UNIVERSIDADE FEDERAL DE SANTA CATARINA
CAMPUS REITOR JOÃO DAVID FERREIRA LIMA
PROGRAMA DE GRADUAÇÃO EM SISTEMAS DE INFORMAÇÃO

João Victhor da Paz Campos

POSGE: PORTAL DE OUTORGA DE SERVIÇOS DO GOVERNO ELETRÔNICO

Florianópolis, Santa Catarina – Brasil
2020

João Victhor da Paz Campos

POSGE: PORTAL DE OUTORGA DE SERVIÇOS DO GOVERNO ELETRÔNICO

Trabalho de Conclusão de Curso submetido ao Programa de Graduação em Sistemas de Informação da Universidade Federal de Santa Catarina para a obtenção do Grau de Bacharel em Sistemas de Informação.
Orientador(a): Ricardo Felipe Custódio, Dr.

Florianópolis, Santa Catarina – Brasil
2020

Catalogação na fonte pela Biblioteca Universitária da Universidade Federal de Santa Catarina.
Arquivo compilado às 01:38h do dia 8 de dezembro de 2020.

João Victhor da Paz Campos
POSGE: Portal de Outorga de Serviços do Governo Eletrônico / João Victhor da Paz Campos; Orientador(a), Ricardo Felipe Custódio, Dr. Florianópolis, Santa Catarina Brasil, 13 de novembro de 2020.
115 p.

Trabalho de Conclusão de Curso Universidade Federal de Santa Catarina, INE Departamento de Informática e Estatística, CTC Centro Tecnológico, Programa de Graduação em Sistemas de Informação.

Inclui referências

1. Assinatura Digital, 2. Procuração Eletrônica, 3. e-gov, I. Ricardo Felipe Custódio, Dr. II. Programa de Graduação em Sistemas de Informação III. POSGE: Portal de Outorga de Serviços do Governo Eletrônico

CDU 02:141:005.7

João Victhor da Paz Campos

POSGE: PORTAL DE OUTORGA DE SERVIÇOS DO GOVERNO ELETRÔNICO

Este(a) Trabalho de Conclusão de Curso foi julgado adequado(a) para obtenção do Título de Bacharel em Sistemas de Informação, e foi aprovado em sua forma final pelo Programa de Graduação em Sistemas de Informação do INE – Departamento de Informática e Estatística, CTC – Centro Tecnológico da Universidade Federal de Santa Catarina.

Florianópolis, Santa Catarina – Brasil, 13 de novembro de 2020.

Renato Cislaghi, Dr.
Coordenador(a) do Programa de Graduação em Sistemas de Informação

Banca Examinadora:

Ricardo Felipe Custódio, Dr.
Orientador(a)
Universidade Federal de Santa Catarina – UFSC

Lucas Perin, Me.
Universidade Federal de Santa Catarina – UFSC

Thiago Leucz Astrizi, Me.
Universidade Federal de Santa Catarina – UFSC

AGRADECIMENTOS

Os agradecimentos principais são direcionados a todas as pessoas que fizeram parte da minha trajetória, aos grandes professores com os quais tive a honra de aprender, ao meu orientador Prof. Custódio e meus queridos colegas do LabSEC, que me auxiliaram e compartilharam o conhecimento que me permitiu desenvolver este trabalho.

Os agradecimentos especiais são direcionados aos meus pais, que me deram o suporte necessário para que eu pudesse alcançar essa conquista, e também a Ivana Moraes, que sempre esteve ao meu lado, me apoiando em meus melhores e piores momentos.

“Todos nós morreremos. O objetivo não é viver para sempre, o objetivo é criar algo que irá.”
Chuck Palahniuk, Diário

“Só há uma coisa que temo: não ser digno do meu sofrimento.”
Fyodor Dostoevsky

RESUMO

Serviços oferecidos pelo Governo de maneira digital têm se tornado cada vez mais comuns com o passar dos anos em vários países do mundo. Nos últimos anos, o Brasil tem investido grandes esforços na iniciativa de criar uma plataforma unificada para acesso a serviços do governo eletrônico através da utilização de certificação digital. Com os avanços dessa iniciativa cada vez mais serviços aderem a plataforma e mais cidadãos são capazes de realizar serviços de maneira totalmente digital. Nesse contexto, faz-se necessária a criação de um complemento a essa plataforma para que usuários possam delegar direitos de uso para determinados serviços de e-gov a terceiros, quando julgarem necessário. O objetivo do seguinte trabalho é desenvolver uma prova de conceito de um sistema gerenciador de procurações eletrônicas que possa ser um suplemento ao portal Gov.br, auxiliando assim a iniciativa do Governo Brasileiro e aumentando as possibilidades de acesso e uso dos serviços e-gov disponíveis.

Palavras-chaves: Assinatura Digital. Procuração Eletrônica. e-gov.

ABSTRACT

Government services provided on the web are becoming common in several countries around the world as time goes by. In the last years, Brazil has invested efforts on the initiative of creating an unified channel for providing access to e-gov services through the use of digital certification. With the advancements made by this initiative, several services are applying to the platform each day and more citizens can use government services in a totally digital manner. In this context, the creation of a complement to this platform is necessary, so users can delegate use rights to third parties for certain e-gov services, when necessary. The goal of this thesis is to develop a proof of concept of a system that can manage electronic proxies that is supplementary to the Brazilian e-gov portal, assisting the Government's initiative and increasing possibilities of access and use of available e-gov services.

Keywords: Digital Signature. Electronic Proxy. e-gov.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

PDF	Portable Document Format (Formato de Documento Portável)
e-gov	Governo Eletrônico
ICP	Infraestrutura de Chaves Públicas
ITI	Instituto Nacional de Tecnologia da Informação
API	Application Programming Interface (Interface de Programação de Aplicações)
HTTP	Hypertext Transfer Protocol (Protocolo de Transferência de Hipertexto)
URL	Uniform Resource Locator (Localizador Uniforme de Recursos)
JSON	JavaScript Object Notation (Notação de Objetos JavaScript)
RFB	Receita Federal do Brasil
ASN.1	Abstract Syntax Notation One
ORM	Object–relational Mapping (Mapeamento Objeto-relacional)
PLATYPUS	Page Layout and TYPography Using Scripts (Layout de Página e Tipografia usando Scripts)
SGBD	Sistema Gerenciador de Banco de Dados
CSS	Cascading Style Sheets (Folhas de Estilo em Cascata)
MVC	Model-View-Controller (Modelo-Visão-Controlador)
SQL	Standard Query Language (Linguágem Padrão de Consultas)
PdS	Provedor de Serviços
MEC	Ministério da Educação

SUMÁRIO

1 INTRODUÇÃO

O advento da internet revolucionou o mundo das telecomunicações trazendo possibilidades nunca antes imaginadas, moldando a maneira como as relações interpessoais poderiam ser realizadas, gerando assim expectativas sobre o que poderia ser feito na área da governança. Nesse contexto globalizado, habitantes de diversas partes do mundo têm acesso à rede mundial de computadores e a utiliza todos os dias como fonte de novas informações, dando abertura não só para novas abordagens dos governos na execução de seus processos, como também na disponibilização de serviços e de conteúdo relevante para os seus cidadãos.

Um novo campo de estudos em Sistemas de Informação nasce, se aproveitando dessas oportunidades, de diferentes governos que iniciam o processo de informatização e modernização de suas iniciativas e de entusiastas vislumbrados com o cenário propício para a perpetuação da democracia nesse meio, gerando uma riquíssima área para pesquisas sobre os impactos sociais e políticos desse novo canal de comunicações.

Com o crescimento do interesse dos governos em disponibilizar serviços no âmbito digital, emerge a possibilidade de uso de procurações eletrônicas e, para tal, é necessário existir uma entidade responsável por intermediar o processo de concessão do direito de uso dos serviços, bem como a segurança dos dados envolvidos no procedimento realizado. Um desses cuidados se refere à autenticação de ambas as partes, concedente e receptor do direito, para a prevenção de fraudes por falsidade ideológica e acesso indevido de hackers e agentes inteligentes. Para a garantia dessa propriedade uma das possibilidades é a utilização de certificação digital, ferramenta capaz de identificar indivíduos no âmbito digital.

O consumo da certificação digital é viabilizado por meio de uma cadeia hierárquica denominada Infraestrutura de Chaves Públicas, responsável por gerir o ciclo de vida desses certificados, realizando sua emissão e posterior extinção, enquanto garante a validade destes através de protocolos de funcionamento pré-estabelecidos entre os softwares verificadores e a Infraestrutura em questão. No Brasil a estrutura é gerida pelo ITI, Instituto Brasileiro de Tecnologia da Informação, e é denominada ICP-Brasil[1].

Um dos objetivos de uso de um certificado digital é a geração de assinaturas digitais para documentos eletrônicos, artefato este que possibilita realizar a identificação do assinante, assim como uma assinatura a punho, constituindo uma ferramenta importantíssima para a viabilidade do presente trabalho.

Com o aumento da quantidade de serviços governamentais disponíveis no portal de serviços eletrônicos do Governo brasileiro, e com a crescente popularização do uso de certificação digital no Brasil, emerge a necessidade de existir uma plataforma que permita aos cidadãos brasileiros que habilitem terceiros a realizarem atividades de serviços de e-gov em seus nomes e que gerencie a concessão dessas permissões.

---

[1] ICP-Brasil. Instituto Nacional de Tecnologia da Informação, 2017. Disponível em: <https://www.iti.gov.br/icp-brasil>. Acesso em: 14 de abr. de 2020

1.1 OBJETIVOS

1.1.1 Objetivo geral

Desenvolver uma aplicação web em Python que gerencie o ciclo de vida de procurações eletrônicas, possibilitando a emissão, visualização e a extinção de outorgas que garantem a usuários a habilidade de conceder permissões para o uso de serviços específicos do governo eletrônico por terceiros através do uso de certificação digital.

1.1.2 Objetivos específicos

- Utilizar certificação digital para a emissão de procurações eletrônicas.
- Desenvolver uma aplicação web em Python para uso da ferramenta.
- Propor uma alternativa de plataforma unificada para emissão de procurações eletrônicas entre os diferentes serviços oferecidos pelo Governo Brasileiro em modalidade digital.

1.2 MOTIVAÇÃO

Em 2017 se iniciou um processo de Transformação Digital por parte do governo brasileiro a fim de fomentar o uso dos serviços de e-gov pela internet. Através de um Censo de Serviços foram identificados cerca de 2,8 mil serviços disponibilizados pelo governo que davam acesso às atividades através de canais independentes entre si. Nessa iniciativa definiu-se um conceito unificado de serviço público e todos os identificados no Censo foram reunidos no Portal de Serviços do Governo Federal. Nos dois anos subsequentes o foco foi centrado em encontrar métodos para incentivar cada vez mais serviços a aderirem ao meio e, em julho de 2019, mais da metade de todos os serviços ofertados pelas entidades governamentais estavam disponíveis de forma totalmente digital.[2]

Em paralelo ao processo de digitalização dos serviços governamentais brasileiros, o número de emissões de certificados digitais ICP-Brasil aumenta, batendo recorde de emissões em janeiro de 2020, totalizando 8,9 milhões de certificados digitais ativos no país. [3] No entanto, apesar do número expressivo, representa menos de 5% dos 210 milhões de brasileiros estimados pelo

---

[2] 5 passos para a Transformação Digital. Governo do Brasil, 2019 Disponível em: <https://www.gov.br/governodigital/pt-br/transformacao-digital/o-que-e/passos-para-transformacao-digital>. Acesso em: 14 de fev. de 2020

[3] ITI: Janeiro bate recorde de emissões ICP-Brasil em relação a-os anos anteriores. Colégio Notarial do Brasil, 2020 Disponível em: <https://www.cnbsp.org.br/index.php?pG=X19leGliZV9ub3RpY2lhcw&in=MTkyODU&filtro=&Data=>. A-cesso em: 05 de dez. de 2020

IBGE em julho de 2019. [4] Com isso, é importante implementar uma alternativa para uso do POSGE por usuários que não sejam detentores de certificado digital.

No momento é possível emitir uma procuração eletrônica para os serviços disponíveis na Lista de Serviços da Secretaria Especial da Receita Federal do Brasil através do Atendimento Virtual (e-CAC), desde que outorgante e outorgado sejam detentores de certificado digital. [5] No entanto esse serviço restringe a emissão das procurações aos serviços oferecidos pela RFB, se opondo ao processo de Transformação Digital no que tange a unificação das plataformas provedoras de serviços de e-gov.

1.3 METODOLOGIA

O trabalho foi elaborado seguindo a seguinte metodologia:

- Estudo sobre os frameworks para a web e bibliotecas de manipulação de PDF disponíveis para a linguagem de programação Python;
- Investigação sobre as normas e boas práticas do processo de emissão e gerência de outorgas digitais;
- Compreensão dos processos relacionados a infraestrutura de chave públicas para a garantia do uso correto de uma assinatura digital em um PDF;
- Implementação do protótipo;
- Análise dos resultados alcançados e proposições para trabalhos futuros.

---

[4] Estimativa da população do Brasil passa de 210 milhões, diz IBGE. Agência Brasil, 2019. Disponível em: <https://agenciabrasil.ebc.com.br/economia/noticia/2019-08/estimativa-da-populacao-do-brasil-passa-de-210-milhoes-diz-ibge>. Acesso em: 05 de dez. de 2020

[5] BRASIL. Instrução Normativa RFB nº 1751, de 16 de outubro de 2017. dispõe sobre o acesso do contribuinte aos serviços disponíveis na Lista de Serviços da Secretaria Especial da Receita Federal do Brasil (RFB) mediante outorga de poderes a pessoa física ou jurídica detentora de certificado digital. Diário Oficial da União: seção 1, página 85, 18 jan. 2017

2 FUNDAMENTAÇÃO TEÓRICA

O foco deste capítulo é abordar os conceitos básicos relacionados ao desenvolvimento do trabalho em questão. Tratando-se de conceitos mais teóricos os quais são integrantes das áreas correlatas ao tema, Governança e Direito, e também de aspectos mais específicos de Sistemas de Informação e suas tecnologias utilizadas na concepção da prova de conceito desenvolvida.

2.1 PDF

O *Portable Document Format* é um formato de arquivos de computador criado no início da década de 90, pela Adobe Systems, com o objetivo de se tornar um formato padronizado para compartilhamento de documentos digitais sem que os usuários do produto tenham de ter o mesmo sistema operacional ou as mesmas dependências a fim de poder acessar seu conteúdo, as contendo em seu interior. De 1993 a 2006 o formato passou por constantes atualizações até o ano de 2008 quando, devido ao constante aumento no índice de aderência por parte dos usuários, se tornou um padrão aberto, deixando de ser propriedade da Adobe Systems.

Todo o conteúdo desta seção foi escrito baseando-se na norma técnica *Document Management – Portable Document Format – Part 1: PDF 1.7*(ISO, 2008), onde se encontram todas as informações disponíveis a respeito da formalização do formato. Todas as figuras de exemplo presentes nessa seção se baseiam em um documento exemplo, gerado pelo protótipo desenvolvido, que pode ser visualizado no Apêndice A.

2.1.1 Estrutura

Figura 1 – Estrutura de um arquivo no formato PDF

2.1.1.1 Header

A primeira linha de um arquivo no formato PDF deve ser um *header* (cabeçalho), o qual consiste nos 5 caracteres "%PDF-" precedidos do número da versão utilizada na concepção do arquivo. A partir da variante 1.4 caso um número de versão esteja presente no catálogo do dicionário do *Trailer* do documento, esta terá preferência em relação a presente no *header*. É possível que um documento com funcionalidades mais recentes seja interpretado por um leitor que só dê suporte a uma versão mais antiga do formato. O projeto é desenvolvido de tal maneira que novas características introduzidas possam ser ignoradas por softwares que não as compreendam.

Figura 2 – *Header* do documento exemplo

```
%PDF-1.4
%“Œ‹ž ReportLab Generated PDF document http://www.reportlab.com
```

Como visto na Figura 2 também é possível realizar comentários no *header* utilizando o caractere **"%"**. O *ReportLab*, um dos frameworks utilizados na implementação do protótipo, insere um comentário no *header* no momento da geração do PDF.

2.1.1.2 Body

O *body* (corpo) de um documento em PDF consiste em uma série *Indirect Objects* (objetos indiretos) que representam o conteúdo do documento, a parte visível para o usuário. O formato suporta vários tipos comuns de objetos como *Booleans*, números inteiros e reais, *Strings*, *Arrays*, *Dictionaries*, *Null*, e também específicos do escopo como *Object Streams*, que por sua vez são compostos por uma série de *Indirect objects*. O conteúdo do *body* depois é referenciado na *Cross-Reference Table*.

2.1.1.3 Cross-Reference Table

A *Cross-Reference Table* (Tabela de Referência Cruzada) ou *x-ref table* contém as informações necessárias para permitir o acesso aleatório a *Indirect Objects* específicos sem que seja necessário processar o arquivo inteiro.

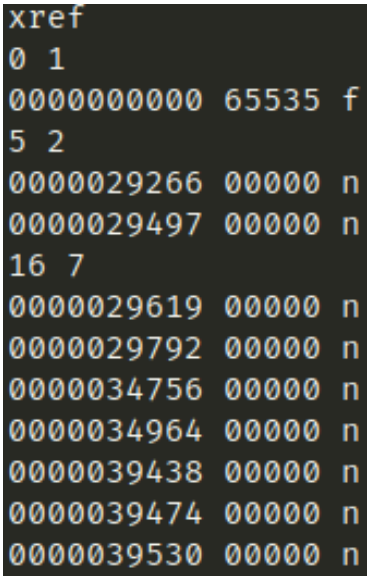
A tabela consiste em uma ou mais seções. Inicialmente a tabela inteira se trata de uma só seção e, na medida que o documento for atualizado, outras mais podem ser adicionadas a tabela. Cada seção deve ser iniciada com a palavra-chave **xref** e pode conter uma ou mais subseções.

A primeira linha de cada subseção deve conter dois números, separados por espaço, sendo esses o número do primeiro objeto dessa subseção e a quantidade de entradas na mesma. Após a primeira linha, cada objeto terá sua entrada na tabela representada por 20 *bytes* em cada linha.

Os primeiros 10 *bytes* são o *offset* do objeto em relação ao início do documento. Após o espaço, os próximos 5 *bytes* representam o *generation number*, que é um número que inicia

em 0 e é incrementado a cada revisão realizada naquele objeto, de acordo com cada atualização realizada no documento. Depois de mais um espaço, o último *byte* antes do fim da linha é uma *flag* que informa se esse objeto está sendo usado no documento ou não, sendo as duas possibilidades representadas por "n", em uso, ou "f", *free* (livre). O *generation number* e a *flag* são persistidos para possibilitar aos documentos revisão e posterior reversão de modificações sem que seja necessário utilizar um sistema de revisão externo.

Figura 3 – *x-ref table* do documento exemplo



Na Figura 3 é possível visualizar a última seção da tabela do documento exemplo — denotada pela palavra-chave **xref** na primeira linha — sobre a qual se observa que:

- Contém um total de três subseções e dez objetos;
- Armazena o objeto inicial na primeira subseção, identificado pela sua *flag* **f** e seu *generation number* **65535** (o valor máximo deste atributo). Esse objeto é inserido em cada seção da tabela;
- Conta com dois objetos em uso na segunda subseção e sete, também em uso, na terceira;
- Nenhum dos objetos listados foi atualizado ou desativado, logo seus *generation numbers* são iguais a **00000** e sua *flag* a **n**, exceto pelo objeto inicial.

2.1.1.4 Trailer

O *trailer*, localizado nas últimas linhas presentes em um arquivo no formato PDF como visto na Figura 1, armazena ponteiros para a Cross-Reference Table — possibilitando que *softwares* leitores de PDFs rapidamente achem sua posição no arquivo ao iniciar a leitura pelo fim — e também para certos objetos especiais que ficam armazenados em seu dicionário.

A última linha contém um marcador *end-of-file* (fim de arquivo) e as duas linhas anteriores apresentam, respectivamente, a palavra-chave**startxref** e o *offset* em *bytes* do início do arquivo até a palavra-chave **xref** da última seção da *x-ref table* presente no arquivo.

Nas linhas acima da palavra-chave**startxref** localiza-se o *trailer dictionary*. Iniciado pela palavra-chave **trailer**, trata-se de uma sucessão de pares "chave-valor" entre aspas em linha, representadas pelos símbolos "«" e "»". O Quadro 1 lista as chaves possíveis e os tipos de valores esperados.

Quadro 1 – Chaves possíveis para o *trailer dictionary*

Chave	**Tipo**	**Requerido**	**Valor**
Size	Inteiro	Sim	O número total de objetos na Cross-Reference Table, combinando as seções iniciais com as atualizadas.
Prev	Inteiro	Não	O *offset* em *bytes* do início do documento até a seção anterior da *xref-table*.
Root	Dictionary	Sim	O catálogo do documento PDF. Armazena uma série de ponteiros para objetos referenciados no arquivo — como figuras, tabelas e nomes de seções — e também informações sobre a exibição do documento — por exemplo a maneira como deve ser exibido, definição de qual página será exibida quando o documento for aberto pela primeira vez, entre outras.
Encrypt	Dictionary	Não	Caso esteja cifrado, informações diversas a cerca da encriptação do documento como o algoritmo utilizado, tamanho das chaves e especificação do *security handler* (responsável por implementar as funções criptográficas utilizadas no documento).
Info	Dictionary	Não	Metadados do documento como título, autor, tema, palavras-chave, etc.
ID	Array	Não	Um identificador para o arquivo formado por dois *byte-strings*. Obrigatório caso *Encrypt* esteja presente neste dicionário, opcional caso contrário.

Fonte: (ISO, 2008)

Pelas variáveis existentes no dicionário do *trailer* presente na Figura 4 é possível observar que o documento exemplo:

- Contém 23 objetos, como indicado em *size*, 10 deles na seção da *Cross-Reference Table* que pode ser visualizada na Figura 3 e o restante em suas seções anteriores;
- Sua *x-ref table* possui, ao menos, duas seções já que a chave *prev* está presente e aponta para um endereço ou seja, o documento foi atualizado ao menos uma vez. (No caso dos documentos emitidos pelo POSGE o PDF é gerado e posteriormente assinado duas vezes, resultando em três seções da tabela);

Figura 4 – *Trailer* do documento exemplo

```
trailer
<<
/Size 23
/Root 6 0 R
/Prev 28905
/ID [ <e34ec505713fe38f80350a2b6afa3e7d> <2be877da5cee8e3c1e1e67d7cf6ffe15> ]
/Info 7 0 R
>>
startxref
39805
%%EOF
```

- O objeto *root* é o sexto *indirect object* registrado na *Cross-Reference Table* do documento e o *info* o sétimo.
- Apesar de a chave *encrypt* não estar no dicionário, a *ID* está. Isso se deve ao fato de este documento ter sido assinado digitalmente;
- A última seção da *Cross-Reference Table* se inicia no *offset* **39205**, como indicado na linha abaixo da palavra-chave **startxref**.

2.1.2 Assinatura Digital

Tendo seu uso formalizado a partir da versão 1.3 do formato PDF, a assinatura digital pode ser usada para autenticar a identidade de uma pessoa e o conteúdo de um documento. A assinatura armazena informações sobre quem realizou a assinatura e o estado do documento no momento em que a assinatura ocorreu.

Existem múltiplas maneiras de criar uma assinatura digital, seja utilizando funções matemáticas, como na **Criptografia de Chaves Públicas**, ou de forma biométrica através de impressões digitais ou reconhecimento de retina. Sua implementação é provida por um módulo denominado *Signature Handler* (Manipulador de Assinatura). A implementação desse módulo pode ser feita por terceiros, provendo assim a possibilidade de se desenvolver uma solução personalizada de acordo com as necessidades de cada domínio, e tem que seguir as especificações da ISO 32000.

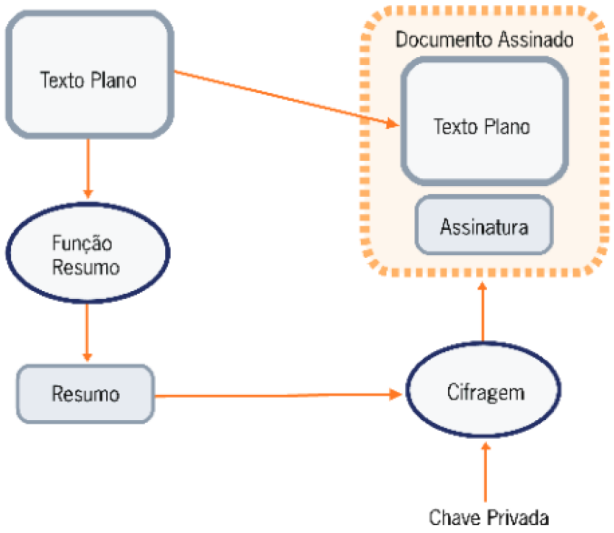
O padrão definido pela ISO 32000 dá suporte a duas atividades relacionadas ao uso de assinatura digital; Inclusão de assinatura digital no documento e posterior verificação da validade da assinatura (Especificamente se o campo está de acordo com o especificado na norma técnica e se a assinatura foi emitida por uma entidade confiável). No entanto a verificação das informações de revocação do certificado e validade da cadeia de certificação no momento da assinatura é de responsabilidade do *software* utilizado para assinar.

Toda a informação relacionada a assinatura deve estar presente no *signature dictionary* (Dicionário de assinatura). *Signature Handlers* podem usar ou omitir determinadas entradas padronizadas da tabela — as quais podem ser visualizadas no Quadro 2 — e também incluir

novas, de acordo com a personalização da implementação, que devem ser prefixadas com o nome do *Signature Handler* seguido de um ponto-final.

Assinaturas digitais são criadas calculando-se um resumo dos dados do documento, utilizando uma função matemática denominada função de *hash* criptográfico, posteriormente cifrando esse resumo utilizando uma senha presente no certificado digital do assinante, designada chave privada (essa chave é integrante de um par de chaves presentes no certificado do usuário, sendo a outra nomeada chave pública. O que uma é capaz de encriptar, a outra é capaz de decriptar). Ao fim da operação o resultado é concatenado ao documento junto com uma série de metadados que serão utilizados no processo de verificação da validade da assinatura pelo *software* leitor de PDF, série essa que inclui o certificado do usuário (sem sua chave privada). O processo pode ser visualizado na Figura 5.

Figura 5 – Fluxo de geração de uma assinatura digital



Fonte: (CARLOS *et al.*, 2010)

Para realizar a verificação da assinatura, o leitor de PDF a decripta utilizando a chave pública do assinante presente no certificado, tendo acesso assim ao resumo criptográfico gerado no momento da assinatura. O leitor de PDF gera um resumo criptográfico do documento recebido e compara o resultado com o resumo presente na assinatura; Se forem iguais, a assinatura é valida, caso contrário isso indica que o documento foi modificado e a assinatura é considerada inválida.

Quadro 2 – Chaves possíveis para o *signature dictionary*.

Chave	**Tipo**	**Requerido**	**Valor**
Type	String	Não	O tipo de objeto que este dicionário descreve. Neste caso **Sig** de *Signature* (Assinatura).

Fonte:(ISO, 2008)

Filter	String	Sim	O nome do *Signature Handler* preferido a ser usado no momento da validação da assinatura. Caso a entrada **Prob_Build** não esteja presente na tabela, indica também o *Signature Handler* usado no momento da criação da assinatura. Caso o *Filter* listado não esteja disponível, o leitor de PDF usará uma alternativa disponível que dê suporte ao *SubFilter* listado na tabela.
Subfilter	String	Não	Um nome que descreve a codificação dos valores da chave e assinatura presentes na entrada *contents* da tabela. Os valores padrão válidos são: **adbe.x509.rsa_sha1**, **adbe.pkcs7.detached** e **adbe.pkcs7.sha1**. Valores personalizados podem ser utilizados e devem ser prefixados por uma identificação do desenvolvedor seguida de ponto-final.
Contents	Byte string	Sim	O valor da assinatura propriamente dita. Caso a entrada *ByteRange* esteja presente na tabela, o resumo criptográfico, gerado no momento da assinatura, computou o conteúdo existente dentro dos limites definidos nesse campo, caso contrário, o documento inteiro foi considerado. O espaço para o conteúdo de *Contents* deve ser alocado antes da geração do resumo.
Cert	Array	Não	Um *array* de *byte strings* representando a cadeia de certificação utilizada ao assinar e verificar assinaturas que utilizam criptografia de chaves públicas. O primeiro certificado deve ser aquele que assinou o documento, o qual será usado para verificar a assinatura, e os demais certificados para verificar a autenticidade do certificado usado para assinar. Obrigatório quando *Subfilter* é**adbe.x509.rsa_sha1**.
ByteRange	Array	Não	Um *array* com pares de números inteiros, representando o *offset* em *bytes* para o início do intervalo e o comprimento do intervalo em *bytes*, que descrevem o exato intervalo em *bytes* que deve ser considerado no cálculo do resumo criptográfico.

Fonte:(ISO, 2008)

Changes	Array	Não	Um *array* com três números inteiros que especificam as modificações que ocorreram no documento entre a última assinatura e a assinatura atual. Cada número representa respectivamente: a quantidade de páginas modificadas, o número de campos alterados e o número de campos preenchidos.
Name	String	Não	O nome da pessoa ou autoridade que assinou o documento. Esse valor só deve ser utilizado quando for impossível extrair essa informação da assinatura.
M	Date	Não	A data e hora da assinatura. Dependendo do *Signature Handler* pode ser o horário do computador no momento da assinatura sem qualquer verificação ou um gerado em um servidor confiável. Esse valor só deve ser usado se for impossível extrair essa informação da assinatura.
Location	String	Não	O nome do *Host* ou a localização em que a assinatura ocorreu.
Reason	String	Não	A razão da assinatura.
ContactInfo	String	Não	Informação disponibilizada pelo assinante para que um recebedor do documento possa entrar em contato para fins de validação, por exemplo um número de telefone.
R	Inteiro	Não	A versão do *Signature Handler* usado para criar a assinatura. Este campo pode estar ausente caso essa informação esteja disponível no *dictionary* da entrada *Prop_Build* da tabela.
Prop_Build	Dictionary	Não	Um *dictionary* que pode ser usado pelo *Signature Handler* para armazenar informações que forem pertinentes sobre o estado do computador no momento da assinatura como, por exemplo, o nome e versão do *Signature Handler*, sistema operacional utilizado, etc;

Fonte:(ISO, 2008)

Analisando a Figura 6 é possível observar que:

- O *signature dictionary* é o 11 ° objeto presente no Body do documento. É possível o identificar pelo valor **Sig** atribuído a chave *Type*;
- O *Filter* utilizado foi o **Adobe.PPKLite**. Esse é o *security handler* padrão;

Figura 6 – Objeto no documento exemplo que contém o *signature dictionary*

```
11 0 obj
<<
/Type /Sig
/Filter /Adobe.PPKLite
/SubFilter /adbe.pkcs7.detached
/Name (POSGE\100POSGE\056com)
/Location (Brasil)
/Reason (Procuracao\040Eletronica\040do\040Portal\040de\040Outorga\040de\040Servicos\040do\040Governo\040Eletronico\040Brasileiro)
/M (20201030080945\05503\04700\047)
/Contents <3082088706092a864886f70d010702a0820878308208740201013 10f300d06096086480165030402010500300b06092a864886f70d010701a082056
< ... >
c1cd5c17c0cf267821034674ee2d81c5d8ff722e4bbd60cd00d89cc5f944481e3293193b8a1049956c24f12a29896e9621798e45ce1201627c9b70ad763db53d9>
/ByteRange [0 3495 7871 21393]
>>
endobj
```

- O SubFilter utilizado foi o **adbe.pkcs7.detached**. Isso indica que a assinatura está no formato **PKCS#7**, informação importante que indica como interpretar o conteúdo da assinatura, e **detached** (desanexado) informa que o campo *Contents* contém uma assinatura e o certificado do assinante com sua chave pública, sem incluir o conteúdo assinado em seu interior, ao contrário do *attached* (anexado);

- A entidade responsável por assinar se identifica como **POSGE@POSGE.com** de acordo com a entrada *Name*. (“\100” e “\056” são caracteres ASCII codificados em base octal e representam, respectivamente, arrouba e ponto final.);

- A data da assinatura está no padrão “AAAAMMDDhhmmss” seguido do fuso horário “-03’00” em *M*.

- O campo *Contents*, o qual contém 4,374 bytes no documento exemplo e na figura foi reduzido a fim de melhorar a visualização, é registrado no *dictionary* representado em base hexadecimal armazenando a assinatura e alguns metadados utilizando uma linguagem de formalização denominada *ASN.1*. Um utilitário *online*[1] foi utilizado para decodificar essa entrada e auxiliar na compreensão desse campo. O resultado poder ser visto no Apêndice B.

2.2 WEB API

Application Programming Interface (Interface de Programação de Aplicações) trata-se de um conjunto de padrões de acesso implementados para que uma aplicação possa se comunicar com outras, provendo a possibilidade de integração entre diferentes sistemas que não necessariamente rodam no mesmo sistema operacional ou são desenvolvidos usando as mesmas linguagens de programação (KOPECKY; FREMANTLE; BOAKES, 2014).

Web APIs são implementadas com o objetivo de conectar diferentes sistemas pertencentes a uma mesma infraestrutura ou para disponibilizar e realizar o controle do compartilhamento das informações disponíveis com terceiros. Para isso, *Web APIs* utilizam o protocolo *HTTP*

[1] MICHAEL HOLSTROM. ASN.1 decoder, 2020. Disponível em: <https://holtstrom.com/michael/tools/asn1decoder.php>. Acesso em: 30 de out. 2020

para realizar a troca de mensagens entre os sistemas, e formatos como o *XML* e o *JSON* para representar as respostas às requisições, possibilitando assim que as informações enviadas possam ser interpretadas e processadas conforme necessário. O servidor que contém os dados a serem compartilhados deve disponibilizar uma *URL* de acesso.

Para que a troca de informações ocorra, o sistema cliente deve enviar uma requisição *HTTP* para o servidor, especificando em seu cabeçalho quaisquer parâmetros definidos pela função presente na interface do servidor. Em resposta, o sistema realiza o processamento da requisição e retorna, em um formato padronizado, as informações requisitadas.

2.3 E-GOV

O termo e-gov, criado na década de noventa, serviu de rótulo para uma nova área, dentro do campo de Sistemas de Informação, que tem como objeto de estudo a "governança digital", tratando de políticas, estratégias e implementação de processos governamentais no meio digital (GRÖNLUND; HORAN, 2004).

Apesar de nomeada na década de 1990, os primeiros estudos dentro dessa área datam desde a década de 1970, onde as pesquisas se concentravam em meios de utilizar a Tecnlogia da Informação dentro do governo (KRAEMER, 1978), diferente da tendência atual de concentrar seus estudos no uso externo, como o de serviços governamentais por parte dos cidadãos de um país.

As publicações realizadas tratam de projetos distintos de e-gov que ocorrem em diferentes países do mundo, analisando dados geográficos e sociais, identificando denominadores comuns e avaliando aspectos governamentais desenvolvidos através das políticas de viés digital implantadas. Segundo Grönlund e Horan (2004), as estratégias nacionais de e-gov implementadas têm, similarmente, três objetivos específicos:

- Mais eficiência governamental;
- Melhores formas de acesso a serviços para os cidadãos, e;
- Melhorias nos processos democráticos.

A fim de complementar a iniciativa federal do portal *gov.br*, o presente trabalho focará seus esforços no segundo objetivo citado.

2.4 PROCURAÇÃO ELETRÔNICA

Segundo (GONÇALVES, 2007), a outorga é um instrumento utilizado por uma pessoa, denominada mandante nesse contexto, a fim de conceder plenos poderes a um terceiro, nomeado procurador, para executar ações e administrar processos em seu nome.

Existem dois tipos de procuração; a particular, que pode ser feita com somente o envolvimento das duas partes, mandante e procurador, ou podendo o reconhecimento de firma

ser exigido por uma delas, e a pública, que é feita e registrada por um tabelionato de notas. Procurações podem ser outorgadas e recebidas por quaisquer pessoas desde que sejam maiores de idade e estejam em conformidade com as leis da justiça eleitoral.

Para que uma procuração seja considerada válida, uma série de informações devem estar inclusas em seu conteúdo:

- Nome e documentos de identificação de ambas as partes, outorgante e outorgado;
- Finalidade e data da procuração;
- Descrição detalhada dos poderes concedidos e sua extensão;
- Designação do lugar da concessão de poderes;
- Assinatura do mandante com reconhecimento em firma, caso requisitado. A assinatura pode ser realizada digitalmente de acordo com o §1º do artigo 105 do Código de Processo Civil de 16 de Março de 2015 (BRASIL, 2017).

O documento exemplo, presente no Apêndice A, apresenta uma proposta de texto a ser usado nas procurações emitidas no POSGE que engloba todos os requisitos listados acima, e pode ser ampliada de acordo com os dados disponíveis no portal e as especifidades de cada aplicação.

Por si, não existe uma exigência a cerca do prazo de extinção de uma procuração, podendo uma data para tal ser especificada caso seja exigido por uma lei específica ou se for da vontade do mandante existir, sendo a outorga válida por tempo indeterminado caso contrário. É permitido o cancelamento de um mandato a qualquer momento por ambas as partes através de uma requisição, quando não quiserem que os direitos sejam mais concedidos, e também em caso de óbito, cabendo outros processos jurídicos caso o receptor do direito tenha interesse em resolver ações em nome do falecido.

No Brasil um exemplo prático do uso de procurações eletrônicas é na concessão de direitos sobre serviços disponibilizados no Centro de Atentimento ao Contribuinte (e-CAC) da Receita Federal Brasileira, homologada pela Instrução Normativa RFB nº 1.751 de 16 de Outubro de 2017 (BRASIL, 2015).

3 DESENVOLVIMENTO

Após o estudo dos conceitos teóricos necessários para a realização do trabalho, iniciou-se uma pesquisa a cerca das ferramentas que poderiam ser utilizadas para o desenvolvimento da prova de conceito em questão. Deixando, a princípio, a escolha da linguagem de programação em aberto, a pesquisa se concentrou em frameworks open-source para a manipulação de documentos PDF com suporte ao uso de assinatura digital. As opções mais acessíveis e com menor curva de aprendizado encontradas, devido a grande complexidade da manipulação de arquivos em PDF, foram bibliotecas implementadas utilizando a linguaguem de programação *Python*.

Outra característica importante considerada na escolha das tecnologias usadas foi a implementação do acesso via web. O objetivo é que o sistema possa ser integrado ao portal gov.br para ser usado em conjunto com os demais serviços de e-gov disponibilizados pelo governo através de uma Web API. Também é imprescindível que o protótipo tenha uma forma de autenticação, a fim de identificar os usuários, outorgantes e outorgados, e para isso escolheu-se um Sistema Gerenciador de Banco de Dados.

3.1 TECNOLOGIAS USADAS

Duas opções de *frameworks* para *Python* dentre os mais populares foram consideradas para realizar a implementação das funcionalidades *web* do protótipo, o *Django* e o *Flask*. Em termos gerais, ambas as bibliotecas contam com diversas funcionalidades em comum; roteamento de *URLs*, autenticação e estabelecimento de sessões, integração com bancos de dados, mecanismo de *templating*, etc. No entanto, o *Django* conta com uma série de funcionalidades extras como interfaces de administração, ferramentas integradas para *bootstrapping*, suporte a múltiplas aplicações e sistema ORM para integração com bancos de dados, tendo seu foco em aplicações complexas(DJANGO, 2019). Em contrapartida, o *Flask* conta com menos funcionalidades, focando em aplicações mais simples e se concentrando em somente prover as funcionalidades necessárias para possibilitar o acesso a aplicações *Python* na web(PALLETS, 2019). Devido a complexidade extra envolvida no uso do *Django* e também ao fato de suas funcionalidades extras não serem necessárias para o desenvolvimento do protótipo, escolheu-se o *Flask*.

No que tange a manipulação de arquivos em PDF, duas bibliotecas foram utilizadas, uma para criar os documentos e uma segunda para assiná-los. O *ReportLab* é uma *engine* de criação de PDFs bem documentada que tem uma versão gratuita e *open-source*, a qual foi utilizada na implementação, que dá suporte a desenhos vetoriais, geração e reutilização de gráficos e ao *PLATYPUS*, uma *engine* que permite criar documentos a partir de elementos textuais como títulos, parágrafos e tabelas (REPORTLAB, 2019). Já o *Endesive* é uma biblioteca *open-source* com foco em assinatura e verificação de assinaturas digitais em PDFs , XMLs e Emails. Dentre as funcionalidades oferecidas para esses formatos, a biblioteca suporta a implementação do *security handler* **Adobe.PPKLite/adbe.pkcs7.detached**(MAKAREWICZ, 2019), sendo este o utilizado no protótipo como visto no exemplo da Subseção 2.1.2.

O objetivo do **POSGE** é ser integrado ao gov.br e, nesse contexto, os cidadãos que forem detentores de certificado digital poderão o utilizar para assinar suas procurações no portal; aqueles que não forem também poderão utilizar a plataforma, o sistema emitirá um novo certificado a cada *login* do usuário no sistema o qual será usado durante aquela sessão e será descartado ao fim da mesma. A nível de prova de conceito se utiliza a biblioteca *OpenSSL*, efetuando chamadas via *shell linux*, para emitir todos certificados dos usuários que serão usados para assinar as procurações(OPENSSL, 2019).

Os certificados de longa duração — que representam, na prova de conceito, os certificados já vinculados ao cadastro do cidadão no gov.br — são emitidos no momento da instalação do sistema, junto ao cadastro dos usuários iniciais. Os usuários cadastrados posteriormente — representando aqueles que não são detentores de certificado digital — têm seus certificados emitidos a cada autenticação no sistema. Também é importante frisar que as senhas dos usuários cadastrados, assim como as chaves de API dos provedores de seviços, não são armazenadas em claro no banco de dados. Para isso utilizou-se a biblioteca *Passlib* a fim de gerar resumos criptográficos das senhas utilizando o algoritmo *sha256* e *salt*, para evitar, por exemplo, ataques de pré-imagem, os quais são armazenados na base.

As informações dos usuários, serviços e procurações estão armazenadas em uma instância do SGBD *MariaDB* e a biblioteca CSS utilizada foi o *Materialize*. Ambos softwares de código-aberto, que realizam o lançamento frequente de versões estáveis, desde 2010 e 2014 respectivamente, e que já foram usados em projetos anteriores. Nos momentos em que se fez necessário realizar o processamento em uma página dinâmica, foi utilizada a biblioteca *jQuery* do *JavaScript*.

3.2 PLANEJAMENTO E ESTRUTURA DO PROTÓTIPO

Uma análise a cerca das funcionalidades que deveriam ser implementadas e sua relação com os atores envolvidos com o uso do software foi realizada, gerando o diagrama de casos de uso presente na Figura 7. Os atores que irão interagir com o ambiente são os cidadãos, usuários do gov.br, que irão emitir, extinguir e acessar suas procurações, e os provedores de serviços do portal que irão acessar o POSGE via Web API para verificar a permissão do uso de certos serviços por determinados usuários em nome de outrem. A prova de conceito utilizou exemplos de provedores e serviços disponíveis no portal gov.br que já utilizam certificação digital, os quais foram populados na base através do *script* de criação do banco de dados, no entanto, em um caso de uso real, o sistema deveria sempre se atualizar com relação aos serviços e provedores disponíveis através de uma conexão com o portal.

Com a compreensão dos atores, suas ações sobre o ambiente e artefatos provenientes de seu uso, um diagrama entidade-relacional foi criado para entender e ilustrar os relacionamentos que deveriam existir entre as entidades provenientes do domínio do problema, o qual pode ser visualizado na Figura 8. Cada entidade e relação identificada tem sua existência representada no banco de dados através de entradas na tabela e ligações por chave-estrangeira.

Figura 7 – Diagrama de casos de uso do protótipo

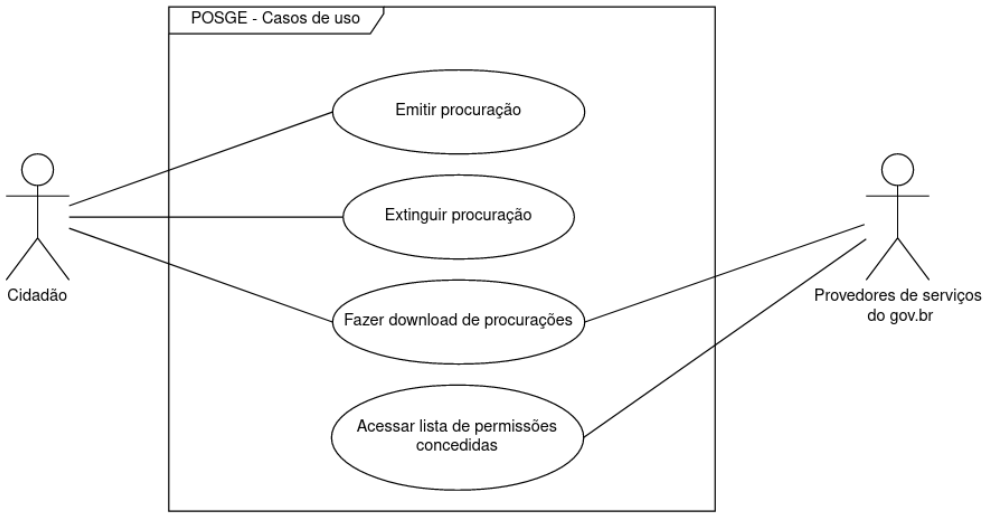
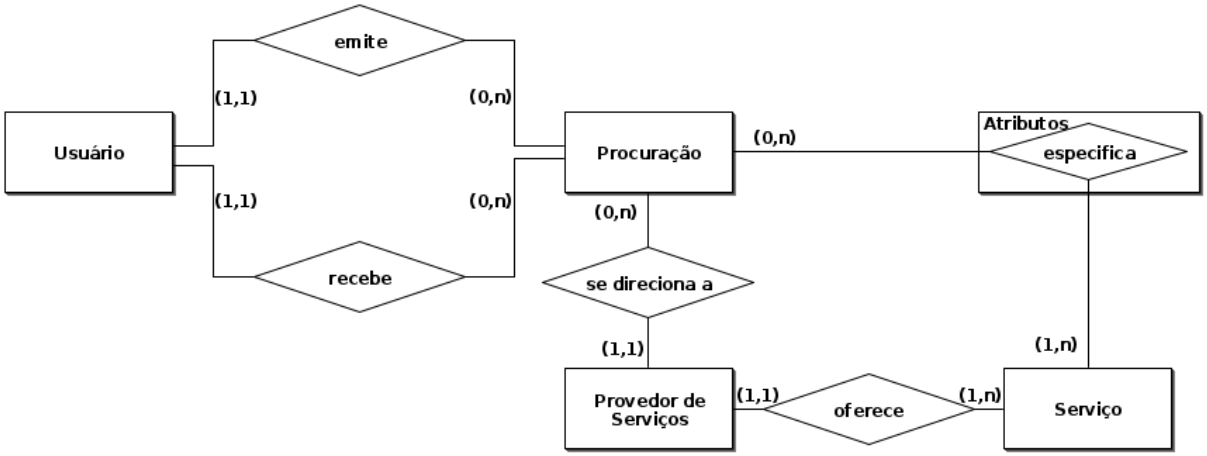


Figura 8 – Diagrama ER do protótipo

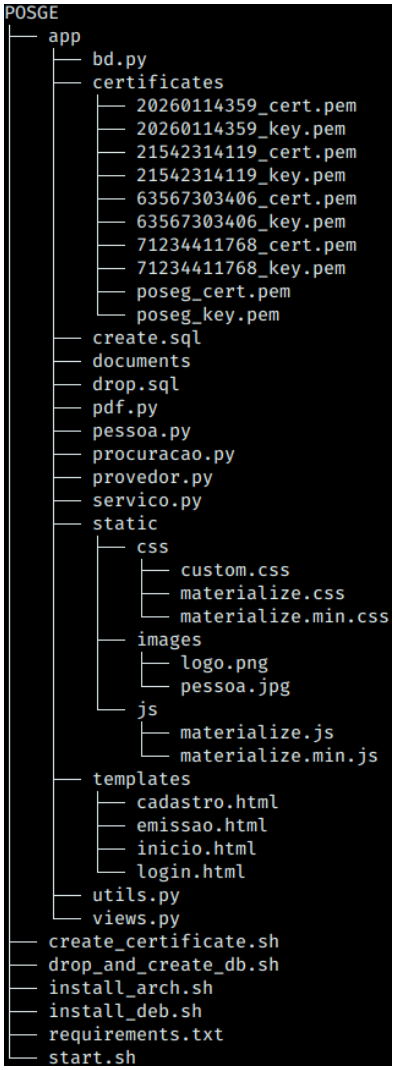


A abordagem aplicada na implementação usou conceitos do paradigma orientado a objetos — sendo cada entidade do banco de dados representada por uma classe, e cada tupla por um objeto em *python*, além de ser modularizado de acordo com o padrão de arquitetura MVC — e se baseou na documentação do framework *Flask*. O protótipo utiliza um *Virtual Environment* (Ambiente Virtual) em conjunto com o gerenciador de pacotes *python-pip* para, respectivamente, isolar e gerenciar as dependências do projeto.

Na Figura 9 é possível visualizar, em formato de árvore, o conteúdo existente no diretório do projeto ao fim do processo de instalação:

- Na raiz encontra-se uma série de *scripts* que automatizam os processos de instalação,

Figura 9 – Estrutura do protótipo



emissão de certificados, repopulação do banco de dados e execução do projeto, além de o arquivo de texto *requirements.txt* que define as dependências do projeto a serem instaladas via *python-pip*;

- *Certificates* armazena os certificados de chaves dos usuários do sistema e do próprio POSGE;

- Os documentos emitidos pelo *software* ficam armazenados em *documents*;

- A pasta *app* armazena o código-fonte do protótipo. As classes-modelo *pessoa.py*, *procuracao.py*, *provedor.py* e *servico.py*; as classes-controlador *bd.py* (se comunica com o banco de dados), *pdf.py* (implementa todas as funções relacionadas a PDF), *utils.py* (armazena métodos utilitários) e *view.py* (responsável por redirecionar as requisições *HTTP* para suas devidas funções) e os scripts SQL *create.sql* e *drop.sql*;

- Em *static* estão os arquivos estáticos separados por tipo, o que inclui os arquivos CSS da biblioteca *materialize* e seus *scripts* em *JavaScript*, o logotipo do portal e um ícone genérico de pessoa de uso livre[1];
- Os *templates* HTML do **Jinja2**, *engine* de *templating* do *Flask*, localizam-se em *templates*.

3.3 CONFIGURAÇÕES INICIAIS

Tendo como base o diagrama entidade-relacional gerado, iniciou-se a redição dos *scripts* de criação e remoção do banco de dados. O momento da criação da base e cadastro inicial registra:

- Quatro usuários, tendo seus CPFs como chave-primária e atributos registrados nome e resumo criptográfico da senha com *salt* utilizando o algoritmo *sha256*, além de uma chave *boolean* indicando se aquele usuário tem um certificado de curta-duração;
- 11 provedores de serviços identificados pela chave-primária auto-incrementada, tendo seu nome e um resumo salgado de sua chave da API como atributos;
- 22 serviços também identificados por um campo auto-incremental, com ligação via chave-estrangeira com seu provedor de serviços e um atributo nome;
- Uma tabela de procurações sem entradas, com seu identificador auto-incremental, data de extinção, emissor e outorgado com chave-estrangeira para usuário, provedor com referência a tabela de provedores de serviços e uma chave *boolean* para sua validade, denotando que foi extinguida caso o valor seja falso. Devido ao relacionamento muitos-para-muitos existente entre as entidades **Procuração** e **Serviço**, uma tabela denominada **Atributos** foi criada para armazenar os serviços permitidos por uma procuração.

É importante observar que no contexto real de uso dos serviços do gov.br, o sistema de autenticação é implementado pelo próprio portal, denominado **Login-Único**, atribuindo ao gov.br o papel de provedor de identidade[2] ou seja, o POSGE armazenaria em seu banco de dados somente as informações necessárias para gerir as procurações, somente acessando dados adicionais via gov.br quando necessário.

Após o cadastro das informações no banco de dados, os certificados digitais dos usuários iniciais são emitidos efetuando chamadas para a biblioteca **OpenSSL** e salvos na pasta *certificates*. As chamadas efetuadas no script, presentes na Figura 10, emitem os certificados utilizando o algoritmo *RSA* com chaves de 4096 *bits*, válidos por um ano, com a extensão de uso para assinatura digital e o nome do detentor do certificado; já os certificados de curta-duração, emitidos para

[1] VECTEEZY. Download person icon for free. Disponível em: <https://www.vecteezy.com/vector-art/566937-person-icon>. Acesso em: 14 de ago. de 2019.

[2] Dúvidas Frequentes da Conta gov.br. Disponível em: <http://faq-login-unico.servicos.gov.br/en/latest/>

Figura 10 – Script de criação de certificados

```
cd app/certificates
openssl req -x509 -newkey rsa:4096 -sha256 -utf8 -keyout poseg_key.pem -out poseg_cert.pem -days 3650
-nodes -addext keyUsage=digitalSignature -subj "/C=BR/O=gov.br/CN=Portal de Outorga de Serviços do Governo Eletrônico"
openssl req -x509 -newkey rsa:4096 -sha256 -utf8 -keyout 21542314119_key.pem -out 21542314119_cert.pem -days 365
-nodes -addext keyUsage=digitalSignature -subj "/C=BR/O=gov.br/CN=Ricardo Ribeiro"
openssl req -x509 -newkey rsa:4096 -sha256 -utf8 -keyout 63567303406_key.pem -out 63567303406_cert.pem -days 365
-nodes -addext keyUsage=digitalSignature -subj "/C=BR/O=gov.br/CN=João da Costa"
openssl req -x509 -newkey rsa:4096 -sha256 -utf8 -keyout 71234411768_key.pem -out 71234411768_cert.pem -days 365
-nodes -addext keyUsage=digitalSignature -subj "/C=BR/O=gov.br/CN=Rafael Carvalho"
openssl req -x509 -newkey rsa:4096 -sha256 -utf8 -keyout 20260114359_key.pem -out 20260114359_cert.pem -days 365
-nodes -addext keyUsage=digitalSignature -subj "/C=BR/O=gov.br/CN=Gabriel Ferreira"
```

usuários que não tenham certificado, têm, além das demais características, validade de um dia (caso o certificado tivesse uma duração maior e não fosse descartado ao fim da sessão, o sistema também teria que desempenhar o papel de gestor do ciclo de vida de certificados digitais, o que sairia do escopo do proposto).

Seguindo a documentação do *framework web* utilizado, o primeiro arquivo criado, *views.py* neste protótipo, contém o código necessário para iniciar o servidor da aplicação. No código-fonte o aplicativo é inicializado definindo-se o atributo *secret_key* (chave secreta) usado para assinar os *cookies* de sessão. A função *run* é invocada, passando por parâmetro o *IP* do *host*, a porta a ser utilizada pela aplicação e uma chave *boolean* para habilitar as mensagens de *debug*, inicializando assim o servidor do *flask*. Nesse arquivo também se encontra a implementação das funções de redirecionamento do sistema. Métodos são executados quando o usuário acessa determinadas *urls*, para isso se utiliza uma marcação acima da função em questão no código-fonte, *@app.route*, informando o *path* que a executa e quais métodos HTTP são permitidos.

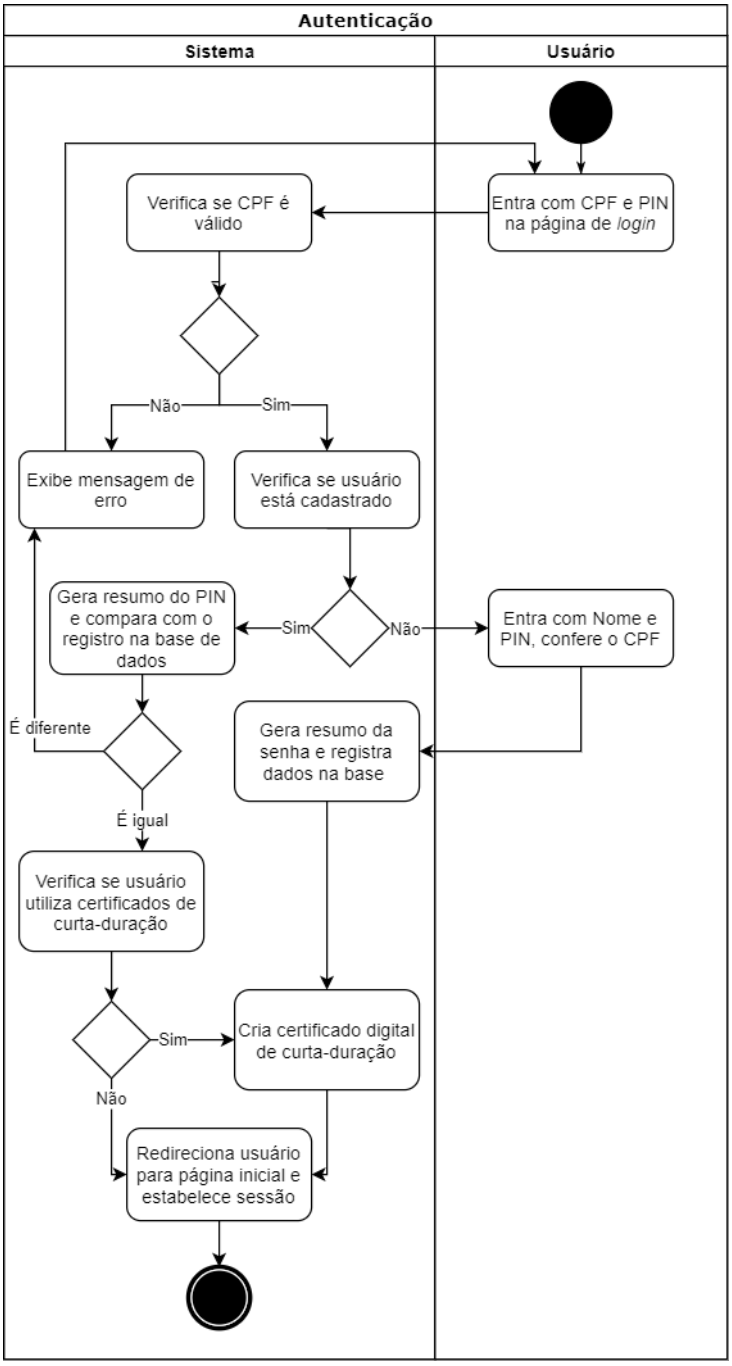
Toda a comunicação com o banco de dados é implementada pelo módulo **bd**. No arquivo *bd.py* se encontram as configurações de conexão com o *MariaDB* (*host*, nome de usuário, senha e nome da base) e todas as consultas em SQL realizadas pelo sistema.

3.4 AUTENTICAÇÃO

Com os dados iniciais carregados na base e os certificados dos usuários emitidos, o próximo passo foi implementar o sistema de autenticação. O diagrama de atividades presente na Figura 11 apresenta uma visão geral do processo.

Ao entrar no site, o usuário é direcionado para a página de *login*, que pode ser vista na Figura 12, onde lhe é requisitado o CPF e um PIN para ter acesso ao sistema. Após a submissão do formulário, o sistema verifica se a sequência numérica entrada pelo usuário constitui um CPF válido, utilizando uma função do módulo *utils* que pode ser visto no Apêndice D, e, em caso positivo, verifica se o usuário está registrado na base. Caso esteja, o resumo da senha entrada pelo usuário é comparado com o armazenado no banco de dados; se o registro do usuário indicar que ele utiliza certificados de curta-duração, um certificado é emitido e armazenado na pasta de certificados, prosseguindo com o estabelecimento da sessão e redirecionamento para a página inicial caso sejam iguais.

Figura 11 – Diagrama de Atividades do processo de autenticação



Caso o CPF não esteja registrado na base, o usuário é redirecionado para uma página de cadastro, Figura 13, onde deverá informar nome e PIN e conferir seu CPF, prosseguindo para a página inicial após o cadastro. Nesse momento sempre é gerado um certificado digital de curta-duração para que o usuário possa usar o sistema. Uma vez que esteja autenticado na aplicação, o usuário poderá realizar seu *logout* clicando no ícone presente no final do menu superior, o qual fica vermelho quando o mouse se sobrepõe, limpando seus dados de sessão que ficam armazenados na variável *session* do módulo *views* e excluindo seu certificado digital do

disco caso seja de curta-duração.

3.5 EMISSÃO

Logo que esteja autenticado na aplicação, o usuário pode proceder a emissão de uma nova procuração a partir da opção correspondente disponível na página inicial.

Na página de emissão de outorgas, Figura 15, o usuário deverá clicar, na lista a esquerda, na opção correspondente do provedor de serviços ao qual a procuração se destina, e então selecionar os serviços que serão autorizados. Do lado direito se deve entrar com o CPF do outorgado, conferindo o nome resultado na última caixa de texto do formulário, e com a data de validade do documento, sendo possível marcar a caixa de seleção para que seja válido por tempo indeterminado. Ao clicar no botão de emissão, os dados passarão por uma série de validações, que pode ser vista na Figura 14, e os dados serão registrados na base de dados.

No formulário de emissão, a biblioteca *jQuery* do *JavaScript* foi utilizada com o objetivo de realizar solicitações assíncronas para o servidor para que o usuário não tenha que enviar o formulário antes de receber uma resposta sobre as informações preenchidas.

A fim de garantir que cada procuração emitida só inclua serviços de um provedor de testes específico, cada vez que o usuário clica em uma opção de provedor uma requisição é enviada ao servidor para que este registre a escolha da opção e, a cada serviço selecionado, todos os *checkboxes* com opções não correspondentes ao provedor selecionado são automaticamente desmarcados.

Uma vez que o usuário tenha preenchido o campo com o CPF do receptor, o sistema envia uma solicitação ao servidor com o CPF, o qual responde com o nome correspondente ao detentor daquele identificador. Caso o usuário não esteja registrado, o campo informará que o usuário não está cadastrado no sistema. Se o usuário colocar seu próprio CPF, o campo também o informará.

Caso o usuário marque a opção **Indeterminado** para o campo "validade", o sistema definirá que a data de validade é o valor máximo para o campo tipo *Date* do *MariaDB*. Em todo o sistema o valor de data "31/12/9999" é interpretado como tempo indeterminado.

Após o cadastro da outorga, a função responsável por criar o documento, a qual é implementada pelo módulo *pdf* e pode ser acompanhada no Apêndice C, recebe por parâmetro um objeto da classe **procuracao** que contém todas as informações da outorga que serão usadas para criar o documento PDF usando o *ReportLab*. Cada parágrafo do texto do documento é escrito e atribuído a variáveis, utilizando-se *tags* para definir o tamanho da fonte e usar negrito, se adequando as informações contantes na outorga. O tamanho do papel do documento é definido (neste caso a4, importado pela variável *SimpleDocTemplate*), passando o tamanho das margens da página. Estilos de texto são definidos e armazenados na variável *styles* de acordo com o espaçamento e o alinhamento do texto. As variáveis de texto criadas são então inseridas em uma lista denominada *story*, informando o estilo a ser utilizado, seguidas de espaços em branco que separam os parágrafos do texto. Por fim, a função insere o rodapé com o link de *download*, utilizando a função *footer*, e o documento é gerado e salvo em disco.

Figura 12 – Tela de *Login* do sistema

Figura 13 – Tela de cadastro do sistema

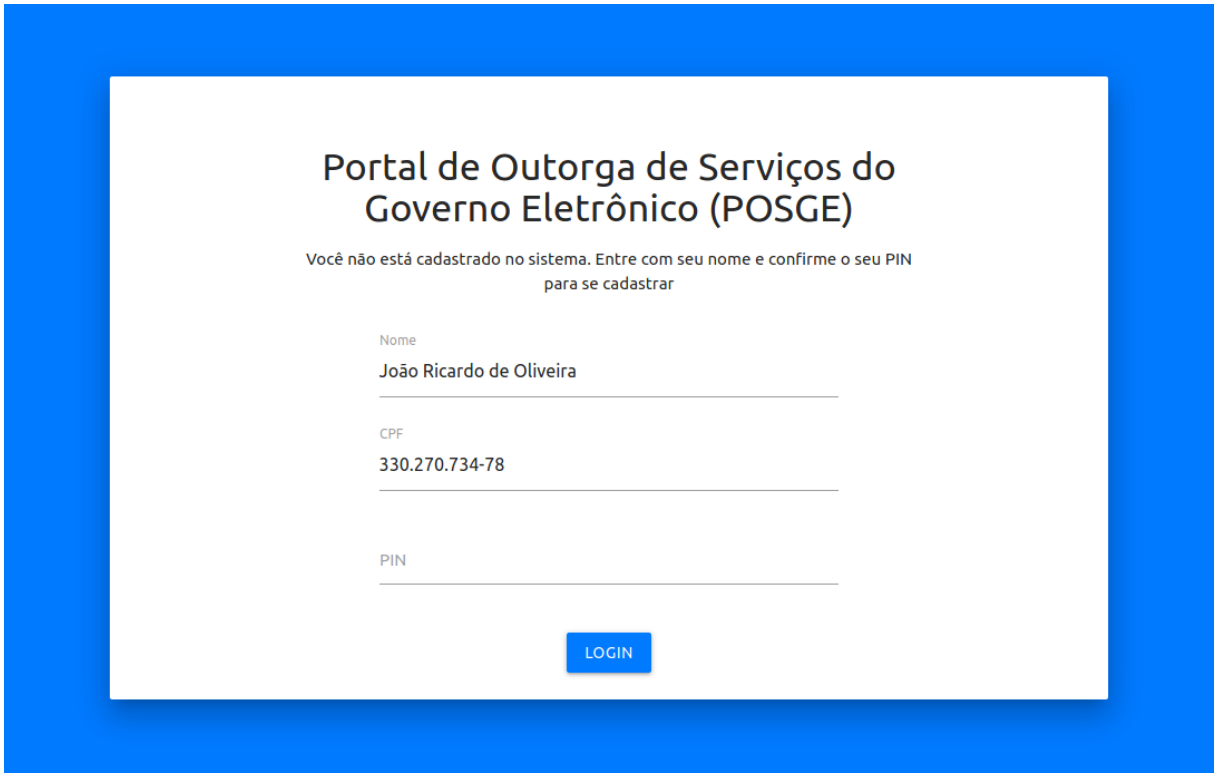
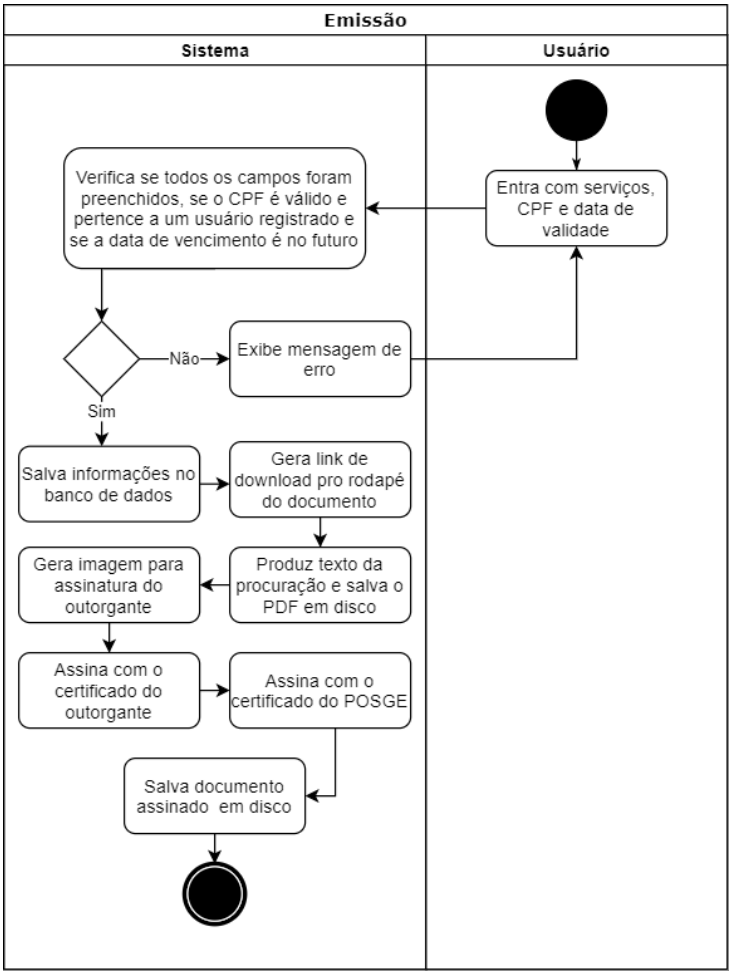


Figura 14 – Diagrama de Atividades do processo de emissão



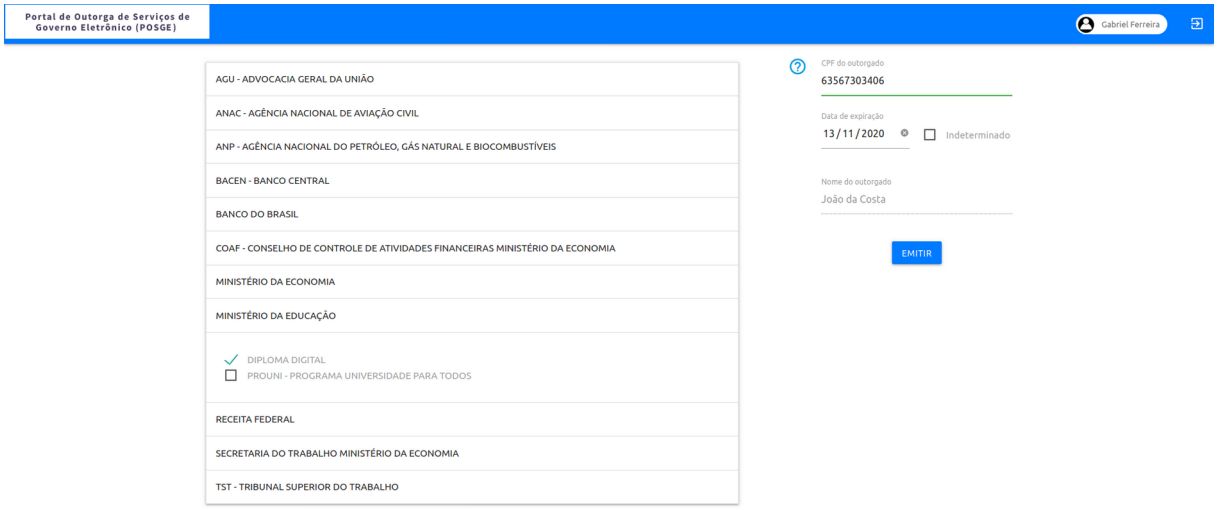
Com o arquivo criado e em disco, uma função do módulo *utils* é invocada para criar uma imagem, com o nome e CPF do futuro assinante, a fim de representar visualmente a assinatura digital do documento. Os certificados digitais e as chaves privadas do outorgante e do POSEG são carregados na memória, metadados da assinatura como contato, localização, hora, nome da entidade emissora e imagem da assinatura são adicionados a uma variável dicionário e esses dados são passados por parâmetro, junto a definição do algoritmo de resumo criptográfico, para a função *sign* do *Endesive* que assina o documento duas vezes, finalizando o processo de emissão. O documento é disponibilizado para *Download* através da página inicial do sistema.

3.6 DOWNLOAD E EXTINÇÃO

A partir da página inicial do sistema, presente na Figura 16, o usuário pode visualizar as procurações que foram emitidas e recebidas por ele, além de baixar e extinguir os documentos emitidos ao clicar nos ícones correspondentes na tabela. Tanto o outorgante quanto o receptor da outorga detém poderes para executar ambas as tarefas.

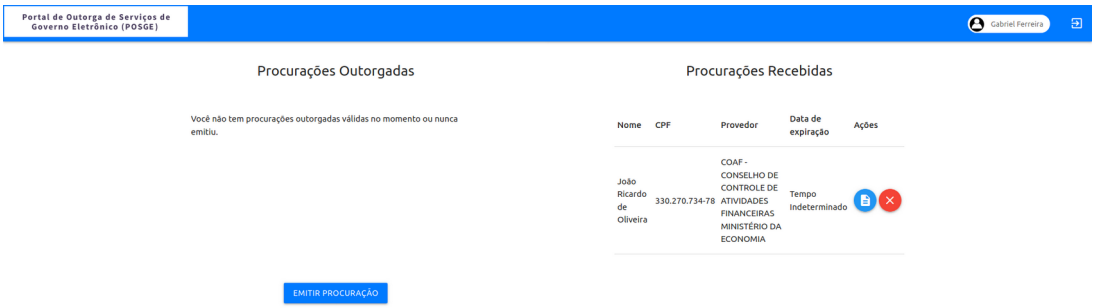
Ao clicar no ícone de *Download*, uma requisição é enviada para o servidor, utilizando

Figura 15 – Tela de emissão de procurações do sistema



jQuery, com o identificador do documento. O sistema verifica se o usuário tem direito de acessar aquele documento, o que só é verdade se tratar-se do outorgante ou outorgado, e gera a *string* correspondente ao nome do arquivo utilizando as informações disponíveis na outorga e libera o *download* da procuração.

Figura 16 – Tela inicial do sistema



No processo de extinção da outorga, que é iniciado pela opção correspondente na tela inicial, uma requisição é enviada ao servidor com o identificador da procuração. A função de extinção verificará se o usuário tem permissão de efetuar a ação e, em caso positivo, exibirá uma *pop-up* requisitando a confirmação do usuário. Em caso confirmado, o sistema irá mudar a *flag* de validade presente na entrada do documento na tabela de procurações para *false* e excluirá o documento do disco-rígido do servidor. A procuração não mais aparecerá na tela inicial de quaisquer usuários e caso alguém tente realizar *download*, o sistema informará que a procuração não é válida mais.

3.7 API

Com a implementação da funcionalidade de emissão de procurações finalizada, é necessário disponibilizar o acesso via API aos demais serviços de governo eletrônico do portal gov.br.

A fim de garantir o controle do acesso dos provedores de serviços as funcionalidades implementadas, cada provedor cadastrado no banco de dados tem uma chave de acesso a API. As chaves foram geradas utilizando a função *urandom* da biblioteca *os* do *python* e são representadas em base hexadecimal. No momento do cadastro do provedor, um resumo criptográfico *sha256* salgado da chave de acesso é armazenado em sua tupla, a fim de verificar a validade da chave no momento do acesso à API.

A autenticação do provedor na API não só é utilizada para ter acesso às funcionalidades do sistema como também para limitar o que pode ser acessado. O sistema só responde as requisições com os dados requisitados se a procuração for destinada ao uso dos serviços do provedor autenticado. As possibilidades de resposta podem ser visualizadas no Quadro 4.

Quadro 3 – Possíveis respostas HTTP do sistema

Código	**Descrição**
200 - OK	A requisição foi um sucesso.
400 - Bad Request	Erro de sintaxe na requisição.
401 - Unauthorized	O conteúdo solicitado só pode ser acessado por usuários autenticados.
403 - Forbidden	O usuário autenticado não tem direito de acesso ao conteúdo solicitado.
406 - Not Acceptable	O filtro definido pelo usuário na requisição não retornou um conteúdo.
410 - Gone	O conteúdo requisitado não existe mais no servidor.

Fonte: (MDN, 2019)

3.7.1 Implementação

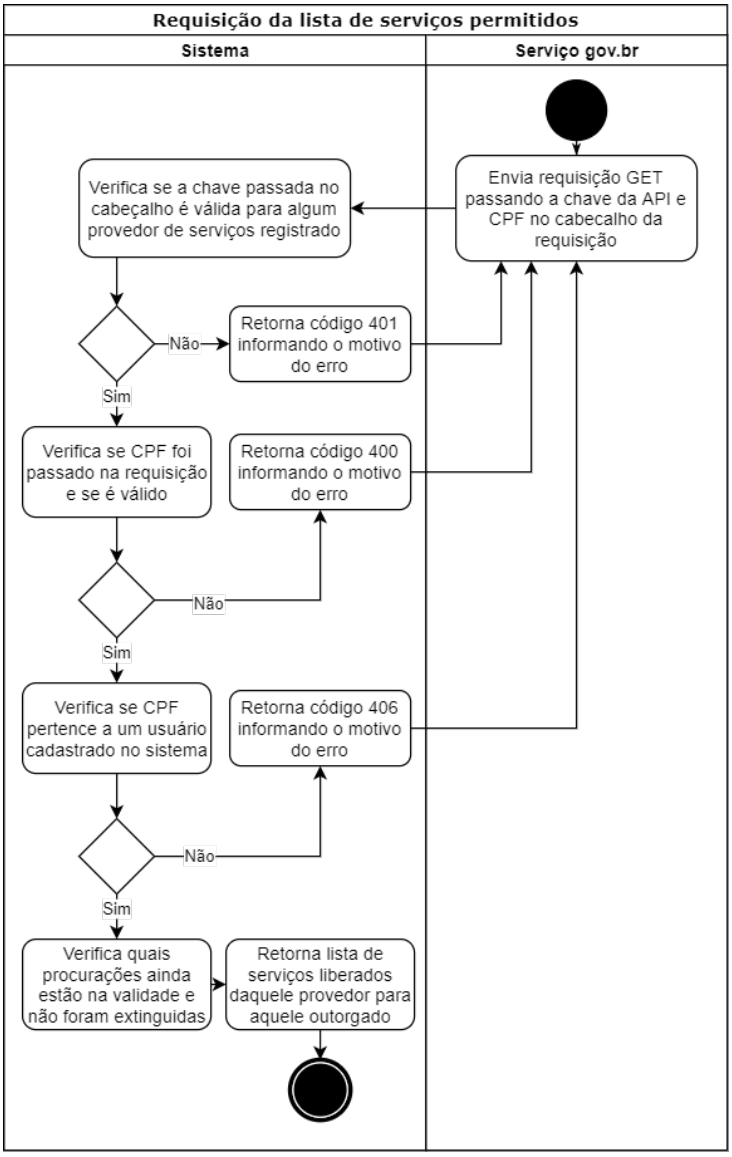
No módulo *views* foram implementadas três funções para uso da API; *verify_api_key*, para realizar a autenticação do provedor, *get_servicos*, que disponibiliza a lista de serviços do provedor liberados para determinado outorgado, e *download_procuracao_api*, a qual responde a requisição com o arquivo PDF da procuração requisitada. Para que as respostas em *JSON* estivessem no formato *UTF-8* foi necessário definir a *flag JSON_AS_ASCII* da variável *app* no módulo *views* para "falso".

A função de autenticação da API é um *decorator* — em *python*, *decorators* são objetos que tem a capacidade de alterar o comportamento de funções e classes; a função ou classe em questão é passada por parâmetro para o objeto que a modifica de acordo com o definido em sua implementação[3] — responsável por alterar o funcionamento das outras duas funções utilizadas

[3] PEP 318 – Decorators for Functions and Methods. Disponível em: <https://www.python.org/dev/peps/pep-0318/> . Acesso em: 31 de out. de 2020.

pela API, ambas precisam passar pela autenticação antes de serem executadas; ela compara um resumo da chave disponibilizada no cabeçalho da requisição com os resumos criptográficos salgados armazenados nas tuplas dos provedores do banco de dados. Se aquela chave pertencer a um provedor, o mesmo é autenticado e pode prosseguir com a requisição, caso contrário, o servidor retorna uma mensagem em *JSON* informando que a chave é inválida e o código HTTP 401.

Figura 17 – Diagrama de atividades do processo de requisicação da lista de serviços permitidos



Para acessar a lista de serviços permitidos, o provedor deve enviar uma requisição *GET* para o endereço da API (*http://host:5000/api*) informando um CPF em seu cabeçalho, que é interpretado pela interface como o identificador do usuário para qual os direitos de uso foram concedidos, e a chave da API em seu cabeçalho. A chave passa pela função de autenticação e o CPF passa por algumas validações, que podem ser visualizadas na Figura 17, retornando códigos HTTP de acordo com o resultado. Caso o CPF passe pela validação e o usuário esteja

cadastrado na base, o sistema irá retornar uma lista com as procurações válidas emitidas por ele para o provedor de serviços autenticado em *JSON* (podendo também retornar uma lista vazia caso não exista nenhuma) e um código HTTP 200. As informações disponíveis na lista são:

- Identificador da procuração no sistema;
- Link para download da procuração via API;
- Data de validade da procuração;
- O nome do provedor de serviços;
- Nomes e CPFs do emissor e do outorgado;
- Lista com o identificador e nome dos serviços liberados.

Com o objetivo de realizar o *download* da procuração, o provedor deve realizar uma requisição *GET* para o endereço de *download* de procurações da API (*http://host:5000/api/download/id*) com a chave da API inserida no *header*. A chave é usada para autenticar o provedor e o identificador da procuração para acessá-la. Para que a API responda com o documento, a procuração precisa ser direcionada ao provedor de serviços autenticado e tem que estar válida, como é possível ver na Figura 18, retornando códigos HTTP de erro caso contrário.

3.7.2 Testes

Para testar o funcionamento da API foram cadastradas na plataforma seis procurações com diferentes características que interferem na decisão de incluir ou não a procuração na lista de serviços liberados. A relação das entradas pode ser visualizada no Quadro 5.

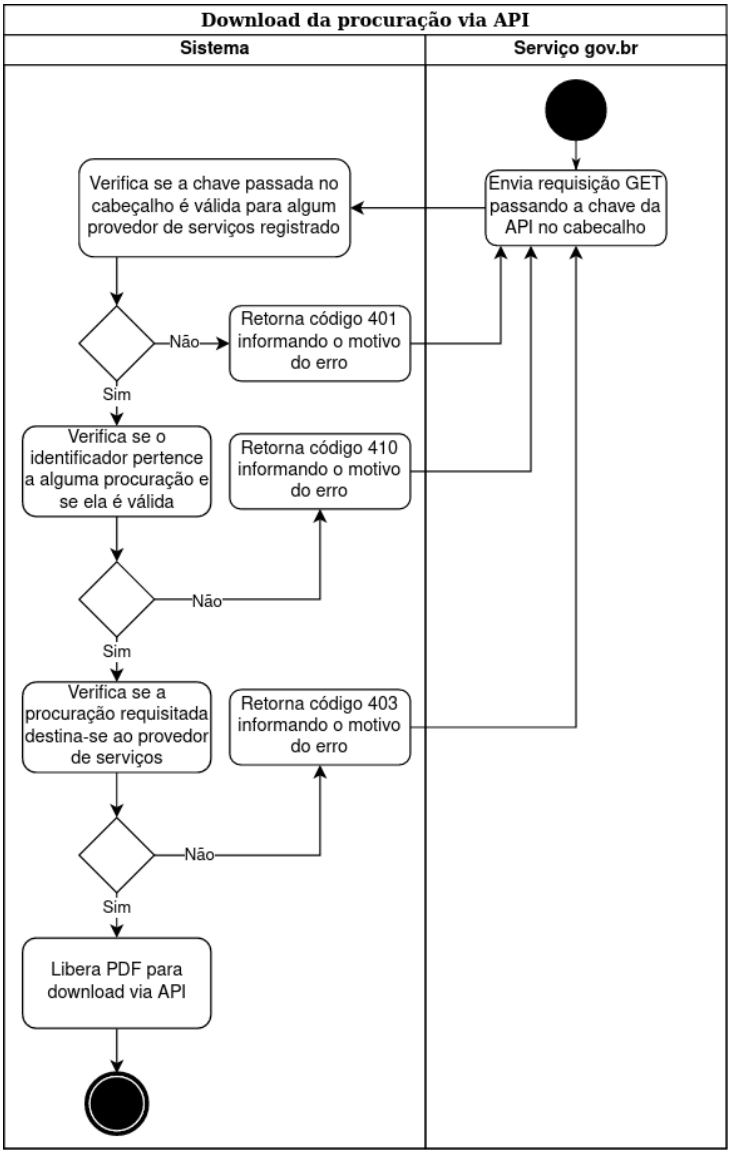
Quadro 4 – Procurações emitidas para teste da API

ID	Emissor	Outorgado	PdS	Validade	Observação
3	33027073478	20260114359	RFB	Indefinida	Data de validade é indefinida
4	21542314119	20260114359	RFB	30 Nov. 2020	-
5	71234411768	20260114359	RFB	10 Dez. 2020	Foi extinguida pelo outorgado
6	20260114359	71234411768	RFB	20 Nov. 2020	O CPF é do emissor
7	33027073478	20260114359	MEC	25 Dez. 2020	O provedor é outro
8	21542314119	20260114359	RFB	10 Out. 2020	A data de validade já passou

Fonte: o autor

Segundo as verificações realizadas pelo *software*, que podem ser visualizadas no diagrama da Figura 17, caso o provedor de serviços **Receita Federal** requisitasse os serviços liberados para o **Gabriel Ferreira**, CPF 202.601.143-59, somente as procurações de *ID* 3 e 4 deveriam aparecer na lista de serviços permitidos, de acordo com as observações presentes no Quadro 5.

Figura 18 – Diagrama de atividades do processo de download da procuração via API



O utilitário **curl** do *linux*, capaz de se comunicar com servidores utilizando uma grande gama de protocolos de comunicação, foi utilizado para enviar requisições *GET* à API para realizar os testes O comando executado em *shell* para testar a funcionalidade de requisição da lista de serviços permitidos envia uma requisição contendo a chave da API em seu cabeçalho e o CPF de um usuário definido na variável**CPF**. O resultado dessa consulta pode ser visualizado na Figura 19.

Para realizar o teste de *download* da procuração pela API foi utilizado o mesmo utilitário do *Linux*. Nessa ocasião a *url* da requisição é diferente e a informação passada é o *ID* da procuração que deseja ser baixada. Foram executados três testes, uma situação de sucesso, Figura 20, e duas de erro, Figura 21 e Figura 22. As verificações realizadas podem ser verificadas na Figura 18.

Figura 19 – Resposta da API à requisição enviada pelo provedor de serviços RFB

```
~ >>> curl -k -H "api_key:689dff9b7078884997ae9ada8bdf129e1e322c39c58c10cc" -H "cpf:20260114359" -X GET "http://localhost:5000/api"
{
  "procuracoes": [
    {
      "cpf_emissor": "21542314119",
      "cpf_outorgado": "20260114359",
      "id_procuracao": 4,
      "link_pra_download": "http://localhost:5000/api/download/4",
      "nome_emissor": "Ricardo Ribeiro",
      "nome_outorgado": "Gabriel Ferreira",
      "provedor": "RECEITA FEDERAL",
      "servicos": [
        {
          "id_servico": 11,
          "nome_servico": "DIRF - DECLARAÇÃO DO IMPOSTO DE RENDA RETIDO NA FONTE"
        },
        {
          "id_servico": 12,
          "nome_servico": "DOI - DECLARAÇÃO DE OPERAÇÕES IMOBILIÁRIAS"
        }
      ],
      "validade": "Mon, 30 Nov 2020 00:00:00 GMT"
    },
    {
      "cpf_emissor": "33027073478",
      "cpf_outorgado": "20260114359",
      "id_procuracao": 3,
      "link_pra_download": "http://localhost:5000/api/download/3",
      "nome_emissor": "João Ricardo de Oliveira",
      "nome_outorgado": "Gabriel Ferreira",
      "provedor": "RECEITA FEDERAL",
      "servicos": [
        {
          "id_servico": 17,
          "nome_servico": "DMED - DECLARAÇÃO DE SERVIÇOS MÉDICOS E DA SAÚDE"
        },
        {
          "id_servico": 18,
          "nome_servico": "DPREV - DECLARAÇÃO SOBRE A OPÇÃO DE TRIBUTAÇÃO DE PLANOS PREVIDENCIÁRIOS"
        }
      ],
      "validade": "Fri, 31 Dec 9999 00:00:00 GMT"
    }
  ]
}
```

Figura 20 – Resposta da API a uma requisição de *download* bem-sucedida

```
~ >>> curl -ki -H "api_key:689dff9b7078884997ae9ada8bdf129e1e322c39c58c10cc" -X GET "http://localhost:5000/api/download/4"
HTTP/1.0 200 OK
Content-Disposition: attachment; filename="procuracao_Ricardo Ribeiro_RECEITA FEDERAL_4.pdf"
Content-Length: 39475
Content-Type: application/pdf
Last-Modified: Wed, 30 Sep 2020 13:29:27 GMT
Cache-Control: no-cache, no-store, must-revalidate
Expires: Mon, 07 Dec 2020 13:55:33 GMT
ETag: "1601472567.8200798-39475-2592679159"
Date: Mon, 07 Dec 2020 01:55:33 GMT
Server: Werkzeug/1.0.1 Python/3.8.5

Warning: Binary output can mess up your terminal. Use "--output -" to tell
Warning: curl to output it to your terminal anyway, or consider "--output
Warning: <FILE>" to save to a file.
```

Figura 21 – Resposta da API a uma requisição de *download* código HTTP 410

```
~ >>> curl -ki -H "api_key:689dff9b7078884997ae9ada8bdf129e1e322c39c58c10cc" -X GET "http://localhost:5000/api/download/5"
HTTP/1.0 410 GONE
Content-Type: application/json
Content-Length: 99
Cache-Control: no-cache, no-store, must-revalidate
Server: Werkzeug/1.0.1 Python/3.8.5
Date: Mon, 07 Dec 2020 01:56:46 GMT

{
  "msg": "O identificador não pertence a nenhuma procuração ou a mesma já foi extinguida."
}
```

Figura 22 – Resposta da API a uma requisição de *download* código HTTP 403

```
~ >>> curl -ki -H "api_key:689dff9b7078884997ae9ada8bdf129e1e322c39c58c10cc" -X GET "http://localhost:5000/api/download/7"
HTTP/1.0 410 GONE
Content-Type: application/json
Content-Length: 99
Cache-Control: no-cache, no-store, must-revalidate
Server: Werkzeug/1.0.1 Python/3.8.5
Date: Mon, 07 Dec 2020 01:57:36 GMT

{
  "msg": "O identificador não pertence a nenhuma procuração ou a mesma já foi extinguida."
}
```

4 CONCLUSÃO

Neste trabalho foi abordada uma proposta de melhoria nos processos democráticos dos serviços eletrônicos do governo brasileiro no que tange a delegação de direitos de uso. É um direito do cidadão, no exercício de sua democracia, permitir a terceiros que efetuem ações em seu nome. Para que isso fosse possível, foi necessário estudar diferentes áreas do conhecimento, devido ao caráter interdisciplinar da obra, levando a compreensão dos processos que regem a concessão de poderes em diferentes instâncias. Com o objetivo de se desenvolver um portal de emissão de procurações eletrônicas utilizando certificação digital, houve muito a se compreender sobre os seus casos de uso e limitações, havendo a necessidade de também lidar com usuários do gov.br que não têm certificados, o que resultou na possibilidade de emitir certificados de curta-duração no momento do cadastro. Para o acesso via API foi proposto o uso de chaves geradas para cada provedor, no entanto a forma de acesso é dependente de como o próprio portal de serviços de e-gov implementa a comunicação entre servidores.

No fim, o trabalho demonstrou que é possível, através do conjunto de tecnologias utilizado, implementar uma solução para a problemática questionada seguindo os objetivos propostos inicialmente. No entanto, é necessário investigar mais a respeito do que é feito no portal de serviços brasileiro para adequar a metodologia aplicada ao padrão implantado para que seja usado na prática.

A realização de trabalhos acadêmicos com foco em melhoria nos processos democráticos brasileiros no âmbito digital são importantes por serem de interesse de toda a sociedade e devem ser fomentados.

4.1 TRABALHOS FUTUROS

- Os *frameworks* para manipulação de *PDF* usados e convenções seguidas na concepção do trabalho se limitaram às definições da norma técnica ISO-32000-1 de 2008, a versão 1.7 do formato PDF. Essa escolha foi tomada devido a dificuldade de se encontrar *frameworks open-source* e gratuitos que manipulassem PDF de acordo com as novas convenções da versão 2.0 de 2017 à época do início da trabalho, principalmente no que tange ao uso de assinaturas digitais.

- Uma melhoria possível, e não prevista inicialmente no escopo do trabalho, é a utilização do protocolo de comunicação *oauth2* em substituição à *WEB API*. O nível de complexidade da utilização desse protocolo é maior, no entanto podem existir uma série de vantagens para a realização da integração com outros serviços e na segurança dos dados.

- Na União Européia existe o conceito de lista de serviços confiáveis, denominada Trusted Service List (TSL)[1], que se refere a uma lista de listas em que cada país integrante da união

[1] https://webgate.ec.europa.eu/tl-browser

tem, em sua própria lista, informações a cerca dos serviços disponibilizados, histórico e políticas de forma padronizada; Se encontram disponíveis Autoridades Certificadoras, para emissão de certificados digitais, Autoridades de Carimbo de Tempo e diversos tipos de serviços de governo eletrônico, existindo a possibilidade de se aplicar esses conceitos ao Brasil.

- É possível implementar uma solução utilizando certificados de atributo para o problema. Seria interessante existir um novo trabalho apresentando as vantagens dessa abordagem.

REFERÊNCIAS

BRASIL. Artigo 105 da Lei nº 13.105. **Código de Processo Civil**, 16 mar. 2015. Disponível em: <https://www.jusbrasil.com.br/topicos/28895561/artigo-105-da-lei-n-13105-de-16-de-marco-de-2015>. Acesso em: 16 out. 2020. Citado na p. 26.

BRASIL. Instrução Normativa RFB nº 1751. **Diário Oficial da União**, 18 out. 2017. Disponível em: <http://normas.receita.fazenda.gov.br/sijut2consulta/link.action?idAto=87210>. Acesso em: 16 out. 2020. Citado na p. 26.

CARLOS, Marcelo Carlomagno *et al.* **Introdução a Infraestrutura de Chaves Públicas e Aplicações**. 1.0.1. ed. [*S.l.*]: Escola Superior de Redes RNP, 2010. Citado na p. 21.

DJANGO. **Django documentation**. [*S.l.*: *s.n.*], ago. 2019. Disponível em: <https://docs.djangoproject.com/en/2.2/>. Acesso em: 13 ago. 2019. Citado na p. 27.

GONÇALVES, Marcus Vinicius Rios. **Novo Curso de Direito Processual Civil**. 4. ed. [*S.l.*]: Editora Saraiva, 2007. v. 1. Citado na p. 25.

GRÖNLUND, Åke; HORAN, Thomas A. Introducing e-gov: History, Definitions, and Issues. **Communications of the Association for Information Systems**, v. 15, p. 713–729, 2004. Citado na p. 25.

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. **ISO 32000-1:2008**: Document management - Portable document format — Part 1: PDF 1.7. [*S.l.*], jul. 2008. P. 747. Citado nas pp. 16, 19, 21–23.

KOPECKY, Jacek; FREMANTLE, Paul; BOAKES, Rich. A History and Future of Web APIs. **Information Technology**, 26 fev. 2014. Disponível em: <https://www.researchgate.net/publication/274527941_A_history_and_future_of_Web_APIs>. Acesso em: 19 out. 2020. Citado na p. 24.

KRAEMER, Kenneth L. Local Government and Information Technology in the United States. **OECD Informatics Studies**, v. 12, 1978. Citado na p. 25.

MAKAREWICZ, Grzegorz. **m32/endesive: en-crypt, de-crypt, si-gn, ve-rify - smime, pdf, xades and plain files in pure python**. [*S.l.*: *s.n.*], out. 2019. Disponível em: <https://github.com/m32/endesive>. Acesso em: 13 out. 2019. Citado na p. 27.

MOZILLA DEVELOPER NETWORK. **Autenticação HTTP**. [*S.l.*: *s.n.*], mar. 2019. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Authentication >. Acesso em: 31 out. 2020. Citado na p. 38.

OPENSSL. **Manpages for 1.1.1**. [*S.l.*: *s.n.*], set. 2019. Disponível em: <https://www.openssl.org/docs/man1.1.1/>. Citado na p. 28.

PALLETS. **Flask documentation (1.1.x)**. [*S.l.*: *s.n.*], ago. 2019. Disponível em: <https://flask.palletsprojects.com/en/1.1.x/ >. Acesso em: 13 ago. 2019. Citado na p. 27.

REPORTLAB. **ReportLab open-source User Guide**. [*S.l.*: *s.n.*], out. 2019. Disponível em: <https://www.reportlab.com/docs/reportlab-userguide.pdf>. Acesso em: 4 out. 2019. Citado na p. 27.

Apêndices

APÊNDICE A – DOCUMENTO EXEMPLO

Procuração

Outorgante: Eu, Gabriel Ferreira, portador do CPF nº 202.601.143-59, pelo presente instrumento, nomeio e constituo como meu (minha) procurador(a)

Outorgado(a): João da Costa, portador do CPF nº 635.673.034-06, com poderes para representaro outorgantejunto ao provedor MINISTÉRIO DA EDUCAÇÃO, na utilização dos seguintes serviços:

- DIPLOMA DIGITAL

Responsabilizando-me por todos os atos praticados no cumprimento deste instrumento, cessando seus efeitos a partir de 13 de novembro de 2020.

Assinado digitalmente via Portal de Outorga de Serviços do Governo Eletrônico em 30 de outubro de 2020 por Gabriel Ferreira

Portal de Outorga de Serviços de Governo Eletrônico (POSGE)

Outorgado via POSGE por:
Nome: Gabriel Ferreira
CPF: 202.601.143-59

A autenticidade deste documento pode ser verificada em: http://localhost:5000/download/3

Abrir documento anexado

APÊNDICE B – CONTEÚDO DA ASSINATURA DO DOCUMENTO EXEMPLO EM ASN.1

```
SEQUENCE {
   OBJECTIDENTIFIER 1.2.840.113549.1.7.2 (signedData)
   [0] {
      SEQUENCE {
         INTEGER 0x01 (1 decimal)
         SET {
            SEQUENCE {
               OBJECTIDENTIFIER 2.16.840.1.101.3.4.2.1 (sha256)
               NULL
            }
         }
         SEQUENCE {
            OBJECTIDENTIFIER 1.2.840.113549.1.7.1 (data)
         }
         [0] {
            SEQUENCE {
               SEQUENCE {
                  [0] {
                     INTEGER 0x02 (2 decimal)
                  }
                  INTEGER 0x290ed638cad83cedd574aea102bb0dd0a38b0b5c
                  SEQUENCE {
                     OBJECTIDENTIFIER 1.2.840.113549.1.1.11
                     (sha256WithRSAEncryption)
                     NULL
                  }
                  SEQUENCE {
                     SET {
                        SEQUENCE {
                           OBJECTIDENTIFIER 2.5.4.6 (countryName)
                           PrintableString 'BR'
                        }
                     }
                     SET {
                        SEQUENCE {
                          OBJECTIDENTIFIER 2.5.4.10 (organizationName)
                           UTF8String 'gov.br'
```

```
37                      }
38                   }
39                   SET {
40                      SEQUENCE {
41                         OBJECTIDENTIFIER 2.5.4.3 (commonName)
42                         UTF8String 'Gabriel Ferreira'
43                      }
44                   }
45                }
46                SEQUENCE {
47                   UTCTime '201029182647Z'
48                   UTCTime '211029182647Z'
49                }
50                SEQUENCE {
51                   SET {
52                      SEQUENCE {
53                         OBJECTIDENTIFIER 2.5.4.6 (countryName)
54                         PrintableString 'BR'
55                      }
56                   }
57                   SET {
58                      SEQUENCE {
59                        OBJECTIDENTIFIER 2.5.4.10 (organizationName)
60                         UTF8String 'gov.br'
61                      }
62                   }
63                   SET {
64                      SEQUENCE {
65                         OBJECTIDENTIFIER 2.5.4.3 (commonName)
66                         UTF8String 'Gabriel Ferreira'
67                      }
68                   }
69                }
70                SEQUENCE {
71                   SEQUENCE {
72                      OBJECTIDENTIFIER 1.2.840.113549.1.1.1
                      (rsaEncryption)
73                      NULL
74                   }
```

```
75                  BITSTRING 0x3082020a0282020100ab4b5b9d6398cf7d7
                    61e4af8d5902b7bc0a98700489d8166082e8e4c45ff9974
                    3cd975e54559668fc001c2c1b8448743b25f837b431e1a0
                    d5df29825d5932ff94e99f41cc040c35e3166387d6e54bc
                    0e23c30f5415a7b5edd06fc5bf5a582c7b786e7ac01efd9
                    a1f21a79f5ba76faefdf83e20d7ee287759ab69b2a3edcc
                    6a438cfd9c9988f0af39ebc825fc296ca1f559340a4de4b
                    b43641689efe410fe133c0e9c458f094df4cf47752845de
                    96262189d3ee1ad66bc765cc59167fca60760c2e613bfee
                    e194232e9cdb5e04ba7033538ce0c338eda6206d59452f4
                    89d20db59caaec9d4a330e8120d13b0f7d26365b92ce2b9
                    56ac8df3935feb4bf6edf3bcb1f4db218328ed7d28474683
                    74506eaf59b0393274250c3cb9de8724672b726a28724bdd
                    9ec0c08a47368a9cd48bcb08ac29f32c4a4918daf8b32f3
                    8753c79929e4e4e72537bcb178b1ca66d7b18f0213471ac
                    214321765faac54345dcb01f7810c89a29e20e4019ccde4
                    31a690bcaf800582e4b42244385de2a5adc79e7c865b54a
                    59f1612dc812920c615e4e67345ab4a1f18ad76dd914d21
                    e9df954857412ea74074843895508a05de708e5f1e672c6
                    ec01bec25d57954e0767cc47c362577fdfef74103e97552
                    506d0e05d02d675ba91e258fe78bb75086bb27e4d72b568
                    2529c683afb696737083b9511ccf050ec58d4529ae4f139
                    d78f4d2309f34397d70203010001 : 0 unused
                    bit(s)
76              }
77              [3] {
78                  SEQUENCE {
79                      SEQUENCE {
80                          OBJECTIDENTIFIER 2.5.29.14
                            (subjectKeyIdentifier)
81                          OCTETSTRING 04147c12f589593351d2ccf18f6be7f
                            2e20fc52c7d1b
82                      }
83                      SEQUENCE {
84                          OBJECTIDENTIFIER 2.5.29.35
                            (authorityKeyIdentifier)
85                          OCTETSTRING 301680147c12f589593351d2ccf18f6
                            be7f2e20fc52c7d1b
86                      }
```

```
87                      SEQUENCE {
88                       OBJECTIDENTIFIER 2.5.29.19 (basicConstraints)
89                          BOOLEAN TRUE
90                          OCTETSTRING 30030101ff
91                      }
92                      SEQUENCE {
93                          OBJECTIDENTIFIER 2.5.29.15 (keyUsage)
94                          OCTETSTRING 03020780
95                      }
96                  }
97              }
98          }
99          SEQUENCE {
100             OBJECTIDENTIFIER 1.2.840.113549.1.1.11
                (sha256WithRSAEncryption)
101             NULL
102         }
103         BITSTRING 0x7458d38c121d69d0bdd049329f087cbbbbf93a3dc8
            957b0421b878879127fe6f17d40d0479f043526b949ce11c29a79
            46b5d4f3873fcade660bcc6e9fd226b42c0ed18e39086c6d656f4f
            30564cc23c32ef00a7cc7e36fb644350c1b8d8d9a07a1f4b1acab
            752c688f7e4e1cffbdc1bae70b9e23b879ecb46f05121b0c760593
            c8695d17ea069208166e660245f49f4b9b1d5542f2d11103c1f075
            3cc4d46622ed9b46ddaab7f0351baa034f0da475a72b743d17731
            39f8d8cd034325ab33837a81fc00a3eb99f33e9e6d8959ed72a2f
            a131a107cee3a5cb625e527b03543b781ff0ebee6178e67e07ca7
            0f5d31ec32c41953d6ab68b20001d24af470bfc38d9ffed888a1a
            737649854634537e4fb0e61c0ec1cc71a03aca9fca54eb0bb76e9e
            3969eb5ffddc89f2f2581830829934f8d256992a98bf58f6d7038
            58445cf5969b0701637ceba2a920531a09e25f372d638c4d9659c
            b01ea81804b892c5c8cce051b802d14fcd01851fdb5b15abf0d53
            79a6c3ff71281d12f170129e1e6a4fa64fd903c60c7cd8af71ae1e
            026e28241b2d16c3dbfbd0fe9613c5dd927e26421ed025795a1b6
            1056140f3e66156506e5795748d4c1aa326a64fb488ede0f2d912d
            a406474bd419bb333829c0a3fc4076ccfbdcd19ed646fa05e7e15
            29f587fd23be4a6073d2b0004bc4ccdb6e1afcfe57fef7b3e6f01
            afe129a968b52b1041c34e2 : 0 unused
            bit(s)
104     }
```

```
105         }
106         SET {
107            SEQUENCE {
108               INTEGER 0x01 (1 decimal)
109               SEQUENCE {
110                  SEQUENCE {
111                     SET {
112                        SEQUENCE {
113                           OBJECTIDENTIFIER 2.5.4.6 (countryName)
114                           PrintableString 'BR'
115                        }
116                     }
117                     SET {
118                        SEQUENCE {
119                          OBJECTIDENTIFIER 2.5.4.10 (organizationName)
120                           UTF8String 'gov.br'
121                        }
122                     }
123                     SET {
124                        SEQUENCE {
125                           OBJECTIDENTIFIER 2.5.4.3 (commonName)
126                           UTF8String 'Gabriel Ferreira'
127                        }
128                     }
129                  }
130                  INTEGER 0x290ed638cad83cedd574aea102bb0dd0a38b0b5c
131               }
132               SEQUENCE {
133                  OBJECTIDENTIFIER 2.16.840.1.101.3.4.2.1 (sha256)
134                  NULL
135               }
136               [0] {
137                  SEQUENCE {
138                     OBJECTIDENTIFIER 1.2.840.113549.1.9.3
                     (contentType)
139                     SET {
140                        OBJECTIDENTIFIER 1.2.840.113549.1.7.1 (data)
141                     }
142                  }
```

```
143                 SEQUENCE {
144                     OBJECTIDENTIFIER 1.2.840.113549.1.9.5
                        (signingTime)
145                     SET {
146                         UTCTime '201030130945Z'
147                     }
148                 }
149                 SEQUENCE {
150                     OBJECTIDENTIFIER 1.2.840.113549.1.9.4
                        (messageDigest)
151                     SET {
152                       OCTETSTRING fff8674c8f25f43a3d190fe32138573f0a
                          4025b1d2722f91e69574664a7a5651
153                     }
154                 }
155             }
156             SEQUENCE {
157               OBJECTIDENTIFIER 1.2.840.113549.1.1.1 (rsaEncryption)
158                NULL
159             }
160             OCTETSTRING 519a27b9b6b53306d4b88b5c247de563bc10e8c2b5
                76b8110e1da749a335f148ba97c79d3a10cbb4a75a3fa703e4a07
                58cfac362e02a75524251eb80b36bae406673eac152e37739df03b
                b0e24c9157a807657121d43ba1c8cc3c4741b251132a6e39649d3
                73d69271c3aae15b739fd9102f148a93247a0e0afcbd3a67ac0639
                e3f114845db95df2cfad0bf066303b8e5f02d4828543f9d48e73c
                b9a7e9ae1d8bda8135967b86b6b0a891b0e79e6ec27b4cc6bc5b3
                710864446186719 36c13e6dbcd79213ed5f64d3cddc111f4703b23
                d30cb24fa26bed090676ef6699035d45cb9c1ed8b804a68d5b22c
                1d1842028ce2646b62c07c842d0687485f099f1b004aec7d59a4d7
                030912e8af6b0de5444e4b8e8624e7f6dd738389e763e822a4c23
                a48429f411401ef1d3c1f959664254de65e2ac04011c768f3fb6e
                905aafd8bd7f3be08128d9e191e6b507a913246d9681f30d522b3
                0f2b5ab67f553a090eb55c9dbdd044ab1e3a1d63ecdf64385385c3
                3832050873dee818949c3e10bd884013163c1b55f004e16f467e5
                67f9987ceaaa6db6c4b45dfabb74ef62ce06f8c7d7e0710bd3c7b
                a73128d6f7212e21b561d0c3b148b7f5cca7d59ee24eed38a08a3
                6c1cd5c17c0cf267821034674ee2d81c5d8ff722e4bbd60cd00d8
                9cc5f944481e3293193b8a1049956c24f12a29896e9621798e45c
                e1201627c9b70ad763db53d9
```

```
161             }
162           }
163       }
164     }
165 }
```

APÊNDICE C – CÓDIGO-FONTE PDF.PY

```
1  from reportlab.lib.enums import TA_JUSTIFY, TA_CENTER,TA_LEFT
2  from reportlab.platypus import SimpleDocTemplate, Paragraph, Spacer,
       Image
3  from reportlab.lib.styles import getSampleStyleSheet, ParagraphStyle
4  from reportlab.lib.units import cm
5  from datetime import datetime
6  from pathlib import Path
7  import utils
8  import locale
9
10 from OpenSSL.crypto import load_certificate, load_privatekey,
       FILETYPE_PEM
11 from endesive import pdf
12
13 locale.setlocale(locale.LC_ALL, 'pt_BR.UTF-8')
14
15 def sign(certificate_path, pkey_path, path_to_image, x_begin,
       file_name, sigfield):
16     signatureTime = datetime.now(tz=None).strftime(
17         "%Y%m%d%H%M%S-03'00'")
18     dct = {
19         'aligned': 0,
20         'sigflags': 3,
21         'sigfield': "Assinatura" + str(sigfield),
22         'contact': b'POSGE@POSGE.com',
23         'location': b'Brasil',
24         'signingdate': signatureTime.encode(),
25         'reason': b'Procuracao Eletronica do Portal de Outorga de
               Servicos do Governo Eletronico Brasileiro',
26         'signaturebox':
               utils.calculate_signature_box_size(path_to_image=str(path_to_image),
               x_begin=x_begin),
27         'signature_img': str(path_to_image)
28     }
29
30     certificate = load_certificate(type=FILETYPE_PEM,
           buffer=open(certificate_path).read())
31     pkey = load_privatekey(type=FILETYPE_PEM,
           buffer=open(pkey_path).read(),passphrase=None)
```

```
32      datau = open ( file_name , 'rb ') . read ()
33      pkey = pkey . to_cryptography_key ()
34      certificate = certificate . to_cryptography ()
35      datas = pdf . cms . sign ( datau , dct , pkey , certificate , [] , ' sha256 ')
36
37      with open ( file_name , ' wb ') as fp :
38          fp . write ( datau )
39          fp . write ( datas )
40
41  def footer ( canvas , doc ) :
42      canvas . saveState ()
43      P = Paragraph ( doc . link , ParagraphStyle ( name = ' Justify ' ,
            alignment = TA_CENTER , leading =9) )
44      w , h = P. wrap ( doc . width , doc . bottomMargin )
45      P . drawOn ( canvas , doc . leftMargin , 10)
46      canvas . restoreState ()
47
48  def create_pdf ( procuracao ) :
49      user_certificate_path = ' certificates / ' + procuracao . emissor . cpf
            + '_cert . pem '
50      user_pkey_path = ' certificates / ' + procuracao . emissor . cpf +
            '_key . pem '
51
52      poseg_certificate_path = ' certificates / poseg_cert . pem '
53      poseg_pkey_path = ' certificates / poseg_key . pem '
54
55      file_name = " procuracao_" + procuracao . emissor . nome+ \
56          "_" + procuracao . provedor + "_" + str ( procuracao . id ) + " . pdf "
57      text1 = " <b > Outorgante </ b >: , " + procuracao . emissor . nome+ " ,
            portador do CPF n° " + \
58          utils . format_cpf ( procuracao . emissor . cpf ) + \
59              " , pelo presente instrumento , nomeio e constituo como
                  meu ( minha ) procurador ( a)"
60      text2 = " <b > Outorgado ( a ) </ b >: " + procuracao . outorgado . nome+ " ,
            portador do CPF n° " +
            utils . format_cpf ( procuracao . outorgado . cpf ) + \
61          " , com poderes para representar o outorgante junto ao
                provedor " + procuracao . provedor + " , "
62      text2 += " na utiliza ç ã o dos seguintes servi ç os : "
63      text3 = " Responsabilizando-me por todos os atos praticados no
            cumprimento deste instrumento "
64      if ( procuracao . validade . strftime ("%Y-%m-%d") =='9999-12-31'):
```

```
65          text3 += "por  tempo indeterminado ."
66      else :
67          text3 += " ,cessando  seus  efeitos   a  partir   de "  +
                procuracao . validade . strftime ( "% d  de  %B de  % Y " )+ " .  "
68      doc  = SimpleDocTemplate (" documents / "+ file_name ,
69                                rightMargin =72 ,  leftMargin =72 ,
70                                topMargin =72 , bottomMargin =18)
71      story   = []
72      styles   = getSampleStyleSheet ()
73      styles . add ( ParagraphStyle ( name = ' Justify ',
74                                      alignment = TA_JUSTIFY , leading =16) )
75      styles . add ( ParagraphStyle ( name = ' Centered ' ,
76                                      alignment = TA_CENTER ,leading =16) )
77      styles . add ( ParagraphStyle ( name = ' Norma ',
78                                      alignment = TA_LEFT ,leading =12) )
79      story . append (
80          Paragraph ( " < fontsize =24 >Procura ç ã o</ font > " ,
                styles [" title " ]) )
81      story . append ( Spacer (1 ,40) )
82      ptext   = ' < font  size =14 >% s </ font >% text1
83      story . append ( Paragraph ( ptext , styles [ " Justify " ]) )
84      story . append ( Spacer (1 ,12) )
85      ptext   = ' < font  size =14 >% s </ font >% text2
86      story . append ( Paragraph ( ptext , styles [ " Justify " ]) )
87      story . append ( Spacer (1 ,8) )
88      for   servico   in   procuracao . servicos :
89          ptext   = ' < font  size =12 >& nbsp%s </ font > % str ( servico . nome )
90          story . append ( Paragraph ( ptext , styles [" Norma "],
                bulletText = '-'))
91          story . append ( Spacer (1 ,8) )
92
93      ptext   = ' < font  size =14 >% s </ font >% text3
94      story . append ( Paragraph ( ptext , styles [ " Justify " ]) )
95      story . append ( Spacer (1 ,18) )
96      ptext   = ' < font  size =14 > '+ " Assinado   digitalmente   via   Portal   de
            Outorga  de  Servi ç os  do  Governo  Eletr ô nico   em "  +
            datetime . now ( tz = None ) . strftime ( " %d% Bde  %Y " )+ "   por   "  + \
97      procuracao . emissor . nome+ ' </ font > '
98      story . append ( Paragraph ( ptext , styles [ " Centered " ]) )
99      doc . link   = '< font   size =8 > Aautenticidade    deste   documento pode
            ser   verificada    em : <a href =" '  + procuracao . link_pra_download
            + '"   color =" blue " > '+ procuracao . link_pra_download   +
```

```
        ' </a > </ font > '
100     doc . build ( story ,  onFirstPage = footer )
101     utils . create_image ( procuracao . emissor . nome
            , procuracao . emissor . cpf )
102     sign ( user_certificate_path ,      user_pkey_path ,  Path (__file__) . parent
            /  ' assinatura_cidadao . png ',    297.5 ,  " documents / "+ file_name ,
            1)
103     sign ( poseg_certificate_path ,     poseg_pkey_path ,
            Path (__file__) . parent    /  ' static '    /  ' images ' /  " logo . png " ,0,
            " documents / "+ file_name ,  0)
```

APÊNDICE D – CÓDIGO-FONTE UTILS.PY

```
from PIL import Image , ImageDraw , ImageFont
import os
import re
import subprocess
import shlex

def check_cpf ( cpf ) :
    cpf = re . match (" (\ d {3}\.?\ d {3}\.?\ d{3}−?\d {2}) " ,cpf )
    if ( cpf is not None ) :
        cpf = re . sub ("[^0−9]" , " " , cpf [0])
        if ( len ( cpf ) ==11 and cpf . isdigit () ):
            i = 10
            somatorio = 0
            for algarismo in cpf [:9]:
                somatorio += int ( algarismo ) i
                i−=1
            resto = somatorio % 11
            if (( resto < 2 and cpf [9] =='0 ') or ( resto >=2and int (
                cpf [9]) ==11−resto )):
                i = 11
                somatorio = 0
                for algarismo in cpf [:10]:
                    somatorio += int ( algarismo ) i
                    i−=1
                resto = somatorio % 11
                if (( resto < 2 and cpf [10] =='0 ') or ( resto >=2and
                    int ( cpf [10]) ==11−resto )):
                    return cpf
    return False

def create_certificate ( cpf , nome ) :
    cmd = ' openssl req −x509 −newkey rsa :4096 −sha256 −utf8 −keyout '
        + shlex . quote ( cpf ) + ' \ textunderscore key . pem \
        −out '+ shlex . quote ( cpf ) + ' \ textunderscore cert . pem −days 1
            −nodes −addext keyUsage = digitalSignature \
        −subj "/ C = BR /O= gov . br / CN=' + shlex . quote ( nome ) + '" '
    subprocess . call ( cmd , shell = True ,cwd =" certificates ")

def format_cpf ( cpf ) :
```

```
36      return  cpf [:3]  + " ."  + cpf [3:6]  + " ."  + cpf [6:9]  + "-"  + cpf [9:]
37
38  def  create_image ( nome ,cpf ) :
39      texto  = " Outorgado via  POSGE por : "
40      cpf  = " CPF : " + format_cpf ( cpf )
41      nome = " Nome :"  + nome
42      size  = 52
43      fnt  = ImageFont . truetype ( font  = '/ usr / share / fonts / TTF / DejaVuSans .
            ttf ',  size  = size )
44
45      width_texto  = fnt . getsize ( texto ) [0]
46      width_cpf  = fnt . getsize ( cpf ) [0]
47      width_nome , height_nome  = fnt . getsize ( nome )
48
49      if ( width_texto  >= width_cpf  and  width_texto  >= width_nome ) :
50          width  = width_texto  + 20
51      elif ( width_cpf  >= width_nome ) :
52          width  = width_cpf  + 20
53      else :
54          width  = width_nome + 20
55      height  = height_nome 3  + 50
56      square_size  = ( width ,  height )
57      image = Image . new ( mode = " RGB " ,size  = square_size ,  color  = "
            white ")
58      draw  = ImageDraw . Draw ( image )
59      draw . rectangle ((0 ,0 , width -1, height -1),  fill  = " white " ,outline  ="
            black ")
60      draw . text ((10 ,10) ,  texto ,  font = fnt ,  fill =(0 ,0 ,0) )
61      draw . text ((10 ,20+ height_nome ) nome ,font = fnt ,  fill =(0 ,0 ,0) )
62      draw . text ((10 ,30+ height_nome 2 ) ,cpf ,  font = fnt ,  fill =(0 ,0 ,0) )
63      image . save (" assinatura_cidadao . png " )
64
65  def  calculate_signature_box_size ( path_to_image ,  x_begin ,  size  = 0.8) :
66      width ,  height  = Image . open ( path_to_image ) . size
67      proportion  = width  /  height
68      half_doc_width  = 297.5
69      width  = half_doc_width  size
70      height  = width  /  proportion
71      return  (x_begin +(( half_doc_width-width ) /2) ,(150- height ) /2 , x_begin
            +(( half_doc_width-width ) /2) + width ,(150- height ) /2+ height )
```

APÊNDICE E – CÓDIGO-FONTE VIEWS.PY

```
from flask import
from functools import wraps
import bd
import utils
from pessoa import usuario
from provedor import provedor
from procuracao import procuracao
from servico import servico
from datetime import datetime
import os
import time
import glob
import pdf

app = Flask (__name__)
app . secret_key = " wak_hxadgu_jgy_ixovzumxglog_buik "
app . config [ ' JSON_AS_ASCII '] = False

@app . route ( '/ ' , methods =[ ' GET '',POST ' ])
def login () :
    if ( not session . get ( ' logged_in ') ):
        if ( request . method == ' POST '):
            return do_the_login ()
        else :
            return show_the_login_form ()
    else :
        return show_inicio ()


def do_the_login () :
    cpf = utils . check_cpf ( request . form [ ' CPF ']) 
    if ( cpf ) :
        try :
            usuario = bd . login ( cpf , request . form [ ' pin ']) 
            if ( usuario ) :
                session [ ' nome '] = usuario . nome
                session [ ' CPF '] = usuario . cpf
                session [ ' cert_eh_curta '] = usuario . cert_eh_curta
                if ( session [ ' cert_eh_curta ']) :
```

```
40                          utils . create_certificate ( session [ ' CPF '] ,       session [
                                ' nome ' ])
41                      session [ ' logged_in ']    = True
42                      return   show_inicio ()
43                  else :
44                      return   cadastrar ()
45              except  Exception   as  e:
46                  flash ( str ( e) )
47                  return   show_the_login_form ()
48          else :
49              flash (" CPF  Inv á lido " )
50          return   show_the_login_form ()
51
52  @app . route ( '/ cadastro ' , methods =[ ' GET ', POST ' ])
53  def  cadastrar () :
54          if ( " login "   in   request . form ) :
55              session [ ' CPF ']  = utils . check_cpf ( request . form [ ' CPF ' ])
56              session [ ' nome '] = request . form [ ' login ']
57              bd . save_usuario ( usuario ( cpf = session [ ' CPF '] ,nome = session [ '
                    nome '] , pin = request . form [ ' pin ']) )
58              utils . create_certificate ( session [ ' CPF '] ,       session [ ' nome ' ])
59              session [ ' cert_eh_curta ']    = 1
60              session [ ' logged_in ']    = True
61              return   show_inicio ()
62          else :
63              return   render_template ( " cadastro . html " , cpf = request . form [ ' CPF
                    ' ])
64
65
66  @app . route ( '/ logout ')
67  def  logout () :
68          try :
69              if ( session [ ' cert_eh_curta ']) :
70                  os . remove ( ' certificates / '   + session [ ' CPF ']  + '_cert . pem ')
71                  os . remove ( ' certificates / '   + session [ ' CPF ']  + '_key . pem ')
72              session . clear ()
73              os . remove (" assinatura_cidadao . png " )
74          except :
75                  print ( " Sem arquivos   para  excluir " )
76                  return   redirect ( '/ ')
77          return   redirect ( '/ ')
78
```

```
79
80 def show_the_login_form () :
81     return render_template (" login . html " , funcao_ajuda = get_usuarios
           _cadastrados )
82
83 def get_procuracoes_pra_tabela ( tipo ) :
84     # Outorgadas = 0 , recebidas = 1
85     if ( tipo ==0) :
86         procuracoes = bd . get_procuracoes_outorgadas (
87             usuario ( cpf = session [ ' CPF ' ]) )
88     elif ( tipo ==1) :
89         procuracoes = bd . get_procuracoes_recebidas ( usuario ( cpf =
              session [ ' CPF ' ]) )
90         print ( procuracoes )
91     return procuracoes
92
93
94 def show_inicio () :
95     try :
96         procuracoes_outorgadas = get_procuracoes_pra_tabela (0)
97         procuracoes_recebidas = get_procuracoes_pra_tabela (1)
98         return render_template ( " inicio . html " , nome = session [ ' nome '] ,
              procuracoes_outorgadas = procuracoes_outorgadas , procuracoes
              _recebidas = procuracoes_recebidas , funcao = utils . format_cpf )
99     except KeyError as e:
100        return show_the_login_form ()
101
102 @app . route ( '/ emissao ' ,methods =[ ' GET '',POST ' ])
103 def show_emissao_procuracao () :
104     if ( request . method ==' GET ') :
105         try :
106             return render_template ( " emissao . html " ,nome = session [ ' nome
                  '] , funcao_provedores = criar_lista_provedores , \
107                   funcao_cpf = get_nome_por_cpf , funcao_ajuda = get_usuarios
                      _cadastrados )
108         except KeyError as e:
109             return show_the_login_form ()
110     elif ( request . method ==' POST ') :
111         if ( check_input ( request . form )) :
112             if ( 'data-fim ' not in request . form ) :
113                 data = datetime . strptime ( '9999-12-31', '%Y-%m-%d ')
114             else :
```

```
115                     data = datetime.strptime(request.form['data-fim'], '%
                            Y-%m-%d')
116                 bd.save_procuracao(procuracao(validade = data.strftime('%Y
                        -%m-%d'),emissor = session['CPF'],outorgado = utils.
                        check_cpf(request.form['CPF'])), session['provedor
                        _selecionado'],\
117                     request.form.getlist('servico'))
118                 outorga = bd.get_procuracao()
119                 outorga.link_pra_download = url_for('download_procuracao
                        _user', procuracao = outorga.id, _external = True)
120                 pdf.create_pdf(outorga)
121                 session['provedor_selecionado'] = None
122                 return show_inicio()
123             else:
124                 return render_template("emissao.html",nome = session['nome
                        '], funcao_provedores = criar_lista_provedores,\
125                     funcao_cpf = get_nome_por_cpf, funcao_ajuda = get_usuarios
                            _cadastrados)
126
127
128 def criar_lista_provedores():
129     provedores = bd.get_provedores()
130     tabela = '<ul class="collapsible" data-collapsible="accordion">
            '
131     for provedor in provedores:
132         tabela +='<li><div class="collapsible-header" data-value="'
                +\
133             str(provedor.id) + '"> ' + provedor.nome+\
134             '</div><divclass="collapsible-body"><span><ul>'
135         for servico in bd.get_servicos_por_provedor(provedor.id):
136             tabela +='<li><label><inputtype="checkbox"name="
                    servico" value="' +\
137                 str(servico.id) + '"/><span>+\
138                 servico.nome + '</span></label></li>'
139         tabela +='</ul></span></div></li>'
140     tabela +='</ul>'
141     return tabela
142
143
144 @app.route('/_get_cpf')
145 def get_nome_por_cpf():
146     cpf = utils.check_cpf(request.args['CPF'])
```

```
147     if ( cpf ) :
148         try :
149             usuario = bd . get_usuario_por_cpf ( cpf )
150             if ( usuario . nome == session [ ' nome ' ]) :
151                 return jsonify ( nome =" Voc ê mesmo " )
152             return jsonify ( nome = usuario . nome )
153         except Exception as e:
154             return jsonify ( nome = " Usu á rio n ão cadastrado ")
155     else :
156         return jsonify ( nome =" CPF inv á lido " )
157
158 def get_usuarios_cadastrados () :
159     usuarios = bd . get_usuarios_cadastrados ()
160     string_usuarios = " "
161     for usuario in usuarios :
162         string_usuarios += " nome= " + usuario . nome + " , CPF = " +
            usuario . cpf + " , PIN = " + usuario . pin + " <br > "
163     return string_usuarios
164
165
166 @app . route ( '/_set_provedor ')
167 def set_provedor () :
168     session [ ' provedor_selecionado '] = request . args [ ' provedor ']
169     return " provedor selecionado com sucesso "
170
171 @app . route ( '/_get_id_servicos ')
172 def get_id_servicos () :
173     provedor_id = session [ ' provedor_selecionado ']
174     lista_id =[]
175     for servico in bd . get_servicos_por_provedor ( provedor_id ) :
176         lista_id . append ( servico . id )
177     return jsonify ( lista_id )
178
179
180 def check_input ( lista ) :
181     for key , item in lista . items () :
182         if ( key == ' action ') :
183             pass
184         else :
185             if ( not len ( item ) ) :
186                 flash (" Preencha todos os campos " )
187                 return False
```

```
188                 elif ( key   == ' CPF ') :
189                     if ( not   utils . check_cpf ( item )) :
190                         flash ( " CPF  inv á lido " )
191                         return   False
192                     elif ( bd . get_usuario_por_cpf ( utils . check_cpf ( item ))     ==
                            None ) :
193                         flash ( " Usu á rio nã o cadastrado " )
194                         return   False
195                 elif ( key   == 'data-fim '):
196                     data_procuracao   = datetime . strptime ( item ,  '%Y-%m-%d ')
197                     data_atual   = datetime . now ()
198                     if ( data_atual   > data_procuracao ) :
199                         flash ( " Escolha   uma data   no  futuro " )
200                         return   False
201         return   True
202
203 @app . route ( '/ remover /< string : procuracao > ')
204 def  remove_procuracao ( procuracao ) :
205         try :
206             if ( session . get ( ' logged_in ') ):
207                 procuracao   = bd . get_procuracao ( procuracao )
208                 if ( procuracao . emissor . cpf   == session [ ' CPF ']  or  procuracao
                        . outorgado . cpf   == session [ ' CPF ' ]) :
209                     bd . anular_procuracao ( str ( procuracao . id ))
210                 else :
211                     raise   KeyError (" Voc ê nã o tem permiss ão  para  executar
                            essa  aç ão . ")
212             else :
213                 raise   KeyError ("É  necess á rio  estar  logado  no  sistema  para
                        executar  essa  aç ã o." )
214         except  Exception  as  e:
215             flash ( str ( e ))
216             return   show_inicio ()
217         return   show_inicio ()
218
219
220 @app . route ( '/ download / < string : procuracao > ')
221 def  download_procuracao_user ( procuracao ) :
222         try :
223             if ( session . get ( ' logged_in ') ):
224                 procuracao   = bd . get_procuracao ( procuracao )
225                 if ( procuracao . emissor . cpf   == session [ ' CPF ']  or  procuracao
```

```
                    . outorgado . cpf  == session [ ' CPF ' ]) :
226                     nome = " procuracao_"  + procuracao . emissor . nome+ \
227                     "_"  + procuracao . provedor  + "_"  + str ( procuracao . id )
                            + " . pdf "
228                     return  send_from_directory ( app . root_path  + " /
                            documents " , nome , as_attachment = True )
229                 else :
230                     raise  Exception (" Voc ê n ã o tem permiss ã o para  acessar
                            esse  conte ú do . " )
231             else :
232                 raise  Exception ( " É necess á rio  se  autenticar  ao  sistema
                        para  acessar  esse  conte ú do . " )
233         except  Exception  as  e :
234             flash ( str ( e ))
235             return  show_inicio ()
236
237 # API
238
239 def  verify_api_key ( view_function ) :
240         @wraps ( view_function )
241         def  decorated_functionargs, **kwargs):try:g.provedor =
                bd.login(*_api ( request . headers [ ' api_key '
242
243 ])
244                 return  view_functionargs, **kwargs)except Exception as e:return
                    jsonify("msg": str(e)), 401return decorated(*_function
245
246 @app . after_request
247 def  add_header ( request ) :
248         request . headers [" Cache-Control "]  = "no-cache , no-store ,  must-
                revalidate "
249         return  request
250
251 @app . route ( '/ api ', methods =[ ' GET '])
252 @verify_api_key
253 # curl -k -H  " api_key : xxx " -H" cpf :012345678910" -X GET " http ://
        localhost :5000/ api "
254 # Documento exemplo : curl -k -H  " api_key :
        cea09d3691d9a3f2ee08451e8b413d59a0599f65dff17fca " -H " cpf
        :63567303406" -X  GET " http :// localhost :5000/ api "
255 def  get_servicos () :
256         if ( request . headers [ ' cpf ' ]) :
```

```
257             if ( utils . check_cpf ( request . headers [ ' cpf ']) ) :
258                 cpf  = request . headers [ ' cpf ']
259                 try :
260                     usuario  = bd . get_usuario_por_cpf ( cpf )
261                 except  Exception  as  e :
262                     return  jsonify ({ " msg " : " Usu á rio nã o cadastrado . " }) ,
                            406
263             else :
264                 return  jsonify ({ " msg " : " CPF inv á lido . " }) , 400
265         else :
266             return  jsonify ({ " msg ":  " É necess á rio  informar  o  CPF  do
                    receptor  do  direito ." }) ,  400
267         procuracoes  = []
268         for  procuracao  in  bd . get_procuracoes_recebidas ( usuario ,  g.
                provedor . id ):
269             servicos  = []
270             for  servico  in  bd . get_servicos_liberados ( procuracao . id ) :
271                 servicos . append ({
272                     'id_servico '      :  servico . id ,
273                     ' nome_servico ':  servico . nome
274                     })
275
276             procuracoes . append ({
277                 'id_procuracao ':  procuracao . id ,
278                 ' link_pra_download '  :  url_for ( ' download_procuracao_api ' ,
                        procuracao  = procuracao .id , _external = True ) ,
279                 ' validade '  :  procuracao . validade ,
280                 ' provedor '  :  procuracao . provedor ,
281                 ' nome_emissor '  :  procuracao . emissor . nome ,
282                 ' cpf_emissor '  :  procuracao . emissor . cpf ,
283                 ' nome_outorgado ':  procuracao . outorgado . nome ,
284                 ' cpf_outorgado '  :  procuracao . outorgado . cpf ,
285                 ' servicos '  :  servicos
286                 })
287         return  jsonify ( procuracoes = procuracoes )
288
289 @app . route ( '/ api / download / < string : procuracao >', methods =[ ' GET ' ])
290 @verify_api_key
291 # Documento exemplo : curl -k -H  " api_key :
        cea09d3691d9a3f2ee08451e8b413d59a0599f65dff17fca " -X GET " http ://
        localhost :5000/ api / download /1" --output  documento . pdf
292 def  download_procuracao_api ( procuracao ) :
```

```
293     try :
294         procuracao = bd . get_procuracao ( procuracao )
295         if ( str ( g . provedor . nome )==str ( procuracao . provedor ) ) :
296             nome= " procuracao_" + procuracao . emissor . nome+ \
297             "_" + procuracao . provedor + "_" + str ( procuracao . id ) + " .
                    pdf "
298             return send_from_directory ( app . root_path + "/ documents " ,
                    nome ,as_attachment = True ,mimetype = ' application / pdf ')
299         else :
300             return jsonify ({ " msg " : " Voc ên ão tem permiss ã o para
                    acessar esse conte ú do . " }) ,403
301     except Exception as e :
302         return jsonify ({ " msg ": " O identificador n ã opertence a
                nenhumaprocura ção ou a mesmaj á foi extinguida ." }) , 410
303
304 os . environ [ ' TZ '] = ' EST +05 EDT , M4 .1.0 , M10 .5.0 '
305 time . tzset ()
306 if __name__ =='__main__':
307     app . run ( host = ' 0.0.0.0 ' ,port = ' 5000 ', debug = True )
```

APÊNDICE F – CÓDIGO-FONTE BD.PY

```
1  import  mysql . connector
2  from  pessoa  import  usuario
3  from  provedor  import  provedor
4  from  procuracao  import  procuracao
5  from  servico  import  servico
6  from  datetime  import  datetime
7  from  passlib . hash  import  pbkdf2_sha256
8
9
10 db  = mysql . connector . connect (
11       host = " localhost " ,
12       user = " root " ,
13       passwd =" 1234 " ,
14       database = ' Posge '
15 )
16 cursor  = db . cursor ()
17
18
19 def  login ( cpf ,  pin ) :
20       cursor . execute ( " SELECT  FROMusuario  WHEREcpf  = '"  + cpf  + " '" )
21       user  = cursor . fetchone ()
22       if ( user  is  not  None ) :
23            if ( pbkdf2_sha256 . verify ( pin ,  user [1]) ) :
24                 return  usuario ( cpf = user [0] , nome = user [2] ,
                         cert_eh_curta = user [3])
25            else :
26                 raise  Exception ( " Senha Incorreta ! " )
27       else :
28            return  False
29
30 def  login_api ( api_key ) :
31       cursor . execute ( " SELECT  FROMprovedor "  )
32       result  = cursor . fetchall ()
33       for  provedor_id ,  nome , api_key_db  in  result :
34            if ( pbkdf2_sha256 . verify ( api_key ,  api_key_db )) :
35                 return  provedor ( id = provedor_id , nome = nome )
36       raise  Exception (" Chave  de  API  inv á lida . " )
37
38
```

```
39 def get_usuario ( nome ):
40     cursor . execute ( " SELECT  FROM usuario  WHERE nome = '"  + nome +
           " '" )
41     result  = cursor . fetchone ()
42     return  usuario ( cpf = result [0] ,  nome = result [2])
43     # 0 = cpf ,  1 = pin ,  2 = nome
44
45 def get_usuarios_cadastrados () :
46     cursor . execute ( " SELECT  FROM usuario " )
47     result  = cursor . fetchall ()
48     usuarios  = []
49     for  cpf ,  pin ,  nome , cert_eh_curta  in  result :
50         usuarios . append ( usuario ( cpf = cpf , pin = pin , nome = nome ,
               cert_eh_curta  = cert_eh_curta ))
51     return  usuarios
52
53 def get_usuario_por_cpf ( cpf ):
54     cursor . execute ( " SELECT  FROM usuario  WHERE cpf  = '"  + cpf  + " '" )
55     result  = cursor . fetchone ()
56     if ( result  != None ) :
57         return  usuario ( cpf = result [0] ,  nome = result [2])
58     else :
59         raise  Exception ( " Usu á rio n ã o cadastrado " )
60
61
62 def save_usuario ( usuario ) :
63     cursor . execute ( " INSERT INTO usuario  ( cpf ,  pin ,  nome ,
           cert_eh_curta )  VALUES ( ' "  +
64                     usuario . cpf  + " ',  '"  +
                           pbkdf2_sha256 . hash ( usuario . pin ) + " ',  \" "  +
                           usuario . nome + " \" ,  1) " )
65     db . commit ()
66
67
68 def get_provedores () :
69     cursor . execute ( " SELECT id ,  nome FROM provedor " )
70     result  = cursor . fetchall ()
71     provedores  = []
72     for  id_prov ,  nome in  result :
73         provedores . append ( provedor ( id = id_prov , nome = nome ) )
74     return  provedores
75
```

```
76
77 def get_servicos_por_provedor ( provedor_id ) :
78     cursor . execute (
79         " SELECT id , nome FROM servico WHERE provedor = " +
              str ( provedor_id ))
80     result = cursor . fetchall ()
81     servicos = []
82     for id_serv , nome in result :
83         servicos . append ( servico ( id = id_serv , nome = nome ))
84     return servicos
85
86
87 def get_provedor_por_id_servico ( id_serv ) :
88     cursor . execute ( " SELECT provedor FROM servico WHERE id = " +
          str ( id_serv ) )
89     return cursor . fetchone () [0]
90
91
92 def get_procuracoes_outorgadas ( user , u = " u1 " , provedor = None ) :
93     select = " SELECT pa . id 'id_procuracao ', pa . validade ' validade ',
          " +\
94     " pa . ehValido ' eh_valido ', u1 . nome ' nome_emissor ', u1 . cpf
          ' cpf_emissor ', " +\
95     " u2 . nome ' nome_outorgado ', u2 . cpf ' cpf_outorgado ', po . nome
          ' nome_provedor '" +\
96     " FROM procuracao as pa JOIN usuario as u1 ON pa . emissor = u1 . cpf
          JOIN usuario " +\
97     " as u2 on pa . outorgado = u2 . cpf JOIN provedor as po ON
          pa . provedor = po . id WHERE " \
98     + u + " . cpf = " + user . cpf
99     if ( provedor != None ) :
100         select += " AND po . id = " + str ( provedor )
101     select += " ORDER BY validade ASC "
102     cursor . execute ( select )
103     result = cursor . fetchall ()
104     procuracoes = []
105     for id_procuracao , validade , eh_valido , nome_emissor ,
          cpf_emissor , nome_outorgado , cpf_outorgado , nome_provedor in
          result :
106         if ( eh_valido and validade > datetime . now ( tz = None ) . date () ) :
107             pessoa_emissora = usuario ( cpf = cpf_emissor ,
                    nome = nome_emissor )
```

```
108                 pessoa_outorgada  = usuario ( cpf = cpf_outorgado ,
                        nome = nome_outorgado )
109                 procuracoes . append ( procuracao ( id = id_procuracao ,
                        validade = validade ,
110                                                     emissor = pessoa_emissora ,
                                                            outorgado = pessoa_outorgada ,
                                                            provedor = nome_provedor ))
111         return  procuracoes
112
113
114 def  get_procuracoes_recebidas ( usuario ,    provedor = None ) :
115         return  get_procuracoes_outorgadas ( usuario ,    " u2 " , provedor )
116
117
118 def  save_procuracao ( procuracao ,  provedor ,  servicos ) :
119         cursor . execute ( ' INSERT INTO procuracao ( validade ,  emissor ,
                outorgado ,  provedor )  VALUES (" '  +
120                             procuracao . validade  + '" ,  \ ' '  +
                                    procuracao . emissor  + " ',  '"  +
                                    procuracao . outorgado  + " ',  '"  + provedor  +
                                    " ') " )
121         id_procuracao  = cursor . lastrowid
122         for  serv  in  servicos :
123             cursor . execute (" INSERT INTO atributos ( procuracao ,  servico )
                    VALUES ("  +
124                                 str ( id_procuracao )  + " ,  "  + str ( serv )  + " ) " )
125         db . commit ()
126
127
128 def  anular_procuracao ( procuracao ) :
129         cursor . execute (
130             " UPDATEprocuracao  SET ehValido  = 0 WHEREid  = "  +
                    procuracao )
131         db . commit ()
132
133
134 def  get_servicos_liberados ( procuracao ) :
135         cursor . execute ( " SELECTs .id ,  s . nomeFROMatributos  as a JOIN
                servico  "  +\
136             " as s ON s. id  = a. servico  JOIN  procuracao  as p ON
                    a. procuracao  = p. id " +\
137             "  WHEREp . id  = "  + str ( procuracao ) )
```

```
138         result    = cursor . fetchall ()
139         servicos    = []
140         for   id ,   nome in   result :
141             servicos . append ( servico ( id =id ,  nome = nome ) )
142         return    servicos
143
144
145 def  get_procuracao ( procuracao_id    = " LAST_INSERT_ID () " ) :
146         cursor . execute ( " SELECT pa . id  ' id_procuracao ',     pa . validade
                ' validade ',"  +\
147             " pa . ehValido  ' eh_valido ',     u1 . nome ' nome_emissor ',  u1 . cpf
                    ' cpf_emissor ',     u2 . nome ' nome_outorgado '"  +\
148             " ,  u2 . cpf  ' cpf_outorgado ',     po . nome ' nome_provedor  ' FROM
                    procuracao   as  pa "  +\
149             "  JOIN   usuario   as  u1  ON pa . emissor = u1 . cpf JOIN   usuario   as
                    u2  on  pa . outorgado "  +\
150             "  = u2 . cpf JOIN   provedor   as  po  ON pa . provedor = po . id  WHERE
                    pa . id  = "   + procuracao_id )
151         result    = cursor . fetchone ()
152         if ( result    !=  None ) :
153             if ( result [2]) :
154                 pessoa_emissora   = usuario ( cpf = result [4] ,   nome = result [3])
155                 pessoa_outorgada   = usuario ( cpf = result [6] ,   nome = result [5])
156                 proc   = procuracao ( id = result [0] ,  validade = result [1] ,
                        emissor = pessoa_emissora ,
157                                     outorgado = pessoa_outorgada ,
                                        provedor = result [7] ,
                                        servicos = get_servicos_liberados ( result [0]) )
158             else :
159                 raise    Exception ( " A procura çã o n ão é  v á lida  mais ." )
160         else :
161             raise    Exception ( " O identificador      nã o pertence   a  nenhuma
                    procura ç ã o." )
162         return    proc
```

APÊNDICE G – TEMPLATES

Inicio.html breaklines

```
<! DOCTYPE html >
< html >
< head >
    < meta charset =" UTF-8" >
    <!--Import Google Icon Font-->
    < link href = " https :// fonts . googleapis . com / icon ? family = Material +
        Icons " rel = " stylesheet " >
    <!--Import materialize . css-->
    < link type = " text / css " rel = " stylesheet " href = " ../ static / css /
        materialize . min . css " media =" screen , projection "/ >
    < link type = " text / css " rel = " stylesheet " href = " ../ static / css / custom
        . css " media = " screen , projection " / >
    <!--Let browser know website is optimized for mobile-->
    < meta name = " viewport " content =" width = device-width , initial-scale
        =1.0 " / >
    < style type = " text / css " >
        # emitir { display : flex ; justify-content : center ; }
    </ style >
    < link href = " https :// fonts . googleapis . com / icon ? family = Material +
        Icons " rel = " stylesheet " >
    < title > Home - POSGE </ title >
    < script type = " text / javascript " >
        function confirmar ( id_procuracao ) {
            if ( confirm ( " Voc êtem certeza ? " ) ) {
                window . location . href = "/ remover /" + id_procuracao ;
                return true ;
            } else {
                return false ;
            }
        }
    </ script >

</ head >
< body >

    < div class = " col s12 m12 l12 " >
        < nav >
            < div class =" nav-wrapper " >
```

```
34                  < div class =" brand-logo " ><a href = "/" > < img src = " ../
                        static / images / logo . png " > </ a ></ div >
35                  < ul id =" nav-mobile "class = " right  hide-on-med-and-down
                        " >
36                      <li ><a href = " / "name = " name " > <div class = " chip " > <
                            img  src =" ../ static / images / pessoa . jpg "  alt = "
                            Usuario  logado " >{{  nome }}  </ div > </ a > </ li >
37                      <li  id = " sair " > <a href = " / logout " > <i class ="
                            material-icons " > exit_to_app </i ></a ></ li >
38                  </ ul >
39              </ div >
40          </ nav >
41      </ div >
42      < div class = " container " >
43          < div class =" row " >
44              {% for  message in  get_flashed_messages () %}
45              < div class =" card-panel  red  lighten -3  center " > <b > {{ message
                    }} </ b > </ div >
46              {% endfor  %}
47              < div class = " row " > < div class = " col s12 " > </ div > </ div >
48              < div class = " row " > < div class = " col s5 " >< h5 align = " center " >
                    Procura ç õ es Outorgadas </ h5 > </ div >
49              < div class = " col s2 " >
50                  < div class = " divider  center " > </ div >
51              </ div >
52              < div class = " col s5 " > <h5 align = " center " > Procura ç õ es
                    Recebidas </ h5 > </ div >
53          </ div >
54          < div class =" row " >
55              < div class = " col s5 " >
56                  < div class = " section  left " >
57                      {% if  procuracoes_outorgadas  is  defined  and
                            procuracoes_outorgadas [0]  is  defined  %}
58                      < table >< thead > < tr > <th > Nome </ th > < th > CPF </ th > <
                            Provedor </ th >
59                      <th > Data de  expira ç ã o </ th > <th >A ç õ es </ th ></ tr > </
                            thead > < tbody >
60                          {% for  procuracao  in  procuracoes_outorgadas
                                %}
61                          < tr > <td >{{ procuracao . outorgado . nome }} </ td > <
                                td > {{ funcao ( procuracao . outorgado . cpf ) }}
62                          </ td >< td >{{ procuracao . provedor }} </ td > <td >
```

```
63                          {% if procuracao.validade.strftime("%Y/%m/%d")=="9999/12/31" %}
64                              Tempo Indeterminado
65                          {% else %}
66                              {{ procuracao.validade.strftime("%d/%m/%Y")}}
67                          {% endif %}
68                      </td><td style='white-space: nowrap'><a class="waves-effect waves-light btn-floating blue" href="/download/{{
69                          procuracao.id }}" style="padding:14;"><i class="material-icons">description</i></a>
70                          <button class="waves-effect waves-light btn-floating red" onclick="confirmar({{ procuracao.id }})"><i class="material-icons">clear</i></button>
71                          {% endfor %}
72                          </td></tr></tbody></table>
73                          {% else %}
74                          <div class="center-block"><p>Você não tem procurações outorgadas válidas no momento ou nunca emitiu.</p></div>
75                          {% endif %}
76                      </div>
77                  </div>
78                  <div class="col s2">
79                      <div class="divider center"></div>
80                  </div>
81
82                  <div class="col s5">
83                      <div class="section right">
84                          {% if procuracoes_recebidas is defined and
85                          procuracoes_recebidas[0] is defined %}
86                  <table><thead><tr><th>Nome</th><th>CPF</th><th>Provedor</th><th>Data de expiração</th><th>Ações</th></tr></thead><tbody>
87                          {% for procuracao in procuracoes_recebidas %}
88                          <tr><td>{{ procuracao.outorgado.nome }}</td><td>{{ funcao(procuracao.outorgado.cpf) }}</td>
```

```
89                          < td > {{ procuracao . provedor }} </ td > < td >
90                              {% if   procuracao . validade . strftime ( " %Y /% m
                                    /% d " )==" 9999/12/31 " %}
91                                  Tempo Indeterminado
92                              {% else   %}
93                                  {{ procuracao . validade . strftime ( "% d /% m
                                        /% Y " )}}
94                              {% endif   %}
95                          </ td >< td style =' white-space :  nowrap ' > <a class
                                =" waves-effect  waves-light   btn-floating
                                blue "  href = " / download /{{
96                                  procuracao . id }} " style = " padding :14; " ><i
                                    class =" material-icons " > description </i >
                                    </a >
97                                  < button class =" waves-effect  waves-light
                                    btn-floating    red "  onclick =" confirmar
                                    ({{ procuracao . id }}) " > <i class = "
                                    material-icons " > clear </i ></ button >
98                                  {% endfor   %}
99                                  </ td > </ tr ></ tbody ></ table >
100                                 {% else   %}
101                                 < div class =" center-block " ><p > Você não tem
                                        procura ções outorgadas  válidas  no
                                        momento ou  nunca  emitiu . </p > </ div >
102                                 {% endif   %}                               </ div >
103                         </ div >
104                 </ div >
105                 < div class = " row " >
106                         < div class =" col  s5 " >
107                             <p class =" center-align " >
108                                 <a class = " btn waves-effect   waves-light "
                                        href = " / emissao 'name = " emitir " > Emitir
                                        procura ção
109                                 </a >
110                             </ p >
111                         </ div >
112
113                 </ div >
114             </ div >
115             <!-- Compiled  and  minified    JavaScript -->
116             < script   type =" text / javascript "   src = " ../ static / js /
                    materialize . min . js " ></ script >
```

```
117             </ body >
118             </ html >
```

login.html

```
1  <! DOCTYPE html >
2  <! DOCTYPE html >
3  < html style = " display :  table ; margin :  auto ; " >
4  < head >
5     < meta charset =" UTF-8" >
6     <!--Import Google  Icon   Font-->
7     < link
          href = " https :// fonts . googleapis . com / icon ? family = Material + Icons "
          rel =" stylesheet " >
8     <!--Import materialize . css-->
9     < link  type =" text / css " rel = " stylesheet "
          href = " ../ static / css / materialize . min . css "
          media = " screen , projection " / >
10    < link  type =" text / css " rel = " stylesheet "
          href = " ../ static / css / custom . css "   media = " screen , projection "/ >
11    < script >
12    function   help ()  {
13       var  x  = document . getElementById (" help " );
14       if   (x. style . display    ===" none " ){
15          x . style . display   = " block " ;
16       }  else  {
17          x . style . display   = " none " ;
18       }
19    }
20    </ script >
21    < title > Login - POSGE </ title >
22    <!--Let browser  know website  is  optimized  for  mobile-->
23    < meta name =" viewport " content =" width = device-width ,
          initial-scale =1.0 "/ >
24 </ head >
25 < body class = " login " >
26    < div class =" container " >
27       < div class = " col s6  offset-s3   z-depth -4 card-panel center
             valign-wrapper " >
28          < form class =" login-form "  method = " post " >
29             < div class =" row " >
30                {% for   message in   get_flashed_messages () %}
31             < div class =" card-panel  red
```

```
            lighten -3" ><b >{{ message }} </b ></ div >
32      {% endfor  %}
33          < div class =" input-field  col  s8  offset-s2  center " >
34            < h4class = " center " > Portalde Outorga de Servi ç os do
                  Governo Eletr ô nico ( POSGE ) </ h4 >
35            <p class = " center " >Entre com seu CPF e seu PIN para
                  utilizar o sistema . </p >
36          </ div >
37          < div class =" input-field  col  s6  offset-s3  center " >
38            < input id = " CPF "name = " CPFtype = " text "class = " " >
39            < label for = " CPF " > CPF </ label >
40          </ div >
41          < div class =" input-field  col  s6  offset-s3  center " >
42            < input id = " pin "name = " pintype = " password "class ="" >
43            < label for = " pin " > PIN </ label >
44          </ div >
45        </ div >
46        < button class = " btn waves-effect  waves-light "  type =" submit "
              name = " action " > Login
47        </ button >
48      </ form >
49    </ div >
50    < div class = " right  valign-wrapper " >
51      <a onclick =" help () " ><i class =" small
            material-icons " > help_outline </i ></a >
52    </ div > <br >< br >
53    < div id =" help " class =" col  s6  offset-s3  z-depth -4 card-panel
          center  valign-wrapper "  style = " display : none " >
54      < div class =" input-field  col  s8  offset-s2  center " >
55        <p style =" text-align : center " >{{ funcao_ajuda () | safe }} </p >
56      </ div >
57    </ div >
58  </ div >
59  <!-- Compiled and minified JavaScript -->
60  < script  type =" text / javascript "
        src =" ../ static / js / materialize . min . js " > </ script >
61 </ body >
62 </ html >
```

cadastro.html

```
1 <! DOCTYPEhtml >
2 < html style = " display :  table ; margin :  auto ; " >
```

```
3  < head >
4    < metacharset =" UTF-8" >
5         <!--Import Google  Icon   Font-->
6         < link
              href = " https :// fonts . googleapis . com / icon ? family = Material + Icons "
              rel = " stylesheet " >
7         <!--Import materialize . css-->
8         < link  type =" text / css "rel = " stylesheet "
              href = " ../ static / css / materialize . min . css "
              media = " screen , projection " / >
9         < link  type =" text / css "rel = " stylesheet "
              href = " ../ static / css / custom . css "   media =" screen , projection " / >
10
11   <!--Let browser  know website  is  optimized  for  mobile-->
12   < metaname =" viewport "content =" width = device-width ,
         initial-scale =1.0 "/ >
13      < title > Cadastro - POSGE </ title >
14
15 </ head >
16 < bodyclass = " login " >
17   < div class =" container " >
18   < div class =" col  s6  offset-s3   z-depth -4 card-panel center
         valign-wrapper " >
19      < formclass =" login-form "  method = " post " >
20
21      < div class = " row " >
22        < div class =" input-field   col  s8  offset-s2   center " >
23          <h4 class =" center " > Portal de  Outorga  de  Servi ç os  do  Governo
                Eletr ô nico  ( POSGE ) </ h4 >
24          <p class = " center " > Voc mê  o est á  cadastrado  no  sistema .  Entre
                com seu  nome e  confirme  o  seu  PIN  para  se  cadastrar </ p >
25        </ div >
26        < div class =" input-field   col  s6  offset-s3   center " >
27            < input  name =" login "type =" text " >
28            < label  for =" login "  class =" center-align " > Nome </ label >
29          </ div >
30
31        < div class =" input-field   col  s6  offset-s3   center " >
32            < input  value =" {{cpf  }} " readonly = "" name =" CPF "
                  type = " text " >
33            < label  for =" CPF "class =" center-align " > CPF </ label >
34          </ div >
```

```
35
36              < div class =" input-field    col   s6  offset-s3    center " >
37                < input  name =" pin " type = " password " class = " validate " >
38                < label  for =" pin " class =" center-align " > PIN </ label >
39              </ div >
40                    </ div >
41           < button class = " btn waves-effect   waves-light "   type =" submit "
                 name = " action " > Login
42     </ button >
43 </ form >
44 </ div >
45 </ div >
46     <!-- Compiled  and  minified    JavaScript -->
47           < script   type = " text / javascript "
                src =" ../ static / js / materialize . min . js " ></ script >
48       </ body >
49 </ html >
```

emissao.html

```
1  <! DOCTYPE html >
2  < html >
3  < head >
4        < meta charset =" UTF-8" >
5        <!--Import Google  Icon   Font-->
6        < link
             href = " https :// fonts . googleapis . com / icon ? family = Material + Icons "
             rel = " stylesheet " >
7        <!--Import materialize . css-->
8        < link  type = " text / css " rel = " stylesheet "
             href = " ../ static / css / materialize . min . css "
             media =" screen , projection " / >
9        < link  type = " text / css " rel = " stylesheet "
             href = " ../ static / css / custom . css "   media =" screen , projection " / >
10       <!--Let browser  know website   is   optimized   for   mobile-->
11       < meta name = " viewport " content =" width = device-width ,
             initial-scale =1.0 "/ >
12       < style  type = " text / css " >
13            # emitir  {  display :   flex ;   justify-content :    center ;   }
14       </ style >
15       < script   type   = " text / javascript "
16       src   = " https :// code . jquery . com / jquery -2.1.1. min . js " ></ script >
17       < script   type = " text / javascript " >
```

```
18          $( document ). ready ( function () {
19              $ ( '. collapsible ') . collapsible () ;
20
21              $ ( '# CPF ') . change ( function (){
22                  $ . getJSON ( $SCRIPT_ROOT + '/_get_cpf ',     {
23                      CPF : $ ( '# CPF ') . val ()
24                  } ,  function ( data ) {
25                      $ ( '# nome-outorgado ') . val ( data . nome );
26                  }) ;
27                  return   false ;
28              }) ;
29
30              $ ( " # eh_indeterminado " ). click ( function () {
31                  if   ( this . checked )  {
32                      $ ("# data " ). hide ()
33                      $ ("# check ") . removeClass ( " s6 " ) . addClass ( " s12 " )
34                      $("# data-fim "). prop ( " disabled " ,  true   ) ;
35                  }
36                  else  {
37                      $ ("# data " ). show ()
38                      $ ("# check ") . removeClass ( " s12 " ) . addClass ( " s6 " )
39                      $("# data-fim "). prop ( " disabled " ,  false   ) ;
40                  }
41              }) ;
42
43              $ ( ' input [ name = servico ] ') . change (
44                  function () {
45                      list    = []
46                      $ . getJSON ( $SCRIPT_ROOT + '/_get_id_servicos ' ,      {
47                      } ,  function ( data ) {
48                          list    = data ;
49                          console . log ($ (' input [ name = servico ] ') )
50                          $ ( ' input [ name = servico ] ') . each ( function()  {
51                              if
                                    (! list . includes ( Number ( $ ( this ) . val () )))
                                    {
52                                  $ ( this ) . prop ( ' checked ', false );
53                              }
54                          }) ;
55                      }) ;
56                  }) ;
57          }) ;
```

```
58      </ script >
59
60      < script  type = " text / javascript " > $SCRIPT_ROOT = {{
            request . script_root | tojson | safe     }}; </ script >
61
62      < script  type = " text / javascript " >
63
64      </ script >
65
66      < script  type = " text / javascript " >
67          $( function ()   {
68              $ ( '. collapsible-header ') . on (" click " , function ( event )    {
69                  $ . ajax ({
70                      url :   $SCRIPT_ROOT+ '/_set_provedor ' ,
71                      data :  { provedor :   $( this ) . data ( ' value ') }
72                  }) ;
73                  console . log ( $ ( this ) . data (' value ') )
74              }) ;
75          }) ;
76      </ script >
77
78      < script  type = " text / javascript " >
79          function   help ()  {
80              var  x  = document . getElementById ( " help " );
81              if   ( x. style . display   ===" none " ){
82                  x . style . display   = " block ";
83              }  else   {
84                  x . style . display   = " none " ;
85              }
86          } </ script >
87          < title > Emiss ão - POSGE </ title >
88      </ head >
89      < body >
90
91          < div class =" col  s12  m12 l12 " >
92              < nav >
93                  < div class =" nav-wrapper " >
94                      < div class =" brand-logo " ><a href =" / " >< img
                            src = " ../ static / images / logo . png " > </ a > </ div >
95                      <ul  id =" nav-mobile "class =" right
                            hide-on-med-and-down " >
96                          < li > <a href ="/ "  name =" name " > < div
```

```
                                class =" chip " >< img
                                src =" ../ static / images / pessoa . jpg "
                                alt =" Usuario  logado " >{{  nome }}
                                </ div ></ a > </ li >
97                          < li  id =" sair " ><a href = " / logout " > <i
                                class =" material-icons " > exit_to_app </i ></a ></ li >
98                      </ ul >
99                  </ div >
100             </ nav >
101         </ div >
102         < div class =" container " >
103             < div class = " row " > < div class = " col s12 " > </ div > </ div >
104             {% for  message in  get_flashed_messages () %}
105             < div class =" card-panel  red  lighten -3
                    center " > <b >{{ message }} </ b ></ div >
106             {% endfor  %}
107             < form method =" post " class =" col  s12 " >
108                 < div class = " row " >
109                     < div class =" col  s8 " >
110                         {{  funcao_provedores () | safe  }}
111                     </ div >
112                     < div class =" col  s1 " >
113                         < div class =" right  valign-wrapper " >
114                             <a onclick = " help () " ><i class = " small
                                    material-icons " > help_outline </i ></a >
115                         </ div >
116                     </ div >
117
118                     < div class =" col  s3 " >
119                         < div class =" row " >
120                             < div class =" input-field  col  s12 " >
121                                 < input  placeholder = " " id =" CPF "
                                        name = " CPF " type = " text "
                                        class = " validate " >
122                                 < label  for = " CPF " > CPF
                                        outorgado </ label >
123                             </ div >
124                             < div class = " col s12 " >
125                                 < div class = " row " >
126                                     < div class =" input-field  col  s6 "
                                            id = " data " >
127                                         < input  value = " "
```

```
                                                placeholder = " "
                                                type =" date "
                                                name =" data-fim "
                                                id =" data-fim " >
128                                            < label  for =" data-fim " > Data
                                                de  expira ç ã o </ label >
129                                         </ div >
130                                         < div class =" input-field    col   s6 "
                                                id = " check " >
131                                            < label >
132                                                < input
                                                    id =" eh_indeterminado "
                                                    type =" checkbox "/ >
133                                                < span > Indeterminado </ span >
134                                            </ label >
135                                         </ div >
136                                         < div class =" input-field    col   s6 "
                                                id = " hid " >
137                                            < br >
138                                         </ div >
139                                      </ div >
140                                   </ div >
141                                   < div class =" input-field    col   s12 " >
142                                      < input disabled    value = ""
                                          placeholder = ""  type = " text "
                                          id =" nome-outorgado " >
143                                      < label  for =" nome-outorgado " > Nome
                                          outorgado </ label >
144                                   </ div >
145                                </ div >
146                                <p class = " center " >< button class =" btn
                                    waves-effect   waves-light   center "
                                    type = " submit " name = " action " > Emitir
147                                </ button ></p >
148                             </ div >
149                          </ div >
150                       </ form >
151                    </ div >
152                    < div id = " help " class =" input-field    col   s8  offset-s2    center "
                          style = " display : none " >
153                       <p style =" text-align : center " >{{ funcao_ajuda () | safe }} </p >
154                    </ div >
```

```
155             <!-- Compiled and minified JavaScript -->
156             <script type="text/javascript"
                    src="../static/js/materialize.min.js"></script>
157         </body>
158         </html>
```

APÊNDICE H – CLASSES-MODELO

usuario.py

```
class   usuario :
    def  __init__( self ,        kargs ) :
        self . cpf   = kargs . get ( ' cpf ')
        self . nome = kargs . get ( ' nome ')
        self . pin   = kargs . get ( ' pin ')
        self . cert_eh_curta     = kargs . get ( ' cert_eh_curta ')
```

procuracao.py

```
class   procuracao :
    def  __init__( self ,        kargs ) :
        self . id   = kargs . get ( 'id ')
        self . validade    = kargs . get ( ' validade ')
        self . emissor   = kargs . get ( ' emissor ')
        self . outorgado   = kargs . get ( ' outorgado ')
        self . provedor   = kargs . get ( ' provedor ')
        self . servicos    = kargs . get ( ' servicos ')
        self . link_pra_download     = kargs . get ( ' link_pra_download ')
```

provedor.py

```
class   provedor :
    def  __init__( self ,        kargs ) :
        self . id   = kargs . get ( 'id ')
        self . nome = kargs . get ( ' nome ')
```

servico.py

```
class   servico :
    def  __init__( self ,        kargs ) :
        self . id   = kargs . get ( 'id ')
        self . nome = kargs . get ( ' nome ')
```

APÊNDICE I – ARQUIVO CUSTOM.CSS

```
body . login  {
   background−color :  #007 aff ;

}

. login   {
   height :  100% ;
   width :  100% ;
   min−height :  100 vh ;
   display :   flex ;
   flex−wrap :  wrap ;
   justify−content :    center ;
   align−items :   center ;
}

. nav−wrapper {
      background−color :  #007 aff ;
}

. chip  {
      background−color :# FFF ;
}

. brand−logo  {
      display :   block ;
      float :   left ;
      margin :  2 px 2 px ;

}

. divider {
      position : absolute ;
      left : 50% ;
      top : 10% ;
      bottom : 10% ;
      border−left :1 px   solid   white ;
}


```

```
40 /  label  focus  color  /
41  .input-field  input:focus  + label  {
42    color:  #007aff !important;
43  }
44  /  label  underline  focus  color  /
45  .row  .input-field  input:focus  {
46    border-bottom:  1px solid  #007aff !important;
47    box-shadow:0  1px 0  0  #007aff !important
48  }
49
50  .btn,  .btn-large,  .btn-small  {
51      background-color:  #007aff;
52  }
53
54  .btn:hover, .btn-large:hover,  .btn-small:hover, .btn-floating:hover
        {
55      background-color:  #0268d6;
56  }
57
58 nav  ul  #sair:hover {
59     background-color:  red;
60 }
```

APÊNDICE J – ARQUIVOS SQL

create.sql

```
1  CREATE DATABASE Posge;
2  USE Posge;
3  CREATE TABLE usuario(cpf CHAR(11), pin VARCHAR(255) NOT NULL, nome
   VARCHAR(255) NOT NULL, cert_eh_curta TINYINT(1), PRIMARY KEY(cpf));
4  CREATE TABLE provedor(id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
5                                                  nome VARCHAR(255) UNIQUE,
                                                   api_key VARCHAR(255) NOT
                                                   NULL UNIQUE);
6
7  CREATE TABLE procuracao(id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
8                                                      validade DATE NOT
                                                       NULL,
9                                                      emissor CHAR(11) NOT
                                                       NULL,
10                                                     outorgado CHAR(11),
11                                                     provedor INT(6)
                                                       UNSIGNED NOT NULL,
12                                                     ehValido TINYINT(1)
                                                       NOT NULL default
                                                       1          ,
13                                                     FOREIGN KEY(emissor)
                                                       REFERENCES
                                                       usuario(cpf),
14                                                     FOREIGN
                                                       KEY(outorgado)
                                                       REFERENCES
                                                       usuario(cpf),
15                                                     FOREIGN KEY(provedor)
                                                       REFERENCES
                                                       provedor(id));
16
17 CREATE TABLE servico(id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
18                                                 nome VARCHAR(255) NOT NULL,
19                                                 provedor INT(6) UNSIGNED NOT
                                                   NULL,
```

```
20                                                  FOREIGN KEY(provedor)
                                                    REFERENCES provedor(id));
21
22  CREATE TABLE atributos(procuracao INT(6) UNSIGNED,
23                                                    servico INT(6) UNSIGNED,
24                                                    PRIMARY KEY (procuracao,
                                                      servico),
25                                                    FOREIGN KEY(procuracao)
                                                      REFERENCES procuracao(id),
26                                                    FOREIGN KEY(servico)
                                                      REFERENCES servico(id));
27
28  #Todas as senhas são "123"
29  INSERT INTO usuario (cpf, pin, nome, cert_eh_curta) VALUES
    ('21542314119', '$pbkdf2-sha256$29000$03ovpRSi9J5TSmntPQegtA$qSoPwUWn
    iYZWkV8T9f.x8tvQiFO4rk7FhyfC7A4RLtg', 'Ricardo Ribeiro',
    0);
30  INSERT INTO usuario (cpf, pin, nome, cert_eh_curta) VALUES
    ('63567303406', '$pbkdf2-sha256$29000$03ovpRSi9J5TSmntPQegtA$qSoPwUWn
    iYZWkV8T9f.x8tvQiFO4rk7FhyfC7A4RLtg', 'João da Costa',
    0);
31  INSERT INTO usuario (cpf, pin, nome, cert_eh_curta) VALUES
    ('71234411768', '$pbkdf2-sha256$29000$03ovpRSi9J5TSmntPQegtA$qSoPwUWn
    iYZWkV8T9f.x8tvQiFO4rk7FhyfC7A4RLtg', 'Rafael Carvalho',
    0);
32  INSERT INTO usuario (cpf, pin, nome, cert_eh_curta) VALUES
    ('20260114359', '$pbkdf2-sha256$29000$03ovpRSi9J5TSmntPQegtA$qSoPwUWn
    iYZWkV8T9f.x8tvQiFO4rk7FhyfC7A4RLtg', 'Gabriel Ferreira',
    0);
33
34  #api_keys logo abaixo
35  INSERT INTO provedor (id, nome, api_key) VALUES (1, 'AGU - ADVOCACIA
    GERAL DA UNIÃO', "$pbkdf2-sha256$29000$WytFqHVOqRWitBYi5FzrnQ$4h9hegE
    dDSW1gSpF057IXfpxosE2woKM4DWPexZSMLk");
36  # b00620a18e73b54d1e626f0b0ed0fc8465a32725e4f25868
37  INSERT INTO provedor (id, nome, api_key) VALUES (2, 'ANAC - AGÊNCIA
    NACIONAL DE AVIAÇÃO CIVIL', "$pbkdf2-sha256$29000$PmdszXmvFYIwRkiJsZY
    Sog$ohq25H.oTSksvILuG.3qhQ4v5r93lL2wZbcJSwan2/o");
38  # 56585d131e3c31fe5feea2f7fa853842a7967c650e8228bd
```

```
39 INSERT INTO provedor (id, nome, api_key) VALUES (3, 'ANP - AGÊNCIA
   NACIONAL DO PETRÓLEO, GÁS NATURAL E BIOCOMBUSTÍVEIS',
   "$pbkdf2-sha256$29000$VaqVUmpNKcX4PwegNCbkXA$Mi2JRaw3nCfPQPkiDZAn5
   lRqP3tgaz2mtuBPtjb8s");
40 # 1234555a76341f76cd2aa04a9bdc3518507aaaa7a332d875
41 INSERT INTO provedor (id, nome, api_key) VALUES (4, 'BACEN - BANCO
   CENTRAL', "$pbkdf2-sha256$29000$R4gxBoCwFkKolVIKwXgPYQ$frneoiSP3DtUwS
   HzATTFL.nSBTvo0KxM5JgYDiBeTR4");
42 # 1f86e83b03d7e7693fe30521003ace6f4f5bc96ea4e98d10
43 INSERT INTO provedor (id, nome, api_key) VALUES (5, 'BANCO DO BRASIL',
   "$pbkdf2-sha256$29000$7t2bk5IS4jxnTMmZsxYCQA$asj4QovMoGVP2f5fxYYCvSV
   cVgnhNz4UrafjWnUZtNw");
44 # d306ec486f3a4bc8358fa99bbc35973fa0e5d277e22cb342
45 INSERT INTO provedor (id, nome, api_key) VALUES (6, 'COAF - CONSELHO
   DE CONTROLE DE ATIVIDADES FINANCEIRAS - MINISTÉRIO DA ECONOMIA',
   '$pbkdf2-sha256$29000$VOodo9Sac.79nzPmPGfsnQ$pPGrwmcZgCdmEOg1B3dD1
   BHv4Gj5x7XCIW70vSHUQ');
46 # fc08b83c9a631bc7e07db078c27b06005d70938c3bf58abd
47 INSERT INTO provedor (id, nome, api_key) VALUES (7, 'MINISTÉRIO DA
   ECONOMIA', '$pbkdf2-sha256$29000$xdibkzKmlPleo/S.FyJEaA$IK0F9jziD2tlc
   H2pvvVXeElUdCt4hoOIsy4wIuLY7tw');
48 # 49562b0d6f581a263cd5b47965b630b3b889773249b51e3b
49 INSERT INTO provedor (id, nome, api_key) VALUES (8, 'MINISTÉRIO DA
   EDUCAÇÃO', '$pbkdf2-sha256$29000$xngvpdT6H8PYO2dMCYGw1g$Vav.DhU1ZEc
   NjeYfIn01FqJaF4uW3JIAozE8V0.BM');
50 # cea09d3691d9a3f2ee08451e8b413d59a0599f65dff17fca
51 INSERT INTO provedor (id, nome, api_key) VALUES (9, 'RECEITA FEDERAL',
   "$pbkdf2-sha256$29000$7B2DUIrRmrPWOickZCyFMA$yLyae3.wfczNfQMC34SXzYv
   O5ntI.dZFSK8GZ4iVuKg");
52 # 689dff9b7078884997ae9ada8bdf129e1e322c39c58c10cc
53 INSERT INTO provedor (id, nome, api_key) VALUES (10, 'SECRETARIA DO
   TRABALHO MINISTÉRIO DA ECONOMIA', "$pbkdf2-sha256$29000$fI9xrjXmnPOec
   67VWsvZmw$e6bDAlwhslB7LctdJZ6bbgvm2SpYL0WgIMmmBMKtnZM");
54 # 3237b297a455fe05926ec3c80eb9e491d5f64f4edcf2671e
55 INSERT INTO provedor (id, nome, api_key) VALUES (11, 'TST - TRIBUNAL
   SUPERIOR DO TRABALHO', "$pbkdf2-sha256$29000$eO/dG.Oc09r7P.ec8x7DGA$O
   4J0JuJVqjhU8UYWh6OfdjPxq1OrqtFm4cdwz51mYZM");
56 # 2356ab59a3bc355ed9e91be6f9d29e60d3001de2ec4cb7c0
57
```

```
58 INSERT INTO servico (id, nome, provedor) VALUES (1, 'SISTEMA SAPIENS',
   1);
59 INSERT INTO servico (id, nome, provedor) VALUES (2, 'SEI-ANAC', 2);
60 INSERT INTO servico (id, nome, provedor) VALUES (3, 'SRD-PR - SISTEMA
   DE REGISTRO DE DOCUMENTOS DOS POSTOS REVENDEDORES', 3);
61 INSERT INTO servico (id, nome, provedor) VALUES (4, 'SPB - SISTEMA DE
   PAGAMENTOS BRASILEIRO', 4);
62 INSERT INTO servico (id, nome, provedor) VALUES (5, 'ASSINATURA DE
   CONTRATOS DE CÂMBIO', 5);
63 INSERT INTO servico (id, nome, provedor) VALUES (6, 'SISCOAF - SISTEMA
   DO CONTROLE DE ATIVIDADES FINANCEIRAS', 6);
64 INSERT INTO servico (id, nome, provedor) VALUES (7, 'RAIS - RELAÇÃO
   ANUAL DE INFORMAÇÕES SOCIAIS', 7);
65 INSERT INTO servico (id, nome, provedor) VALUES (8, 'CAGED - CADASTRO
   GERAL DE EMPREGADOS E DESEMPREGADOS', 7);
66 INSERT INTO servico (id, nome, provedor) VALUES (9, 'DIPLOMA DIGITAL',
   8);
67 INSERT INTO servico (id, nome, provedor) VALUES (10, 'PROUNI -
   PROGRAMA UNIVERSIDADE PARA TODOS', 8);
68 INSERT INTO servico (id, nome, provedor) VALUES (11, 'DIRF -
   DECLARAÇÃO DO IMPOSTO DE RENDA RETIDO NA FONTE', 9);
69 INSERT INTO servico (id, nome, provedor) VALUES (12, 'DOI - DECLARAÇÃO
   DE OPERAÇÕES IMOBILIÁRIAS', 9);
70 INSERT INTO servico (id, nome, provedor) VALUES (13, 'PER/DCOMP -
   PEDIDO ELETRÔNICO DE RESTITUIÇÃO, RESSARCIMENTO OU REEMBOLSO E
   DECLARAÇÃO DE COMPENSAÇÃO', 9);
71 INSERT INTO servico (id, nome, provedor) VALUES (14, 'DBF - DECLARAÇÃO
   DE BENEFÍCIOS FISCAIS', 9);
72 INSERT INTO servico (id, nome, provedor) VALUES (15, 'DCTF -
   DECLARAÇÕES DE DÉBITOS E CRÉDITOS TRIBUTÁRIOS', 9);
73 INSERT INTO servico (id, nome, provedor) VALUES (16, 'DECRED -
   DECLARAÇÃO DE OPERAÇÕES COM CARTÃO DE CRÉDITO', 9);
74 INSERT INTO servico (id, nome, provedor) VALUES (17, 'DMED -
   DECLARAÇÃO DE SERVIÇOS MÉDICOS E DA SAÚDE', 9);
75 INSERT INTO servico (id, nome, provedor) VALUES (18, 'DPREV -
   DECLARAÇÃO SOBRE A OPÇÃO DE TRIBUTAÇÃO DE PLANOS PREVIDENCIÁRIOS', 9)
76 INSERT INTO servico (id, nome, provedor) VALUES (19, 'WEB MIGRANTE',
   10);
77 INSERT INTO servico (id, nome, provedor) VALUES (20, 'e-SOCIAL', 10);
```

```
78 INSERT INTO servico (id, nome, provedor) VALUES (21, 'e-DOC', 11);
79 INSERT INTO servico (id, nome, provedor) VALUES (22, 'e-PET', 11);
```

drop.sql

```
1  USE Posge;
2  DELETE FROM atributos;
3  DELETE FROM procuracao;
4  DELETE FROM servico;
5  DELETE FROM provedor;
6  DELETE FROM procuracao;
7  DELETE FROM usuario;
8
9  INSERT INTO usuario (cpf, pin, nome, cert_eh_curta) VALUES
   ('21542314119', '$pbkdf2-sha256$29000$03ovpRSi9J5TSmntPQegtA$qSoPwUWn
   iYZWkV8T9f.x8tvQiFO4rk7FhyfC7A4RLtg', 'Ricardo Ribeiro',
   0);
10 INSERT INTO usuario (cpf, pin, nome, cert_eh_curta) VALUES
   ('63567303406', '$pbkdf2-sha256$29000$03ovpRSi9J5TSmntPQegtA$qSoPwUWn
   iYZWkV8T9f.x8tvQiFO4rk7FhyfC7A4RLtg', 'João da Costa',
   0);
11 INSERT INTO usuario (cpf, pin, nome, cert_eh_curta) VALUES
   ('71234411768', '$pbkdf2-sha256$29000$03ovpRSi9J5TSmntPQegtA$qSoPwUWn
   iYZWkV8T9f.x8tvQiFO4rk7FhyfC7A4RLtg', 'Rafael Carvalho',
   0);
12 INSERT INTO usuario (cpf, pin, nome, cert_eh_curta) VALUES
   ('20260114359', '$pbkdf2-sha256$29000$03ovpRSi9J5TSmntPQegtA$qSoPwUWn
   iYZWkV8T9f.x8tvQiFO4rk7FhyfC7A4RLtg', 'Gabriel Ferreira',
   0);
13
14 #api_keys logo abaixo
15 INSERT INTO provedor (id, nome, api_key) VALUES (1, 'AGU - ADVOCACIA
   GERAL DA UNIÃO', "$pbkdf2-sha256$29000$WytFqHVOqRWitBYi5FzrnQ$4h9hegE
   dDSW1gSpF057IXfpxosE2woKM4DWPexZSMLk");
16 # b00620a18e73b54d1e626f0b0ed0fc8465a32725e4f25868
17 INSERT INTO provedor (id, nome, api_key) VALUES (2, 'ANAC - AGÊNCIA
   NACIONAL DE AVIAÇÃO CIVIL', "$pbkdf2-sha256$29000$PmdszXmvFYIwRkiJsZY
   Sog$ohq25H.oTSksvILuG.3qhQ4v5r93lL2wZbcJSwan2/o");
18 # 56585d131e3c31fe5feea2f7fa853842a7967c650e8228bd
```

```
19 INSERT INTO provedor (id, nome, api_key) VALUES (3, 'ANP - AGÊNCIA
   NACIONAL DO PETRÓLEO, GÁS NATURAL E BIOCOMBUSTÍVEIS',
   "$pbkdf2-sha256$29000$VaqVUmpNKcX4PwegNCbkXA$Mi2JRaw3nCfPQPkiDZAn5
   lRqP3tgaz2mtuBPtjb8s");
20 # 1234555a76341f76cd2aa04a9bdc3518507aaaa7a332d875
21 INSERT INTO provedor (id, nome, api_key) VALUES (4, 'BACEN - BANCO
   CENTRAL', "$pbkdf2-sha256$29000$R4gxBoCwFkKolVIKwXgPYQ$frneoiQP3DtUwS
   HzATTFL.nSBTvo0KxM5JgYDiBeTR4");
22 # 1f86e83b03d7e7693fe30521003ace6f4f5bc96ea4e98d10
23 INSERT INTO provedor (id, nome, api_key) VALUES (5, 'BANCO DO BRASIL',
   "$pbkdf2-sha256$29000$7t2bk5IS4jxnTMmZsxYCQA$asj4QovMoGVP2f5fxYYCvSV
   cVgnhNz4UrafjWnUZtNw");
24 # d306ec486f3a4bc8358fa99bbc35973fa0e5d277e22cb342
25 INSERT INTO provedor (id, nome, api_key) VALUES (6, 'COAF - CONSELHO
   DE CONTROLE DE ATIVIDADES FINANCEIRAS - MINISTÉRIO DA ECONOMIA',
   '$pbkdf2-sha256$29000$VOodo9Sac.79nzPmPGfsnQ$pPGrwmcZgCdmEOg1B3dD1
   BHv4Gj5x7XCIW70vSHUQ');
26 # fc08b83c9a631bc7e07db078c27b06005d70938c3bf58abd
27 INSERT INTO provedor (id, nome, api_key) VALUES (7, 'MINISTÉRIO DA
   ECONOMIA', '$pbkdf2-sha256$29000$xdibkzKmlPleo/S.FyJEaA$IK0F9jziD2tlc
   H2pvvVXeElUdCt4hoOIsy4wIuLY7tw');
28 # 49562b0d6f581a263cd5b47965b630b3b889773249b51e3b
29 INSERT INTO provedor (id, nome, api_key) VALUES (8, 'MINISTÉRIO DA
   EDUCAÇÃO', '$pbkdf2-sha256$29000$xngvpdT6H8PYO2dMCYGw1g$Vav.DhU1ZEc
   NjeYfIn01FqJaF4uW3JIAozE8V0.BM');
30 # cea09d3691d9a3f2ee08451e8b413d59a0599f65dff17fca
31 INSERT INTO provedor (id, nome, api_key) VALUES (9, 'RECEITA FEDERAL',
   "$pbkdf2-sha256$29000$7B2DUIrRmrPWOickZCyFMA$yLyae3.wfczNfQMC34SXzYv
   O5ntI.dZFSK8GZ4iVuKg");
32 # 689dff9b7078884997ae9ada8bdf129e1e322c39c58c10cc
33 INSERT INTO provedor (id, nome, api_key) VALUES (10, 'SECRETARIA DO
   TRABALHO MINISTÉRIO DA ECONOMIA', "$pbkdf2-sha256$29000$fI9xrjXmnPOec
   67VWsvZmw$e6bDAlwhslB7LctdJZ6bbgvm2SpYL0WgIMmmBMKtnZM");
34 # 3237b297a455fe05926ec3c80eb9e491d5f64f4edcf2671e
35 INSERT INTO provedor (id, nome, api_key) VALUES (11, 'TST - TRIBUNAL
   SUPERIOR DO TRABALHO', "$pbkdf2-sha256$29000$eO/dG.Oc09r7P.ec8x7DGA$O
   4J0JuJVqjhU8UYWh6OfdjPxq1OrqtFm4cdwz51mYZM");
36 # 2356ab59a3bc355ed9e91be6f9d29e60d3001de2ec4cb7c0
37
```

```
38 INSERT INTO servico (id, nome, provedor) VALUES (1, 'SISTEMA SAPIENS',
   1);
39 INSERT INTO servico (id, nome, provedor) VALUES (2, 'SEI-ANAC', 2);
40 INSERT INTO servico (id, nome, provedor) VALUES (3, 'SRD-PR - SISTEMA
   DE REGISTRO DE DOCUMENTOS DOS POSTOS REVENDEDORES', 3);
41 INSERT INTO servico (id, nome, provedor) VALUES (4, 'SPB - SISTEMA DE
   PAGAMENTOS BRASILEIRO', 4);
42 INSERT INTO servico (id, nome, provedor) VALUES (5, 'ASSINATURA DE
   CONTRATOS DE CÂMBIO', 5);
43 INSERT INTO servico (id, nome, provedor) VALUES (6, 'SISCOAF - SISTEMA
   DO CONTROLE DE ATIVIDADES FINANCEIRAS', 6);
44 INSERT INTO servico (id, nome, provedor) VALUES (7, 'RAIS - RELAÇÃO
   ANUAL DE INFORMAÇÕES SOCIAIS', 7);
45 INSERT INTO servico (id, nome, provedor) VALUES (8, 'CAGED - CADASTRO
   GERAL DE EMPREGADOS E DESEMPREGADOS', 7);
46 INSERT INTO servico (id, nome, provedor) VALUES (9, 'DIPLOMA DIGITAL',
   8);
47 INSERT INTO servico (id, nome, provedor) VALUES (10, 'PROUNI -
   PROGRAMA UNIVERSIDADE PARA TODOS', 8);
48 INSERT INTO servico (id, nome, provedor) VALUES (11, 'DIRF -
   DECLARAÇÃO DO IMPOSTO DE RENDA RETIDO NA FONTE', 9);
49 INSERT INTO servico (id, nome, provedor) VALUES (12, 'DOI - DECLARAÇÃO
   DE OPERAÇÕES IMOBILIÁRIAS', 9);
50 INSERT INTO servico (id, nome, provedor) VALUES (13, 'PER/DCOMP -
   PEDIDO ELETRÔNICO DE RESTITUIÇÃO, RESSARCIMENTO OU REEMBOLSO E
   DECLARAÇÃO DE COMPENSAÇÃO', 9);
51 INSERT INTO servico (id, nome, provedor) VALUES (14, 'DBF - DECLARAÇÃO
   DE BENEFÍCIOS FISCAIS', 9);
52 INSERT INTO servico (id, nome, provedor) VALUES (15, 'DCTF -
   DECLARAÇÕES DE DÉBITOS E CRÉDITOS TRIBUTÁRIOS', 9);
53 INSERT INTO servico (id, nome, provedor) VALUES (16, 'DECRED -
   DECLARAÇÃO DE OPERAÇÕES COM CARTÃO DE CRÉDITO', 9);
54 INSERT INTO servico (id, nome, provedor) VALUES (17, 'DMED -
   DECLARAÇÃO DE SERVIÇOS MÉDICOS E DA SAÚDE', 9);
55 INSERT INTO servico (id, nome, provedor) VALUES (18, 'DPREV -
   DECLARAÇÃO SOBRE A OPÇÃO DE TRIBUTAÇÃO DE PLANOS PREVIDENCIÁRIOS', 9)
56 INSERT INTO servico (id, nome, provedor) VALUES (19, 'WEB MIGRANTE',
   10);
57 INSERT INTO servico (id, nome, provedor) VALUES (20, 'e-SOCIAL', 10);
```

```
58 INSERT INTO servico (id, nome, provedor) VALUES (21, 'e-DOC', 11);
59 INSERT INTO servico (id, nome, provedor) VALUES (22, 'e-PET', 11);
```

APÊNDICE K – SHELL SCRIPTS

install_arch.sh

```
sudo pacman -Syu
sudo pacman -S --noconfirm mariadb python3 python-pip
    python-virtualenv python-passlib swig gcc
echo "A senha do root do BD definida no projeto é \"1234\", mude no
    arquivo bd.py pra qual voc ê for usar"
sudo mysql_install_db --user=mysql --basedir=/usr
    --datadir=/var/lib/mysql
sudo systemctl start mysqld
sudo /usr/bin/mysql_secure_installation
sudo systemctl restart mysql
sudo systemctl enable mariadb
mysql -u root -p < app/create.sql
# Para criar o Virtual Environment:
sudo pip3 install --upgrade pip
sudo pip3 install virtualenv
python3 -m venv env
# Para acessar o Virtual Environment:
source env/bin/activate
# Atualizar o pip
# Para instalar as dependências:
python3 -m pip install --upgrade pip
pip3 install -r requirements.txt
./create_certificate.sh
```

install_deb.sh

```
sudo apt update
sudo apt install -y mariadb-server python3 python3-pip python3-venv
    python-passlib swig
# A senha do root padrão é "1234", mude no bd.py pra qual voc ê for
    usar
mysqld --skip-grant-tables
sudo mysql -e "use mysql; update user set
    plugin='mysql_native_password' where User='root'"
sudo mysql -e "UPDATE mysql.user SET authentication_string =
    PASSWORD('1234') WHERE User='root' AND Host='localhost';"
sudo mysql -e "update user set plugin='mysql_native_password' where
    User='root'"
sudo mysql -e "FLUSH PRIVILEGES"
```

```
/ etc / init .d / mysql  restart
sudo  systemctl  enable  mariadb
mysql -u  root -p  < app / create . sql
# Para  criar  o  Virtual  Environment :
sudo  pip3  install --upgrade  pip
sudo  pip3  install  virtualenv
python3 -m venv  env
# Para  acessar  o  Virtual  Environment :
source  env / bin / activate
# Atualizar  o  pip
# Para  instalar  as  depend ê ncias :
pip3  install -r  requirements . txt
./ create_certificate . sh
```

create_certificate.sh

```
mkdir  app / documents
mkdir  app / certificates
cd  app / certificates
openssl  req -x509 -newkey  rsa :4096 -sha256 -utf8 -keyout
    poseg_key . pem -outposeg_cert . pem -days 3650 -nodes -addext
    keyUsage = digitalSignature -subj  "/ C = BR / O = gov . br / CN = Portal
    Outorga  de  Servi ç os  do  Governo  Eletr ô nico "
openssl  req -x509 -newkey  rsa :4096 -sha256 -utf8 -keyout
    21542314119_key . pem -out21542314119_cert . pem -days365 -nodes
    -addext  keyUsage = digitalSignature -subj
    " /C= BR /O= gov . br / CN = Ricardo Ribeiro "
openssl  req -x509 -newkey  rsa :4096 -sha256 -utf8 -keyout
    63567303406_key . pem -out63567303406_cert . pem -days365 -nodes
    -addext  keyUsage = digitalSignature -subj  " / C = BR /O = gov . br / CN = Jo ã o
    da  Costa "
openssl  req -x509 -newkey  rsa :4096 -sha256 -utf8 -keyout
    71234411768_key . pem -out71234411768_cert . pem -days365 -nodes
    -addext  keyUsage = digitalSignature -subj  " / C = BR /O = gov . br / CN = Rafael
    Carvalho "
openssl  req -x509 -newkey  rsa :4096 -sha256 -utf8 -keyout
    20260114359_key . pem -out20260114359_cert . pem -days365 -nodes
    -addext  keyUsage = digitalSignature -subj
    " /C= BR /O= gov . br / CN = Gabriel Ferreira "
```

drop_and_create_db.sh

```
mysql -u  root -p  < app / drop . sql
rm -r  app / documents
```

```
rm -r app/certificates
./create_certificate.sh
```

start.sh

```
source env/bin/activate
cd app
python3 views.py
```

APÊNDICE L – ARQUIVO REQUIREMENTS.TXT

```
flask
mysql−connector−python
reportlab
endesive
chardet
pytz
passlib
```

APÊNDICE M – ARTIGO SBC

POSGE: PORTAL DE OUTORGA DE SERVIC¸ OS DO GOVERNO ELETRÔNICO

João Victhor da Paz Campos[1]

[1]Departamento de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)
Florianópolis – SC – Brazil

joao.victhor@grad.ufsc.br

Abstract. *In the last years, Brazil has invested efforts on the initiative of creating an unified channel for providing access to e-gov services through the use of digital certification. Several services are applying to the platform each day and more citizens can use government services in a totally digital manner. In this context, the creation of a complement to this platform is necessary, so users can delegate use rights to third parties for certain e-gov services, when necessary. The goal of this thesis is to develop a proof of concept of a system that can manage electronic proxies that is supplementary to the Brazilian e-gov portal, assisting the Government's initiative.*

Resumo. *Nos últimos anos, o Brasil tem investido esforc¸os na iniciativa de criar uma plataforma unificada para acesso a servic¸os do governo eletr ônico através da utilizac¸ão de certificac¸ão digital. Cada vez mais servic¸os aderem a plataforma e mais cidad ãos são capazes de realizar servic¸os de maneira totalmente digital. Nesse contexto, é necessário a criar um complemento a essa plataforma para que usu ários possam delegar direitos de uso para determinados servic¸os de e-gov a terceiros. O objetivo do seguinte trabalho é desenvolver uma prova de conceito de um sistema gerenciador de procurac¸ões eletrônicas que possa ser um suplemento ao portal Gov.br, auxiliando assim a iniciativa do Governo Brasileiro.*

1. Introdução

Em 2017 se iniciou um processo de Transformação Digital por parte do governo brasileiro a fim de fomentar o uso dos servic¸os de e-gov pela internet. Através de um Censo de Servic¸os foram identificados cerca de 2,8 mil servic¸os disponibilizados pelo governo que davam acesso às atividades atrav és de canais independentes entre si. Nessa iniciativa definiu-se um conceito unificado de servic¸o p úblico e todos os identificados no Censo foram reunidos no Portal de Servic¸os do Governo Federal. Nos dois anos subsequentes o foco foi centrado em encontrar métodos para incentivar cada vez mais servic¸os a aderirem ao meio e, em julho de 2019, mais da metade de todos os servic¸os ofertados pelas entidades governamentais estavam disponíveis de forma totalmente digital.

Com o crescimento do interesse dos governos em disponibilizar servic¸os no âmbito digital, emerge a possibilidade de uso de procurac¸ões eletrônicas e, para tal, é necessário existir uma entidade respons ável por intermediar o processo de concess ão do direito de uso dos servic¸os, bem como a seguranc¸a dos dados envolvidos no procedimento realizado. Um desses cuidados se refere à autenticação de ambas as partes, concedente e

receptor do direito, para a prevenção de fraudes por falsidade ideológica e acesso indevido de hackers e agentes inteligentes. Para a garantia dessa propriedade uma das possibilidades é a utilização de certificação digital, ferramenta capaz de identificar indivíduos no âmbito digital.

Em paralelo ao processo de digitalização dos serviços governamentais brasileiros, o número de emissões de certificados digitais ICP-Brasil aumenta, batendo recorde de emissões em janeiro de 2020, totalizando 8,9 milhões de certificados digitais ativos no país. No entanto, apesar do número expressivo, representa menos de 5% dos 210 milhões de brasileiros estimados pelo IBGE em julho de 2019. Com isso, é importante implementar uma alternativa para uso do POSGE por usuários que não sejam detentores de certificado digital.

2. PDF

O *Portable Document Format* é um formato de arquivos de computador criado no início da década de 90, pela Adobe Systems, com o objetivo de se tornar um formato padronizado para compartilhamento de documentos digitais sem que os usuários do produto tenham de ter o mesmo sistema operacional ou as mesmas dependências a fim de poder acessar seu conteúdo, as contendo em seu interior. De 1993 a 2006 o formato passou por constantes atualizações até o ano de 2008 quando, devido ao constante aumento no índice de aderência por parte dos usuários, se tornou um padrão aberto, deixando de ser propriedade da Adobe Systems.[ISO 2008]

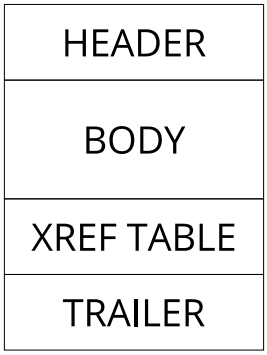


Figura 1. Estrutura de um arquivo no formato PDF

A primeira linha de um arquivo no formato PDF deve ser um *header* (cabeçalho), o qual consiste nos 5 caracteres "%PDF-" precedidos do número da versão utilizada na concepção do arquivo. A partir da variante 1.4 caso um número de versão esteja presente no catálogo do dicionário do *trailer* do documento, esta terá preferência em relação a presente no *header*. É possível que um documento com funcionalidades mais recentes seja interpretado por um leitor que só dê suporte a uma versão mais antiga do formato. O projeto é desenvolvido de tal maneira que novas características introduzidas possam ser ignoradas por softwares que não as compreendam.

O *body* (corpo) de um documento em PDF consiste em uma série *Indirect Objects* (objetos indiretos) que representam o conteúdo do documento, a parte visível para o

usuário. O formato suporta vários tipos comuns de objetos como *Booleans*, números inteiros e reais, *Strings*, *Arrays*, *Dictionaries*, *Null*, e também específicos do escopo como *Object Streams*, que por sua vez são compostos por uma série de *Indirect objects*. O conteúdo do *body* depois é referenciado na *Cross-Reference Table*.

A *Cross-Reference Table* (Tabela de Referência Cruzada) ou *x-ref table* contém as informações necessárias para permitir o acesso aleatório a *Indirect Objects* específicos sem que seja necessário processar o arquivo inteiro. A tabela consiste em uma ou mais seções. Inicialmente a tabela inteira se trata de uma só seção e, na medida que o documento for atualizado, outras mais podem ser adicionadas a tabela. Cada seção deve ser iniciada com a palavra-chave **xref** e pode conter uma ou mais subseções.

O *trailer*, localizado nas últimas linhas presentes em um arquivo no formato PDF como visto na Figura 1, armazena ponteiros para a *Cross-Reference Table* — possibilitando que *softwares* leitores de PDFs rapidamente achem sua posição no arquivo ao iniciar a leitura pelo fim — e também para certos objetos especiais que ficam armazenados em seu dicionário como, por exemplo, a assinatura digital do documento.

3. Assinatura Digital

Existem múltiplas maneiras de criar uma assinatura digital, seja utilizando funções matemáticas, como na **Criptografia de Chaves Públicas**, ou de forma biométrica através de impressões digitais ou reconhecimento de retina. Sua implementação é provida por um módulo denominado *Signature Handler* (Manipulador de Assinatura). A implementação desse módulo pode ser feita por terceiros, provendo assim a possibilidade de se desenvolver uma solução personalizada de acordo com as necessidades de cada domínio, e tem que seguir as especificações da ISO 32000.

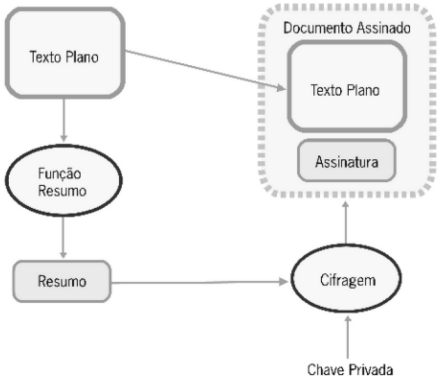


Figura 2. Estrutura de um arquivo no formato PDF

Assinaturas digitais são criadas calculando-se um resumo dos dados do documento, utilizando uma função matemática denominada função de *hash* criptográfico, posteriormente cifrando esse resumo utilizando uma senha presente no certificado digital do assinante, designada chave privada (essa chave é integrante de um par de chaves presentes

no certificado do usuário, sendo a outra nomeada chave pública. O que uma é capaz de encriptar, a outra é capaz de decriptar). Ao fim da operação o resultado é concatenado ao documento junto com uma série de metadados que serão utilizados no processo de verificação da validade da assinatura pelo *software* leitor de PDF, série essa que inclui o certificado do usuário (sem sua chave privada).

Para realizar a verificação da assinatura, o leitor de PDF a decripta utilizando a chave pública do assinante presente no certificado, tendo acesso assim ao resumo criptográfico gerado no momento da assinatura. O leitor de PDF gera um resumo criptográfico do documento recebido e compara o resultado com o resumo presente na assinatura; Se forem iguais, a assinatura é valida, caso contrário isso indica que o documento foi modificado e a assinatura é considerada inválida.

4. WEB API

Web APIs são implementadas com o objetivo de conectar diferentes sistemas pertencentes a uma mesma infraestrutura ou para disponibilizar e realizar o controle do compartilhamento das informações disponíveis com terceiros. Para isso, *Web APIs* utilizam o protocolo *HTTP* para realizar a troca de mensagens entre os sistemas, e formatos como o *XML* e o *JSON* para representar as respostas às requisições, possibilitando assim que as informações enviadas possam ser interpretadas e processadas conforme necessário. O servidor que contém os dados a serem compartilhados deve disponibilizar uma *URL* de acesso.

5. e-gov

O termo e-gov, criado na década de noventa, serviu de rótulo para uma nova área, dentro do campo de Sistemas de Informação, que tem como objeto de estudo a "governança digital", tratando de políticas, estratégias e implementação de processos governamentais no meio digital [Åke Grönlund and Horan 2004].

Apesar de nomeada na década de 1990, os primeiros estudos dentro dessa área datam desde a década de 1970, onde as pesquisas se concentravam em meios de utilizar a Tecnlogia da Informação dentro do governo [Kraemer 1978], diferente da tendência atual de concentrar seus estudos no uso externo, como o de serviços governamentais por parte dos cidadãos de um país.

6. Procuração Eletrônica

Segundo [Gonçalves 2007], a outorga é um instrumento utilizado por uma pessoa, denominada mandante nesse contexto, a fim de conceder plenos poderes a um terceiro, nomeado procurador, para executar ações e administrar processos em seu nome.

Existem dois tipos de procuração; a particular, que pode ser feita com somente o envolvimento das duas partes, mandante e procurador, ou podendo o reconhecimento de firma ser exigido por uma delas, e a pública, que é feita e registrada por um tabelionato de notas. Procurações podem ser outorgadas e recebidas por quaisquer pessoas desde que sejam maiores de idade e estejam em conformidade com as leis da justiça eleitoral.

Para que uma procuração seja considerada válida, uma série de informações devem estar inclusas em seu conteúdo:

- Nome e documentos de identificação de ambas as partes, outorgante e outorgado;
- Finalidade e data da procuração;
- Descrição detalhada dos poderes concedidos e sua extensão;
- Designação do lugar da concessão de poderes;
- Assinatura do mandante com reconhecimento em firma, caso requisitado. A assinatura pode ser realizada digitalmente de acordo com o §1º do artigo 105 do Código de Processo Civil de 16 de Março de 2015 [Brasil 2017].

7. Tecnologias Usadas

Duas opções de *frameworks* para *Python* dentre os mais populares foram consideradas para realizar a implementação das funcionalidades *web* do protótipo, o *Django* e o *Flask*. Em termos gerais, ambas as bibliotecas contam com diversas funcionalidades em comum; roteamento de *URLs*, autenticação e estabelecimento de sessões, integração com bancos de dados, mecanismo de *templating*, etc. No entanto, o *Django* conta com uma série de funcionalidades extras como interfaces de administração, ferramentas integradas para *bootstrapping*, suporte a múltiplas aplicações e sistema ORM para integração com bancos de dados, tendo seu foco em aplicações complexas[Django 2019]. Em contrapartida, o *Flask* conta com menos funcionalidades, focando em aplicações mais simples e se concentrando em somente prover as funcionalidades necessárias para possibilitar o acesso a aplicações *Python* na web[Pallets 2019]. Devido a complexidade extra envolvida no uso do *Django* e também ao fato de suas funcionalidades extras não serem necessárias para o desenvolvimento do protótipo, escolheu-se o *Flask*.

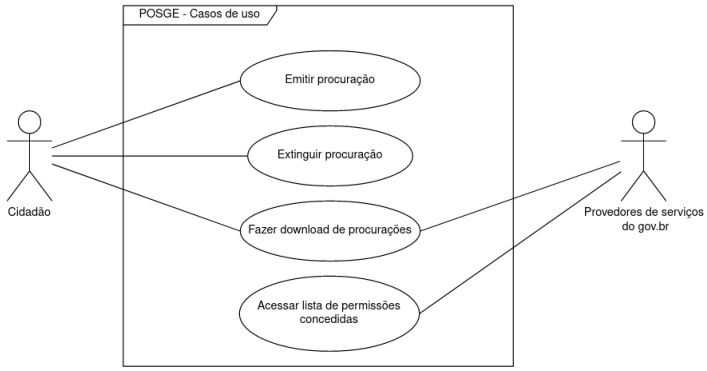
No que tange a manipulação de arquivos em PDF, duas bibliotecas foram utilizadas, uma para criar os documentos e uma segunda para assiná-los. O *ReportLab* é uma *engine* de criação de PDFs bem documentada que tem uma versão gratuita e *open-source*, a qual foi utilizada na implementação, que dá suporte a desenhos vetoriais, geração e reutilização de gráficos e ao *PLATYPUS*, uma *engine* que permite criar documentos a partir de elementos textuais como títulos, parágrafos e tabelas [ReportLab 2019]. Já o *Endesive* é uma biblioteca *open-source* com foco em assinatura e verificação de assinaturas digitais em PDFs , XMLs e Emails. Dentre as funcionalidades oferecidas para esses formatos, a biblioteca suporta a implementação do *security handler* **Adobe.PPKLite/adbe.pkcs7.detached**[Makarewicz 2019], sendo este o utilizado no protótipo.

Os certificados de longa duração — que representam, na prova de conceito, os certificados já vinculados ao cadastro do cidadão no gov.br — são emitidos no momento da instalação do sistema, junto ao cadastro dos usuários iniciais. Os usuários cadastrados posteriormente — representando aqueles que não são detentores de certificado digital — têm seus certificados emitidos a cada autenticação no sistema. Também é importante frisar que as senhas dos usuários cadastrados, assim como as chaves de API dos provedores de sevicços, não são armazenadas em claro no banco de dados. Para isso utilizou-se a biblioteca *Passlib* a fim de gerar resumos criptográficos das senhas utilizando o algoritmo *sha256* e *salt*, para evitar, por exemplo, ataques de pré-imagem, os quais são armazenados na base.

8. Planejamento

Uma análise a cerca das funcionalidades que deveriam ser implementadas e sua relação com os atores envolvidos com o uso do software foi realizada. Os atores que irão interagir com o ambiente são os cidadãos, usuários do gov.br, que irão emitir, extinguir e acessar suas procurações, e os provedores de serviços do portal que irão acessar o POSGE via API para verificar a permissão do uso de certos serviços por determinados usuários em nome de outrem. A prova de conceito utilizou exemplos de provedores e serviços disponíveis no portal gov.br que já utilizam certificação digital, os quais foram populados na base através do *script* de criação do banco de dados, no entanto, em um caso de uso real, o sistema deveria sempre se atualizar com relação aos serviços e provedores disponíveis através de uma conexão com o portal.

Figura 3. Diagrama de casos de uso do protótipo



Seguindo a documentação do *framework web* utilizado, o primeiro arquivo criado, *views.py* neste protótipo, contém o código necessário para iniciar o servidor da aplicação. No código-fonte o aplicativo é inicializado definindo-se o atributo *secret_key* (chave secreta) usado para assinar os *cookies* de sessão. A função *run* é invocada, passando por parâmetro o *IP* do *host*, a porta a ser utilizada pela aplicação e uma chave *boolean* para habilitar as mensagens de *debug*, inicializando assim o servidor do *flask*. Nesse arquivo também se encontra a implementação das funções de redirecionamento do sistema. Métodos são executados quando o usuário acessa determinadas *urls*, para isso se utiliza uma marcação acima da função em questão no código-fonte, *@app.route*, informando o *path* que a executa e quais métodos HTTP são permitidos.

9. Autenticação

Ao entrar no site, o usuário é direcionado para a página de *login*, onde lhe é requisitado o CPF e um PIN para ter acesso ao sistema. Após a submissão do formulário, o sistema verifica se a sequência numérica entrada pelo usuário constitui um CPF válido e, em caso

positivo, verifica se o usu ário est á registrado na base. Caso esteja, o resumo da senha entrada pelo usu ário é comparado com o armazenado no banco de dados; se o registro do usuário indicar que ele utiliza certificados de curta-durac¸ão, um certificado é emitido e armazenado na pasta de certificados, prosseguindo com o estabelecimento da sess ão e redirecionamento para a página inicial caso sejam iguais.

Caso o CPF n ão esteja registrado na base, o usu ário é redirecionado para uma página de cadastro onde dever á informar nome e PIN e conferir seu CPF, prosseguindo para a p ágina inicial ap ós o cadastro. Nesse momento sempre é gerado um certificado digital de curta-duração para que o usuário possa usar o sistema. Uma vez que esteja autenticado na aplicação, o usuário poderá realizar seu *logout* clicando no ícone presente no final do menu superior, o qual fica vermelho quando o mouse se sobrepõe, limpando seus dados de sessão e excluindo seu certificado digital do disco caso seja de curta-duração.

10. Emissão

Na página de emiss ão de outorgas o usu ário deverá clicar, na lista a esquerda, na opc¸ão correspondente do provedor de servic¸os ao qual a procurac¸ão se destina, e então selecionar os servic¸os que serão autorizados. Do lado direito se deve entrar com o CPF do outorgado, conferindo o nome resultado na última caixa de texto do formul ário, e com a data de validade do documento, sendo possível marcar a caixa de seleção para que seja válido por tempo indeterminado. Ao clicar no botão de emissão, os dados passarão por uma série de validações e os dados serão registrados na base de dados.

Após o cadastro da outorga, a função responsável por criar o documento recebe por parâmetro um objeto da classe **procuracao** que contém todas as informações da outorga que serão usadas para criar o documento PDF usando o *ReportLab*. Cada parágrafo do texto do documento é escrito e atribu ído a vari áveis, utilizando-se *tags* para definir o tamanho da fonte e usar negrito, se adequando as informac¸ões contantes na outorga. O tamanho do papel do documento é definido (neste caso a4, importado pela vari ável *SimpleDocTemplate*), passando o tamanho das margens da p ágina. Estilos de texto s ão definidos e armazenados na variável *styles* de acordo com o espac¸amento e o alinhamento do texto. As vari áveis de texto criadas s ão ent ão inseridas em uma lista denominada *story*, informando o estilo a ser utilizado, seguidas de espac¸os em branco que separam os parágrafos do texto. Por fim, a função insere o rodapé com o link de *download*, utilizando a função *footer*, e o documento é gerado e salvo em disco.

Com o arquivo criado e em disco, uma func¸ão do m ódulo *utils* é invocada para criar uma imagem, com o nome e CPF do futuro assinante, a fim de representar visualmente a assinatura digital do documento. Os certificados digitais e as chaves privadas do outorgante e do POSEG são carregados na memória, metadados da assinatura como contato, localização, hora, nome da entidade emissora e imagem da assinatura s ão adicionados a uma variável dicionário e esses dados são passados por parâmetro, junto a definição do algoritmo de resumo criptogr áfico, para a func¸ão *sign* do *Endesive* que assina o documento duas vezes, finalizando o processo de emiss ão. O documento é disponibilizado para *Download* através da página inicial do sistema.

11. Download e Extinção

A partir da página inicial do sistema, o usuário pode visualizar as procurações que foram emitidas e recebidas por ele, além de baixar e extinguir os documentos emitidos ao clicar nos ícones correspondentes na tabela. Tanto o outorgante quanto o receptor da outorga detém poderes para executar ambas as tarefas.

Ao clicar no ícone de *Download*, uma requisição é enviada para o servidor, utilizando *jQuery*, com o identificador do documento. O sistema verifica se o usuário tem direito de acessar aquele documento, o que só é verdade se tratar-se do outorgante ou outorgado, e gera a *string* correspondente ao nome do arquivo utilizando as informações disponíveis na outorga e libera o *download* da procuração.

No processo de extinção da outorga, que é iniciado pela opção correspondente na tela inicial, uma requisição é enviada ao servidor com o identificador da procuração. A função de extinção verificará se o usuário tem permissão de efetuar a ação e, em caso positivo, exibirá uma *pop-up* requisitando a confirmação do usuário. Em caso confirmado, o sistema irá mudar a *flag* de validade presente na entrada do documento na tabela de procurações para *false* e excluirá o documento do disco-rígido do servidor. A procuração não mais aparecerá na tela inicial de quaisquer usuários e caso alguém tente realizar *download*, o sistema informará que a procuração não é válida mais.

12. API

A fim de garantir o controle do acesso dos provedores de serviços as funcionalidades implementadas, cada provedor cadastrado no banco de dados tem uma chave de acesso a API. As chaves foram geradas utilizando a função *urandom* da biblioteca *os* do *python* e são representadas em base hexadecimal. No momento do cadastro do provedor, um resumo criptográfico *sha256* salgado da chave de acesso é armazenado em sua tupla, a fim de verificar a validade da chave no momento do acesso à API.

A autenticação do provedor na API não só é utilizada para ter acesso às funcionalidades do sistema como também é para limitar o que pode ser acessado. O sistema só responde as requisições com os dados requisitados se a procuração for destinada ao uso dos serviços do provedor autenticado.

12.1. Implementação

Foram implementadas três funções para uso da API; *verify_api_key*, para realizar a autenticação do provedor, *get_servicos*, que disponibiliza a lista de serviços do provedor liberados para determinado outorgado, e *download_procuracao_api*, a qual responde a requisição com o arquivo PDF da procuração requisitada.

A função de autenticação da API é responsável por alterar o funcionamento das outras duas funções disponibilizadas, ambas precisam passar pela autenticação antes de serem executadas; ela compara um resumo da chave disponibilizada no cabeçalho da requisição com os resumos criptográficos salgados armazenados nas tuplas dos provedores do banco de dados. Se aquela chave pertencer a um provedor, o mesmo é autenticado e pode prosseguir com a requisição, caso contrário, o servidor retorna uma mensagem em *JSON* informando que a chave é inválida.

Para acessar a lista de serviços permitidos, o provedor deve enviar uma requisição *GET* para o endereço da API (*http://host:5000/api*) informando um CPF em seu

cabeçalho,que é interpretado pela interface como o identificador do usuário para qual os direitos de uso foram concedidos, e a chave da API em seu cabeçalho. A chave passa pela função de autenticação e o CPF passa por algumas validações, retornando códigos HTTP de acordo com o resultado. Caso o CPF passe pela validação e o usuário esteja cadastrado na base, o sistema irá retornar uma lista com as procurações válidas emitidas por ele para o provedor de serviços autenticado em *JSON* (podendo também retornar uma lista vazia caso não exista nenhuma).

Com o objetivo de realizar o *download* da procuração, o provedor deve realizar uma requisição *GET* para o endereço de *download* de procurações da API (*http://host:5000/api/download/id*) com a chave da API inserida no *header*. A chave é usada para autenticar o provedor e o identificador da procuração para acessá-la. Para que a API responda com o documento, a procuração precisa ser direcionada ao provedor de serviços autenticado e tem que estar válida, retornando códigos HTTP de erro caso contrário.

12.2. Testes

Para testar o funcionamento da API foram cadastradas na plataforma seis procurações com diferentes características que interferem na decisão de incluir ou não a procuração na lista de serviços liberados.

ID	Emissor	Outorgado	PdS	Validade	Observação
3	33027073478	20260114359	RFB	Indefinida	Data de validade é indefinida
4	21542314119	20260114359	RFB	30 Nov. 2020	-
5	71234411768	20260114359	RFB	10 Dez. 2020	Foi extinguida pelo outorgado
6	20260114359	71234411768	RFB	20 Nov. 2020	O CPF é do emissor
7	33027073478	20260114359	MEC	25 Dez. 2020	O provedor é outro
8	21542314119	20260114359	RFB	10 Out. 2020	A data de validade já passou

Segundo as verificações realizadas pelo *software*, caso o provedor de serviços **Receita Federal** requisitasse os serviços liberados para o **Gabriel Ferreira**, CPF 202.601.143-59, somente as procurações de *ID* 3 e 4 deveriam aparecer na lista de serviços permitidos.

O utilitário **curl** do *linux*, capaz de se comunicar com servidores utilizando uma grande gama de protocolos de comunicação, foi utilizado para enviar requisições *GET* à API para realizar os testes. O comando executado em *shell* para testar a funcionalidade de requisição da lista de serviços permitidos envia uma requisição contendo a chave da API em seu cabeçalho e o CPF de um usuário definido na variável **CPF**. Para realizar o teste de *download* da procuração pela API foi utilizado o mesmo utilitário do *Linux*. Nessa ocasião a *url* da requisição é diferente e a informação passada é o *ID* da procuração que deseja ser baixada. Foram executados três testes, uma situação de sucesso e duas de erro.

13. Conclusão

Neste trabalho foi abordada uma proposta de melhoria nos processos democráticos dos serviços eletrônicos do governo brasileiro no que tange a delegação de direitos de uso. É

um direito do cidadão, no exercício de sua democracia, permitir a terceiros que efetuem ações em seu nome. Para que isso fosse possível, foi necessário estudar diferentes áreas do conhecimento, devido ao caráter interdisciplinar da obra, levando a compreensão dos processos que regem a concessão de poderes em diferentes instâncias. No fim, o trabalho demonstrou que é possível, através do conjunto de tecnologias utilizado, implementar uma solução para a problemática questionada seguindo os objetivos propostos inicialmente. No entanto, é necessário investigar mais a respeito do que é feito no portal de serviços brasileiro para adequar a metodologia aplicada ao padrão implantado para que seja usado na prática.

References

(2008). *ISO 32000-1:2008*. International Organization for Standardization.

Brasil (2017). Instrução normativa rfb nº 1751. *Diário Oficial da União*.

Django (2019). Django documentation.

Gonçalves, M. V. R. (2007).*Novo Curso de Direito Processual Civil*, volume 1. Editora Saraiva, 4 edition.

Kraemer, K. L. (1978). Local government and information technology in the united states. *OECD Informatics Studies*, 12.

Makarewicz, G. (2019). m32/endesive: en-crypt, de-crypt, si-gn, ve-rify - smime, pdf, xades and plain files in pure python.

Pallets (2019). Flask documentation (1.1.x).

ReportLab (2019). Reportlab open-source user guide.

Åke Grönlund and Horan, T. A. (2004). Introducing e-gov: History, definitions, and issues. *Communications of the Association for Information Systems*, 15:713–729.