properties.size()]);

        if (this.alias != null) {
            return new QChildBean(this.resultType, array).as(this.alias);
        } else {
            return Projections.bean(this.resultType, array);
        }
    }

    @SuppressWarnings({ "unchecked", "rawtypes" })
    private MetaField<?> select(SimpleExpression<?> expression, MetaField<?> path) {
        /**
         * Verifica se path composto e ignora o path raiz.
         * Ex: profissional.cbo.nome, o parent do "nome" o "cbo", porm o
         *     ↪ parent("profissional") do "cbo" ignorado por ser a raiz do select
         */
        if (path.getParent() != null && !path.getParent().getType().equals(this.resultType))
            ↪ {
                ↪ // Cria selection para os nveis acima
            }
    }
}

```

```

        MetaField<?> startingField = this.select(null, path.getParent());

        if (expression != null) {
            // Caso uma expresso relacionada for informada, fazer o bind no
            // selection criado para o parent
            ((Select) this.selectionMap.get(startingField)).select(expression,
                this.truncatePath(path, startingField));
        }

        return startingField;
    } else {
        if (!this.selectionMap.containsKey(path)) {
            if (expression != null) {
                // Caso uma expresso relacionada for informada, fazer um
                // bind simples
                this.selectionMap.put(path, new
                    SimpleSelectExpression(expression, path));
            } else {
                // Caso uma expresso no for informada, criar uma selecion
                // para os paths filhos
                this.selectionMap.put(path, new Select<>(path.getType(),
                    path.getAlias()));
            }
        }
        return path;
    }
}

/*
 * Extrai parte de um path composto.
 * Ex: pathToTruncate = profissional.cbo.nome, startingPath = profissional.cbo, resultado
 *     = nome
 */
@SuppressWarnings({ "rawtypes", "unchecked" })
private MetaField<?> truncatePath(MetaField<?> pathToTruncate, MetaField<?>
    startingPath) {
    if (pathToTruncate != startingPath) {
        MetaField<?> returnPath = this.truncatePath(pathToTruncate.getParent(),
            startingPath);
        if (returnPath == null) {
            returnPath = new MetaField(pathToTruncate.getType(),
                pathToTruncate.getAlias());
        } else {
            returnPath = new MetaField(returnPath,
                pathToTruncate.getAlias());
        }
        return returnPath;
    }
    return null;
}

public class SimpleSelectExpression implements SelectExpression<Object> {
    private SimpleExpression<?> expression;
    private MetaField<?> constant;

    protected SimpleSelectExpression(SimpleExpression<?> expression, MetaField<?>
        constant) {
        super();
        this.expression = expression;
        this.constant = constant;
    }

    @SuppressWarnings("unchecked")
    @Override
    public Expression<Object> createSelection() {
        return (Expression<Object>) this.expression.as(this.constant.getAlias());
    }
}

private QBean<T> qbean;

@Override
public <R, C> R accept(Visitor<R, C> v, C context) {
    return v.visit(this, context);
}

@Override
public Class<? extends T> getType() {
    return this.resultType;
}
}

```



```

@Override
public List<Expression<?>> getArgs() {
    this.qbean = (QBean<T>) this.createSelection();
    return this.qbean.getArgs();
}

@Override
public T newInstance(Object... args) {
    return this.qbean.newInstance(args);
}
}

package alpc.ufsc.util.querydsl;

import br.ufsc.bridge.metafy.MetaField;

import com.mysema.query.support.Expressions;
import com.mysema.query.types.Path;
import com.mysema.query.types.expr.SimpleExpression;
import com.mysema.query.types.path.NumberPath;
import com.mysema.query.types.path.StringPath;

public class ExpressionsUtils {

    public static <T> SimpleExpression<T> path(MetaField<T> field, Path<?> parent) {
        return Expressions.path(field.getType(), parent, field.getAlias());
    }

    public static StringPath stringPath(MetaField<String> field, Path<?> parent) {
        return Expressions.stringPath(parent, field.getAlias());
    }

    public static NumberPath<Long> longPath(MetaField<Long> field, Path<?> parent) {
        return Expressions.numberPath(Long.class, parent, field.getAlias());
    }

    public static NumberPath<Integer> integerPath(MetaField<Integer> field, Path<?>
        ↪ parent) {
        return Expressions.numberPath(Integer.class, parent, field.getAlias());
    }

    public static <T> void addAs(Select<?> select, MetaField<T> field, Path<?> parent) {
        select.as(path(field, parent), field);
    }
}

package alpc.ufsc.util;

import java.io.IOException;

import com.fasterxml.jackson.core.JsonGenerator;
import com.fasterxml.jackson.core.JsonProcessingException;
import com.fasterxml.jackson.databind.JsonSerializer;
import com.fasterxml.jackson.databind.SerializerProvider;

import alpc.ufsc.entity.util.QueryUtils;

public class AnoJsonSerialize extends JsonSerializer<Integer> {

    @Override
    public void serialize(Integer value, JsonGenerator gen, SerializerProvider serializers) throws
        ↪ IOException, JsonProcessingException {
        gen.writeNumber(value);
    }

    @Override
    public boolean isEmpty(SerializerProvider provider, Integer value) {
        return value == null || QueryUtils.MIN_YEAR.equals(value) ||
            ↪ QueryUtils.MAX_YEAR.equals(value);
    }
}

package alpc.ufsc.util;

import java.util.Collection;

public class CollectionsUtilsInternal {

    @SuppressWarnings("rawtypes")
    public static void turnCollectionsEqual(Collection one, Collection other) {
        if (one.size() < other.size()) {
            addToOne(one, other);
        }
    }
}

```

```

        } else if (one.size() > other.size()) {
            addToOne(other, one);
        }
    }

    @SuppressWarnings({ "rawtypes", "unchecked" })
    private static void addToOne(Collection one, Collection other) {
        int size = other.size() - one.size();
        for (int i = 0; i < size; i++) {
            one.add(null);
        }
    }
}

package alpc.ufsc.util;

import java.util.Calendar;
import java.util.Date;

public class DateUtilsInternal {

    public static Date applyTimeToDate(Date date, String time) {
        Calendar calendar = Calendar.getInstance();
        calendar.setTime(date);
        calendar.set(Calendar.HOUR_OF_DAY, Integer.parseInt(time.substring(0, 2)));
        calendar.set(Calendar.MINUTE, Integer.parseInt(time.substring(2, 4)));
        calendar.set(Calendar.SECOND, Integer.parseInt(time.substring(4, 6)));
        calendar.set(Calendar.MILLISECOND, 0);
        return calendar.getTime();
    }

    public static boolean isYearNotBefore(int ano, Date dataInicio) {
        Calendar calendar = Calendar.getInstance();
        calendar.setTime(dataInicio);
        return ano >= calendar.get(Calendar.YEAR);
    }
}

package alpc.ufsc.util;

import java.util.concurrent.ThreadFactory;
import java.util.concurrent.atomic.AtomicInteger;

public class NamedThreadFactory implements ThreadFactory {
    ThreadGroup group;
    AtomicInteger threadNumber = new AtomicInteger(1);
    String namePrefix;

    public NamedThreadFactory(String name) {
        SecurityManager s = System.getSecurityManager();
        this.group = s != null ? s.getThreadGroup() :
            ↪ Thread.currentThread().getThreadGroup();
        this.namePrefix = name + "-thread-";
    }

    @Override
    public Thread newThread(Runnable r) {
        Thread t = new Thread(this.group, r, this.namePrefix +
            ↪ this.threadNumber.getAndIncrement(), 0);
        if (t.isDaemon()) {
            t.setDaemon(false);
        }
        if (t.getPriority() != Thread.NORM_PRIORITY) {
            t.setPriority(Thread.NORM_PRIORITY);
        }
        return t;
    }
}

package alpc.ufsc.server;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;

import alpc.ufsc.config.web.security.PasswordEncoderWrapper;

@RestController
public class ServerController {

    @Autowired
    private PasswordEncoderWrapper passwordEncoder;

```

```

    @RequestMapping(value = "/serverstatus", method = RequestMethod.GET)
    public ResponseEntity<Void> checkStatus() {
        return new ResponseEntity<>(HttpStatus.OK);
    }

    @RequestMapping(value = "/public/key", method = RequestMethod.GET)
    public @ResponseBody String getKey() {
        return this.passwordEncoder.getPublicKey();
    }
}

}package alpc.ufsc.common.exception;

public interface ErrorMessages {

    public static final String OBRIGATORIO = "Campo de preenchimento obrigatrio";
}

package alpc.ufsc.common.exception;

public class EmptyXmlException extends Exception {

    private static final long serialVersionUID = 654161031289830610L;

    public EmptyXmlException(String message) {
        super(message);
    }
}

package alpc.ufsc.common.exception;

import java.util.LinkedHashMap;
import java.util.Map;

import lombok.Getter;
import br.ufsc.bridge.metafy.MetaField;

import com.fasterxml.jackson.core.JsonProcessingException;
import com.fasterxml.jackson.databind.ObjectMapper;

@Getter
public class InvalidDTOException extends RuntimeException {

    private static final long serialVersionUID = 6151525291759922758L;

    private Map<String, Object> errors = new LinkedHashMap<>();

    public void putError(MetaField<?> field, Object error) {
        this.errors.put(field.getAlias(), error);
    }

    public boolean contains(MetaField<?> field) {
        return this.errors.containsKey(field.getAlias());
    }

    public void checkThrow() throws InvalidDTOException {
        if (!this.errors.isEmpty()) {
            throw this;
        }
    }

    public String toJson() {
        try {
            return new ObjectMapper().writeValueAsString(this.errors);
        } catch (JsonProcessingException e) {
            throw new RuntimeException("Error ao montar json", e);
        }
    }
}

package alpc.ufsc.common.exception;

public class InvalidOperationException extends Exception {

    private static final long serialVersionUID = 2242923640302534668L;

    public InvalidOperationException(String error) {
        super(error);
    }
}

package alpc.ufsc.common.exception;

public class ProcessAlreadyRunningException extends Exception {

    private static final long serialVersionUID = 1L;
}

```

```

        public ProcessAlreadyRunningException() {
            super("O Processo j esta em execucao");
        }
    }
}
package alpc.ufsc.common;

public interface FormDTO {

    Long getId();

    default String response() {
        return "{\"id\": " + this.getId() + "}";
    }
}
package alpc.ufsc.common;

import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
public class PeriodoFilterDTO {

    private Integer inicio;
    private Integer fim;

    public boolean hasInicio() {
        return this.inicio != null;
    }

    public boolean hasFim() {
        return this.fim != null;
    }
}
package alpc.ufsc;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.boot.SpringApplication;
import org.springframework.context.ConfigurableApplicationContext;

public class SpringBootTestStarter {

    static final Logger logger = LoggerFactory.getLogger(SpringBootTestStarter.class);

    public static void main(String[] args) {
        SpringApplication application = new SpringApplication(Application.class);
        ConfigurableApplicationContext run = application.run(args);
        logger.info("Number of beans definition: " + run.getBeanDefinitionCount());
    }
}
package alpc.ufsc.login.user;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RestController;

import alpc.ufsc.common.exception.InvalidDTOException;
import alpc.ufsc.common.exception.InvalidOperationException;
import alpc.ufsc.config.web.security.PasswordEncoderWrapper;
import alpc.ufsc.login.user.dto.UserFormDTO;

@RestController
@RequestMapping("/public/services/user")
public class UserPublicRestController {

    @Autowired
    private UserService service;

    @Autowired
    private PasswordEncoderWrapper passwordEncoder;

    @RequestMapping(method = RequestMethod.PUT)
    public ResponseEntity<String> save(@RequestBody UserFormDTO form) {
        try {
            form.setPassword(this.passwordEncoder.decrypt(form.getPassword());
            form.setConfirmPassword(this.passwordEncoder.decrypt(form.getConfirmPassword());

```

```

        form.setId(this.service.save(form));
        return new ResponseEntity<>(form.response(), HttpStatus.CREATED);
    } catch (InvalidDTOException e) {
        return new ResponseEntity<>(e.toJson(), HttpStatus.BAD_REQUEST);
    } catch (InvalidOperationException e) {
        return new ResponseEntity<>(e.getMessage(),
            ↪ HttpStatus.UNAUTHORIZED);
    }
}
}
package alpc.ufsc.login.user;

import static alpc.ufsc.entity.login.user.QProgramaUserEntity.programaUserEntity;
import static alpc.ufsc.entity.login.user.QUserEntity.userEntity;
import static alpc.ufsc.entity.xml.QXmlDataEntity.xmlDataEntity;

import java.util.Date;
import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.apache.commons.lang3.BooleanUtils;
import org.apache.commons.lang3.StringUtils;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageImpl;
import org.springframework.data.domain.Pageable;
import org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.Tuple;
import com.mysema.query.jpql.impl.JPAQuery;
import com.mysema.query.jpql.impl.JPAUpdateClause;
import com.mysema.query.types.expr.CaseBuilder;
import com.mysema.query.types.expr.NumberExpression;

import alpc.ufsc.common.exception.InvalidDTOException;
import alpc.ufsc.common.exception.InvalidOperationException;
import alpc.ufsc.entity.login.user.ProgramaUserEntity;
import alpc.ufsc.entity.login.user.QProgramaUserEntity;
import alpc.ufsc.entity.login.user.UserEntity;
import alpc.ufsc.entity.programa.ProgramaEntity;
import alpc.ufsc.entity.programa.QProgramaEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.entity.xml.ProgramaXmlDataEntity;
import alpc.ufsc.entity.xml.QProgramaXmlDataEntity;
import alpc.ufsc.entity.xml.QXmlDataEntity;
import alpc.ufsc.entity.xml.XmlDataEntity;
import alpc.ufsc.login.user.dto.MUserDetailsImpl;
import alpc.ufsc.login.user.dto.MUserFormDTO;
import alpc.ufsc.login.user.dto.MUserRowDTO;
import alpc.ufsc.login.user.dto.ProgramaUserSaveDTO;
import alpc.ufsc.login.user.dto.UserDetailsImpl;
import alpc.ufsc.login.user.dto.UserFilterDTO;
import alpc.ufsc.login.user.dto.UserFormDTO;
import alpc.ufsc.login.user.dto.UserRowDTO;
import alpc.ufsc.util.querydsl.Select;
import alpc.ufsc.xml.UploadCurriculoService;

@Service
@Transactional
public class UserService {

    @PersistenceContext
    private EntityManager em;

    @Autowired
    private UploadCurriculoService curriculoService;

    @Autowired
    private PasswordEncoder passwordEncoder;

    public Long save(UserFormDTO dto) throws InvalidDTOException,
        ↪ InvalidOperationException {
        if (dto.getId() != null) {
            throw new InvalidOperationException("Atualizacao de usuario no permitida");
        }
        this.validateUserForm(dto, true);
        UserEntity user = new UserEntity();

```

```

        user.setNome(dto.getName());
        user.setPassword(this.passwordEncoder.encode(dto.getPassword()));
        user.setEmail(dto.getEmail().toLowerCase());
        user.setAdm(BooleanUtils.isTrue(dto.getAdm()));
        user.setCriarPrograma(BooleanUtils.isTrue(dto.getCriarPrograma()));
        this.em.persist(user);
        return user.getId();
    }

    public void update(UserFormDTO dto) throws InvalidDTOException,
        ↪ InvalidOperationException {
        if (dto.getId() == null) {
            throw new InvalidOperationException("Criação de usuário não permitida");
        }
        this.validateUserForm(dto, StringUtils.isNotBlank(dto.getPassword()));
        UserEntity user = this.em.find(UserEntity.class, dto.getId());
        user.setNome(dto.getName());
        if (StringUtils.isNotBlank(dto.getPassword())) {
            user.setPassword(this.passwordEncoder.encode(dto.getPassword()));
        }
        user.setEmail(dto.getEmail().toLowerCase());
        this.em.persist(user);
    }

    private void validateUserForm(UserFormDTO dto, boolean validatePassword) {
        boolean alreadyHasEmail = false;
        if (StringUtils.isNotBlank(dto.getEmail())) {
            JPAQuery query = new JPAQuery(this.em).from(userEntity);
            query.where(userEntity.email.eq(dto.getEmail().toLowerCase()));
            if (dto.getId() != null) {
                query.where(userEntity.id.ne(dto.getId()));
            }
            alreadyHasEmail = query.exists();
        }
        dto.validate(alreadyHasEmail, validatePassword);
    }

    public UserFormDTO load(Long id) {
        JPAQuery query = new JPAQuery(this.em).from(userEntity);
        query.where(userEntity.id.eq(id));

        Select<UserFormDTO> select = new Select<>(UserFormDTO.class);
        MUserFormDTO meta = MUserFormDTO.meta;
        select.as(userEntity.id, meta.id);
        select.as(userEntity.nome, meta.name);
        select.as(userEntity.email, meta.email);
        return query.uniqueResult(select);
    }

    public UserDetailsImpl loadUserDetailsByUsername(String login) throws
        ↪ UsernameNotFoundException {
        JPAQuery query = new JPAQuery(this.em).from(userEntity);
        query.where(userEntity.email.eq(login));

        Select<UserDetailsImpl> select = new Select<>(UserDetailsImpl.class);
        MUserDetailsImpl meta = MUserDetailsImpl.meta;
        select.as(userEntity.id, meta.id);
        select.as(userEntity.nome, meta.name);
        select.as(userEntity.email, meta.email);
        select.as(userEntity.password, meta.password);
        select.as(userEntity.adm, meta.adm);
        select.as(userEntity.criarPrograma, meta.criarPrograma);
        UserDetailsImpl userDto = query.uniqueResult(select);

        if (userDto != null) {
            return userDto;
        } else {
            throw new UsernameNotFoundException("Usuário ou senha inválidos.");
        }
    }

    public Page<UserRowDTO> getPage(UserFilterDTO filter, Pageable pageable) {
        QProgramaUserEntity qProgUser = QProgramaUserEntity.programaUserEntity;
        QProgramaEntity qProg = QProgramaEntity.programaEntity;

        JPAQuery query = new JPAQuery(this.em).from(userEntity);
        query.leftJoin(userEntity.programaUser, qProgUser);
        query.leftJoin(qProgUser.programa(), qProg);

        Select<UserRowDTO> select = new Select<>(UserRowDTO.class);
        MUserRowDTO meta = MUserRowDTO.meta;

```

```

select.as(qProgUser.id, meta.programaUserId);
select.as(userEntity.id, meta.userId);
select.as(userEntity.nome, meta.nome);
select.as(userEntity.email, meta.email);
select.as(userEntity.adm, meta.adm);
select.as(userEntity.criarPrograma, meta.criarPrograma);
select.as(qProg.nome, meta.nomePrograma);
select.as(qProgUser.grantedAuthority, meta.grantedAuthority);
select.as(qProgUser.requestAuthority, meta.requestAuthority);

if (filter.hasProgramaId()) {
    query.where(qProg.id.eq(filter.getProgramaId()));
}
if (filter.hasNome()) {
    query.where(userEntity.nomeFiltro.like(QueryUtils.like(filter.getNome())));
}
if (filter.hasEmail()) {
    query.where(userEntity.email.like(QueryUtils.like(filter.getEmail())));
}

query.offset(pageable.getOffset());
query.limit(pageable.getPageSize());

if (filter.hasAuthority()) {
    NumberExpression<Integer> caseRequest = new
        CaseBuilder().when(qProgUser.requestAuthority.eq(filter.getAuthority())).then(0)
    NumberExpression<Integer> caseGranted = new
        CaseBuilder().when(qProgUser.grantedAuthority.eq(filter.getAuthority())).then(0)
    query.orderBy(caseRequest.asc(), caseGranted.asc(), userEntity.nome.asc());
} else {
    query.orderBy(userEntity.nome.asc());
}
List<UserRowDTO> list = query.list(select);
return new PageImpl<>(list, pageable, query.count());
}

public String getAuthority(Long userId, Long programaId) {
    QProgramaUserEntity qProgramaUser = QProgramaUserEntity.programaUserEntity;

    JPAQuery query = new JPAQuery(this.em).from(qProgramaUser);
    query.where(qProgramaUser.user().id.eq(userId).and(qProgramaUser.programa().id.eq(programaId)))

    return query.uniqueResult(qProgramaUser.grantedAuthority);
}

public Long saveUserPrograma(ProgramaUserSaveDTO dto) {
    QProgramaUserEntity qProgUser = QProgramaUserEntity.programaUserEntity;

    JPAQuery query = new JPAQuery(this.em).from(qProgUser);
    if (dto.hasId()) {
        query.where(qProgUser.id.eq(dto.getId()));
    } else {
        query.where(qProgUser.programa().id.eq(dto.getProgramaId()));
        query.where(qProgUser.user().id.eq(dto.getUserId()));
    }

    ProgramaUserEntity programaUser = query.uniqueResult(qProgUser);
    if (programaUser == null) {
        programaUser = new ProgramaUserEntity();
    }

    if (dto.getProgramaId() != null) {
        programaUser.setPrograma(this.em.getReference(ProgramaEntity.class,
            dto.getProgramaId()));
    }
    if (dto.getUserId() != null) {
        programaUser.setUser(this.em.getReference(UserEntity.class,
            dto.getUserId()));
    }
    programaUser.setGrantedAuthority(dto.getGrantedAuthority());
    if (StringUtils.equals(dto.getGrantedAuthority(), dto.getRequestAuthority())) {
        programaUser.setRequestAuthority(null);
    } else {
        programaUser.setRequestAuthority(dto.getRequestAuthority());
    }
    this.em.persist(programaUser);
    return programaUser.getId();
}

public void saveUserProgramaAndUpdateXml(ProgramaUserSaveDTO dto) {
    this.updateProgramaUserXml(this.saveUserPrograma(dto));
}

```

```

private void updateProgramaUserXml(Long programaUserId) {
    Tuple tuple = new JPAQuery(this.em).from(programaUserEntity)
        .where(programaUserEntity.id.eq(programaUserId))
        .uniqueResult(programaUserEntity.user().id,
            ↪ programaUserEntity.programa().id,
            ↪ programaUserEntity.grantedAuthority);

    JPAQuery query = new JPAQuery(this.em).from(userEntity);
    query.join(userEntity.xmlData(), xmlDataEntity);
    query.where(userEntity.id.eq(tuple.get(programaUserEntity.user().id)));
    Tuple result = query.uniqueResult(xmlDataEntity.id,
        ↪ xmlDataEntity.dataAtualizacao, xmlDataEntity.identificador);
    // se eu tenho curriculo e estou sendo adicionado no programa, inserir meu curriculo
    ↪ no programa
    if (result != null && tuple.get(programaUserEntity.grantedAuthority) != null) {
        this.updateProgramaXml(
            tuple.get(programaUserEntity.user().id),
            result.get(xmlDataEntity.id),
            result.get(xmlDataEntity.identificador),
            result.get(xmlDataEntity.dataAtualizacao),
            tuple.get(programaUserEntity.programa().id));
    }
}

private void updateProgramaXml(Long userId, Long xmlId, String identificador, Date
    ↪ dataAtualizacao, Long programaId) {
    QXmlDataEntity qXml = QXmlDataEntity.xmlDataEntity;
    QProgramaXmlDataEntity qProgXml =
        ↪ QProgramaXmlDataEntity.programaXmlDataEntity;

    // verificar se meu curriculo j foi importado para o programa
    JPAQuery query = new JPAQuery(this.em).from(qXml);
    query.join(qXml.programaXml, qProgXml);
    query.where(qXml.identificador.eq(identificador));
    query.where(qProgXml.programa().id.eq(programaId));
    Tuple tuple = query.uniqueResult(qXml.id, qXml.dataAtualizacao, qProgXml.id);
    if (tuple != null) {
        // caso tenha sido, comparasse as datas de atualizacao dos curriculos
        if (dataAtualizacao.compareTo(tuple.get(qXml.dataAtualizacao)) < 0) {
            // se o curriculo do programa for mais novo
            // seta o curriculo do programa no meu usuario
            JPAUpdateClause updateClause = new JPAUpdateClause(this.em,
                ↪ userEntity);
            updateClause.set(userEntity.xmlData().id, tuple.get(qXml.id));
            updateClause.where(userEntity.id.eq(userId));
            updateClause.execute();

            // e atualiza os programas que apontavam para meu curriculo antigo
            // e seta para processar
            updateClause = new JPAUpdateClause(this.em, qProgXml);
            updateClause.set(qProgXml.xmlData().id, tuple.get(qXml.id));
            updateClause.where(qProgXml.xmlData().id.eq(xmlId));
            updateClause.execute();

            this.curriculoService.deleteXmlWithoutReference(xmlId);
        } else {
            // se o meu curriculo for o mais novo
            // s atualiza para o programa apontar para o meu curriculo
            JPAUpdateClause updateClause = new JPAUpdateClause(this.em,
                ↪ qProgXml);
            updateClause.set(qProgXml.xmlData().id, xmlId);
            updateClause.where(qProgXml.id.eq(tuple.get(qProgXml.id)));
            updateClause.execute();

            // caso o curriculo remanescente no tenha referencia deletar
            // se possuir referencia, provavelmente significa usuario duplicado ou
            // que outro usuario importou o msm curriculo como seu
            this.curriculoService.deleteXmlWithoutReference(tuple.get(qXml.id));
        }
    } else {
        // caso no tenha sido apenas adicionar seu curriculo no programa
        ProgramaXmlDataEntity progXmlData = new ProgramaXmlDataEntity();
        progXmlData.setPrograma(this.em.getReference(ProgramaEntity.class,
            ↪ programaId));
        progXmlData.setXmlData(this.em.getReference(XmlDataEntity.class,
            ↪ xmlId));
        this.em.persist(progXmlData);
    }
}

public void setAdm(Long userId, Boolean adm) throws InvalidOperationException {

```



```

        if (BooleanUtils.isNotTrue(adm)) {
            JPAQuery query = new JPAQuery(this.em).from(userEntity);
            query.where(userEntity.adm.eq(true));
            if (query.count() <= 1) {
                throw new InvalidOperationExcepção("Este o nico administrador
                ↳ do sistema, voc no pode desativalo.");
            }
        }

        JPAUpdateClause updateClause = new JPAUpdateClause(this.em, userEntity);
        updateClause.set(userEntity.adm, adm);
        updateClause.where(userEntity.id.eq(userId));
        updateClause.execute();
    }

    public void setCriarPrograma(Long userId, Boolean criarPrograma) {
        JPAUpdateClause updateClause = new JPAUpdateClause(this.em, userEntity);
        updateClause.set(userEntity.criarPrograma, criarPrograma);
        updateClause.where(userEntity.id.eq(userId));
        updateClause.execute();
    }
}
package alpc.ufsc.login.user;

import java.lang.annotation.Documented;
import java.lang.annotation.ElementType;
import java.lang.annotation.Retention;
import java.lang.annotation.RetentionPolicy;
import java.lang.annotation.Target;

import org.springframework.security.core.annotation.AuthenticationPrincipal;

@Target({ ElementType.PARAMETER, ElementType.TYPE })
@Retention(value = RetentionPolicy.RUNTIME)
@Documented
@AuthenticationPrincipal
public @interface CurrentUser {
}

package alpc.ufsc.login.user.dto;

import lombok.Getter;
import lombok.Setter;
import br.ufsc.bridge.metafy.Metafy;

@Getter
@Setter
@Metafy
public class UserInfoDTO {

    private Long id;
    private Long programaId;
    private String name;
    private String authority;
    private Boolean adm;
    private Boolean criarPrograma;
}

package alpc.ufsc.login.user.dto;

import java.util.Objects;

import lombok.Getter;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;
import org.apache.commons.validator.routines.EmailValidator;

import br.ufsc.bridge.metafy.Metafy;

import alpc.ufsc.common.FormDTO;
import alpc.ufsc.common.exception.ErrorMessage;
import alpc.ufsc.common.exception.InvalidDTOException;

@Getter
@Setter
@Metafy
public class UserFormDTO implements FormDTO {

    private Long id;
    private String name;
    private String email;
    private Boolean adm;

```

```

private Boolean criarPrograma;

private String password;
private String confirmPassword;

public void validate(boolean alreadyHasEmail, boolean validatePassword) throws
↳ InvalidDTOException {
    InvalidDTOException dtoException = new InvalidDTOException();
    MUserFormDTO meta = MUserFormDTO.meta;
    if (StringUtils.isBlank(this.name)) {
        dtoException.putError(meta.name, ErrorMessages.OBRIGATORIO);
    }
    if (StringUtils.isBlank(this.email)) {
        dtoException.putError(meta.email, ErrorMessages.OBRIGATORIO);
    } else if (alreadyHasEmail) {
        dtoException.putError(meta.email, "Este email j esta cadastrado");
    }
    if (validatePassword && StringUtils.isBlank(this.password)) {
        dtoException.putError(meta.password, ErrorMessages.OBRIGATORIO);
    }
    if (validatePassword && StringUtils.isBlank(this.confirmPassword)) {
        dtoException.putError(meta.confirmPassword,
↳ ErrorMessages.OBRIGATORIO);
    }
    if (!dtoException.contains(meta.email) &&
↳ !EmailValidator.getInstance().isValid(this.email)) {
        dtoException.putError(meta.email, "Email invalido");
    }
    if (!dtoException.contains(meta.password) &&
↳ !dtoException.contains(meta.confirmPassword) &&
↳ !Objects.equals(this.password, this.confirmPassword)) {
        dtoException.putError(meta.password, "A senha no corresponde");
        dtoException.putError(meta.confirmPassword, "A senha no corresponde");
    }
    if (validatePassword && !dtoException.contains(meta.password) &&
↳ this.password.length() < 6) {
        dtoException.putError(meta.password, "Senha deve conter no mnimo 6
↳ caracteres");
    }
    }
    dtoException.checkThrow();
}
}
package alpc.ufsc.login.user.dto;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import alpc.ufsc.common.FormDTO;
import alpc.ufsc.login.authority.Authority;

@Getter
@Setter
@NoArgsConstructor
public class ProgramaUserSaveDTO implements FormDTO {

    private Long id;
    private Long userId;
    private Long programaId;
    private String grantedAuthority;
    private String requestAuthority;

    public ProgramaUserSaveDTO(Long userId, Long programaId, String grantedAuthority) {
        this.userId = userId;
        this.programaId = programaId;
        this.grantedAuthority = grantedAuthority;
    }

    public void setGrantedAuthority(String grantedAuthority) {
        this.grantedAuthority = grantedAuthority == null ? Authority.DISABLE :
↳ grantedAuthority;
    }

    public boolean hasId() {
        return this.id != null;
    }
}
package alpc.ufsc.login.user.dto;

```

```

import lombok.Getter;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;

@Getter
@Setter
public class UserFilterDTO {

    private String nome;
    private String email;
    private String authority;
    private Long programaId;

    public boolean hasProgramaId() {
        return this.programaId != null;
    }

    public boolean hasNome() {
        return !StringUtils.isBlank(this.nome);
    }

    public boolean hasEmail() {
        return !StringUtils.isBlank(this.email);
    }

    public boolean hasAuthority() {
        return !StringUtils.isBlank(this.authority);
    }
}
package alpc.ufsc.login.user.dto;

import java.util.HashSet;
import java.util.Set;

import lombok.Getter;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;
import org.springframework.security.core.CredentialsContainer;
import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.SpringSecurityCoreVersion;
import org.springframework.security.core.userdetails.UserDetails;

import br.ufsc.bridge.metafy.Metafy;

import alpc.ufsc.login.authority.Authority;
import alpc.ufsc.login.authority.AuthorityUtils;

@Getter
@Setter
@Metafy
public class UserDetailsImpl implements UserDetails, CredentialsContainer {

    private static final long serialVersionUID =
        ↪ SpringSecurityCoreVersion.SERIAL_VERSION_UID;

    private Long id;
    private String name;
    private Long programaId;

    private String password;
    private String email;
    private Set<GrantedAuthority> authorities;
    private Boolean adm;
    private Boolean criarPrograma;

    public void addAuthority(String authority) {
        if (this.authorities == null) {
            this.authorities = new HashSet<>();
        }
        if (!StringUtils.isBlank(authority)) {
            this.authorities.add(AuthorityUtils.getGrantedAuthority(authority));
        }
    }

    public void setAdm(Boolean adm) {
        this.adm = adm;
        if (adm) {
            this.addAuthority(Authority.ADM);
        }
    }
}

```

```

@Override
public boolean equals(Object rhs) {
    if (rhs instanceof UserDetailsImpl) {
        return this.email.equals(((UserDetailsImpl) rhs).email);
    }
    return false;
}

@Override
public int hashCode() {
    return this.email.hashCode();
}

@Override
public void eraseCredentials() {
    this.password = null;
}

@Override
public boolean isAccountNonExpired() {
    return true;
}

@Override
public boolean isAccountNonLocked() {
    return true;
}

@Override
public boolean isCredentialsNonExpired() {
    return true;
}

@Override
public boolean isEnabled() {
    return true;
}

@Override
public String getUsername() {
    return this.email;
}
}
package alpc.ufsc.login.user.dto;

import java.util.Collection;

import lombok.AllArgsConstructor;
import lombok.experimental.Delegate;

import org.springframework.security.core.Authentication;
import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.userdetails.UserDetails;

@AllArgsConstructor
public class AuthenticationWrapper implements Authentication {
    private static final long serialVersionUID = -7264256157456161573L;

    @Delegate(excludes = UserDetails.class)
    private Authentication original;
    private UserDetailsImpl currentUser;

    @Override
    public Collection<? extends GrantedAuthority> getAuthorities() {
        return this.currentUser.getAuthorities();
    }
}
package alpc.ufsc.login.user.dto;

import lombok.Getter;
import lombok.Setter;
import br.ufsc.bridge.metafy.Metafy;

import com.fasterxml.jackson.annotation.JsonInclude;
import com.fasterxml.jackson.annotation.JsonInclude.Include;

@Getter
@Setter
@Metafy
@JsonInclude(Include.NON_EMPTY)

```

```

public class UserRowDTO {
    private Long programaUserId;
    private Long userId;
    private Boolean adm;
    private Boolean criarPrograma;
    private String nome;
    private String email;
    private String nomePrograma;
    private String grantedAuthority;
    private String requestAuthority;
}
package alpc.ufsc.login.user;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.security.access.annotation.Secured;
import org.springframework.security.core.context.SecurityContext;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;

import alpc.ufsc.common.exception.InvalidDTOException;
import alpc.ufsc.common.exception.InvalidOperationException;
import alpc.ufsc.config.web.security.PasswordEncoderWrapper;
import alpc.ufsc.login.authority.Authority;
import alpc.ufsc.login.user.dto.AuthenticationWrapper;
import alpc.ufsc.login.user.dto.ProgramaUserSaveDTO;
import alpc.ufsc.login.user.dto.UserDetailsImpl;
import alpc.ufsc.login.user.dto.UserFilterDTO;
import alpc.ufsc.login.user.dto.UserFormDTO;
import alpc.ufsc.login.user.dto.UserInfoDTO;
import alpc.ufsc.login.user.dto.UserRowDTO;

@RestController
@RequestMapping("/services/user")
public class UserRestController {

    @Autowired
    private UserService service;

    @Autowired
    private PasswordEncoderWrapper passwordEncoder;

    @RequestMapping(value = "current", method = RequestMethod.GET)
    public @ResponseBody UserInfoDTO loadInfoCurrentUser(Long programaId, @CurrentUser
        ↪ UserDetailsImpl user) {
        UserInfoDTO userDto = new UserInfoDTO();

        if (programaId != null) {
            String authority = this.service.getAuthority(user.getId(), programaId);
            user.addAuthority(authority);
            user.setProgramaId(programaId);

            SecurityContext context = SecurityContextHolder.getContext();
            if (!(context.getAuthentication() instanceof AuthenticationWrapper) {
                context.setAuthentication(new
                    ↪ AuthenticationWrapper(context.getAuthentication(),
                    ↪ user));
            }

            userDto.setAuthority(authority);
        }
        userDto.setId(user.getId());
        userDto.setProgramaId(user.getProgramaId());
        userDto.setAdm(user.getAdm());
        userDto.setCriarPrograma(user.getCriarPrograma());
        userDto.setName(user.getName());
        return userDto;
    }

    @RequestMapping(value = "update", method = RequestMethod.POST)
    public ResponseEntity<String> updateUser(@CurrentUser UserDetailsImpl user,
        ↪ @RequestBody UserFormDTO dto) {
        try {

```

```

        dto.setId(user.getId());
        dto.setPassword(this.passwordEncoder.decrypt(dto.getPassword()));
        dto.setConfirmPassword(this.passwordEncoder.decrypt(dto.getConfirmPassword()));
        this.service.update(dto);
        return new ResponseEntity<>(HttpStatus.OK);
    } catch (InvalidDTOException e) {
        return new ResponseEntity<>(e.toJson(), HttpStatus.BAD_REQUEST);
    } catch (InvalidOperationException e) {
        return new ResponseEntity<>(e.getMessage(),
            ↪ HttpStatus.UNAUTHORIZED);
    }
}

@RequestMapping(value = "load", method = RequestMethod.GET)
public @ResponseBody UserFormDTO load(@CurrentUser UserDetailsImpl user) {
    return this.service.load(user.getId());
}

@Secured({ Authority.ADM, Authority.COORDENADOR })
@RequestMapping(value = "grantAuthority/{programaUserId}", method =
    ↪ RequestMethod.POST)
public void updateProgramaUser(@PathVariable Long programaUserId, @RequestBody
    ↪ ProgramaUserSaveDTO dto) {
    dto.setId(programaUserId);
    this.service.saveUserProgramaAndUpdateXml(dto);
}

@Secured({ Authority.ADM })
@RequestMapping(value = "setAdm/{userId}/{adm}", method = RequestMethod.POST)
public ResponseEntity<String> setAdm(@PathVariable Long userId, @PathVariable Boolean
    ↪ adm) {
    try {
        this.service.setAdm(userId, adm);
        return new ResponseEntity<>(HttpStatus.OK);
    } catch (InvalidOperationException e) {
        return new ResponseEntity<>(e.getMessage(),
            ↪ HttpStatus.BAD_REQUEST);
    }
}

@Secured({ Authority.ADM })
@RequestMapping(value = "setCriarPrograma/{userId}/{criarPrograma}", method =
    ↪ RequestMethod.POST)
public void setCriarPrograma(@PathVariable Long userId, @PathVariable Boolean
    ↪ criarPrograma) {
    this.service.setCriarPrograma(userId, criarPrograma);
}

@Secured({ Authority.COORDENADOR })
@RequestMapping(method = RequestMethod.GET)
public @ResponseBody Page<UserRowDTO> getPage(@CurrentUser UserDetailsImpl user,
    ↪ UserFilterDTO filter, Pageable pageable) {
    filter.setProgramaId(user.getProgramaId());
    return this.service.getPage(filter, pageable);
}

@Secured({ Authority.ADM })
@RequestMapping(value = "coordenadores", method = RequestMethod.GET)
public @ResponseBody Page<UserRowDTO> getPageCoordenadores(@CurrentUser
    ↪ UserDetailsImpl user, UserFilterDTO filter, Pageable pageable) {
    filter.setProgramaId(null);
    filter.setAuthority(Authority.COORDENADOR);
    return this.service.getPage(filter, pageable);
}
}
package alpc.ufsc.login.user;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.stereotype.Component;
import org.springframework.transaction.annotation.Transactional;

@Component("userDetailsService")
public class UserDetailsServiceImpl implements UserDetailsService {

    @Autowired
    private UserService service;

    @Override
    @Transactional

```

```

    public UserDetails loadUserByUsername(String username) throws
    ↪ UsernameNotFoundException {
        return this.service.loadUserDetailsByUsername(username);
    }
}
package alpc.ufsc.login.authority;

import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.authority.SimpleGrantedAuthority;

public class AuthorityUtils {

    public static final GrantedAuthority ADM = new SimpleGrantedAuthority(Authority.ADM);
    public static final GrantedAuthority COORDENADOR = new
    ↪ SimpleGrantedAuthority(Authority.COORDENADOR);
    public static final GrantedAuthority DOCENTE = new
    ↪ SimpleGrantedAuthority(Authority.DOCENTE);
    public static final GrantedAuthority DISCENTE = new
    ↪ SimpleGrantedAuthority(Authority.DISCENTE);
    public static final GrantedAuthority DISABLE = new
    ↪ SimpleGrantedAuthority(Authority.DISABLE);

    public static boolean isGreaterThanOrEqual(String one, String other) {
        if (Authority.ADM.equals(other)) {
            return Authority.ADM.equals(one);
        }

        if (Authority.COORDENADOR.equals(other)) {
            return Authority.COORDENADOR.equals(one) ||
            ↪ Authority.ADM.equals(one);
        }

        if (Authority.DOCENTE.equals(other)) {
            return !Authority.DISCENTE.equals(one);
        }

        return true;
    }

    public static GrantedAuthority getGrantedAuthority(String authority) {
        switch (authority) {
            case Authority.ADM:
                return ADM;
            case Authority.COORDENADOR:
                return COORDENADOR;
            case Authority.DOCENTE:
                return DOCENTE;
            case Authority.DISCENTE:
                return DISCENTE;
            default:
                return null;
        }
    }
}
package alpc.ufsc.login.authority;

public class Authority {
    public static final String ADM = "ADM";
    public static final String COORDENADOR = "COORDENADOR";
    public static final String DOCENTE = "DOCENTE";
    public static final String DISCENTE = "DISCENTE";
    public static final String DISABLE = "DISABLE";
}
package alpc.ufsc.xml;

import static alpc.ufsc.entity.login.user.QProgramaUserEntity.programaUserEntity;
import static alpc.ufsc.entity.login.user.QUserEntity.userEntity;
import static alpc.ufsc.entity.xml.QProgramaXmlDataEntity.programaXmlDataEntity;
import static alpc.ufsc.entity.xml.QXmlDataEntity.xmlDataEntity;

import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.util.Date;
import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import javax.xml.bind.JAXBException;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageImpl;

```

```

import org.springframework.data.domain.Pageable;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.Tuple;
import com.mysema.query.jpa.JPASubQuery;
import com.mysema.query.jpa.impl.JPADeleteClause;
import com.mysema.query.jpa.impl.JPAQuery;
import com.mysema.query.jpa.impl.JPAUpdateClause;
import com.mysema.query.sql.SQLQuery;
import com.mysema.query.sql.SQLQueryFactory;
import com.mysema.query.types.expr.CaseBuilder;
import com.mysema.query.types.expr.NumberExpression;

import alpc.ufsc.entity.login.user.QUserEntity;
import alpc.ufsc.entity.programa.ProgramaEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.entity.xml.QProgramaXmlDataEntity;
import alpc.ufsc.entity.xml.QProgramaXmlDataEntity;
import alpc.ufsc.entity.xml.QXmlDataEntity;
import alpc.ufsc.entity.xml.XmlDataEntity;
import alpc.ufsc.querydsl.sql.QTbProgramaUser;
import alpc.ufsc.querydsl.sql.QTbProgramaXmlData;
import alpc.ufsc.querydsl.sql.QTbXmlData;
import alpc.ufsc.util.querydsl.Select;
import alpc.ufsc.xml.dto.MUUploadCurriculoRowDTO;
import alpc.ufsc.xml.dto.UploadCurriculoFilterDTO;
import alpc.ufsc.xml.dto.UploadCurriculoRowDTO;
import ufsc.alpc.converter.LattesXmlConverter;
import ufsc.alpc.xml.dto.CurriculoVitaeXmlDto;

@Service
@Transactional
public class UploadCurriculoService {

    @PersistenceContext
    private EntityManager em;

    @Autowired
    private SQLQueryFactory queryFactory;

    public Long save(Long programaId, byte[] curriculo) throws JAXBException, IOException {
        CurriculoVitaeXmlDto xmlDto = LattesXmlConverter.unmarshal(new
            ↳ ByteArrayInputStream(curriculo));

        QXmlDataEntity qXml = QXmlDataEntity.xmlDataEntity;
        QProgramaXmlDataEntity qProgXml =
            ↳ QProgramaXmlDataEntity.programaXmlDataEntity;

        JPAQuery query = new JPAQuery(this.em).from(qXml);
        query.join(qXml.programaXml, qProgXml);
        query.where(qXml.identificador.eq(xmlDto.getNumeroIdentificador()).and(qProgXml.programa().id.eq(programaId)));
        Tuple tuple = query.uniqueResult(qXml.id, qXml.dataAtualizacao, qProgXml.id);

        XmlDataEntity xmlDataEntity;
        if (tuple != null) {
            if
                ↳ (xmlDto.getDataAtualizacao().compareTo(tuple.get(qXml.dataAtualizacao))
                ↳ > 0) {
                    xmlDataEntity = this.em.getReference(XmlDataEntity.class,
                        ↳ tuple.get(qXml.id));
                } else {
                    return tuple.get(qXml.id);
                }
            } else {
                xmlDataEntity = new XmlDataEntity();
            }
        }
        this.curriculoDtoToXmlData(curriculo, xmlDto, xmlDataEntity);
        this.em.persist(xmlDataEntity);

        if (tuple == null) {
            ProgramaXmlDataEntity programaXmlData = new
                ↳ ProgramaXmlDataEntity();
            programaXmlData.setPrograma(this.em.getReference(ProgramaEntity.class,
                ↳ programaId));
            programaXmlData.setXmlData(xmlDataEntity);
            this.em.persist(programaXmlData);
        }

        return xmlDataEntity.getId();
    }
}

```



```

public Long saveUserFile(Long userId, Long programaId, byte[] curriculo) throws
↳ JAXBException, IOException {
    CurriculoVitaeXmlDto xmlDto = LattesXmlConverter.unmarshal(new
↳ ByteArrayInputStream(curriculo));

    QtbProgramaUser qTbProgUser = QtbProgramaUser.tbProgramaUser;
    QtbProgramaXmlData qTbProgXml = QtbProgramaXmlData.tbProgramaXmlData;
    QtbXmlData qTbXml = QtbXmlData.tbXmlData;

    SQLQuery query = this.queryFactory.query();
    query.from(qTbProgUser);
    query.join(qTbProgXml).on(qTbProgUser.idPrograma.eq(qTbProgXml.idPrograma));
    query.join(qTbXml).on(qTbProgXml.idXmlData.eq(qTbXml.idXmlData));

    query.where(qTbProgUser.idUser.eq(userId));
    query.where(qTbXml.identificador.eq(xmlDto.getNumeroIdentificador()));
    query.where(qTbXml.dataAtualizacao.gt(new
↳ java.sql.Date(xmlDto.getDataAtualizacao().getTime())));
    query.orderBy(qTbXml.dataAtualizacao.desc());

    // ve se possui um curriculo mais atual em algum dos programas que o usuario
↳ participa
    Long idXmlMostRecentlyInProg = query.singleResult(qTbXml.idXmlData);

    if (idXmlMostRecentlyInProg == null) {
        // o curriculo importado o mais atual
        // importa esse curriculo
        XmlDataEntity xmlDataEntity = new XmlDataEntity();
        this.curriculoDtoToXmlData(curriculo, xmlDto, xmlDataEntity);
        this.em.persist(xmlDataEntity);
        this.em.flush();
        idXmlMostRecentlyInProg = xmlDataEntity.getId();
    }
    // trais todos os programas onde o usuario participa
    List<Long> programas = new JPAQuery(this.em).from(programaUserEntity)
        .where(programaUserEntity.user().id.eq(userId))
        .list(programaUserEntity.programa().id);

    for (Long participaProgramaId : programas) {
        Long progXmlId = new JPAQuery(this.em).from(programaXmlDataEntity)
            .join(programaXmlDataEntity.xmlData(), xmlDataEntity)
            .where(programaXmlDataEntity.programa().id.eq(participaProgramaId))
            .where(xmlDataEntity.identificador.eq(xmlDto.getNumeroIdentificador()))
            .uniqueResult(programaXmlDataEntity.id);

        if (progXmlId == null) {
            // criar o xml no programa se no possuir
            ProgramaXmlDataEntity programaXmlData = new
↳ ProgramaXmlDataEntity();
            programaXmlData.setPrograma(this.em.getReference(ProgramaEntity.class,
↳ participaProgramaId));
            programaXmlData.setXmlData(this.em.getReference(XmlDataEntity.class,
↳ idXmlMostRecentlyInProg));
            this.em.persist(programaXmlData);
        } else {
            // atualizar o programa para usar o novo xml
            QProgramaXmlDataEntity qProgXml =
↳ QProgramaXmlDataEntity.programaXmlDataEntity;
            JPAUpdateClause updateClause = new JPAUpdateClause(this.em,
↳ qProgXml);
            updateClause.set(qProgXml.xmlData().id,
↳ idXmlMostRecentlyInProg);
            updateClause.where(qProgXml.id.eq(progXmlId));
            updateClause.execute();
        }
    }

    // atualizar o usuario para usar o novo curriculo
    QUserEntity qUser = QUserEntity.userEntity;
    JPAUpdateClause updateClause = new JPAUpdateClause(this.em, qUser);
    updateClause.set(qUser.xmlData().id, idXmlMostRecentlyInProg);
    updateClause.where(qUser.id.eq(userId));
    updateClause.execute();
    return idXmlMostRecentlyInProg;
}

private void curriculoDtoToXmlData(byte[] curriculo, CurriculoVitaeXmlDto xmlDto,
↳ XmlDataEntity xmlDataEntity) {
    xmlDataEntity.setIdentificador(xmlDto.getNumeroIdentificador());
    xmlDataEntity.setNomeCompleto(xmlDto.getDadosGerais().getNomeCompleto());
    xmlDataEntity.setNomeCompletoFiltro(QueryUtils.lowerCaseStripAccents(xmlDto.getDadosGerais())
    xmlDataEntity.setDataAtualizacao(xmlDto.getDataAtualizacao());
}

```

```

xmlDataEntity.setDataImportacao(new Date());
xmlDataEntity.setXml(curriculo);
}

public Page<UploadCurriculoRowDTO> getPage(UploadCurriculoFilterDTO filter, Pageable
↪ pageable) {
    QXmlDataEntity qXml = QXmlDataEntity.xmlDataEntity;
    QProgramaXmlDataEntity qProgXml =
    ↪ QProgramaXmlDataEntity.programaXmlDataEntity;

    JPAQuery query = new JPAQuery(this.em).from(qXml);
    query.join(qXml.programaXml, qProgXml);
    query.leftJoin(qXml.user(), userEntity);

    query.where(qProgXml.programa().id.eq(filter.getProgramaId()));
    if (filter.hasNome()) {
        query.where(qXml.nomeCompletoFiltro.like(QueryUtils.like(filter.getNome()));)
    }

    if (filter.hasImportacaoInicio()) {
        query.where(qXml.dataImportacao.goe(filter.getImportacaoInicio()));
    }
    if (filter.hasImportacaoFim()) {
        query.where(qXml.dataImportacao.loe(filter.getImportacaoInicio()));
    }

    if (filter.hasAtualizacaoInicio()) {
        query.where(qXml.dataAtualizacao.goe(filter.getAtualizacaoInicio()));
    }
    if (filter.hasAtualizacaoFim()) {
        query.where(qXml.dataAtualizacao.loe(filter.getAtualizacaoFim()));
    }
    query.offset(pageable.getOffset());
    query.limit(pageable.getPageSize());

    NumberExpression<Integer> owner = new
    ↪ CaseBuilder().when(userEntity.id.eq(filter.getUserId())).then(1).otherwise(0);
    query.orderBy(owner.desc(), qXml.nomeCompletoFiltro.asc());

    Select<UploadCurriculoRowDTO> select = new
    ↪ Select<>(UploadCurriculoRowDTO.class);
    MUploadCurriculoRowDTO meta = MUploadCurriculoRowDTO.meta;
    select.as(qXml.id, meta.id);
    select.as(qProgXml.id, meta.programaXmlId);
    select.as(qXml.nomeCompleto, meta.nomeCompleto);
    select.as(qXml.identificador, meta.identificador);
    select.as(qXml.dataAtualizacao, meta.dataAtualizacao);
    select.as(qXml.dataImportacao, meta.dataImportacao);
    select.as(owner, meta.owner);

    return new PageImpl<>(query.list(select), pageable, query.count());
}

public void deleteXmlWithoutReference(Long xmlId) {
    QUserEntity qUser = QUserEntity.userEntity;
    QXmlDataEntity qXml = QXmlDataEntity.xmlDataEntity;
    QProgramaXmlDataEntity qProgXml =
    ↪ QProgramaXmlDataEntity.programaXmlDataEntity;

    JPASubQuery subQueryUser = new JPASubQuery().from(qUser);
    subQueryUser.where(qUser.xmlData().id.eq(xmlId));

    JPASubQuery subQueryProgXml = new JPASubQuery().from(qProgXml);
    subQueryProgXml.where(qProgXml.xmlData().id.eq(xmlId));

    JPADeleteClause deleteClause = new JPADeleteClause(this.em, qXml);
    deleteClause.where(qXml.id.eq(xmlId));
    deleteClause.where(subQueryUser.notExists());
    deleteClause.where(subQueryProgXml.notExists());
    deleteClause.execute();
}

public void deleteXmlFromProgram(Long programaXmlId, Long xmlId) {
    QProgramaXmlDataEntity qProgXml =
    ↪ QProgramaXmlDataEntity.programaXmlDataEntity;

    JPADeleteClause deleteClause = new JPADeleteClause(this.em, qProgXml);
    deleteClause.where(qProgXml.id.eq(programaXmlId));
    deleteClause.execute();

    this.deleteXmlWithoutReference(xmlId);
}

```

```

}
package alpc.ufsc.xml;

import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.util.zip.ZipEntry;
import java.util.zip.ZipInputStream;

import javax.xml.bind.JAXBException;

import lombok.Cleanup;
import lombok.extern.slf4j.Slf4j;

import org.apache.commons.io.IOUtils;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.messaging.simp.SimpMessagingTemplate;
import org.springframework.security.access.annotation.Secured;
import org.springframework.util.MimeTypeUtils;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.context.request.async.DeferredResult;
import org.springframework.web.multipart.MultipartFile;

import alpc.ufsc.common.exception.ProcessAlreadyRunningException;
import alpc.ufsc.config.web.util.nonblocking.NonBlockingIOThreadPool;
import alpc.ufsc.login.authority.Authority;
import alpc.ufsc.login.user.CurrentUser;
import alpc.ufsc.login.user.dto.UserDetailsImpl;
import alpc.ufsc.xml.ProcessCurriculoService.ProcessCurriculoStatus;
import alpc.ufsc.xml.dto.UploadCurriculoFilterDTO;
import alpc.ufsc.xml.dto.UploadCurriculoRowDTO;
import alpc.ufsc.xml.dto.UploadCurriculoSaveDTO;
import alpc.ufsc.xml.message.UploadCurriculoMessageManager;

@Slf4j
@RestController
@RequestMapping(value = "/services/curriculo")
public class UploadCurriculoRestController {

    private static final String APPLICATION_ZIP = "application/zip";

    private static final String INVALID_TYPE_MESSAGE = "Tipo invlido";
    private static final String IO_ERROR_MESSAGE = "Erro na leitura do arquivo";
    private static final String XML_ERROR_MESSAGE = "XML invlido";

    @Autowired
    private UploadCurriculoService uploadService;

    @Autowired
    private ProcessCurriculoService processService;

    @Autowired
    private NonBlockingIOThreadPool threadPool;

    @Autowired
    private SimpMessagingTemplate template;

    @Secured({ Authority.COORDENADOR, Authority.DOCENTE })
    @RequestMapping(method = RequestMethod.POST)
    public DeferredResult<String> uploadFiles(@CurrentUser UserDetailsImpl user,
        ↪ UploadCurriculoSaveDTO uploadCurriculo) {
        DeferredResult<String> result = new DeferredResult<>(10L * 60L * 1000L);
        UploadCurriculoMessageManager messenger = new
            ↪ UploadCurriculoMessageManager(this.template, user.getId(), true);

        this.threadPool.execute(() -> {
            for (MultipartFile file : uploadCurriculo.getFiles()) {
                try {
                    if
                    ↪ (MimeTypeUtils.TEXT_XML_VALUE.equals(file.getContentType()))
                    ↪ {
                        @Cleanup
                        InputStream inputStream = file.getInputStream();
                        this.uploadService.save(user.getProgramId(),

```

```

        ↪ IOUtils.toByteArray(inputStream));
        messenger.sendMessage(file.getOriginalFilename());
    } else if
        ↪ (APPLICATION_ZIP.equals(file.getContentType()))
        ↪ {
            this.unzip(messenger, user.getProgramId(), file);
    } else {
        messenger.sendMessage(file.getOriginalFilename(),
            ↪ INVALID_TYPE_MESSAGE);
    }
} catch (IOException e) {
    log.error(file.getOriginalFilename() + " | " +
        ↪ IO_ERROR_MESSAGE, e);
    messenger.sendMessage(file.getOriginalFilename(),
        ↪ IO_ERROR_MESSAGE);
} catch (JAXBException e) {
    log.error(file.getOriginalFilename() + " | " +
        ↪ XML_ERROR_MESSAGE, e);
    messenger.sendMessage(file.getOriginalFilename(),
        ↪ XML_ERROR_MESSAGE);
}
}

messenger.sendFinishMessage();
this.refreshModule(user);
result.setResult("success");
});
return result;
}

private void unzip(UploadCurriculoMessageManager messenger, Long programId,
    ↪ MultipartFile file) throws IOException {
    @Cleanup
    ZipInputStream zipInputStream = new ZipInputStream(file.getInputStream());
    ZipEntry entry = null;
    while ((entry = zipInputStream.getNextEntry()) != null) {
        if (entry.getName().endsWith(".xml")) {
            try {
                ByteArrayOutputStream baos = new
                    ↪ ByteArrayOutputStream();
                byte[] buffer = new byte[32768];
                int numBytes;
                while ((numBytes = zipInputStream.read(buffer, 0,
                    ↪ buffer.length)) != -1) {
                    baos.write(buffer, 0, numBytes);
                }
                this.uploadService.save(programId, baos.toByteArray());
                messenger.sendMessage(entry.getName());
            } catch (IOException e) {
                log.error(entry.getName() + " | " +
                    ↪ IO_ERROR_MESSAGE, e);
                messenger.sendMessage(entry.getName(),
                    ↪ IO_ERROR_MESSAGE);
            } catch (JAXBException e) {
                log.error(entry.getName() + " | " +
                    ↪ XML_ERROR_MESSAGE, e);
                messenger.sendMessage(entry.getName(),
                    ↪ XML_ERROR_MESSAGE);
            }
        } else {
            messenger.sendMessage(entry.getName(),
                ↪ INVALID_TYPE_MESSAGE);
        }
        zipInputStream.closeEntry();
    }
}

@Secured({ Authority.COORDENADOR, Authority.DOCENTE, Authority.DISCENTE })
@RequestMapping(value = "userFile", method = RequestMethod.POST)
public DeferredResult<String> uploadYourFile(@CurrentUser UserDetailsImpl user,
    ↪ MultipartFile files) {
    DeferredResult<String> result = new DeferredResult<>(60l * 1000l);
    UploadCurriculoMessageManager messenger = new
        ↪ UploadCurriculoMessageManager(this.template, user.getId(), false);

    this.threadPool.execute() -> {
        try {
            if
                ↪ (MimeTypeUtils.TEXT_XML_VALUE.equals(files.getContentType()))

```

```

        ↪ {
        @Cleanup
        InputStream inputStream = files.getInputStream();
        this.uploadService.saveUserFile(user.getId(),
        ↪ user.getProgramId(),
        ↪ IOUtils.toByteArray(inputStream));
        messenger.sendMessage(files.getOriginalFilename());
    } else {
        messenger.sendMessage(files.getOriginalFilename(),
        ↪ INVALID_TYPE_MESSAGE);
    }
} catch (IOException e) {
    log.error(files.getOriginalFilename() + " | " +
    ↪ IO_ERROR_MESSAGE, e);
    messenger.sendMessage(files.getOriginalFilename(),
    ↪ IO_ERROR_MESSAGE);
} catch (JAXBException e) {
    log.error(files.getOriginalFilename() + " | " +
    ↪ XML_ERROR_MESSAGE, e);
    messenger.sendMessage(files.getOriginalFilename(),
    ↪ XML_ERROR_MESSAGE);
}

    messenger.sendFinishMessage();
    this.refreshModule(user);
    result.setResult("success");
});
return result;
}

@Secured({ Authority.COORDENADOR, Authority.DOCENTE, Authority.DISCENTE })
@RequestMapping(method = RequestMethod.GET)
public @ResponseBody Page<UploadCurriculoRowDTO> getPage(@CurrentUser
    ↪ UserDetailsImpl user, UploadCurriculoFilterDTO filter, Pageable pageable) {
    filter.setProgramId(user.getProgramId());
    filter.setUserId(user.getId());
    return this.uploadService.getPage(filter, pageable);
}

@Secured({ Authority.COORDENADOR })
@RequestMapping(value = "process/status", method = RequestMethod.GET)
public @ResponseBody ProcessCurriculoStatus getProcessStatus(@CurrentUser
    ↪ UserDetailsImpl user) {
    return this.processService.getProcessStatus(user.getProgramId());
}

@Secured({ Authority.COORDENADOR })
@RequestMapping(value = "{programaXmlId}/{xmlId}", method =
    ↪ RequestMethod.DELETE)
public void delete(@CurrentUser UserDetailsImpl user, @PathVariable Long programaXmlId,
    ↪ @PathVariable Long xmlId) {
    this.uploadService.deleteXmlFromProgram(programaXmlId, xmlId);
    this.refreshModule(user);
}

@Secured({ Authority.COORDENADOR })
@RequestMapping(method = RequestMethod.DELETE)
public ResponseEntity<String> deleteAllProgramData(@CurrentUser UserDetailsImpl user)
    ↪ {
    try {
        this.processService.deleteAllProgramData(user.getProgramId());
        return new ResponseEntity<>("Proceso iniciado", HttpStatus.OK);
    } catch (ProcessAlreadyRunningException e) {
        return new ResponseEntity<>(e.getMessage(),
        ↪ HttpStatus.BAD_REQUEST);
    }
}

@Secured({ Authority.COORDENADOR })
@RequestMapping(value = "process", method = RequestMethod.POST)
public ResponseEntity<String> processAll(@CurrentUser UserDetailsImpl user) {
    try {
        this.processService.executeProcessTask(user.getProgramId(), null);
        return new ResponseEntity<>("Proceso iniciado", HttpStatus.OK);
    } catch (ProcessAlreadyRunningException e) {
        return new ResponseEntity<>(e.getMessage(),
        ↪ HttpStatus.BAD_REQUEST);
    }
}

@Secured({ Authority.COORDENADOR })
@RequestMapping(value = "process/{id}", method = RequestMethod.POST)

```

```

public ResponseEntity<String> process(@CurrentUser UserDetailsImpl user, @PathVariable
↳ Long id) {
    try {
        this.processService.executeProcessTask(user.getProgramaId(), id);
        return new ResponseEntity<>("Processo iniciado", HttpStatus.OK);
    } catch (ProcessAlreadyRunningException e) {
        return new ResponseEntity<>(e.getMessage(),
↳ HttpStatus.BAD_REQUEST);
    }
}

private void refreshModule(UserDetailsImpl user) {
    this.template.convertAndSend("/topic/" + user.getProgramaId() + "/curriculo",
↳ "foi");
}
}

package alpc.ufsc.xml;

import static alpc.ufsc.entity.xml.QProgramaXmlDataEntity.programaXmlDataEntity;
import static alpc.ufsc.entity.xml.QXmlDataEntity.xmlDataEntity;

import java.io.ByteArrayInputStream;
import java.util.List;
import java.util.Map;
import java.util.concurrent.ArrayBlockingQueue;
import java.util.concurrent.ConcurrentHashMap;
import java.util.concurrent.ThreadPoolExecutor;
import java.util.concurrent.TimeUnit;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;
import lombok.extern.slf4j.Slf4j;

import org.springframework.beans.factory.InitializingBean;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.messaging.simp.SimpMessagingTemplate;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Propagation;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.Tuple;
import com.mysema.query.jpa.impl.JPAQuery;
import com.mysema.query.types.path.NumberPath;

import alpc.ufsc.common.exception.ProcessAlreadyRunningException;
import alpc.ufsc.config.web.ProcessCurriculoProperties;
import alpc.ufsc.pessoa.atuacaoprofissional.linhaPesquisa.LinhaPesquisaService;
import alpc.ufsc.pessoa.producao.service.ProducaoService;
import alpc.ufsc.programa.ProgramaService;
import alpc.ufsc.programa.dto.ProgramaFormDTO;
import alpc.ufsc.util.MinMaxPagination;
import alpc.ufsc.util.MinMaxPagination.FromTo;
import alpc.ufsc.xml.message.ProcessCurriculoMessageManager;
import ufsc.alpc.converter.LattesXmlConverter;

@Slf4j
@Service
@Transactional(propagation = Propagation.SUPPORTS)
public class ProcessCurriculoService implements InitializingBean {
    private static final String waiting = "waiting";
    private static final String processing = "processing";
    private static final String excluding = "excluding";
    private static final String finalized = "finalized";

    @PersistenceContext
    private EntityManager em;

    @Autowired
    private ProgramaService service;

    @Autowired
    private LinhaPesquisaService linhaPesquisaService;

    @Autowired
    private ProjetoPesquisaService projetoService;

```

```

@Autowired
private ProducaoService producaoService;

@Autowired
private SimpMessagingTemplate template;

@Autowired
private ProcessCurriculoProperties props;

private ThreadPoolExecutor threadPool;
private Map<Long, ProcessCurriculoStatus> runningTasks;

@Override
public void afterPropertiesSet() throws Exception {
    //@formatter:off
    this.threadPool = new ThreadPoolExecutor(
        this.props.getMinThreads(),
        this.props.getMaxThreads(),
        60, TimeUnit.SECONDS,
        new ArrayBlockingQueue<>(this.props.getWorkQueueSize()));
    //@formatter:on
    this.runningTasks = new ConcurrentHashMap<>();
}

public void executeProcessTask(Long programaId, Long xmlId) throws
↳ ProcessAlreadyRunningException {
    if (this.runningTasks.containsKey(programaId)) {
        throw new ProcessAlreadyRunningException();
    }

    ProcessCurriculoMessageManager messageManager = new
↳ ProcessCurriculoMessageManager(this.template, programaId);

    this.updateStatus(programaId, messageManager, new
↳ ProcessCurriculoStatus(waiting, 0L));
    this.threadPool.execute(() -> {
        this.process(messageManager, programaId, xmlId);
    });
}

private void process(ProcessCurriculoMessageManager messageManager, Long programaId,
↳ Long xmlId) {
    try {
        this.updateStatus(programaId, messageManager, new
↳ ProcessCurriculoStatus(processing, 0L));

        ProgramaFormDTO programaDTO = this.service.load(programaId);

        NumberPath<Long> xmlPathId = xmlDataEntity.id;
        Tuple result = this.getQuery(programaId,
↳ xmlId).uniqueResult(xmlPathId.min(), xmlPathId.max());

        MinMaxPagination paginator = new
↳ MinMaxPagination(result.get(xmlPathId.min()),
↳ result.get(xmlPathId.max()), 5);
        for (FromTo page : paginator) {
            JPAQuery query = this.getQuery(programaId,
↳ xmlId).where(xmlPathId.between(page.from, page.to));
            List<Tuple> list = query.list(programaXmlDataEntity.id,
↳ xmlDataEntity.xml);

            Long current = page.to + 1 - paginator.getMin();
            if (!list.isEmpty()) {
                int size = 0;
                int gap = paginator.getBatch() / list.size();

                for (Tuple xmls : list) {
                    try {
                        this.service.importLattes(LattesXmlConverter.unmarshal(n
↳ ByteArrayInputStream(xmls.get(xmlDataEntity.xml),
↳ programaDTO));

                        size = size + gap;
                        double percent = (current.doubleValue() -
↳ (paginator.getBatch() - size)) /
↳ paginator.getTotal().doubleValue()
↳ * 100;

                        this.updateStatus(programaId,
↳ messageManager, new
↳ ProcessCurriculoStatus(processing,
↳ Math.round(percent)));
                    } catch (Exception e) {

```

```

        log.error("Erro inesperado, no deveria haver
        ↪ curriculos invalidos no
        ↪ processamento. ", e);
    }
}
}
double percent = current.doubleValue() /
    ↪ paginador.getTotal().doubleValue() * 100;
this.updateStatus(programaId, messageManager, new
    ↪ ProcessCurriculoStatus(processing, Math.round(percent)));
}
this.service.refreshDeleteMestrado(programaId);
this.service.refreshDeleteDoutorado(programaId);
this.service.refreshViewPertencePrograma();
} finally {
    this.updateStatus(programaId, messageManager, new
        ↪ ProcessCurriculoStatus(finalized, 100L));
    this.runningTasks.remove(programaId);
}
}

private JPAQuery getQuery(Long programaId, Long xmlId) {
    JPAQuery query = new JPAQuery(this.em).from(xmlDataEntity);
    query.join(xmlDataEntity.programaXml, programaXmlDataEntity);
    query.where(programaXmlDataEntity.programa().id.eq(programaId));
    if (xmlId != null) {
        query.where(xmlDataEntity.id.eq(xmlId));
    }
    return query;
}

private void updateStatus(Long programaId, ProcessCurriculoMessageManager
    ↪ messageManager, ProcessCurriculoStatus message) {
    this.runningTasks.put(programaId, message);
    messageManager.sendMessage(message);
}

public ProcessCurriculoStatus getProcessStatus(Long programaId) {
    if (this.runningTasks.containsKey(programaId)) {
        return this.runningTasks.get(programaId);
    }
    return new ProcessCurriculoStatus();
}

public void deleteAllProgramData(Long programaId) throws
    ↪ ProcessAlreadyRunningException {
    if (this.runningTasks.containsKey(programaId)) {
        throw new ProcessAlreadyRunningException();
    } else {
        this.threadPool.execute() -> {
            try {
                ProcessCurriculoMessageManager messageManager = new
                    ↪ ProcessCurriculoMessageManager(this.template,
                    ↪ programaId);
                this.updateStatus(programaId, messageManager, new
                    ↪ ProcessCurriculoStatus(excluding, 0L));

                this.linhaPesquisaService.deleteLinhaPesquisa(programaId);
                this.updateStatus(programaId, messageManager, new
                    ↪ ProcessCurriculoStatus(excluding, 10L));

                this.projetoService.deleteProjetoPesquisa(programaId);
                this.updateStatus(programaId, messageManager, new
                    ↪ ProcessCurriculoStatus(excluding, 20L));

                this.producaoService.deleteProjetoPesquisa(programaId);
                this.updateStatus(programaId, messageManager, new
                    ↪ ProcessCurriculoStatus(excluding, 30L));

                this.service.deleteGraduacao(programaId);
                this.updateStatus(programaId, messageManager, new
                    ↪ ProcessCurriculoStatus(excluding, 40L));

                this.service.deleteMestrado(programaId);
                this.updateStatus(programaId, messageManager, new
                    ↪ ProcessCurriculoStatus(excluding, 50L));

                this.service.deleteDoutorado(programaId);
                this.updateStatus(programaId, messageManager, new
                    ↪ ProcessCurriculoStatus(excluding, 60L));

                this.service.deletePosDoutorado(programaId);

```



```

        this.updateStatus(programaId, messageManager, new
            ↪ ProcessCurriculoStatus(excluding, 75L));

        this.service.forceProcessamentoPessoa(programaId);
        this.updateStatus(programaId, messageManager, new
            ↪ ProcessCurriculoStatus(excluding, 87L));

        this.service.refreshViewPertencePrograma();
        this.updateStatus(programaId, messageManager, new
            ↪ ProcessCurriculoStatus(finalized, 100L));
    } finally {
        this.runningTasks.remove(programaId);
    }
    }
    });
}

@Getter
@Setter
@AllArgsConstructor
@NoArgsConstructor
public class ProcessCurriculoStatus {
    String status = "";
    Long percent = 0L;
}
}
package alpc.ufsc.xml.dto;

import java.util.Date;

import lombok.Getter;
import lombok.Setter;

import br.ufsc.bridge.metafy.Metafy;

@Getter
@Setter
@Metafy
public class UploadCurriculoRowDTO {

    private Long id;
    private Long programaXmlId;
    private String nomeCompleto;
    private String identificador;
    private Date dataImportacao;
    private Date dataAtualizacao;
    private Integer owner;
}
package alpc.ufsc.xml.dto;

import java.util.Date;

import lombok.Getter;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;
import org.springframework.format.annotation.DateTimeFormat;

@Getter
@Setter
public class UploadCurriculoFilterDTO {

    private Long programaId;
    private Long userId;
    private String nome;

    @DateTimeFormat(pattern = "yyyy-MM-dd")
    private Date importacaoInicio;
    @DateTimeFormat(pattern = "yyyy-MM-dd")
    private Date importacaoFim;

    @DateTimeFormat(pattern = "yyyy-MM-dd")
    private Date atualizacaoInicio;
    @DateTimeFormat(pattern = "yyyy-MM-dd")
    private Date atualizacaoFim;

    public boolean hasNome() {
        return !StringUtils.isBlank(this.nome);
    }

    public boolean hasImportacaoInicio() {
        return this.importacaoInicio != null;
    }
}

```

```

    }

    public boolean hasImportacaoFim() {
        return this.importacaoFim != null;
    }

    public boolean hasAtualizacaoInicio() {
        return this.atualizacaoInicio != null;
    }

    public boolean hasAtualizacaoFim() {
        return this.atualizacaoFim != null;
    }
}
package alpc.ufsc.xml.dto;

import lombok.Getter;
import lombok.Setter;

@Getter
@Setter
public class UploadCurriculoProcessKey {

    public static final String PROCESSANDO = "Processando";
    public static final String DELETANDO = "Deletando";

    private Long programaId;
    private String tipe;
}
package alpc.ufsc.xml.dto;

import java.util.List;

import lombok.Getter;
import lombok.Setter;

import org.springframework.web.multipart.MultipartFile;

@Getter
@Setter
public class UploadCurriculoSaveDTO {

    private List<MultipartFile> files;
}
package alpc.ufsc.xml.message;

import java.util.LinkedHashMap;
import java.util.Map;

import lombok.Getter;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;
import org.springframework.messaging.simp.SimpMessagingTemplate;

import com.fasterxml.jackson.annotation.JsonInclude;
import com.fasterxml.jackson.annotation.JsonInclude.Include;

@Getter
@Setter
public class UploadCurriculoMessageManager {

    private static final String reading = "reading";
    private static final String success = "sucess";
    private static final String error = "error";

    private SimpMessagingTemplate template;
    private String url;

    private Integer totalFiles;
    private Map<String, String> filesErrors;

    public UploadCurriculoMessageManager(SimpMessagingTemplate template, Long userId,
        ↪ Boolean mutiple) {
        this.template = template;
        this.url = "/topic/" + userId + "/curriculo/upload" + (mutiple ? "/multiple" : "");
        this.totalFiles = 0;
        this.filesErrors = new LinkedHashMap<>();
    }

    public void sendMessage(String filename) {
        this.sendMessage(filename, null);
    }
}

```

```

    }

    public void sendMessage(String filename, String error) {
        this.totalFiles++;
        if (!StringUtils.isBlank(error)) {
            this.filesErrors.put(filename, error);
        }
        this.template.convertAndSend(this.url, new UploadCurriculoMessage(filename));
    }

    public void sendFinishMessage() {
        this.template.convertAndSend(this.url, new UploadCurriculoMessage());
    }

    @Getter
    @Setter
    @JsonInclude(Include.NON_EMPTY)
    class UploadCurriculoMessage {
        String fileName;
        Integer totalFiles;
        Integer totalErrorFiles;
        Map<String, String> filesErrors;
        String status;

        public UploadCurriculoMessage() {
            this.totalFiles = UploadCurriculoMessageManager.this.totalFiles;
            this.totalErrorFiles = UploadCurriculoMessageManager.this.filesErrors.size();
            this.filesErrors = UploadCurriculoMessageManager.this.filesErrors;
            if (this.totalErrorFiles > 0) {
                this.status = error;
            } else {
                this.status = sucess;
            }
        }

        public UploadCurriculoMessage(String filename) {
            this.fileName = filename;
            this.totalFiles = UploadCurriculoMessageManager.this.totalFiles;
            this.totalErrorFiles = UploadCurriculoMessageManager.this.filesErrors.size();
            this.status = reading;
        }
    }
}

package alpc.ufsc.xml.message;

import lombok.Setter;

import org.springframework.messaging.simp.SimpMessagingTemplate;

import alpc.ufsc.xml.ProcessCurriculoService.ProcessCurriculoStatus;

@Setter
public class ProcessCurriculoMessageManager {

    private SimpMessagingTemplate template;
    private String url;

    public ProcessCurriculoMessageManager(SimpMessagingTemplate template, Long
    ↪ programaId) {
        this.template = template;
        this.url = "/topic/" + programaId + "/curriculo/process";
    }

    public void sendMessage(ProcessCurriculoStatus message) {
        this.template.convertAndSend(this.url, message);
    }
}

package alpc.ufsc.domain.instituicao;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.ResponseBody;
import org.springframework.web.bind.annotation.RestController;

import alpc.ufsc.domain.instituicao.dto.InstituicaoFilterDTO;
import alpc.ufsc.domain.instituicao.dto.InstituicaoRowDTO;

```

```

@RestController
@RequestMapping("/services/instituicao")
public class InstituicaoRestController {

    @Autowired
    private InstituicaoService service;

    @ResponseBody
    @RequestMapping(method = RequestMethod.GET)
    public List<InstituicaoRowDTO> getPage(InstituicaoFilterDTO filter) {
        return this.service.getPage(filter);
    }

    @ResponseBody
    @RequestMapping(value = "{instituicaoId}", method = RequestMethod.GET)
    public InstituicaoRowDTO load(@PathVariable Long instituicaoId) {
        return this.service.loadRow(instituicaoId);
    }
}
package alpc.ufsc.domain.instituicao.dto;

import java.util.List;

import lombok.Getter;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;

@Getter
@Setter
public class InstituicaoFilterDTO {

    private String query;
    private Integer pageSize;
    private List<Long> ids;

    public boolean hasQuery() {
        return !StringUtils.isBlank(this.query);
    }

    public boolean hasPageSize() {
        return this.pageSize != null && this.pageSize.equals(0);
    }

    public boolean hasPaisesIds() {
        return this.ids != null && !this.ids.isEmpty();
    }
}
package alpc.ufsc.domain.instituicao.dto;

import lombok.Getter;
import lombok.Setter;
import br.ufsc.bridge.metafy.Metafy;

@Getter
@Setter
@Metafy
public class InstituicaoRowDTO {

    private Long id;
    private String sigla;
    private String description;
}
package alpc.ufsc.domain.instituicao.dto;

import java.util.List;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;

import br.ufsc.bridge.metafy.Metafy;

import alpc.ufsc.common.exception.EmptyXmlException;
import alpc.ufsc.common.exception.ErrorMessages;
import alpc.ufsc.common.exception.InvalidDTOException;
import alpc.ufsc.entity.domain.estado.Estado;
import alpc.ufsc.entity.util.QueryUtils;
import ufsc.alpc.xml.dto.CurriculoVitaeXmlDto;

```

```

import ufsc.alpc.xml.dto.dados.complementares.informacoes.InformacaoAdicionalCursoXmlDto;
import ufsc.alpc.xml.dto.dados.complementares.informacoes.InformacaoAdicionalInstituicaoXmlDto;

@Getter
@Setter
@Metafy
@NoArgsConstructor
public class InstituicaoSaveDTO {

    private String nome;

    private String nomeFiltro;

    private String sigla;

    private String nomePais;

    private String nomePaisFiltro;

    private Estado estado;

    private String codigo;

    public InstituicaoSaveDTO(String codigoCurso, String codigoInst, String nome,
        ↪ CurriculoVitaeXmlDto curriculoXmlDto) throws EmptyXmlException {
        if (StringUtils.isBlank(nome)) {
            ↪ throw new EmptyXmlException("Instituio no pode ter nome nulo.");
        }

        InformacaoAdicionalInstituicaoXmlDto instituicao =
            ↪ this.findInstituicaoByCurso(codigoCurso, curriculoXmlDto);
        if (instituicao == null) {
            instituicao = this.findInstituicao(codigoInst, curriculoXmlDto);
        }

        this.nome = nome;
        this.nomeFiltro = QueryUtils.lowerCaseStripAccents(nome);
        if (instituicao != null) {
            this.codigo = codigoInst;
            this.sigla = instituicao.getSiglaInstituicao();
            this.nomePais = instituicao.getNomePaisInstituicao();
            this.nomePaisFiltro = QueryUtils.lowerCaseStripAccents(this.nomePais);
            this.estado = Estado.getBySigla(instituicao.getSiglaUFInstituicao());
        }
    }

    public void setNome(String nome) {
        this.nome = nome;
        this.nomeFiltro = QueryUtils.lowerCaseStripAccents(nome);
    }

    public void setNomePais(String nomePais) {
        this.nomePais = nomePais;
        this.nomePaisFiltro = QueryUtils.lowerCaseStripAccents(nomePais);
    }

    public boolean isSetEstado() {
        return this.estado != null;
    }

    public boolean isSetCodigo() {
        return !StringUtils.isBlank(this.codigo);
    }

    public boolean hasNomePais() {
        return !StringUtils.isBlank(this.nomePais);
    }

    public boolean hasSigla() {
        return !StringUtils.isBlank(this.sigla);
    }

    private InformacaoAdicionalInstituicaoXmlDto findInstituicaoByCurso(String codigoCurso,
        ↪ CurriculoVitaeXmlDto curriculoXmlDto) {
        if (!StringUtils.isBlank(codigoCurso) &&
            ↪ curriculoXmlDto.isSetDadosComplementares() &&
            ↪ curriculoXmlDto.getDadosComplementares().isSetInfoAdicionaisCursos()
                &&
            ↪ curriculoXmlDto.getDadosComplementares().getInfoAdicionaisCursos().isSe
            ↪ {

        for (InformacaoAdicionalCursoXmlDto curso :

```

```

        ↪ curriloXmlDto.getDadosComplementares().getInfoAdicionaisCursos().getInfAdicionalCurso()
        ↪ {
            if (codigoCurso.equals(curso.getCodigoCurso())) {
                return this.findInstituicao(curso.getCodigoInstituicao(),
                    ↪ curriloXmlDto);
            }
        }
    }
    ↪ return null;
}

private InformacaoAdicionalInstituicaoXmlDto findInstituicao(String codigoInstituicao,
    ↪ CurriculoVitaeXmlDto curriloXmlDto) {
    ↪ if (!StringUtils.isBlank(codigoInstituicao) &&
        ↪ curriloXmlDto.isSetDadosComplementares() &&
        ↪ curriloXmlDto.getDadosComplementares().isSetInfoAdicionaisInstituicoes()
            &&
            ↪ curriloXmlDto.getDadosComplementares().getInfoAdicionaisInstituicoes().isSetInfo
            ↪ {
                List<InformacaoAdicionalInstituicaoXmlDto> instituicoes =
                    ↪ curriloXmlDto.getDadosComplementares().getInfoAdicionaisInstituicoes().getInfoAdicional
                for (InformacaoAdicionalInstituicaoXmlDto instituicao : instituicoes) {
                    if (codigoInstituicao.equals(instituicao.getCodigoInstituicao())) {
                        ↪ return instituicao;
                    }
                }
            }
    ↪ return null;
}

public void validate() throws InvalidDTOException {
    InvalidDTOException dtoException = new InvalidDTOException();
    MInstituicaoSaveDTO meta = MInstituicaoSaveDTO.meta;
    ↪ if (StringUtils.isBlank(this.nome) {
        ↪ dtoException.putError(meta.nome, ErrorMessages.OBRIGATORIO);
    }
    ↪ if (StringUtils.isBlank(this.sigla) {
        ↪ dtoException.putError(meta.sigla, ErrorMessages.OBRIGATORIO);
    }
    ↪ if (StringUtils.isBlank(this.nomePais) {
        ↪ dtoException.putError(meta.nomePais, ErrorMessages.OBRIGATORIO);
    }
    ↪ if (this.estado == null) {
        ↪ dtoException.putError(meta.estado, ErrorMessages.OBRIGATORIO);
    }
    ↪ dtoException.checkThrow();
}
}

package alpc.ufsc.domain.instituicao;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.cache.annotation.Cacheable;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.BooleanBuilder;
import com.mysema.query.jpa.impl.JPAQuery;

import alpc.ufsc.domain.instituicao.dto.InstituicaoFilterDTO;
import alpc.ufsc.domain.instituicao.dto.InstituicaoRowDTO;
import alpc.ufsc.domain.instituicao.dto.InstituicaoSaveDTO;
import alpc.ufsc.domain.instituicao.dto.MInstituicaoRowDTO;
import alpc.ufsc.domain.pais.PaisService;
import alpc.ufsc.entity.domain.instituicao.InstituicaoEntity;
import alpc.ufsc.entity.domain.instituicao.QInstituicaoEntity;
import alpc.ufsc.entity.domain.pais.PaisEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.util.querydsl.Select;

@Service
@Transactional
public class InstituicaoService {

    @PersistenceContext
    private EntityManager em;

```

```

@Autowired
private PaisService paisService;

private static final String CACHE = "instituicoes";

public List<InstituicaoRowDTO> getPage(InstituicaoFilterDTO filter) {
    QInstituicaoEntity qInst = QInstituicaoEntity.instituicaoEntity;

    JPAQuery query = new JPAQuery(this.em).from(qInst);

    BooleanBuilder where = new BooleanBuilder();
    if (filter.hasQuery()) {
        where.and(qInst.nomeSiglaFiltro.like(QueryUtils.like(filter.getQuery())));
    }
    if (filter.hasPaisesIds()) {
        where.and(qInst.id.in(filter.getIds()));
    }
    if (filter.hasPageSize()) {
        query.limit(filter.getPageSize());
    }
    query.orderBy(qInst.nomeFiltro.asc());

    Select<InstituicaoRowDTO> select = this.getSelectedInstituicaoRowDTO();
    return query.where(where).list(select);
}

public InstituicaoRowDTO loadRow(Long id) {
    QInstituicaoEntity qInst = QInstituicaoEntity.instituicaoEntity;

    JPAQuery query = new JPAQuery(this.em).from(qInst).where(qInst.id.eq(id));

    Select<InstituicaoRowDTO> select = this.getSelectedInstituicaoRowDTO();
    return query.uniqueResult(select);
}

private Select<InstituicaoRowDTO> getSelectedInstituicaoRowDTO() {
    QInstituicaoEntity qInst = QInstituicaoEntity.instituicaoEntity;
    Select<InstituicaoRowDTO> select = new Select<>(InstituicaoRowDTO.class);
    MInstituicaoRowDTO meta = MInstituicaoRowDTO.meta;
    select.as(qInst.id, meta.id);
    select.as(qInst.sigla, meta.sigla);
    select.as(qInst.nome, meta.description);
    return select;
}

@Cacheable(value = CACHE, key = "{ #dto.nomeFiltro }", unless =
    ↪ "!#dto.hasNomePais()")
public Long save(InstituicaoSaveDTO dto) {
    QInstituicaoEntity qInst = QInstituicaoEntity.instituicaoEntity;
    JPAQuery query = new JPAQuery(this.em).from(qInst);
    query.where(qInst.nomeFiltro.eq(dto.getNomeFiltro()));

    InstituicaoEntity instituicaoEntity = query.uniqueResult(qInst);
    if (instituicaoEntity == null) {
        instituicaoEntity = new InstituicaoEntity();
    }

    instituicaoEntity.setNome(dto.getNome());
    instituicaoEntity.setNomeFiltro(dto.getNomeFiltro());

    if (dto.hasNomePais()) {
        instituicaoEntity.setPais(this.em.getReference(PaisEntity.class,
            ↪ this.paisService.save(dto.getNomePais()),
            ↪ dto.getNomePaisFiltro()));
    }

    String nomeSiglaFiltro = dto.getNomeFiltro();
    if (dto.hasSigla()) {
        instituicaoEntity.setSigla(dto.getSigla());
        nomeSiglaFiltro += " " + QueryUtils.lowerCaseStripAccents(dto.getSigla());
    }
    instituicaoEntity.setNomeSiglaFiltro(nomeSiglaFiltro);

    if (dto.isSetEstado()) {
        instituicaoEntity.setEstado(dto.getEstado());
    }
    this.em.persist(instituicaoEntity);
    return instituicaoEntity.getId();
}
}
package alpc.ufsc.domain.area;

```

```

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.springframework.cache.annotation.Cacheable;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.jpa.impl.JPAQuery;

import alpc.ufsc.domain.area.dto.AreaConhecimentoSaveDTO;
import alpc.ufsc.entity.domain.area.AreaConhecimentoEntity;
import alpc.ufsc.entity.domain.area.QAreaConhecimentoEntity;

@Service
@Transactional
public class AreaConhecimentoService {

    private static final String CACHE = "areasConhecimento";

    private QAreaConhecimentoEntity qAreaConhecimento =
        ↪ QAreaConhecimentoEntity.areaConhecimentoEntity;

    @PersistenceContext
    private EntityManager em;

    @Cacheable(value = CACHE, key = "{ #dto.areaConhecimentoFiltro }")
    public Long save(AreaConhecimentoSaveDTO dto) {
        JPAQuery query = new JPAQuery(this.em).from(this.qAreaConhecimento);
        query.where(this.qAreaConhecimento.areaConhecimentoFiltro.eq(dto.getAreaConhecimentoFiltro()));
        AreaConhecimentoEntity entity = query.uniqueResult(this.qAreaConhecimento);
        if (entity != null) {
            return entity.getId();
        } else {
            entity = new AreaConhecimentoEntity();
            entity.setGrandeAreaConhecimento(dto.getGrandeAreaConhecimento());
            entity.setNomeAreaConhecimento(dto.getNomeAreaConhecimento());
            entity.setNomeSubAreaConhecimento(dto.getNomeSubAreaConhecimento());
            entity.setNomeEspecialidade(dto.getNomeEspecialidade());
            entity.setAreaConhecimentoFiltro(dto.getAreaConhecimentoFiltro());
            this.em.persist(entity);
            return entity.getId();
        }
    }
}

package alpc.ufsc.domain.area.dto;

import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;

import com.fasterxml.jackson.annotation.JsonGetter;
import com.fasterxml.jackson.annotation.JsonIgnore;

import ufsc.alpc.xml.dto.domain.GrandeAreaDoConhecimento;

@Getter
@Setter
@NoArgsConstructor
@AllArgsConstructor
public class AreaConhecimentoRowDTO {

    private GrandeAreaDoConhecimento grandeAreaConhecimento;

    private String nomeAreaConhecimento;

    private String nomeSubAreaConhecimento;

    private String nomeEspecialidade;

    @JsonGetter(value = "grandeAreaConhecimento")
    public String getGrandeAreaConhecimentoDescricao() {
        return this.grandeAreaConhecimento == null ? "" :
            ↪ this.grandeAreaConhecimento.getDescricao();
    }

    @JsonIgnore
    public boolean isEmpty() {
        return this.grandeAreaConhecimento == null &&

```



```

        ↪ StringUtils.isBlank(this.nomeAreaConhecimento) &&
        ↪ StringUtils.isBlank(this.nomeSubAreaConhecimento)
            && StringUtils.isBlank(this.nomeEspecialidade);
    }
}
package alpc.ufsc.domain.area.dto;

import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;

import alpc.ufsc.common.exception.EmptyXmlException;
import alpc.ufsc.entity.util.QueryUtils;

import ufsc.alpc.xml.dto.common.area.conhecimento.AreaDoConhecimentoXmlDto;
import ufsc.alpc.xml.dto.domain.GrandeAreaDoConhecimento;

@Getter
@Setter
@NoArgsConstructor
public class AreaConhecimentoSaveDTO {

    private GrandeAreaDoConhecimento grandeAreaConhecimento;

    private String nomeAreaConhecimento;

    private String nomeSubAreaConhecimento;

    private String nomeEspecialidade;

    private String areaConhecimentoFiltro = "";

    public AreaConhecimentoSaveDTO(AreaDoConhecimentoXmlDto xmlDto) throws
        ↪ EmptyXmlException {
        boolean hasValue = false;
        this.grandeAreaConhecimento = xmlDto.getGrandeAreaDoConhecimento();
        if (this.grandeAreaConhecimento != null) {
            this.areaConhecimentoFiltro += "grande:" +
                ↪ QueryUtils.lowerCaseStripAccents(this.grandeAreaConhecimento.getDescricao())
                ↪ + " ";
            hasValue = true;
        }

        this.nomeAreaConhecimento = xmlDto.getNomeDaAreaDoConhecimento();
        if (!StringUtils.isBlank(this.nomeAreaConhecimento)) {
            this.areaConhecimentoFiltro += "area:" +
                ↪ QueryUtils.lowerCaseStripAccents(this.nomeAreaConhecimento)
                ↪ + " ";
            hasValue = true;
        }

        this.nomeSubAreaConhecimento = xmlDto.getNomeDaSubAreaDoConhecimento();
        if (!StringUtils.isBlank(this.nomeSubAreaConhecimento)) {
            this.areaConhecimentoFiltro += "sub:" +
                ↪ QueryUtils.lowerCaseStripAccents(this.nomeSubAreaConhecimento)
                ↪ + " ";
            hasValue = true;
        }

        this.nomeEspecialidade = xmlDto.getNomeDaEspecialidade();
        if (!StringUtils.isBlank(this.nomeEspecialidade)) {
            this.areaConhecimentoFiltro += "especialidade:" +
                ↪ QueryUtils.lowerCaseStripAccents(this.nomeEspecialidade);
            hasValue = true;
        }

        if (!hasValue) {
            throw new EmptyXmlException("Area do conhecimento sem valores");
        }
    }
}
package alpc.ufsc.domain.pais;

import java.util.List;

import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;

import org.springframework.cache.annotation.Cacheable;
import org.springframework.stereotype.Service;

```

```

import org.springframework.transaction.annotation.Transactional;

import com.mysema.query.BooleanBuilder;
import com.mysema.query.jpa.impl.JPAQuery;

import alpc.ufsc.domain.pais.dto.MPaisRowDTO;
import alpc.ufsc.domain.pais.dto.PaisFilterDTO;
import alpc.ufsc.domain.pais.dto.PaisRowDTO;
import alpc.ufsc.entity.domain.pais.PaisEntity;
import alpc.ufsc.entity.domain.pais.QPaisEntity;
import alpc.ufsc.entity.util.QueryUtils;
import alpc.ufsc.util.querydsl.Select;

@Service
@Transactional
public class PaisService {

    @PersistenceContext
    private EntityManager em;

    private static final String CACHE = "paises";

    @Cacheable(value = CACHE, key = "{ #nomeFiltro }")
    public Long save(String nome, String nomeFiltro) {
        QPaisEntity qPais = QPaisEntity.paisEntity;
        PaisEntity paisEntity = new
            ↪ JPAQuery(this.em).from(qPais).where(qPais.nomeFiltro.eq(nomeFiltro)).uniqueResult(qPais);
        if (paisEntity == null) {
            paisEntity = new PaisEntity();
            paisEntity.setNome(nome);
            paisEntity.setNomeFiltro(nomeFiltro);
            this.em.persist(paisEntity);
        }
        return paisEntity.getId();
    }

    public List<PaisRowDTO> getPage(PaisFilterDTO filter) {
        QPaisEntity qPais = QPaisEntity.paisEntity;

        JPAQuery query = new JPAQuery(this.em).from(qPais);

        BooleanBuilder where = new BooleanBuilder();
        if (filter.hasQuery()) {
            where.and(qPais.nomeFiltro.like(QueryUtils.like(filter.getQuery())));
        }
        if (filter.hasPaisesIds()) {
            where.and(qPais.id.in(filter.getIds()));
        }
        if (filter.hasPageSize()) {
            query.limit(filter.getPageSize());
        }
        query.orderBy(qPais.nomeFiltro.asc());

        Select<PaisRowDTO> select = this.getSelectedPaisRowDTO();
        return query.where(where).list(select);
    }

    public PaisRowDTO load(Long paisId) {
        QPaisEntity qPais = QPaisEntity.paisEntity;

        JPAQuery query = new JPAQuery(this.em).from(qPais);

        Select<PaisRowDTO> select = this.getSelectedPaisRowDTO();
        return query.where(qPais.id.eq(paisId)).uniqueResult(select);
    }

    private Select<PaisRowDTO> getSelectedPaisRowDTO() {
        QPaisEntity qPais = QPaisEntity.paisEntity;
        Select<PaisRowDTO> select = new Select<>(PaisRowDTO.class);
        MPaisRowDTO meta = MPaisRowDTO.meta;
        select.as(qPais.id, meta.id);
        select.as(qPais.nome, meta.description);
        return select;
    }
}

package alpc.ufsc.domain.pais;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;

```

```

import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RestController;

import alpc.ufsc.domain.pais.dto.PaisFilterDTO;
import alpc.ufsc.domain.pais.dto.PaisRowDTO;

@RestController
@RequestMapping(value = "/services/pais")
public class PaisRestController {

    @Autowired
    private PaisService service;

    @RequestMapping(method = RequestMethod.GET)
    public ResponseEntity<List<PaisRowDTO>> getPage(PaisFilterDTO filter) {
        return new ResponseEntity<>(this.service.getPage(filter), HttpStatus.OK);
    }

    @RequestMapping(value = "{paisId}", method = RequestMethod.GET)
    public ResponseEntity<PaisRowDTO> load(@PathVariable Long paisId) {
        return new ResponseEntity<>(this.service.load(paisId), HttpStatus.OK);
    }
}

package alpc.ufsc.domain.pais.dto;

import lombok.Getter;
import lombok.Setter;
import br.ufsc.bridge.metafy.Metafy;

@Getter
@Setter
@Metafy
public class PaisRowDTO {

    private Long id;
    private String description;
}

package alpc.ufsc.domain.pais.dto;

import java.util.List;

import lombok.Getter;
import lombok.Setter;

import org.apache.commons.lang3.StringUtils;

@Getter
@Setter
public class PaisFilterDTO {

    private String query;
    private Integer pageSize;
    private List<Long> ids;

    public boolean hasQuery() {
        return !StringUtils.isBlank(this.query);
    }

    public boolean hasPageSize() {
        return this.pageSize != null && this.pageSize.equals(0);
    }

    public boolean hasPaisesIds() {
        return this.ids != null && !this.ids.isEmpty();
    }
}

package alpc.ufsc;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

@Controller
public class IndexController {

    private String getIndex() {
        return "index";
    }

    @RequestMapping(value = "/", method = RequestMethod.GET)

```

```
public String index() {
    return this.getIndex();
}

@RequestMapping(value = "/login/**", method = RequestMethod.GET)
public String login() {
    return this.getIndex();
}

@RequestMapping(value = "/home/**", method = RequestMethod.GET)
public String showIndex() {
    return this.getIndex();
}
}
```