UNIVERSIDADE FEDERAL DE SANTA CATARINA

DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

SISTEMAS DE INFORMAÇÃO

CIÊNCIA AMBIENTAL CIDADÃ: UM APLICATIVO PARA ENGAJAR A PARTICIPAÇÃO DAS PESSOAS NOS PARQUES AMBIENTAIS.

JACKSON DENER WRUBLAK

Florianópolis, 2021

UNIVERSIDADE FEDERAL DE SANTA CATARINA

DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

SISTEMAS DE INFORMAÇÃO

CIÊNCIA AMBIENTAL CIDADÃ: UM APLICATIVO PARA ENGAJAR A PARTICIPAÇÃO DAS PESSOAS NOS PARQUES AMBIENTAIS.

JACKSON DENER WRUBLAK

ORIENTADOR: PROF. JOSÉ EDUARDO DE LUCCA

Proposta de trabalho de conclusão de curso apresentado à Universidade Federal de Santa Catarina para obtenção de grau em Bacharel em Sistemas de Informação.

Florianópolis, 2021

JACKSON DENER WRUBLAK

CIÊNCIA AMBIENTAL CIDADÃ: UM APLICATIVO PARA ENGAJAR A PARTICIPAÇÃO DAS PESSOAS NOS PARQUES AMBIENTAIS.

Trabalho de conclusão de curso submetido ao Departamento de Informática e Estatística da Universidade Federal de Santa Catarina para a obtenção do Grau de Bacharelado em Sistemas de Informação.

Orientador:

_____

Prof. José Eduardo de Lucca

Orientador

Universidade Federal de Santa Catarina

Banca Examinadora:

_____

Patrícia Maria Schubert Peres

_____

Francisco Sacco Flores Almeida Teixeira

SUMÁRIO

RESUMO

Este trabalho de conclusão de curso tem como objetivo a viabilização e implementação de uma ferramenta computacional em formato de aplicativo, que segue a linha da Ciência Cidadã adotando medidas de crowdsourcing, com foco ambiental. A principal função da ferramenta é coletar informações do ambiente e disponibilizá-las para que o cidadão comum e até mesmo os gestores possam tomar decisões, ou compreender melhor o entorno. Serão realizadas pesquisas em formato de enquete com as pessoas, com o intuito de medir o nível de desengajamento em relação às áreas verdes urbanas. A solução proposta consiste na utilização de métodos de cadastro e leitura de elementos como árvores, animais, lagos; localização de recursos como lixeiras, banquinhos, cadeiras; localização de áreas de lazer como trilhas, mesas, campos/quadras esportivas; detecção de melhores áreas para se levar crianças. A principal motivação para o desenvolvimento deste trabalho, é aumentar o interesse dos cidadãos nos parques ecológicos, aproveitando o avanço da tecnologia móvel e o aumento da frequência de uso dos smartphones.

Palavras-chaves: Ecologia, Parque, Ambiente, Ciência Cidadã, Crowdsourcing, Aplicativo, *App*, *Smartphone*.

ABSTRACT

This dissertation aims to enable and implement a computational tool, an application format, that follows the Citizen Science line, adopting crowdsourcing measures, with an environmental focus. The main function of the tool is to collect information from the environment and make it available so that the common citizen and even managers can make decisions, or better understand the environment. Surveys will be carried out in the form of a survey with people, in order to measure the level of disengagement in relation to urban green areas. The proposed solution consists of using methods of registration and reading of elements such as trees, animals, lakes; location of resources such as trash cans, stools, chairs; location of recreational areas such as trails, tables, fields / sports courts; detection of best areas to bring children. The main motivation for the development of this work is to increase the interest of citizens in ecological parks, taking advantage of the advance of mobile technology and the increase in the frequency of use of smartphones.

Key-words: Ecology, Park, Environment, Citizen Science, Crowdsourcing, App, Smartphone.

1. INTRODUÇÃO

As mudanças tecnológicas ocorridas nas últimas décadas têm possibilitado o compartilhamento de bens, serviços e informações entre as pessoas de maneira mais dinâmica, esta afirmação consta no artigo “Efetividade do Crowdsourcing como Apoio à Segurança Pública", de João Moisés Brito Mota e Afonso Carneiro Lima, e servirá como inspiração para o desenvolvimento do presente estudo, que visa resolver o problema de desengajamento das pessoas em relação aos parques ambientais através da viabilização e implementação de uma ferramenta computacional em formato de aplicativo que segue a linha da Ciência Cidadã, adotando medidas de CrowdSourcing.

A Ciência Cidadã tem como base a participação informada, consciente e voluntária dos cidadãos, que através de análises de dados partilham seu conhecimento com cientistas profissionais, visando utilizar esses dados para auxiliar a sociedade nas tomadas de decisão. Focando esse conceito especificamente em parques ambientais ecológicos e aproveitando o avanço tecnológico juntamente com o uso de aparelhos smartphones, há uma grande possibilidade de aumentar o engajamento e o interesse dos cidadãos nos parques ambientais ecológicos [1].

Parques ambientais são unidades de conservação, terrestres e/ou aquáticas, normalmente extensas, destinadas à proteção de áreas representativas de ecossistemas, podendo também ser áreas dotadas de atributos naturais ou paisagísticos notáveis, sítios geológicos de grande interesse científico, educacional, recreativo ou turístico. A finalidade dos parques é resguardar atributos excepcionais da natureza, conciliando a proteção integral da flora, da fauna e das belezas naturais com a utilização para objetivos científicos, educacionais e recreativo. Assim, os parques são áreas destinadas para fins de conservação, pesquisa e turismo. Podem ser criados no âmbito nacional, estadual ou municipal, em terras de seu domínio, ou que devem ser desapropriadas para esse fim [2].

Segundo o Art. 8o., parágrafo 1, da Resolução do CONAMA No. 369/2006, os parques ambientais resguardam áreas verdes públicas urbanas, cuja suas funções são ecológicas, estéticas e sociais. Ecológica é a função principal da floresta bem como a recuperação de ambientes degradados pela industrialização. A fauna da cidade, como as aves, por exemplo, geralmente dependem da arborização para abrigo e alimentação. A estética é a harmonização dos diferentes estilos arquitetônicos existentes nas cidades. A função social é a democratização dos espaços públicos destinados ao lazer e recreação. Além disso, as árvores fazem parte do cotidiano das pessoas, gerando um vínculo delas com a natureza [3].

1.1. Objetivos

1.1.1. Objetivo Geral

Este trabalho tem como objetivo identificar o nível de desengajamento das pessoas em relação aos parques ambientais, modelar e oferecer uma ferramenta computacional para smartphones que estimule e apóie o interesse neste tema, trazendo informações e propondo a participação da comunidade, ao mesmo tempo coletando dados para auxílio à tomada de decisões para os gestores.

1.1.2. Objetivos Específicos

Para alcançar os objetivos gerais, os objetivos específicos deste trabalho são:

1. Fazer um levantamento bibliográfico sobre trabalhos, artigos e teorias existentes relacionados à pesquisa;
2. Estimar o número de pessoas que conhece, visita, utiliza os recursos do(s) parque(s);
3. Estudar e analisar tecnologias, técnicas e teorias existentes para a implementação de um aplicativo para celular que promove conhecimento sobre os parques;
4. Desenvolver um sistema prático, objetivo e simplificado que permita a realização de testes de funcionalidade;

1.2. Revisão Bibliográfica

Alguns artigos e trabalhos anteriores foram estudados, destaque para: o artigo “La percepción que tienen los padres de las oportunidades para los niños en la naturaleza” de Patrícia Maria Schubert Peres, Luana dos Santos Raymundo, Maíra Longhinotti Felippe & Ariane Kuhnen (2017), onde é destacado que existe diferença entre “lugar de criança” e “lugar para criança”. Segundo o artigo, “lugar para criança” é o lugar que os adultos construíram para as crianças, enquanto “lugar de criança”, são lugares que as crianças exploram de maneira autônoma [13].

Com o objetivo de identificar as barreiras de uso dos parques por pais e mães, o artigo PERCEPÇÃO PARENTAL DAS BARREIRAS PARA O CONTATO DA CRIANÇA COM A NATUREZA, de Peres, Felippe e Kuhnen, 2019, mostra um estudo onde em um parque urbano de Florianópolis, 72 pais de crianças entre 6 e 9 anos responderam um questionário auto-aplicado, detectando como principal barreira a falta de disponibilidade dos pais para acompanhar os filhos a esses lugares, porém mostraram-se satisfeitos com a

estrutura física e de modo geral não consideram a distância entre sua residência e a área verde, um fator limitante para acessá la [22].

Outro trabalho utilizado como importante referência, é “RENOVAR: UM MVP PARA MONITORAR A QUALIDADE DO AR” de Francisco Sacco Flores Almeida Teixeira, onde os conceitos de Ciência Cidadã, Cidades Inteligentes e Internet das Coisas são genialmente aplicados conjuntamente na elaboração de uma rede cidadã para controlar indicadores de qualidade do ar, através da participação dos cidadãos, com o objetivo de tornar uma cidade cada vez mais inteligente[14].

Existe uma ferramenta lançada pelo MMA (Ministério do Meio Ambiente), que visa a preservação da biodiversidade, estabelecendo mecanismos que regulamentam a participação da sociedade na gestão das unidades de conservação, potencializando a relação entre o Estado, os cidadãos e o meio ambiente[12]. O aplicativo chama-se Parques do Brasil e pode ser encontrado no Google play[15].

Outro artigo que merece destaque é “French citizens monitoring ordinary birds provide tools for conservation and ecological sciences” de Frédéric Jiguet, Vincent Devictor, Romain Julliard e Denis Couvert, que tem como foco mostrar o monitoramento da migração das aves, levantados dados que possam medir as tendências temporais para detectar possíveis espécies em declínios Combinando índices anuais de espécies que compartilham afinidades ecológicas ou um status de lista protegida / vermelha fornece ainda indicadores de biodiversidade para decisões políticas. O artigo frisa que esse levantamento de dados se dá através da Ciência Cidadã, ou seja, uma parceria entre cientistas e leigos voluntários [5].

O artigo “Efetividade do Crowdsourcing como Apoio à Segurança Pública", de João Moisés Brito Mota e Afonso Carneiro Lima, trata da averiguação do potencial de utilização do crowdsourcing no âmbito da segurança pública no Estado do Ceará, em um período de 5 anos, porém pode ser utilizado em outros contextos, pois o artigo trata o crowdsourcing como um estilo de vida colaborativo, destacando-se por usar o conhecimento das multidões para desenvolver produtos, serviços ou promover a solução de problemas por meio de feedbacks [17].

O website Vivoverde, através da postagem de Daiane Santana, lista alguns aplicativos e plataformas online que possuem o objetivo de conectar as pessoas com o meio ambiente. Santana os trata como Ideias inovadoras aliadas ao avanço de novas tecnologias. É argumentado também que a partir da tela do smartphone ou do computador, todos nós podemos contribuir com levantamentos científicos e ter acesso, por exemplo, a informações para

reconhecer espécies de animais e plantas, o que proporciona experiências ricas e diferenciadas com o meio ambiente. Porém na lista dos *apps* não constam *apps* com as abordagens de Ciência Cidadã, nem de Crowdsourcing. Os apps da lista serão abordados na seção “Soluções Existentes”.[18]

1.3. Método de Pesquisa

Para alcançar os objetivos deste projeto, foi adotada uma metodologia de trabalho que consiste nas seguintes etapas:

- Revisão bibliográfica (item anterior): Estudo sobre artigos, trabalhos anteriores, matérias de jornais relacionadas ao tema. A revisão tem o intuito de validar o tema estudado através de análises e comparativos com trabalhos paralelos;
- Coleta de dados: Coleta de dados: Realização de enquetes com pessoas, visitas a parques ecológicos. O objetivo da coleta de dados é estimar o nível de conhecimento, utilização e interesse em relação aos parques, validando um possível problema ou uma causa social e, posteriormente comparada com ocasiões encontradas na revisão bibliográfica.
- Análise de tecnologias: Etapa de estudo sobre plataformas, portabilidade, ambiente de desenvolvimento, linguagens e banco de dados.
- Analisar e projetar a aplicação: Etapa de idealização do aplicativo juntamente com a definição de seus requisitos.;
- Desenvolvimento do aplicativo: Após o esclarecimento do que o aplicativo deve fazer, quais tecnologias utilizar, o aplicativo estará apto a ser desenvolvido;

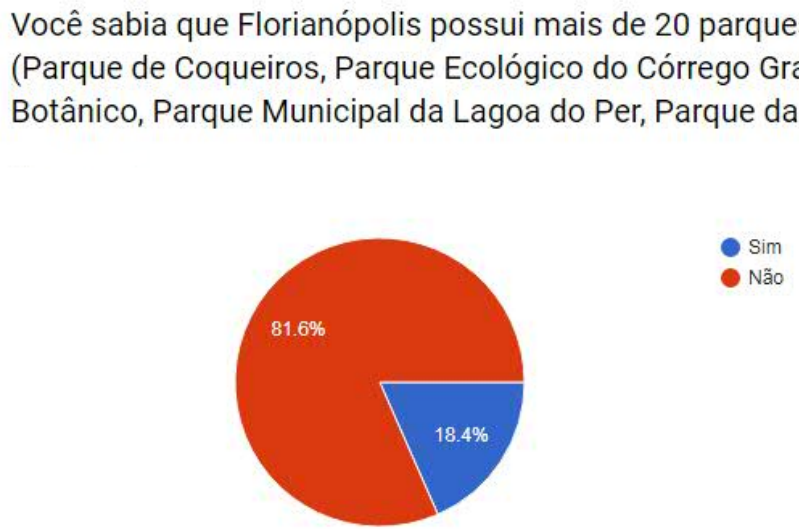
Existem vários projetos de Ciência Cidadã [7] no portfólio do SiBBr (Sistema de Informação sobre a Biodiversidade Brasileira) que se encontra em seu portal, bem como uma forma de participar voluntariamente destes projetos. Como o trabalho está sendo desenvolvido em Florianópolis-SC, a cidade foi tomada como base de pesquisa, até mesmo por possuir algumas opções de parques ou reservas ambientais disponíveis para visitação, sendo alguns deles:

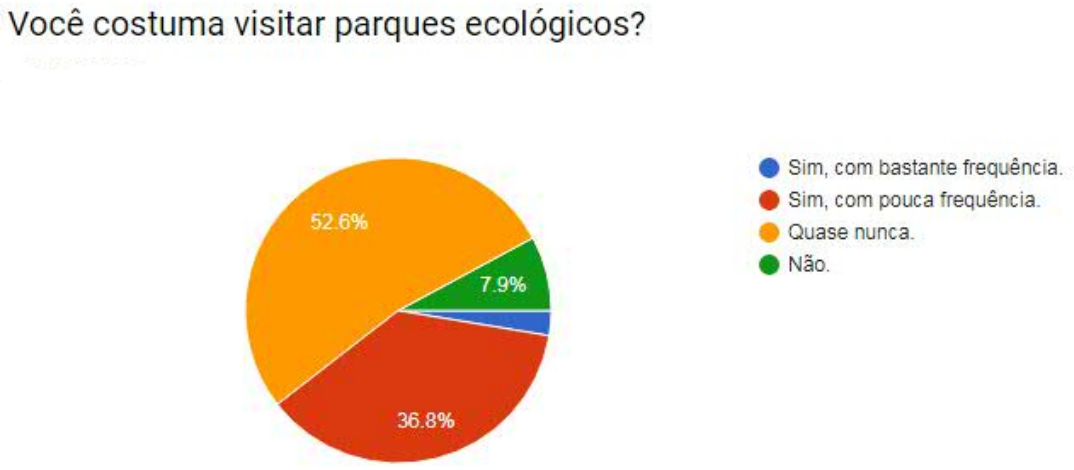
- Parque Ecológico do Córrego Grande, com área de 21,3 hectares, integralmente em área urbana. Na década de 1990 ficou alguns anos fechado ao público, sendo reaberto à visitação no dia 3 de dezembro de 2001 [8];
- Jardim Botânico de Florianópolis é um parque público localizado no bairro Itacorubi, próximo ao Manguezal do Itacorubi. Possui 190 mil metros quadrados, está em funcionamento desde 2016, porém está em fase inicial e com estrutura inacabada [9];

- Parque Municipal da Lagoa do Peri, criado por lei municipal em 1982, possui três trilhas ecológicas e cachoeiras, ocupa uma área de 2030 hectares, sendo 5 quilômetros quadrados de espelho de água, cuja profundidade máxima é de 11 metros [10];

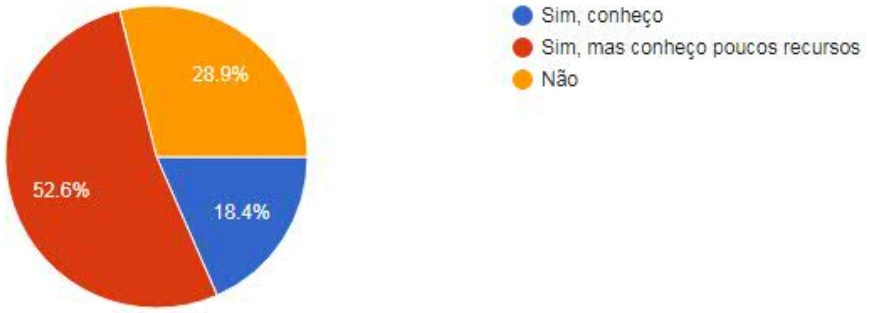
1.3.1. Análise do Problema

Com o objetivo de detectar se há desengajamento das pessoas em relação a parques ambientais, tendo como base as informações contidas nos artigos citados na revisão bibliográfica, foi realizada uma enquete entre os moradores de Florianópolis e região, visando coletar algumas informações relativas ao nível de informação dos cidadãos em relação aos parques ambientais. As respostas foram coletadas através de um formulário virtual divulgado nas redes sociais em dois momentos, primeiramente entre os meses de maio e junho de 2019, depois entre janeiro e maio de 2021. Em ambas as coletas o padrão de respostas manteve-se o mesmo, mostrando que a pandemia do novo coronavírus não impactou no conhecimento e utilização dos parques:

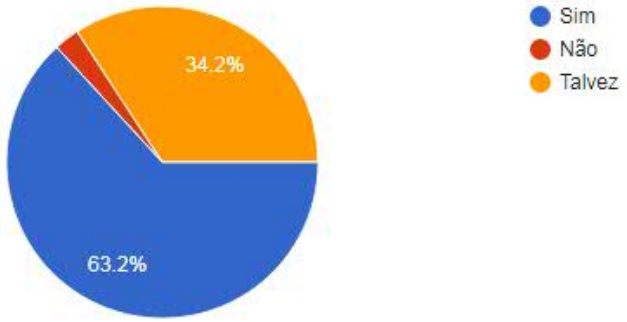




Você conhece todos os recursos do parque que você visita? (Lixeiras, locais para crianças, piqueniques, trilhas, locais esportivos, mesas, bancos, onde encontrar determinado animal ou vegetal)



Se existisse alguma ferramenta que coletasse informações sobre o parque e te auxiliasse na tomada de decisão sobre os recursos, você utilizaria essa ferramenta?



Com base nas respostas dos cidadãos, percebe-se pouco conhecimento entre os entrevistados em saber se existe uma variedade de parques em Florianópolis, apesar de apenas 8,6% dos entrevistados não terem o hábito de visitá-los. Nota-se também que apenas 2,6% dos entrevistados visitam os parques ecológicos com bastante frequência e 52,6% quase nunca faz a visitação. Esse alto índice de não visitação pode ter relação com a barreira encontrada no artigo PERCEPÇÃO PARENTAL DAS BARREIRAS PARA O CONTATO DA CRIANÇA COM A NATUREZA, de Peres, Felippe e Kuhnen, 2019, barreira esta que é a falta de disponibilidade dos pais para acompanhar os filhos a áreas verdes urbanas. Assim como as pessoas que participaram do questionário no estudo citado (pais de crianças de 6 a 9 anos de idade), há possibilidade de que outras pessoas também possuam essa mesma barreira.

Tratando-se dos recursos que os parques dispõem, destacamos que apenas 18,4% dos entrevistados conhece totalmente, e 28,9% não conhece nenhum, essa pode ser a possível causa do baixo índice de interesse.

Em relação à utilização da ferramenta proposta, 63,2% dos entrevistados a utilizaria, e 34,2% talvez a utilizasse. Com base nas respostas, pode-se afirmar que possivelmente a ferramenta contribuiria com o aumento do interesse nos parques, pois o valor percentual de quem a utilizaria é maior do que o valor percentual da baixa frequência nos parques.

1.4. Área de Estudo

Em Florianópolis, segundo o site da prefeitura da cidade, existem mais de vinte parques ecológicos, são alguns deles: Jardim Botânico de Florianópolis, Parque Ponta do Ataliba, Parque da Luz, Parque de Coqueiros, Bosque Pedro Medeiros, Parque Municipal do Córrego Grande, Parque Municipal da Lagoa do Peri, Poção, além de vários outros parques ou reservas ambientais na região. Pelo fato deste projeto estar sendo desenvolvido em Florianópolis-SC, um parque ecológico da cidade foi escolhido como ambiente de teste do aplicativo, o Parque Municipal do Córrego Grande. O principal critério utilizado para a escolha foi a localização, situa-se próximo à UFSC (Universidade Federal de Santa Catarina).

1.5. Ciência Cidadã, Gestão Ambiental e Crowdsourcing

1.5.1. Ciência Cidadã

Cidadão é o habitante da cidade, e tem o direito de gozar de seus direitos civis e políticos do Estado em que nasceu, ou no desempenho de seus deveres para com este. O cidadão ao ter consciência e exercer seus direitos e deveres para com a pátria está praticando a cidadania [11].

Pode-se definir Ciência Cidadã como uma parceria entre cidadão leigo e cientista na coleta de dados. Ao redor do mundo, há centenas de milhares de “cidadãos cientistas”, pessoas que, em seu tempo livre, decidiram se dedicar à ciência [4].

O principal objetivo da Ciência Cidadã é a transformação de dados de linguagem técnica/científica do ambiente, de difícil interpretação aos cidadãos comuns, em dados de linguagem leiga, auxiliando sua tomada de decisão. A Ciência Cidadã é uma ferramenta científica eficiente e, além disso, tem um grande custo-benefício, pois, segundo Jiguet ET AL. 2012, um monitoramento de longo prazo de larga escala em um grupo taxonômico completo (Aves) oferece a oportunidade de comparar diferentes medidas de diversidade biológica, tais como: estudos taxonômicos, filogenéticos; e diversidade

funcional, tal atividade necessitaria de uma equipe reduzida na instituição que analisa os dados, enquanto um trabalho similar conduzido por uma equipe contratada para a coleta de dados custaria mais de um milhão de euros anualmente[5]. Através dessa ponte entre os cientistas profissionais e o público, a ciência cidadã apresenta resultados positivos tanto para a ciência quanto para o aprendizado do público (Dickinson and Bonney, 2012) [4].

1.5.2. Gestão Ambiental

Um “sistema que inclui atividades de planejamento, responsabilidades, processos e recursos para desenvolver, implementar, atingir, analisar criticamente e manter a política ambiental”. Esse é o conceito de Gestão Ambiental, segundo Tinoco, no livro Contabilidade e Gestão Ambiental [6].

O gestor ambiental é responsável por estudar a relação do homem com a natureza com o objetivo de possibilitar o uso adequado dos recursos naturais de modo a preservar o meio ambiente. Além de analisar os impactos das atividades humanas no meio ambiente, o desenvolvimento de programas de educação e proteção ambiental é outra atividade que pode ser desempenhada por esse profissional [6].

Através da motivação pela participação do público na pesquisa científica, a Ciência Cidadã tem potencial para aumentar a participação do público na Gestão Ambiental [4].

1.5.3. Crowdsourcing

Crowdsourcing é um modelo de produção e de estruturação de processos que utiliza a sabedoria e os aprendizados coletivos para a resolução de problemas ou desenvolvimento de uma solução. Segundo o portal Liga Insights, a primeira aparição do termo ocorreu na revista Wired, em uma artigo de Mark Robinson e Jeff Howe. Os autores usaram as palavras crowd (multidão) e outsourcing (terceirização) para representar algo que é construído por meio da união de um grupo de pessoas, ou seja, da construção coletiva. Portanto, os projetos elaborados utilizando crowdsourcing de multidisciplinaridade podem contemplar as seguintes características:

- Colaboração: Envolvimento de mais pessoas na coleta e troca de dados e informações;
- Conhecimento Compartilhado: A Troca e participação coletiva das informações ficam disponíveis para todos;
- Fazer parte: Por estar colaborando, a pessoa passa a ser e sentir-se parte do projeto; [16].

.1.6. Soluções Existentes

O Ministério do Meio Ambiente (MMA) lançou no final de 2018 o aplicativo Parques do Brasil, que reúne informações sobre as principais unidades de conservação (UCs) do país, o *app* Parques do Brasil. O Aplicativo reúne informações sobre as principais unidades de conservação do país, aquelas responsáveis pelo maior fluxo de visitantes. Pela ferramenta, é possível pesquisar informações sobre as unidades de conservação mais próximas do usuário, incluindo orientações sobre como chegar, atrativos, descrição das trilhas, atividades disponíveis, o bioma da unidade, as principais espécies protegidas, condições de acessibilidade e preços de ingressos [12].

Atualmente o app encontra-se em versão Beta, contendo informações sobre 30 unidades de conservação, não sendo possível cadastrar outras.

Conforme visto na seção Revisão Bibliográfica deste trabalho, o website Vivoverde, através da postagem de Daiane Santana, lista alguns aplicativos e plataformas online que possuem o objetivo de conectar as pessoas com o meio ambiente, porém esses aplicativos não utilizam as abordagens de Ciência Cidadã e Crowdsourcing:

- iNaturalist: Criada para contribuir com o banco de dados ambiental, esta plataforma permite que o usuário compartilhe a foto de um animal ou uma planta para que outros usuários ajudem na identificação das espécies. O aplicativo tem versões exclusivas para alguns estados brasileiros para estimular a observação e o mapeamento da biodiversidade em cada região. Para usar, é preciso fazer o download gratuito do aplicativo iNaturalist, preencher o cadastro e procurar pelo projeto em cada região. Disponível em português;

- Pl@ntNet Identificação Planta: Com um sistema de reconhecimento visual, o Pl@ntNet ou Leafsnap é uma ferramenta usada para a identificação de plantas. Basta o usuário fotografar a folha ou a flor da planta desconhecida e o aplicativo identifica a espécie. A plataforma é gratuita e foi desenvolvida por investigadores da Universidade de Columbia. Disponível em português;

- WikiParques: Plataforma on-line interativa e colaborativa para compartilhar conhecimentos, explorar e debater sobre parques

nacionais e outras áreas protegidas brasileiras. Todos os parques nacionais do Brasil estão cadastrados no sistema e o usuário pode consultá-lo para programar passeios às áreas naturais. Disponível em português;

- All Trails: Com GPS integrado, o aplicativo possui mapas de 50 mil trilhas, incluindo avaliações, fotos e informações de acessibilidade. Ele possui recomendações sobre quais trilhas são pet friendly e quais são ideais para passeios com crianças. O download do app é gratuito, mas algumas operações dentro dele são pagas, com custos variáveis. Disponível em inglês.

- Sistema Urubu (aplicativo e site): O Sistema Urubu tem a proposta de reunir, sistematizar e disponibilizar informações sobre a mortalidade de fauna selvagem em rodovias e ferrovias com o objetivo de auxiliar governos e concessionárias na tomada de decisão para a redução desses impactos. Com acesso gratuito, o sistema é alimentado com dados vindos de vários públicos e analisados por especialistas em identificação de espécies. Disponível em português.

- Offline Survival Manual: Gratuito, o Manual de Sobrevivência Offline é indicado para os mais aventureiros. O aplicativo ensina a viver na natureza, com informações sobre primeiros socorros básicos, construção de abrigo, busca por alimentos, entre outras. Disponível em inglês. [18]

2. ANÁLISE E PROJETO

2.1. Solução Proposta

O projeto Ciência Ambiental Cidadã visa tratar os problemas encontrados nos dois primeiros questionamentos, sobre o conhecimento da diversidade de parques e a frequência nas visitações, respectivamente. A solução tem como premissa as respostas das duas últimas perguntas, sobre o conhecimento dos recursos que os parques dispõem e sobre a possibilidade de utilização de uma ferramenta de coleta e disponibilização de dados sobre os recursos dos parques, respectivamente.

A solução consiste em desenvolver um aplicativo móvel prático, objetivo e simplificado para que a comunidade como um todo tenha condições de utilizá-lo.

2.2. Requisitos do Aplicativo

2.2.1. Requisitos Funcionais

- Logar e deslogar: Utilizar o app apenas se estiver logado;
- Cadastrar novos parques: Cada parque deve possuir: nome, estado, cidade, bairro, descrição, id da pessoa que o cadastrou e uma lista de elementos. O parque será cadastrado se houver cinco solicitações de cadastro para o mesmo nome, estado e cidade;
- Remover parques: O parque será removido se houver cinco solicitações para remoção;
- Exibir parques: O parque será exibido, bem como suas informações, se estiver devidamente cadastrado (a partir de cinco solicitações de cadastro);
- Cadastrar novo elemento ao parque: Cada elemento deve possuir: nome, tipo (animal, vegetal ou utensílio), informações e foto. No ato de cadastrar o elemento, será obrigatório o acesso à câmera para registrar a foto do elemento;
- Exibir elemento: Serão exibidos todos os elementos relacionados ao parque selecionado, bem como seus dados (nome, tipo, informações e foto);
- Remover elemento: O elemento selecionado será removido;

2.2.2. Requisitos Não funcionais

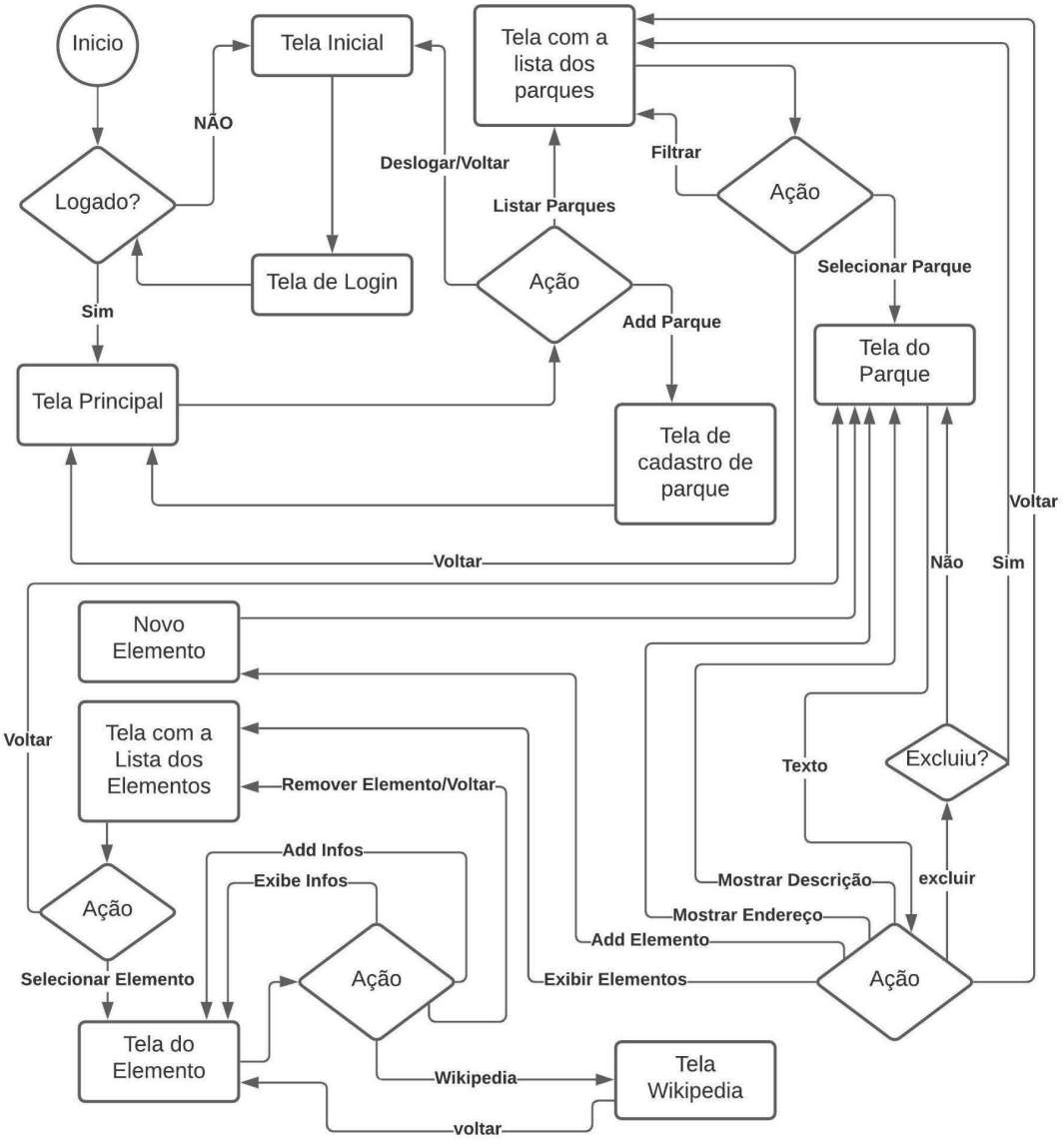
- Arquitetura: Cliente-Servidor;
- Idioma: Portugês-BR;
- Plataforma: Android;
- IDE de implementação: Android Studio;
- Linguagens de implementação: Java, XML;
- Banco de Dados: Firebase Realtime Database;
- Storage para armazenamento de fotos: Firebase Storage;
- API para autenticação: Firebase Authentication;

2.3. Fluxo de Usuário (User Flow)

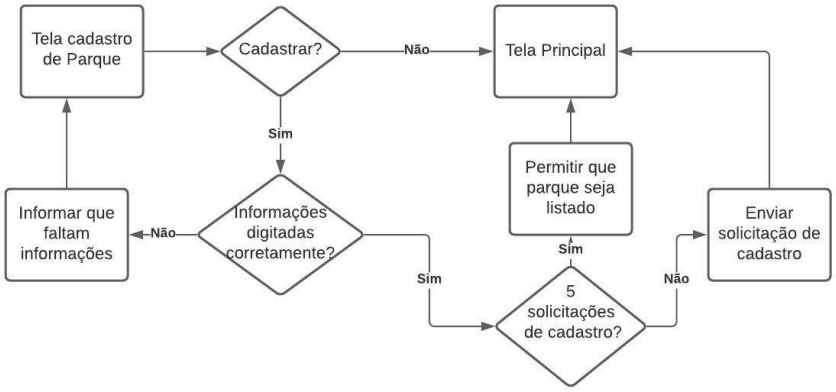
Com o intuito de construir o mapeamento total do fluxo de telas referente ao app projetado, tendo como base os requisitos funcionais acima, foi

modelado fluxo de usuário abaixo (user flow). O mapeamento servirá como representação gráfica, englobando todas as interações que o usuário terá com o aplicativo.

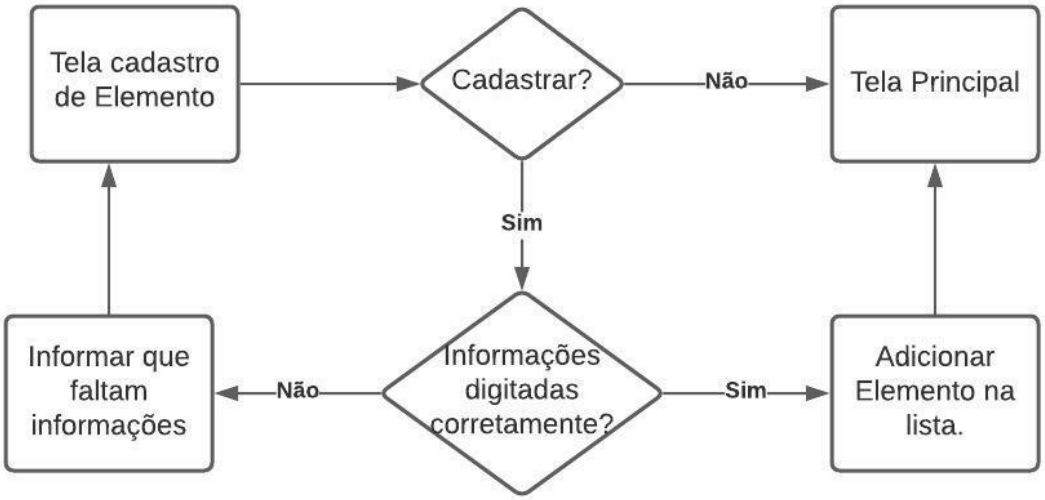
Abaixo segue o user flow de visão geral do aplicativo.



Segue abaixo o user flow do cadastramento de parque.



Abaixo segue o user flow do cadastramento de novo elemento.



3. Desenvolvimento da Aplicação

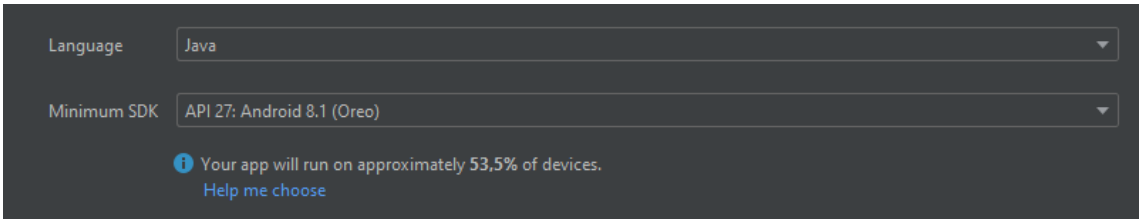
3.1. Front-end

O lado front-end da aplicação foi totalmente implementado utilizando a IDE Android Studio, optou-se pela plataforma android pois, segundo o site canaltech, em publicação feita por Bruna Rasmussen, 70,1% dos smartphones vendidos no mundo possuem esse sistema operacional. [19]

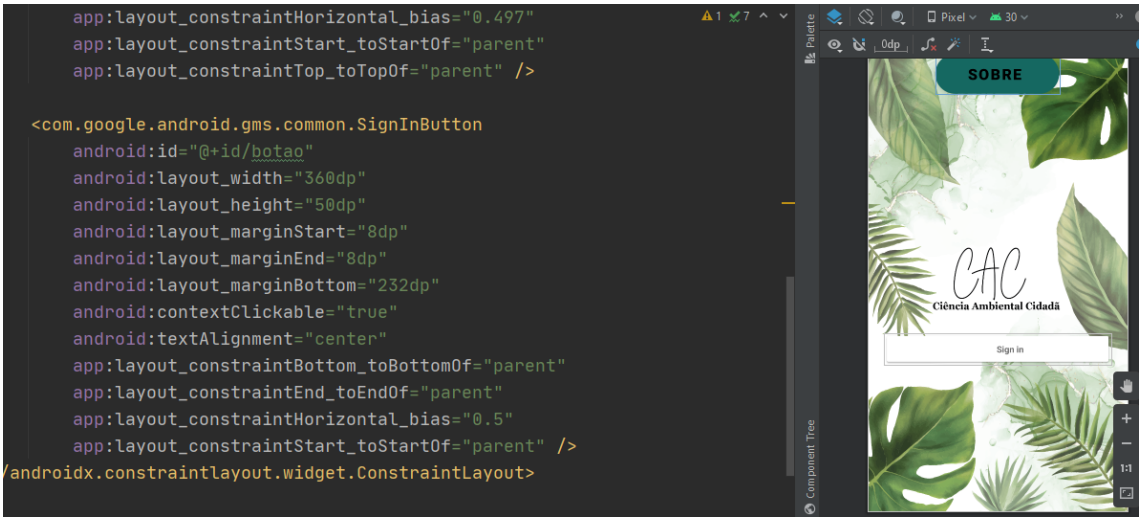
A linguagem escolhida para implementar a parte lógica do aplicativo foi Java juntamente com o XML para a parte de GUI, por se tratar da linguagem mais trabalhada durante o período de graduação, além de ser nativa em programação para Android. Após escolher a plataforma, IDE e linguagem (previstas nos requisitos não funcionais), foram analisadas algumas

possibilidades de Android SDK, que trata-se do Kit de Desenvolvimento de Software para Android. Esse kit permite aos desenvolvedores criarem aplicativos para a plataforma Android de forma nativa. O Android SDK inclui projetos de exemplo com código-fonte, ferramentas de desenvolvimento, emuladores e bibliotecas necessárias para criar os aplicativos Android. [20]

Por fim, o SDK escolhido foi API 27: Oreo, ou Android 8.1, Optou-se por essa SDK, pelo fato de ter um desempenho relativamente bom, e pode ser executada por cerca de 53,5% dos dispositivos, segundo a própria IDE na imagem a seguir.



Ao iniciar o app, será verificado se o smartphone está com alguma conta google logado, caso não esteja, carregará a tela MainActivity:



O botão “sobre” tem como função exibir uma mensagem em tela informando algumas informações sobre o aplicativo, já o botão Sign In irá acionar o Back-End solicitando login através da API do Google Firebase Authentication, carregando a tela de login da própria google.

Caso o smartphone já esteja com alguma conta google logada, bastará se logar uma vez no app, após isso, todas as vezes em que for inicializado irá carregar direto a TelaInicial, como mostram os métodos a seguir:

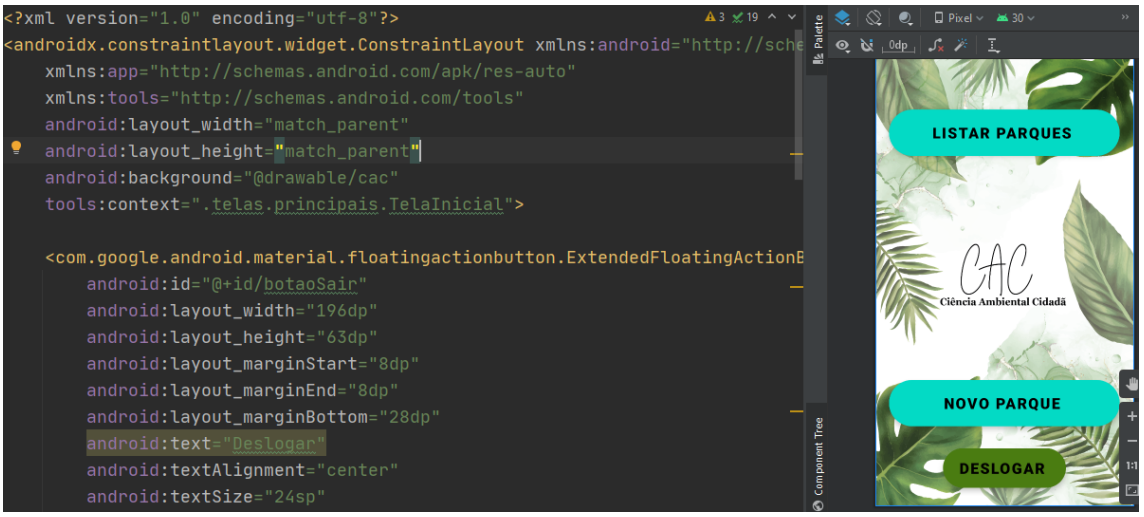
```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    getSupportActionBar().hide();
    mAuth = FirebaseAuth.getInstance();
    ad = new AlertDialog.Builder( context: this);
    botao = (SignInButton) findViewById(R.id.botao);
    botao.setOnClickListener(view -> {
        signIn();
    });
    requisita();
}

private void requisita() {
    GoogleSignInOptions gso = new GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
            .requestIdToken("[redacted]")
            .requestEmail()
            .build();
    mGoogleSignInClient = GoogleSignIn.getClient( activity: this, gso);
}
```

```
private void firebaseAuthWithGoogle(String idToken) {
    AuthCredential credential = GoogleAuthProvider.getCredential(idToken,  accessToken: null);
    mAuth.signInWithCredential(credential)
            .addOnCompleteListener( activity: this, new OnCompleteListener<AuthResult>() {
                @Override
                public void onComplete(@NonNull Task<AuthResult> task) {
                    if (task.isSuccessful()) {
                        Intent i = new Intent( packageContext: MainActivity.this, TelaInicial.class);
                        startActivity(i);
                    } else {
                        ad.setTitle("Falha de Autenticação!");
                        ad.setMessage("Usuário ou Senha Incorreta");
                        ad.show();
                    }
                }
            });
}
```

```
@Override
public void onStart() {
    super.onStart();
    FirebaseUser currentUser = mAuth.getCurrentUser();
    if (currentUser != null) {
        Intent i = new Intent( packageContext: MainActivity.this, TelaInicial.class);
        startActivity(i);

    }
}
```

```java
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == RC_SIGN_IN) {
        Task<GoogleSignInAccount> task = GoogleSignIn.getSignedInAccountFromIntent(data);
        try {
            GoogleSignInAccount account = task.getResult(ApiException.class);
            firebaseAuthWithGoogle(account.getIdToken());
            SharedPreferences.Editor editor = getApplicationContext()
                    .getSharedPreferences( name: "Pref", MODE_PRIVATE)
                    .edit();
            editor.putString("username", account.getDisplayName());
            editor.putString("userid", account.getIdToken());
            editor.putString("useremail", account.getEmail());
            //editor.putString(("foto"),account.getPhotoUrl().toString());
            editor.apply();
        } catch (ApiException e) {
            ad.setTitle("Falha de Login!");
            ad.setMessage("Algo inesperado aconteceu!");
            ad.show();
        }
    }
```

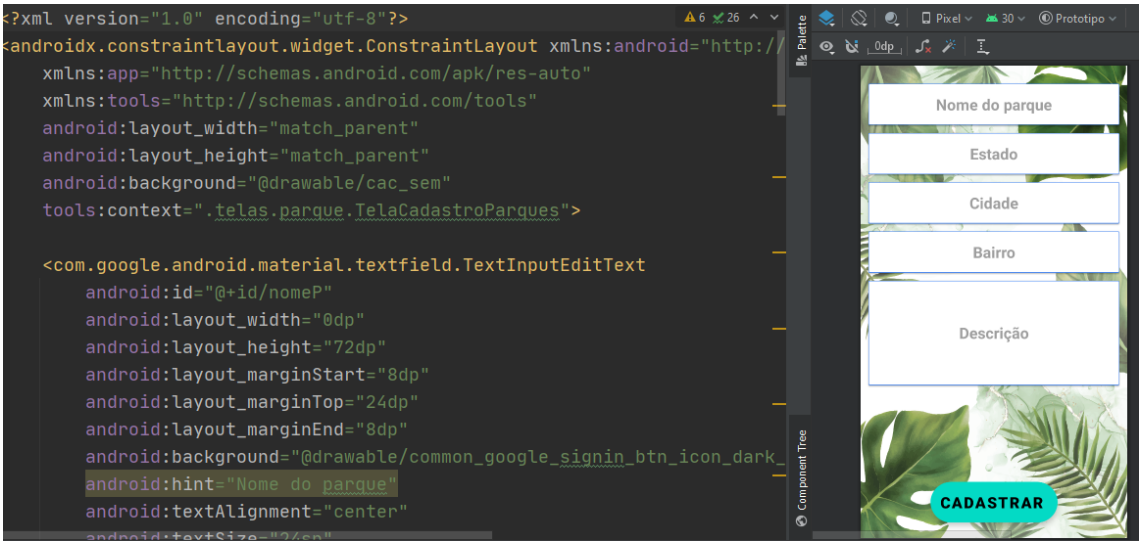
Já na TelaInicial, temos três opções: listar os parques que estão cadastrados, cadastrar um novo parque, ou deslogar, como mostra a imagem abaixo.



Ao clicar em ‘DESLOGAR’, o app solicitará ao back-end novamente uma interação do Firebase Authentication, executando o método “signOut()” e retornará para a MainActivity. Se clicar em ‘NOVO PARQUE’ mudará para a TelaCadastroParques, e se clicar em ‘LISTAR PARQUES’ mudará para a TelaListaParques, como mostra o código abaixo.

```
mAuth = FirebaseAuth.getInstance();
deslogar.setOnClickListener( view->{
    mAuth.signOut();
    Intent i = new Intent( packageContext: TelaInicial.this, MainActivity.class);
    startActivity(i);
});
novo.setOnClickListener(view ->{
    Intent i = new Intent( packageContext: TelaInicial.this, TelaCadastroParques.class);
    startActivity(i);
});
lista.setOnClickListener(view ->{
    Intent i = new Intent( packageContext: TelaInicial.this, TelaListaParques.class);
    startActivity(i);
});
```

A TelaCadastroParques é responsável por criar um objeto do tipo Parque, inicializar os atributos preenchidos nos campos, conforme imagem abaixo.



Durante o ato de cadastro, é verificado no banco de dados se já existe um objeto com mesmo nome, cidade e estado cadastrado. Caso exista, é verificado se o ID do usuário que o cadastrou é o mesmo ID do usuário logado, caso seja, o app irá informar que o usuário já solicitou o cadastro do referido parque. Caso o parque esteja cadastrado, mas o ID do usuário seja diferente, o app irá incrementar o contador de ocorrências de cadastro do parque, abrirá um alerta com texto editável para confirmar a descrição do parque e, por fim salvar no banco, quando o contador chegar no valor 5 o parque estará “aprovado” e será listado na TelaListaParques. O contador de ocorrências foi implementado com o intuito de minimizar o cadastro de parques inexistentes. Já, se o nome, cidade e estado (simultaneamente) não constam no banco, o objeto criado será inserido no mesmo, através do método “salvar()”.

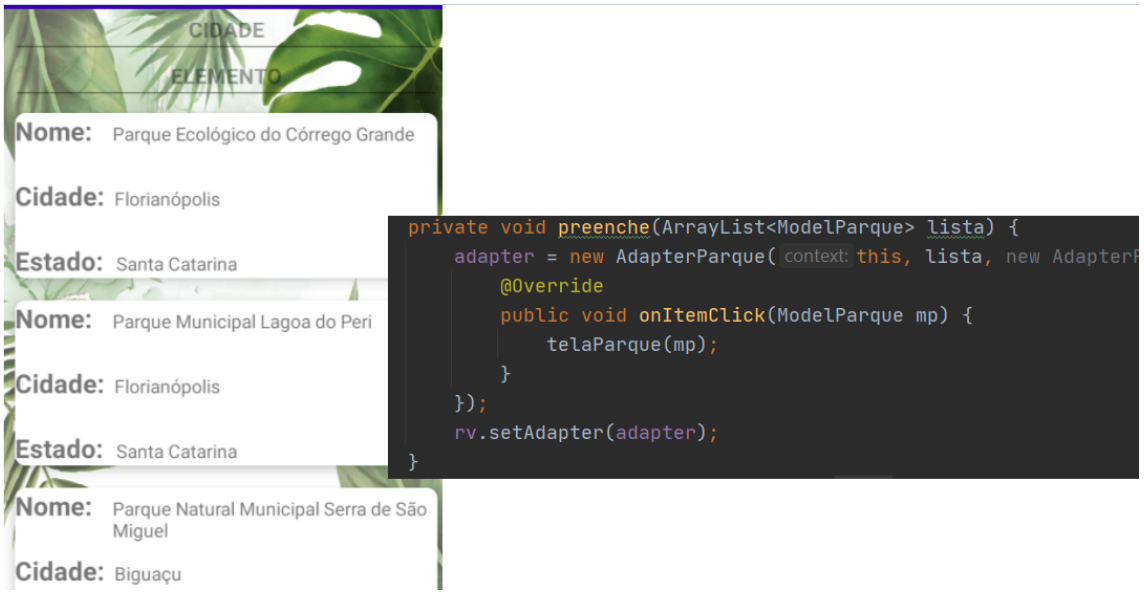
```
if (verificaSeExiste(nome, estado, cidade)) {                    ⚠ 12  ✓ 45
    if (!p.getUserId().equals(userId)) {
        mensagem = "O cadastro desse parque já foi solicitado!";
        if (p.getCont() == 4) {
            p.incrementaCont();
            mensagem = "Parque aprovado!";

        } else if (p.getCont() > 4) {
            mensagem = "Esse parque já está cadastrado!";
        } else {
            p.setUserId(userId);
            p.incrementaCont();
        }
        editaInfos();
        p.salvar();
    } else {
        mensagem = "Você já solicitou o cadastro desse parque!";
    }
} else {
    Parque p = new Parque(nome, estado, cidade, bairro, userId, descricao);
    p.salvar();
```

O método “salvar()” encontra-se na classe "Parque". Sua função é fazer o upload do objeto do tipo parque, carregando os atributos que possuem o método ‘get’, é mais uma interação com o Firebase, conforme o código abaixo.

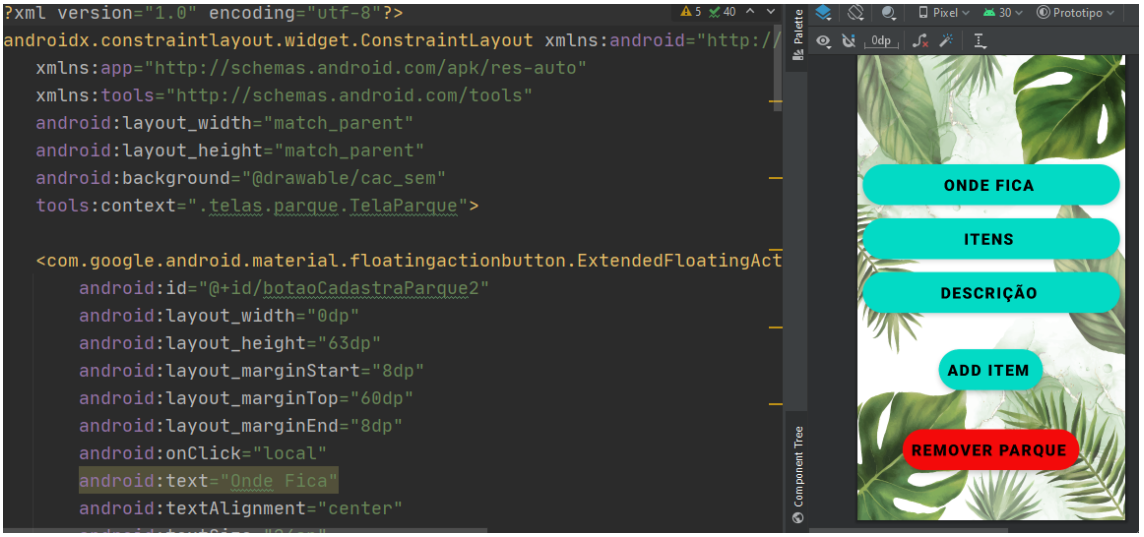
```
public void salvar() {
    DatabaseReference reference = FirebaseDatabase.getInstance().getReference();
    reference.child("parques").child(nome).setValue(this);
}
```

A TelaListaParques é responsável por listar os parques no formato de RecyclerView, um widget do android semelhante à uma lista, mas possui a possibilidade de exibir mais de um atributo em sua visualização, como na imagem a seguir. Nesta tela também constam dois modos de filtro, por cidade e por elemento.

No código abaixo, o ‘rv’ é o objeto referente ao RecyclerView e ‘adapter’ é o objeto responsável por adaptar o objeto referente ao Parque, para que seja compatível com o RecyclerView. Ao clicar em cima de um item da lista, irá chamar o método ‘OnItemClick(), carregando a TelaParque.

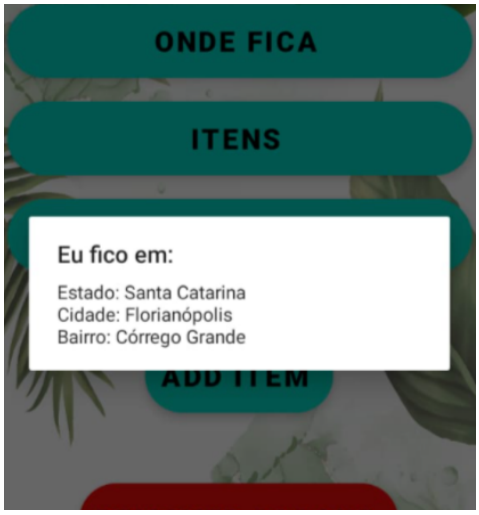


Abaixo temos a TelaParque.

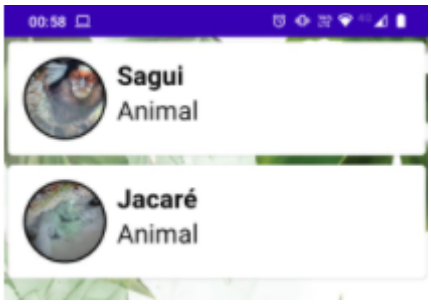


A TelaParque possui cinco funções.

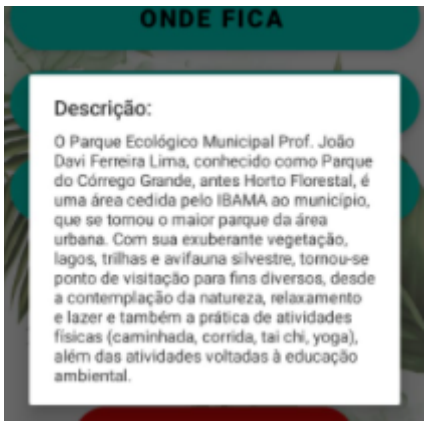
- Botão “ONDE FICA” exibe uma mensagem com estado, cidade e bairro do parque;



- Botão “ITENS” abre a TelaListaElementos que possui um RecyclerView contendo os itens do parque, essa tela possui função semelhante à TelaListaParques. Ao clicar em cima de um item, irá chamar o método ‘onItemClick()’ carregado a TelaFotoElemento referente ao item clicado;



- Botão “DESCRIÇÃO” exibe uma mensagem contendo a descrição atribuída ao parque no ato de cadastro;



- Botão “ADD ITEM” abre a TelaCadastroElemento abaixo;

```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/cac_sem_opacidade"
    tools:context=".telas.parque.TelaCadastroElemento">

    <de.hdodenhof.circleimageview.CircleImageView
        android:id="@+id/foto"
        android:layout_width="340dp"
        android:layout_height="340dp"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:layout_marginStart="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="24dp"
        android:onClick="foto"
```

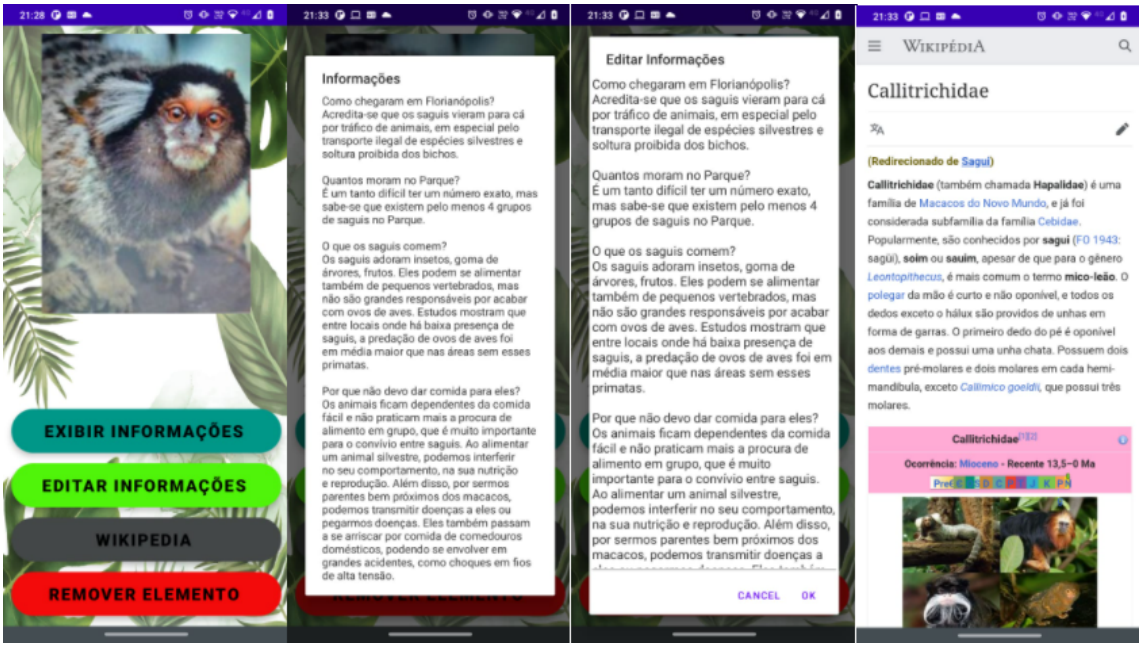
Ao clicar na imagem da câmera, será aberta a função de câmera do celular do usuário, só sendo possível cadastrar um novo elemento, tendo todos os valores preenchidos (inclusive o da foto). O botão “CADASTRAR” tem como função criar um novo objeto da classe ‘Elemento’, adicionar à um arraylist contido dentro do objeto referente ao parque e fazer o upload da imagem utilizando o Firebase Storage, em seguida retorna para a TelaParque, conforme código a seguir.

```java
public void cadastra(View view) {
    if (imagem != null && nome.getText() != null && (vegetal.isChecked() ||
            animal.isChecked() || utensilio.isChecked())) {
        atualizaCaminhoFoto();
        String tipo;
        if (vegetal.isChecked()) {
            tipo = "Vegetal";
        } else if (animal.isChecked()) {
            tipo = "Animal";
        } else {
            tipo = "Utensílio";
        }
        Elemento e = new Elemento(nome.getText().toString(), tipo, caminhofoto);
        list.add(e);
        SharedPreferences preferences = getSharedPreferences( name: "Pref", MODE_PRIVATE);
        String userId = preferences.getString( key: "userid",  defValue: "");
        Parque p = new Parque(list, mp.getNome(), mp.getEstado(), mp.getCidade(), mp.getBairro(), userId,
        p.salvar();
        telaParque();
    } else {
        Toast.makeText( context: TelaCadastroElemento.this,  text: "Você deixou algum campo em branco,\nou esque
```

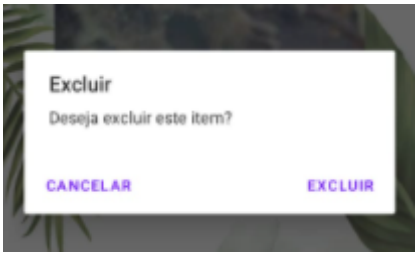
- Botão “REMOVER PARQUE” da TelaParque incrementa o contador de ocorrências contido no objeto referente ao parque, caso chegue até dez (cinco ocorrências), o parque será deletado do banco, conforme código a seguir.

```java
public void deletaParque(View view) {
    if (mp.getCont() < 10) {
        SharedPreferences preferences = getSharedPreferences( name: "Pref", MODE_PRIVATE);
        String userId = preferences.getString( key: "userid",  defValue: "");
        Parque p = new Parque(mp.getElementos(), mp.getNome(), mp.getEstado(),
                mp.getCidade(), mp.getBairro(), userId,
                mp.getDescricao(), mp.getCont());
        p.incrementaCont();
        p.salvar();
        Toast.makeText( context: TelaParque.this,  text: "Solicitação de remoção realizada com sucesso!",
                Toast.LENGTH_SHORT).show();
        Intent i = new Intent( packageContext: TelaParque.this, TelaInicial.class);
        startActivity(i);
    } else {
        DatabaseReference reference = FirebaseDatabase.getInstance().
                getReference().child("parques/" + mp.getNome());
        Toast.makeText( context: TelaParque.this,  text: "Solicitação de remoção atendida com sucesso!",
                Toast.LENGTH_SHORT).show();
        reference.removeValue();
        Intent i = new Intent( packageContext: TelaParque.this,TelaInicial.class);
        startActivity(i);
    }
```

A TelaFotoElemento é responsável por exibir a foto do elemento, bem como exibir e editar informações sobre o elemento, pesquisar o nome do elemento no site da Wikipedia, e por fim, excluir o elemento.



Ao clicar no botão “REMOVER ELEMENTO”, aparecerá uma tela de confirmação, questionando se o usuário realmente deseja deletar aquele item, conforme imagem a seguir.



Se o usuário optar por excluir, o app removerá o elemento do arraylist contido no objeto referente ao parque, também criará uma interação com o Firebase Storge para remover a foto do elemento, conforme código abaixo.

```
adb.setPositiveButton( text: "Excluir", new DialogInterface.OnClickListener()
    @Override
    public void onClick(DialogInterface dialog, int which) {
        deleteImagem(me);
        for (Elemento e : mp.getElementos()) {
            if (e.getCaminhoFoto() == me.getCaminhoFoto()) {
                mp.getElementos().remove(e);
                break;
            }
        }
        //mp.getElementos().remove(me);
        atualizaParque();
        telaParque();
    }
});
adb.show();
```

```
public void deleteImagem(Elemento me) {
    StorageReference storageRef = FirebaseStorage.getInstance()
            .getReference()
            .child(me.getCaminhoFoto());
    storageRef.delete().addOnSuccessListener(new OnSuccessListener<Void>() {
        @Override
        public void onSuccess(Void unused) {
            Toast.makeText( context: TelaFotoElemento.this,  text: "Item removido
            finish();
        }
    }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {
            Toast.makeText( context: TelaFotoElemento.this,  text: "Falha ao remov
        }
    });
}
```

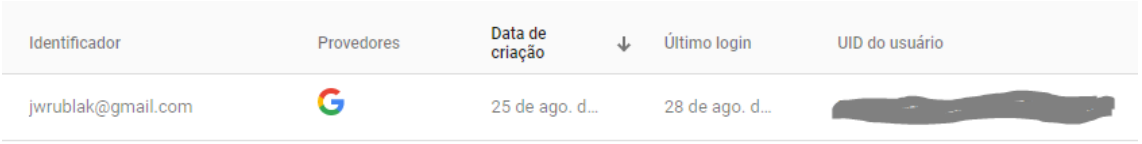
3.2. Back-End - Google Firebase

O Firebase é a plataforma de desenvolvimento de aplicativos móveis do Google. Essa plataforma possui versatilidade e os tipos de aplicativos que podem ser desenvolvidos com Firebase são: Android, iOS e Web.

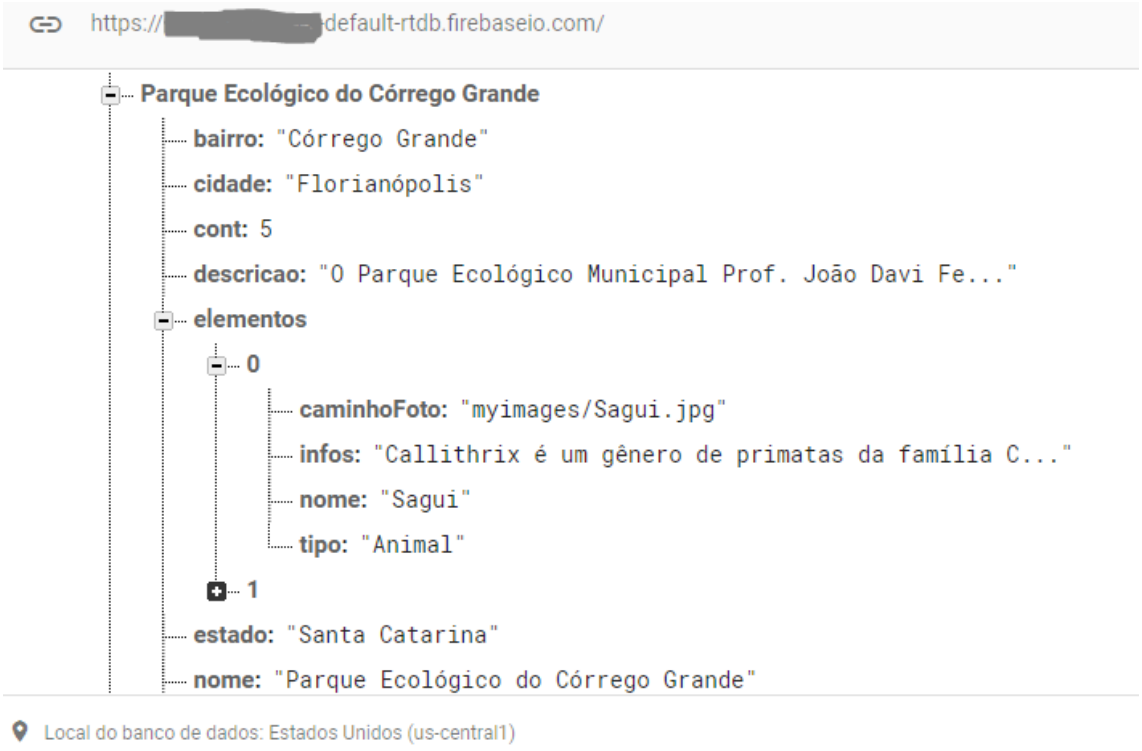
Toda sua base é construída na infraestrutura do Google, sendo categorizado como um programa de banco de dados NoSQL, que armazena dados em documentos do tipo JSON. [21]

O Firebase conta um grande conjunto de ferramentas de desenvolvimento. Destas, foram utilizadas três neste trabalho, são elas:

- Firebase Authentication: API de autenticação utilizada na mainActivity e na TelaInicial, conforme mostrado na seção de Front-End. Ao se logar, o usuário manda um ID para o servidor, onde é possível realizar o controle de usuários do sistema.



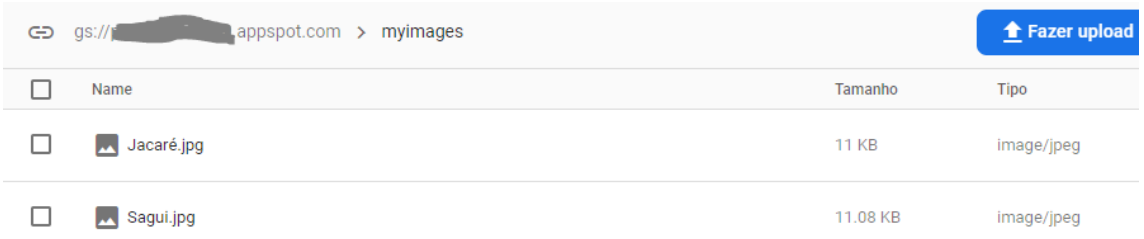
- Firebase Realtime Database: API de gerenciamento de banco de dados não relacional, foi utilizado na TelaCadastroParque, mostrado na seção de Front-End, ao executar o método “salvar()” que fica na classe Parque, é criado uma nova “child” ou “nó” no banco, com os atributos que possuem o método ‘get’, em seguida é executado o “setValue” que atualiza as informações no banco.




```
public void salvar() {
    DatabaseReference reference = FirebaseDatabase.getInstance().getReference();
    reference.child("parques").child(nome).setValue(this);
}
```

- Firebase Storage: API de armazenamento de arquivos, utilizado pela TelaCadastroElemento, como repositório das fotos dos elementos cadastrados, onde a imagem foi convertida para o formato jpeg (imagem.compress), em seguida foi colocada num array do tipo byte.O storage recebe como referência o caminho (URI) do destino, em seguida envia o arquivo através do método "putBytes(arquivo)".

```java
public void atualizaCaminhoFoto() {
    ByteArrayOutputStream bytes = new ByteArrayOutputStream();
    imagem.compress(Bitmap.CompressFormat.JPEG, quality: 90, bytes);
    byte bb[] = bytes.toByteArray();
    uploadToFirebase(bb);
}

private void uploadToFirebase(byte[] bb) {
    caminhofoto = "myimages/" + nome.getText().toString() + ".jpg";
    StorageReference sr = storageRef.child(caminhofoto);
    sr.putBytes(bb).addOnSuccessListener(new OnSuccessListener<UploadTa
        @Override
        public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {
            Toast.makeText( context: TelaCadastroElemento.this, text: "Upload
        }
    }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull  Exception e) {
            Toast.makeText( context: TelaCadastroElemento.this, text: "Algo d
        }
    });
```



A codificação do projeto pode ser acessada no seguinte repositório: https://github.com/jackdnr/TCC

4. Conclusão

O desenvolvimento do projeto Ciência Ambiental Cidadã possibilitou realizar uma análise sobre as diversas possibilidades e benefícios que áreas verdes podem nos proporcionar. O aplicativo criado visa mostrar esses benefícios e ocasionar bem-estar para a comunidade, engajando a participação e fazendo a comunidade se sentir parte do ambiente, utilizando-se de temas como Crowdsourcing na abordagem de Ciência Cidadã.

Pelo fato de ser possível obter dados e construir informações em comunidade, a decisão sobre qual o “melhor” parque para cada ocasião ficou

mais fácil de ser tomada, pois há diferentes possibilidades de atrativos para diferentes objetivos: Se o objetivo é fazer um lanche, é possível saber quais parques possuem mesa, chão regular para piquenique, se o objetivo é levar crianças, é possível ver quais tem parquinhos e brinquedos, se o objetivo é caminhar em contato com a natureza, é possível saber quais possuem trilhas, os animais que circundam, entre outras possibilidades.

Por fim, o projeto como um todo possibilitou o aprofundamento em temas como desenvolvimento de sistemas mobile, integração com APIs, banco de dados e storages, agregando valor de forma considerável ao conhecimento técnico-científico em temas que são tendência mundial.

Ficam as seguintes sugestões para trabalhos futuros:

1. Por tratar-se de uma ferramenta exclusiva para a plataforma Android, elaborar a versão multiplataforma ocasionaria maior abrangência ao uso do aplicativo por não restringir-se apenas a celulares com o sistema Android..
2. Adicionar a localização do parque através do google maps também pode colaborar, pois os usuários do app poderão saber as melhores rotas a se fazer para chegar até o parque, ou até mesmo rotas internas do parque, além de aumentar a confiabilidade no ato de cadastramento de parques e elementos o estipular uma distância mínima do usuário em relação ao parque.

5. REFERÊNCIAS

[1] Ciêcia cidadã. 2019. Disponível em: <https://pt.wikipedia.org/wiki/Ci%C3%AAncia_cidad%C3%A3>. Acesso em: 15 maio 2019.

[2] PARQUES - Conceito. 2019. Disponível em: <https://www.infraestruturameioambiente.sp.gov.br/fundacaoflorestal/pagina-inicial/parques-estaduais/parques-conceito/>. Acesso em: 30 jun. 2019.

[3] SCANAVACA JUNIOR, L.. Importância dos parques urbanos: o exemplo do Parque Alfredo Volpi. 2012. Disponível em: <https://www.embrapa.br/busca-de-publicacoes/-/publicacao/944395/importancia-dos-parques-urbanos-o-exemplo-do-parque-alfredo-volpi>. Acesso em: 30 jun. 2019.

[4] SOUZA, Tatiana Pongiluppi. O que é Ciência Cidadã? Disponível em: <http://www.sibbr.gov.br/cienciacidada/#/about>. Acesso em: 15 maio 2019.

[5] JIGUET et al. French citizens monitoring ordinary birds provide tools for conservation and ecological sciences. 2012. Disponível em: <http://vincent.devictor.free.fr/Articles/Jiguet%20et%20al%20ActaEco%202012.pdf>. Acesso em: 30 jun. 2019.

[6] MELO, Elisandra. SAIBA O QUE É GESTÃO AMBIENTAL E COMO SE TORNAR UM PROFISSIONAL DESSA ÁREA!2019. Disponível em: <https://ead.catolica.edu.br/blog/saiba-o-que-e-gestao-ambiental-e-como-se-tornar-um-profissional-dessa-area>. Acesso em: 30 jun. 2019.

[7] PROJETOS. 2019. Disponível em: <http://www.sibbr.gov.br/cienciacidada/#/portfolio>. Acesso em: 15 maio 2019.

[8] PARQUE Ecológico do Córrego Grande. 2019. Disponível em: <https://pt.wikipedia.org/wiki/Parque_Ecol%C3%B3gico_do_C%C3%B3rrego_Grande>. Acesso em: 15 maio 2019.

[9] JARDIM Botânico de Florianópolis. 2019. Disponível em: <https://pt.wikipedia.org/wiki/Jardim_Bot%C3%A2nico_de_Florian%C3%B3polis>. Acesso em: 15 maio 2019.

[10] PARQUE em Florianópolis abriga maior lagoa de água doce do Litoral de SC. 2013. Disponível em: <http://g1.globo.com/sc/santa-catarina/nossa-terra/2013/noticia/2013/09/parque-em-florianopolis-abriga-maior-lagoa-de-agua-doce-do-litoral-de-sc.html>. Acesso em: 15 maio 2019

[11] SIGNIFICADO de Cidadão: O que significa Cidadão:. O que significa Cidadão:. 2015. Disponível em: <https://www.significados.com.br/cidadao/>. Acesso em: 15 maio 2019.

[12] MMA lança APP Parques do Brasil. 2018. Disponível em: <http://www.icmbio.gov.br/portal/ultimas-noticias/20-geral/10146-mma-lanca-app-parques-do-brasil>. Acesso em: 30 jun. 2019.

[13] PERES, Patrícia Maria Schubert et al. La percepción que tienen los padres de las oportunidades para los niños en la naturaleza. 2017. Disponível em: <https://www.tandfonline.com/doi/abs/10.1080/21711976.2017.1291185?journalCode=rprb20>. Acesso em: 20 maio 2019.

[14] TEIXEIRA, Francisco Sacco Flores Almeida. RENOVAR: UM MVP PARA MONITORAR A QUALIDADE DO AR. 2019. 157 f. TCC (Graduação) - Curso de Sistemas de Informação, Centro Tecnológico, Universidade Federal de Santa Catarina, Florianópolis, 2018.

[15] PARQUES do Brasil. 2018. Disponível em: <https://play.google.com/store/apps/details?id=br.gov.mma.parquesdobrasil&hl=pt_BR>. Acesso em: 30 jun. 2019.

[16] CROWDSOURCING: o que é, benefícios e exemplos para você se inspirar. [Liga Insights], 19 ago. 2019. Disponível em: https://insights.liga.ventures/inovacao/crowdsourcing/. Acesso em: 5 jan. 2021.

[17] MOTA, João Moisés Brito; LIMA, Afonso Carneiro. Efetividade do Crowdsourcing como Apoio à Segurança Pública. Universidade de Fortaleza, Programa de Pós-graduação em Administração, Fortaleza, CE, Brasil, [s. l.], 19 ago. 2018. Disponível em: https://www.scielo.br/j/rac/a/G8yYVspTFRNGVZGBHj388hy/?lang=pt&format=pdf. Acesso em: 19 mar. 2021.

[18] SANTANA, Daiane. SEIS aplicativos e plataformas on-line para se conectar com a natureza. 16 abr. 2019. Disponível em: https://vivoverde.com.br/seis-aplicativos-e-plataformas-on-line-para-se-conectar-com-a-natureza/. Acesso em: 5 ago. 2019.

[19] RASMUSSEN, Bruna. OS NÚMEROS não mentem: Android ou iOS, qual é o melhor?. [*S. l.*], 22 abr. 2013. Disponível em: https://canaltech.com.br/produtos/os-numeros-nao-mentem-android-ou-ios-qual-e-o-melhor-7657/. Acesso em: 20 jan. 2021.

[20] CORDERO, Filipe. ANDROID SDK: O que é? Para que Serve? Como Usar?. [S. l.], 21 fev. 2017. Disponível em: https://www.androidpro.com.br/blog/android-studio/android-sdk/. Acesso em: 20 jan. 2021.

[21] SILVA, Eduardo. Firebase: o que é e quando usar no desenvolvimento mobile?. [S. l.], 6 maio 2021. Disponível em: https://blog.geekhunter.com.br/firebase-o-que-e-e-quando-usar-no-desenvolvimento-mobile/. Acesso em: 27 maio 2021.

[22] Peres, Felippe e Kuhnen: PERCEPÇÃO PARENTAL DAS BARREIRAS PARA O CONTATO DA CRIANÇA COM A NATUREZA, 2019.

APÊNDICE A - Código Java

```
package com.example.prototipo.classes.objetos;


import com.google.firebase.database.DatabaseReference;

import com.google.firebase.database.FirebaseDatabase;

import java.util.ArrayList;


public class Parque {

   ArrayList<Elemento> elementos = new ArrayList<Elemento>();

   String nome;

   String estado;

   String cidade;

   String bairro;

   String userId;

   String descricao;

   int cont = 1;


   public Parque(String nome, String estado, String cidade, String bairro, String userId, String
descricao) {

      this.nome = nome;

      this.estado = estado;

      this.cidade = cidade;

      this.bairro = bairro;

      this.userId = userId;

      this.descricao = descricao;

   }

   public Parque() {
```

```
}
public void setDescricao(String descricao) {
   this.descricao = descricao;
}
public void incrementaCont() {
   this.cont++;
}
public String getUserId() {
   return userId;
}
public void setUserId(String userId) {
   this.userId = userId;
}
public ArrayList getElementos() {
   return elementos;
}
public String getNome() {
   return nome;
}
public String getEstado() {
   return estado;
}
public String getCidade() {
   return cidade;
}
public String getBairro() {
   return bairro;
```

```
    }
    public String getDescricao() {
        return descricao;
    }
    public int getCont() {
        return cont;
    }
    public Parque(ArrayList<Elemento> elementos, String nome, String estado, String cidade,
String bairro, String userId, String descricao, int cont) {
        this.elementos = elementos;
        this.nome = nome;
        this.estado = estado;
        this.cidade = cidade;
        this.bairro = bairro;
        this.userId = userId;
        this.descricao = descricao;
        this.cont = cont;
    }
    public void salvar() {
        DatabaseReference reference = FirebaseDatabase.getInstance().getReference();
        reference.child("parques").child(nome).setValue(this);
    }
}
package com.example.prototipo.classes.objetos;


public class Elemento {
    String nome;
```

```
String tipo;
String caminhoFoto;
String infos;

public Elemento(String nome, String tipo, String caminhoFoto) {
   this.nome = nome;
   this.tipo = tipo;
   this.caminhoFoto = caminhoFoto;
}
public Elemento() {
}
public String getInfos() {
   return infos;
}
public Elemento(String nome, String tipo, String caminhoFoto, String infos) {
   this.nome = nome;
   this.tipo = tipo;
   this.caminhoFoto = caminhoFoto;
   this.infos = infos;
}
public void setInfos(String infos) {
   this.infos = infos;
}
public String getNome() {
   return nome;
}
public void setNome(String nome) {
```

```
        this.nome = nome;
    }
    public String getTipo() {
        return tipo;
    }
    public void setTipo(String tipo) {
        this.tipo = tipo;
    }
    public String getCaminhoFoto() {
        return caminhoFoto;
    }
    public void setCaminhoFoto(String caminhoFoto) {
        this.caminhoFoto = caminhoFoto;
    }
}
package com.example.prototipo.classes.models;

import com.example.prototipo.classes.objetos.Elemento;
import java.util.ArrayList;

public class ModelParque {
    String nome, estado, cidade, bairro, descricao;
    int cont;
    ArrayList<Elemento> elementos = new ArrayList<Elemento>();
    public ModelParque() {
    }
```

```
    public ModelParque(String nome, String estado, String cidade, String bairro, String
descricao) {
        this.nome = nome;
        this.estado = estado;
        this.cidade = cidade;
        this.bairro = bairro;
        this.descricao = descricao;
    }
    public String getNome() {
        return nome;
    }
    public void setNome(String nome) {
        this.nome = nome;
    }
    public String getEstado() {
        return estado;
    }
    public void setEstado(String estado) {
        this.estado = estado;
    }
    public String getCidade() {
        return cidade;
    }
    public void setCidade(String cidade) {
        this.cidade = cidade;
    }
    public String getBairro() {
```

```
        return bairro;
    }
    public void setBairro(String bairro) {
        this.bairro = bairro;
    }
    public String getDescricao() {
        return descricao;
    }
    public void setDescricao(String descricao) {
        this.descricao = descricao;
    }
    public int getCont() {
        return cont;
    }
    public ArrayList<Elemento> getElementos() {
        return elementos;
    }
}
package com.example.prototipo.classes.utilitario;
import androidx.annotation.NonNull;
import com.example.prototipo.classes.objetos.Parque;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import java.util.ArrayList;
```

```
public class UtilitarioBanco {
   public static ArrayList download() {
      ArrayList<Parque> parques = new ArrayList<Parque>();
      DatabaseReference reference = FirebaseDatabase.getInstance().getReference();
      reference.child("parques").addListenerForSingleValueEvent(new ValueEventListener() {
         @Override
         public void onDataChange(@NonNull DataSnapshot snapshot) {
            for (DataSnapshot dn : snapshot.getChildren()) {
               Parque p = (Parque) dn.getValue(Parque.class);
               parques.add(p);
            }
         }
         @Override
         public void onCancelled(@NonNull DatabaseError error) {
         }
      });
      return parques;
   }
}
package com.example.prototipo.classes.adapters;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import androidx.annotation.NonNull;
```

```
import androidx.recyclerview.widget.RecyclerView;

import com.example.prototipo.R;

import com.example.prototipo.classes.models.ModelParque;

import java.util.ArrayList;


public class AdapterParque extends RecyclerView.Adapter<AdapterParque.MyViewHolder>{

   Context context;

   ArrayList<ModelParque> pList;

   ArrayList<ModelParque> listFull;

   OnItemClickListener listener;


   public AdapterParque(ArrayList<ModelParque> pList, OnItemClickListener listener) {

      this.pList = pList;

      this.listener = listener;

   }

   public AdapterParque(Context context, ArrayList<ModelParque> pList, OnItemClickListener
listener) {

      this.context = context;

      this.pList = pList;

      this.listener = listener;

      this.listFull = new ArrayList<>(pList);

   }

   public AdapterParque(Context context, ArrayList<ModelParque> pList) {

      this.context = context;

      this.pList = pList;

   }

   @NonNull
```

```
    @Override
    public MyViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View v = LayoutInflater.from(context).inflate(R.layout.parque,parent,false);
        return new MyViewHolder(v);
    }
    @Override
    public void onBindViewHolder(@NonNull AdapterParque.MyViewHolder holder, int
position) {
        ModelParque model = pList.get(position);
        holder.nome.setText(model.getNome());
        holder.estado.setText(model.getEstado());
        holder.cidade.setText(model.getCidade());
        holder.itemView.setOnClickListener(view->{
            listener.onItemClick(pList.get(position));
        });
    }
    @Override
    public int getItemCount() {
        return pList.size();
    }
    public interface OnItemClickListener {
        void onItemClick(ModelParque mp);
    }
    public static class MyViewHolder extends  RecyclerView.ViewHolder{
        TextView nome, estado, cidade;
        public MyViewHolder(@NonNull View itemView) {
            super(itemView);
```

```
            nome = itemView.findViewById(R.id.valorNomeE);
            estado = itemView.findViewById(R.id.valorEstado);
            cidade = itemView.findViewById(R.id.valorTipoE);
        }
    }
}
package com.example.prototipo.classes.adapters;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import android.widget.Toast;
import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;
import com.bumptech.glide.Glide;
import com.bumptech.glide.Registry;
import com.bumptech.glide.annotation.GlideModule;
import com.bumptech.glide.module.AppGlideModule;
import com.example.prototipo.R;
import com.example.prototipo.classes.objetos.Elemento;
//import com.example.prototipo.classes.models.Elemento;
import com.firebase.ui.storage.images.FirebaseImageLoader;
import com.google.android.gms.tasks.OnFailureListener;
```

```
import com.google.android.gms.tasks.OnSuccessListener;

import com.google.firebase.storage.FileDownloadTask;

import com.google.firebase.storage.FirebaseStorage;

import com.google.firebase.storage.StorageReference;

import java.io.File;

import java.io.IOException;

import java.io.InputStream;

import java.util.ArrayList;

import de.hdodenhof.circleimageview.CircleImageView;


public class AdapterElemento extends
RecyclerView.Adapter<AdapterElemento.MyViewHolder> {

   Context context;

   ArrayList<Elemento> eList;

   AdapterElemento.OnItemClickListener listener;

   StorageReference reference;

   Bitmap img;


   public AdapterElemento(ArrayList<Elemento> eList, AdapterElemento.OnItemClickListener
listener) {

      this.eList = eList;

      this.listener = listener;

   }

   public AdapterElemento(Context context, ArrayList<Elemento> eList,
AdapterElemento.OnItemClickListener listener) {

      this.context = context;

      this.eList = eList;

      this.listener = listener;
```

```
    }


    public AdapterElemento(Context context, ArrayList<Elemento> eList) {
        this.context = context;
        this.eList = eList;
    }
    @GlideModule
    public static class MyAppGlideModule extends AppGlideModule {
        @Override
        public void registerComponents(Context context, Glide glide, Registry registry) {
            // Register FirebaseImageLoader to handle StorageReference
            registry.append(StorageReference.class, InputStream.class,
                    new FirebaseImageLoader.Factory());
        }
    }
    @NonNull
    @Override
    public AdapterElemento.MyViewHolder onCreateViewHolder(@NonNull ViewGroup
parent, int viewType) {
        View v = LayoutInflater.from(context).inflate(R.layout.elemento,parent,false);
        return new AdapterElemento.MyViewHolder(v);
    }
    @Override
    public void onBindViewHolder(@NonNull AdapterElemento.MyViewHolder holder, int
position) {
        Elemento model = eList.get(position);
        reference = FirebaseStorage.getInstance()
                .getReference()
```

```
            .child(model.getCaminhoFoto());

        holder.nome.setText(model.getNome());

        holder.tipo.setText(model.getTipo());

        final File localFile;

        try {

            localFile = File.createTempFile("foto","jpg");

            reference.getFile(localFile).addOnSuccessListener(new
OnSuccessListener<FileDownloadTask.TaskSnapshot>() {

                @Override

                public void onSuccess(FileDownloadTask.TaskSnapshot taskSnapshot) {

                    img = BitmapFactory.decodeFile(localFile.getAbsolutePath());

                    holder.foto.setImageBitmap(img);

                }

            }).addOnFailureListener(new OnFailureListener() {

                @Override

                public void onFailure(@NonNull Exception e) {

                    Toast.makeText(context,"Imagem não pôde ser
carregada",Toast.LENGTH_SHORT).show();

                }

            });

        } catch (IOException e) {

            e.printStackTrace();

        }

//      Glide.with(holder.foto.getContext())

//              .load(reference)

//              .into(holder.foto);


        holder.itemView.setOnClickListener(view->{
```

```
            listener.onItemClick(eList.get(position));
        });
    }
    @Override
    public int getItemCount() {
        return eList.size();
    }
    public interface OnItemClickListener {
        void onItemClick(Elemento mp);
    }
    public static class MyViewHolder extends  RecyclerView.ViewHolder{
        TextView nome, tipo;
        CircleImageView foto;
        public MyViewHolder(@NonNull View itemView) {
            super(itemView);
            nome = itemView.findViewById(R.id.valorNomeE);
            tipo = itemView.findViewById(R.id.valorTipoE);
            foto = itemView.findViewById(R.id.foto);
        }
    }
}
package com.example.prototipo.telas.principais;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
```

```
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;
import com.example.prototipo.R;
import com.google.android.gms.auth.api.signin.GoogleSignIn;
import com.google.android.gms.auth.api.signin.GoogleSignInAccount;
import com.google.android.gms.auth.api.signin.GoogleSignInClient;
import com.google.android.gms.auth.api.signin.GoogleSignInOptions;
import com.google.android.gms.common.SignInButton;
import com.google.android.gms.common.api.ApiException;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthCredential;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.auth.GoogleAuthProvider;

public class MainActivity extends AppCompatActivity {

    private GoogleSignInClient mGoogleSignInClient;
    public static final int RC_SIGN_IN = 123;
    private FirebaseAuth mAuth;
    AlertDialog.Builder ad;
    SignInButton botao;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
```

```
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    getSupportActionBar().hide();
    mAuth = FirebaseAuth.getInstance();
    ad = new AlertDialog.Builder(this);
    botao = (SignInButton) findViewById(R.id.botao);
    botao.setOnClickListener(view -> {
       signIn();
    });
    requisita();
  }
  private void requisita() {
    GoogleSignInOptions gso = new
GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
        .requestIdToken(getString(R.string.default_web_client_id))
        .requestEmail()
        .build();
    mGoogleSignInClient = GoogleSignIn.getClient(this, gso);
  }
  private void signIn() {
    Intent signInIntent = mGoogleSignInClient.getSignInIntent();
    startActivityForResult(signInIntent, RC_SIGN_IN);
  }
  @Override
  public void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == RC_SIGN_IN) {
```

```
        Task<GoogleSignInAccount> task =
GoogleSignIn.getSignedInAccountFromIntent(data);
        try {
            GoogleSignInAccount account = task.getResult(ApiException.class);
            firebaseAuthWithGoogle(account.getIdToken());
            SharedPreferences.Editor editor = getApplicationContext()
                    .getSharedPreferences("Pref", MODE_PRIVATE)
                    .edit();
            editor.putString("username", account.getDisplayName());
            editor.putString("userid", account.getIdToken());
            editor.putString("useremail", account.getEmail());
            //editor.putString(("foto"),account.getPhotoUrl().toString());
            editor.apply();
        } catch (ApiException e) {
            ad.setTitle("Falha de Login!");
            ad.setMessage("Algo inesperado aconteceu!");
            ad.show();
        }
    }
}
private void firebaseAuthWithGoogle(String idToken) {
    AuthCredential credential = GoogleAuthProvider.getCredential(idToken, null);
    mAuth.signInWithCredential(credential)
            .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
                @Override
                public void onComplete(@NonNull Task<AuthResult> task) {
                    if (task.isSuccessful()) {
```

```
                    Intent i = new Intent(MainActivity.this, TelaInicial.class);
                    startActivity(i);
                } else {
                    ad.setTitle("Falha de Autenticação!");
                    ad.setMessage("Usuário ou Senha Incorreta");
                    ad.show();
                }
            }
        });
    }
    @Override
    public void onStart() {
        super.onStart();
        FirebaseUser currentUser = mAuth.getCurrentUser();
        if (currentUser != null) {
            Intent i = new Intent(MainActivity.this, TelaInicial.class);
            startActivity(i);
        }
    }
    public void sobre(View view){
        ad.setTitle("Sobre o CAC");
        ad.setMessage("O Ciência Ambiental Cidadã é um aplicativo voltado à parques ambientais, "+
                "adotando temas como Ciência Cidadã e Crowdsourcing. Trata-se de um app aberto "+
                "à comunidade, onde qualquer pessoa pode contribuir.\nProduzido como Trabalho "+
                "de Conclusão do curso de Sistemas de Informação na Universidade Federal de Santa "+
```

```
            "Catarina, por Jackson Dener Wrublak.");
        ad.show();
    }
}
package com.example.prototipo.telas.principais;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import com.example.prototipo.R;
import com.example.prototipo.telas.parque.TelaCadastroParques;
import com.example.prototipo.telas.parque.TelaFotoElemento;
import com.example.prototipo.telas.parque.TelaListaElementos;
import com.example.prototipo.telas.parque.TelaListaParques;
import com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton;
import com.google.firebase.auth.FirebaseAuth;

public class TelaInicial extends AppCompatActivity {
    ExtendedFloatingActionButton deslogar;
    ExtendedFloatingActionButton novo;
    ExtendedFloatingActionButton lista;
    FirebaseAuth mAuth;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_tela_inicial);
        getSupportActionBar().hide();
```

```
        deslogar = (ExtendedFloatingActionButton) findViewById(R.id.botaoSair);
        novo = (ExtendedFloatingActionButton) findViewById(R.id.botaoNovoParque);
        lista = (ExtendedFloatingActionButton) findViewById(R.id.botaoListarParques);
        mAuth = FirebaseAuth.getInstance();
        deslogar.setOnClickListener( view->{
            mAuth.signOut();
            Intent i = new Intent(TelaInicial.this, MainActivity.class);
            startActivity(i);
        });
        novo.setOnClickListener(view ->{
            Intent i = new Intent(TelaInicial.this, TelaCadastroParques.class);
            startActivity(i);
        });
        lista.setOnClickListener(view ->{
            Intent i = new Intent(TelaInicial.this, TelaListaParques.class);
            startActivity(i);
        });
    }
    @Override
    public void onBackPressed() {
        Intent i = new Intent(TelaInicial.this, MainActivity.class);
        startActivity(i);
        super.onBackPressed();
    }
}
```

```
package com.example.prototipo.telas.parque;


import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.text.InputType;
import android.widget.EditText;
import android.widget.Toast;
import com.example.prototipo.R;
import com.example.prototipo.classes.objetos.Elemento;
import com.example.prototipo.classes.objetos.Parque;
import com.example.prototipo.classes.utilitario.UtilitarioBanco;
import com.example.prototipo.telas.principais.TelaInicial;
import com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton;
import com.google.android.material.textfield.TextInputEditText;
import java.util.ArrayList;


public class TelaCadastroParques extends AppCompatActivity {
    Parque p;
    ArrayList<Parque> parques;
    TextInputEditText nomeP;
    TextInputEditText estadoP;
    TextInputEditText cidadeP;
    TextInputEditText bairroP;
```

```
    TextInputEditText descricaoP;
    ExtendedFloatingActionButton botaoCadastraP;
    String userId;
    String descricao;
    AlertDialog.Builder adb;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_tela_cadastro_parques);
        getSupportActionBar().hide();
        SharedPreferences preferences = getSharedPreferences("Pref", MODE_PRIVATE);
        userId = preferences.getString("userid", "");
        parques = UtilitarioBanco.download();
        nomeP = (TextInputEditText) findViewById(R.id.nomeP);
        estadoP = (TextInputEditText) findViewById(R.id.estadoP);
        cidadeP = (TextInputEditText) findViewById(R.id.cidadeP);
        bairroP = (TextInputEditText) findViewById(R.id.bairroP);
        descricaoP = (TextInputEditText) findViewById(R.id.descricaoP);
        botaoCadastraP = (ExtendedFloatingActionButton)
findViewById(R.id.botaoCadastraParque);
        botaoCadastraP.setOnClickListener(view -> {
            cadastra();
        });
    }
    private void cadastra() {
        String nome = nomeP.getText().toString();
```

```
String estado = estadoP.getText().toString();
String cidade = cidadeP.getText().toString();
String bairro = bairroP.getText().toString();
descricao = descricaoP.getText().toString();
String mensagem = "";
if (verificaSeExiste(nome, estado, cidade)) {
   if (!p.getUserId().equals(userId)) {
      mensagem = "O cadastro desse parque já foi solicitado!";
      if (p.getCont() == 4) {
         p.incrementaCont();
         mensagem = "Parque aprovado!";

      } else if (p.getCont() > 4) {
         mensagem = "Esse parque já está cadastrado!";
      } else {
         p.setUserId(userId);
         p.incrementaCont();
      }
      editaInfos();
      p.salvar();
   } else {
      mensagem = "Você já solicitou o cadastro desse parque!";
   }
} else {
   Parque p = new Parque(nome, estado, cidade, bairro, userId, descricao);
   p.salvar();
   mensagem = "Solicitação de cadastro feita com sucesso!";
```

```
        Intent i = new Intent(TelaCadastroParques.this, TelaInicial.class);
        startActivity(i);
      }
      Toast.makeText(TelaCadastroParques.this, mensagem, Toast.LENGTH_LONG).show();
    }

    public void editaInfos() {
      adb = new AlertDialog.Builder(this);
      adb.setTitle("Solicitação de cadastro já realizada, por favor verifique as informações:");
      final EditText input = new EditText(this);
      input.setText(p.getDescricao());
      input.setInputType(InputType.TYPE_CLASS_TEXT |
InputType.TYPE_TEXT_FLAG_MULTI_LINE);
      adb.setView(input);
      adb.setPositiveButton("OK", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
          descricao = input.getText().toString();
        }
      });
      adb.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
          dialog.cancel();
        }
      });
```

```
package com.example.prototipo.telas.parque;


import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import android.content.Intent;
import android.os.Bundle;
import android.text.Editable;
import android.text.TextWatcher;
import android.widget.EditText;
import android.widget.SearchView;
import com.example.prototipo.R;
import com.example.prototipo.classes.adapters.AdapterParque;
import com.example.prototipo.classes.models.ModelParque;
import com.example.prototipo.classes.objetos.Elemento;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import java.util.ArrayList;


public class TelaListaParques extends AppCompatActivity {
    RecyclerView rv;
    AdapterParque adapter;
    ArrayList<ModelParque> list;
```

```
ArrayList<ModelParque> listFiltrada;
EditText cidade;
EditText elemento;
DatabaseReference reference;
@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_tela_lista_parques);
   getSupportActionBar().hide();
   reference = FirebaseDatabase.getInstance().getReference().child("parques");
   cidade = (EditText) findViewById(R.id.pesquisaCidade);
   elemento = (EditText) findViewById(R.id.pesquisaElemento);
   rv = (RecyclerView) findViewById(R.id.rv);
   rv.setHasFixedSize(true);
   rv.setLayoutManager(new LinearLayoutManager(this));
   baixaListaDoBanco();
   preenche(list);
   defineFiltro();
}

private void baixaListaDoBanco() {
   list = new ArrayList<>();
   reference.addValueEventListener(new ValueEventListener() {
      @Override
      public void onDataChange(@NonNull  DataSnapshot snapshot) {
         for (DataSnapshot dn : snapshot.getChildren()) {
            ModelParque model = dn.getValue(ModelParque.class);
```

```
            if(model.getCont()>4){
               list.add(model);
               adapter.notifyDataSetChanged();
            }
          }
       }
       @Override
       public void onCancelled(@NonNull  DatabaseError error) {
       }
    });
}
private void preenche(ArrayList<ModelParque> lista) {
    adapter = new AdapterParque(this, lista, new AdapterParque.OnItemClickListener() {
       @Override
       public void onItemClick(ModelParque mp) {
          telaParque(mp);
       }
    });
    rv.setAdapter(adapter);
}
public void telaParque(ModelParque p){
    Intent i = new Intent(TelaListaParques.this, TelaParque.class);
    startActivity(i);
    TelaParque.mp=p;
}
public void defineFiltro(){
    cidade.addTextChangedListener(new TextWatcher() {
```

```
    @Override
    public void beforeTextChanged(CharSequence s, int start, int count, int after) {
    }
    @Override
    public void onTextChanged(CharSequence s, int start, int before, int count) {
    }
    @Override
    public void afterTextChanged(Editable s) {
        if(isEmpty(elemento)){
            listFiltrada = new ArrayList<>(list);
        }
        String textoDigitado = s.toString();
        ArrayList filtro = new ArrayList();
        for(ModelParque parque : listFiltrada){
            if(parque.getCidade().toLowerCase().contains(textoDigitado.toLowerCase())){
                filtro.add(parque);
            }
        }
        if(isEmpty(elemento)){
            listFiltrada = filtro;
        }
        preenche(filtro);
    }
});
elemento.addTextChangedListener(new TextWatcher() {
    @Override
    public void beforeTextChanged(CharSequence s, int start, int count, int after) {
```

```
        }
        @Override
        public void onTextChanged(CharSequence s, int start, int before, int count) {
        }
        @Override
        public void afterTextChanged(Editable s) {
          if(isEmpty(cidade)){
            listFiltrada = new ArrayList<>(list);
          }
          String textoDigitado = s.toString();
          ArrayList filtro = new ArrayList();
          for(ModelParque parque : listFiltrada){
            for(Elemento elemento : parque.getElementos()) {
              if (elemento.getNome().toLowerCase().contains(textoDigitado.toLowerCase())) {
                filtro.add(parque);
              }
            }
          }
          if(isEmpty(cidade)){
            listFiltrada = filtro;
          }
          preenche(filtro);
        }
      });
    }
    private boolean isEmpty(EditText etText) {
      return etText.getText().toString().trim().length() == 0;
```

```
    }
}


        adb.show();
    }


    private boolean verificaSeExiste(String nome, String estado, String cidade) {
        for (Parque parque : parques) {
            if (parque.getNome().toLowerCase().equals(nome.toLowerCase()) &&
                    parque.getEstado().toLowerCase().equals(estado.toLowerCase()) &&
                    parque.getCidade().toLowerCase().equals(cidade.toLowerCase())) {
                p = (Parque) parque;
                return true;
            }
        }
        return false;
    }
    @Override
    public void onBackPressed() {
        Intent i = new Intent(TelaCadastroParques.this, TelaInicial.class);
        startActivity(i);
        super.onBackPressed();
    }
}
```

```
package com.example.prototipo.telas.parque;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;
import android.widget.Toast;
import com.example.prototipo.R;
import com.example.prototipo.classes.models.ModelParque;
import com.example.prototipo.classes.objetos.Parque;
import com.example.prototipo.classes.utilitario.UtilitarioBanco;
import com.example.prototipo.telas.principais.TelaInicial;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import java.util.ArrayList;

public class TelaParque extends AppCompatActivity {
    static ModelParque mp;
    TextView nomeParque;
    AlertDialog.Builder adb;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.activity_tela_parque);
        getSupportActionBar().hide();
        adb = new AlertDialog.Builder(this);
        nomeParque = (TextView) findViewById(R.id.nomeParque);
        nomeParque.setText(mp.getNome());
    }
    public void local(View view) {
        adb.setTitle("Eu fico em:");
        adb.setMessage("Estado: " + mp.getEstado() + "\n" + "Cidade: " + mp.getCidade() + "\n"
+ "Bairro: " + mp.getBairro());
        adb.show();
    }
    public void descricao(View view) {
        adb.setTitle("Descrição:");
        adb.setMessage(mp.getDescricao());
        adb.show();
    }
    public void elementos(View view) {
        Intent i = new Intent(TelaParque.this, TelaListaElementos.class);
        startActivity(i);
        TelaListaElementos.mp = mp;
    }
    public void colaborar(View view) {
        Intent i = new Intent(TelaParque.this, TelaCadastroElemento.class);
        startActivity(i);
        TelaCadastroElemento.mp = mp;
    }
```

```
public void deletaParque(View view) {
    SharedPreferences preferences = getSharedPreferences("Pref", MODE_PRIVATE);
    String userId = preferences.getString("userid", "");
    if(verificaUserid(userId)){
        Toast.makeText(TelaParque.this, "Você já solicitou a remoção deste parque!",
                Toast.LENGTH_SHORT).show();
    }
    else {
        if (mp.getCont() < 10) {

            Parque p = new Parque(mp.getElementos(), mp.getNome(), mp.getEstado(),
                    mp.getCidade(), mp.getBairro(), userId,
                    mp.getDescricao(), mp.getCont());
            p.incrementaCont();
            p.setUserId(userId);
            p.salvar();
            Toast.makeText(TelaParque.this, "Solicitação de remoção realizada com sucesso!",
                    Toast.LENGTH_SHORT).show();
            Intent i = new Intent(TelaParque.this, TelaInicial.class);
            startActivity(i);
        } else {
            DatabaseReference reference = FirebaseDatabase.getInstance().
                    getReference().child("parques/" + mp.getNome());
            Toast.makeText(TelaParque.this, "Solicitação de remoção atendida com sucesso!",
                    Toast.LENGTH_SHORT).show();
            reference.removeValue();
            Intent i = new Intent(TelaParque.this, TelaInicial.class);
```

```
            startActivity(i);
        }
    }
}
public boolean verificaUserid(String userId){
    ArrayList<Parque> parques = UtilitarioBanco.download();
    for(Parque p : parques){
        if(p.getNome().equals(mp.getNome()) && p.getUserId().equals(userId)){
            return true;
        }
    }
    return false;
}
@Override
public void onBackPressed() {
    Intent i = new Intent(TelaParque.this, TelaListaParques.class);
    startActivity(i);
    super.onBackPressed();
}
}
package com.example.prototipo.telas.parque;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import android.Manifest;
```

```
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.pm.PackageManager;
import android.graphics.Bitmap;
import android.graphics.Matrix;
import android.os.Bundle;
import android.provider.MediaStore;
import android.view.View;
import android.widget.RadioButton;
import android.widget.TextView;
import android.widget.Toast;
import com.example.prototipo.R;
import com.example.prototipo.classes.objetos.Elemento;
import com.example.prototipo.classes.models.ModelParque;
import com.example.prototipo.classes.objetos.Parque;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.android.material.textfield.TextInputEditText;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;
import com.google.firebase.storage.UploadTask;
import java.io.ByteArrayOutputStream;
import java.util.ArrayList;
import de.hdodenhof.circleimageview.CircleImageView;

public class TelaCadastroElemento extends AppCompatActivity {
    static ModelParque mp;
```

```
TextInputEditText nome;
RadioButton vegetal;
RadioButton animal;
RadioButton facilidades;
RadioButton outroElementoNatural;
RadioButton outro;
ArrayList<Elemento> list;
CircleImageView foto;
StorageReference storageRef;
Bitmap imagem;
String caminhofoto;

@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_tela_cadastro_elemento);
   acessoCamera();
   getSupportActionBar().hide();
   try {
      list = mp.getElementos();
   } catch (Exception e) {
      list = new ArrayList<Elemento>();
   }
   storageRef = FirebaseStorage.getInstance().getReference();
   nome = findViewById(R.id.nomeE);
   vegetal = findViewById(R.id.radioVegetal);
   animal = findViewById(R.id.radioAnimal);
```

```
    facilidades = findViewById(R.id.radioFacilidade);
    outroElementoNatural = findViewById(R.id.radioOutroElementoNatural);
    outro = findViewById(R.id.radioOutro);
    foto = findViewById(R.id.foto);
    trataRadioButtons();
  }

  public void foto(View view) {
    Toast.makeText(TelaCadastroElemento.this, "Tirar foto em modo RETRATO",
Toast.LENGTH_LONG).show();
    Intent i = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    startActivityForResult(i, 1);
  }

  public void cadastra(View view) {
    if (imagem != null && nome.getText() != null && (vegetal.isChecked() ||
        animal.isChecked() || facilidades.isChecked() ||
        outroElementoNatural.isChecked() || outro.isChecked())) {
      atualizaCaminhoFoto();
      String tipo = "";
      if (vegetal.isChecked()) {
        tipo = "Vegetal";
      } else if (animal.isChecked()) {
        tipo = "Animal";
      } else if (facilidades.isChecked()) {
        tipo = "Facilidade";
      } else if (outroElementoNatural.isChecked()) {
```

```
            tipo = "Outro elemento natural";
        } else {
            tipo = "Outro";
        }

        Elemento e = new Elemento(nome.getText().toString(), tipo, caminhofoto);
        list.add(e);
        SharedPreferences preferences = getSharedPreferences("Pref", MODE_PRIVATE);
        String userId = preferences.getString("userid", "");
        Parque p = new Parque(list, mp.getNome(), mp.getEstado(), mp.getCidade(),
mp.getBairro(), userId, mp.getDescricao(), 5);
        p.salvar();
        telaParque();
      } else {
        Toast.makeText(TelaCadastroElemento.this, "Você deixou algum campo em
branco,\nou esqueceu de tirar foto", Toast.LENGTH_LONG).show();
      }
    }

    public void telaParque() {
      Intent i = new Intent(TelaCadastroElemento.this, TelaParque.class);
      startActivity(i);
      TelaParque.mp = mp;
    }

    public void acessoCamera() {
      if (ActivityCompat.checkSelfPermission(this, Manifest.permission.CAMERA) !=
PackageManager.PERMISSION_GRANTED) {
```

```
        ActivityCompat.requestPermissions(this, new String[]{
                Manifest.permission.CAMERA
        }, 0);
    }
}

@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == 1 && resultCode == RESULT_OK) {
        Bundle extras = data.getExtras();
        imagem = rotateBitmap((Bitmap) extras.get("data"), 90);

        foto.setImageBitmap(imagem);
    }
}

public void atualizaCaminhoFoto() {
    ByteArrayOutputStream bytes = new ByteArrayOutputStream();
    imagem.compress(Bitmap.CompressFormat.JPEG, 90, bytes);
    byte bb[] = bytes.toByteArray();
    uploadToFirebase(bb);
}

private void uploadToFirebase(byte[] bb) {
    int i = 0;
    String nomeFoto = nome.getText().toString();
```

```
    //while(verificaNomeFoto(nomeFoto)){
    //    i++;
    //}
    nomeFoto += i;
    caminhofoto = "myimages/" + nomeFoto +".jpg";
    StorageReference sr = storageRef.child(caminhofoto);
    sr.putBytes(bb).addOnSuccessListener(new
OnSuccessListener<UploadTask.TaskSnapshot>() {
        @Override
        public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {
            Toast.makeText(TelaCadastroElemento.this, "Upload feito com sucesso!",
Toast.LENGTH_LONG).show();
        }
    }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {
            Toast.makeText(TelaCadastroElemento.this, "Algo deu errado!",
Toast.LENGTH_LONG).show();
        }
    });
  }

  public static Bitmap rotateBitmap(Bitmap source, float angle) {
    Matrix matrix = new Matrix();
    matrix.postRotate(angle);
    return Bitmap.createBitmap(source, 0, 0, source.getWidth(), source.getHeight(), matrix,
true);
  }
  public boolean verificaNomeFoto(String name){
```

```
    for(Elemento e : list){
        if(e.getNome().equals(name)){
            return true;
        }
    }
    return false;
}
public void trataRadioButtons(){
    animal.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            animal.setChecked(true);
            vegetal.setChecked(false);
            facilidades.setChecked(false);
            outroElementoNatural.setChecked(false);
            outro.setChecked(false);
        }
    });
    vegetal.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            animal.setChecked(false);
            vegetal.setChecked(true);
            facilidades.setChecked(false);
            outroElementoNatural.setChecked(false);
            outro.setChecked(false);
        }
```

```
});
facilidades.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        animal.setChecked(false);
        vegetal.setChecked(false);
        facilidades.setChecked(true);
        outroElementoNatural.setChecked(false);
        outro.setChecked(false);
    }
});
outroElementoNatural.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        animal.setChecked(false);
        vegetal.setChecked(false);
        facilidades.setChecked(false);
        outroElementoNatural.setChecked(true);
        outro.setChecked(false);
    }
});
outro.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        animal.setChecked(false);
        vegetal.setChecked(false);
        facilidades.setChecked(false);
```

```
            outroElementoNatural.setChecked(false);
            outro.setChecked(true);
        }
    });
  }
  @Override
  public void onBackPressed() {
    Intent i = new Intent(TelaCadastroElemento.this, TelaParque.class);
    startActivity(i);
    super.onBackPressed();
  }
}
package com.example.prototipo.telas.parque;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import android.content.Intent;
import android.os.Bundle;
import com.example.prototipo.R;
import com.example.prototipo.classes.adapters.AdapterElemento;
import com.example.prototipo.classes.objetos.Elemento;
import com.example.prototipo.classes.models.ModelParque;
import java.util.ArrayList;

public class TelaListaElementos extends AppCompatActivity {
```

```
RecyclerView rv;
AdapterElemento adapter;
ArrayList<Elemento> list;
static ModelParque mp;
AlertDialog.Builder adb;


@Override
protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_tela_lista_elementos);
   adb = new AlertDialog.Builder(this);
   getSupportActionBar().hide();
   list = new ArrayList<Elemento>();
   rv = (RecyclerView) findViewById(R.id.rve);
   rv.setHasFixedSize(true);
   rv.setLayoutManager(new LinearLayoutManager(this));
   populaLista();
   adapter = new AdapterElemento(this, list, new AdapterElemento.OnItemClickListener() {
      @Override
      public void onItemClick(Elemento me) {
         Intent i = new Intent(TelaListaElementos.this,TelaFotoElemento.class);
         startActivity(i);
         TelaFotoElemento.me = me;
         TelaFotoElemento.mp = mp;
      }
   });
   rv.setAdapter(adapter);
```

```
        adapter.notifyDataSetChanged();
    }

    public void populaLista() {
        for (Elemento e : mp.getElementos()) {
            Elemento me = new Elemento(e.getNome(), e.getTipo(), e.getCaminhoFoto(),
e.getInfos());
            list.add(me);
        }
    }
    @Override
    public void onBackPressed() {
        Intent i = new Intent(TelaListaElementos.this, TelaParque.class);
        startActivity(i);
        super.onBackPressed();
    }
}
package com.example.prototipo.telas.parque;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
```

```
import android.os.Bundle;
import android.text.InputType;
import android.view.View;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.Toast;
import com.example.prototipo.R;
import com.example.prototipo.classes.objetos.Elemento;
import com.example.prototipo.classes.models.ModelParque;
import com.example.prototipo.classes.objetos.Parque;
import com.example.prototipo.telas.principais.MainActivity;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.storage.FileDownloadTask;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;
import java.io.File;
import java.io.IOException;

public class TelaFotoElemento extends AppCompatActivity {
    Bitmap img;
    AlertDialog.Builder adb;
    static ModelParque mp;
    static Elemento me;
    String userId;
    SharedPreferences preferences;
    ImageView iv;
```

```
StorageReference reference;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_tela_foto_elemento);
    reference = FirebaseStorage.getInstance().getReference().child(me.getCaminhoFoto());
    carregaFoto();
    adb = new AlertDialog.Builder(this);
    preferences = getSharedPreferences("Pref", MODE_PRIVATE);
    userId = preferences.getString("userid", "");
    getSupportActionBar().hide();
    iv = findViewById(R.id.fotoElemento);
}

public void deleta(View view) {
    adb.setTitle("Excluir");
    adb.setMessage("Deseja excluir este item?");
    adb.setNeutralButton("Cancelar", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            Toast.makeText(TelaFotoElemento.this, "\uD83D\uDE0E",
Toast.LENGTH_SHORT).show();
        }
    });
    adb.setPositiveButton("Excluir", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
```

```
            deleteImagem(me);
            for (Elemento e : mp.getElementos()) {
               if (e.getNome() == me.getNome()) {
                  mp.getElementos().remove(e);
                  break;
               }
            }
            atualizaParque();
            telaParque();
         }
      });
      adb.show();
   }

   public void deleteImagem(Elemento me) {
      StorageReference storageRef = FirebaseStorage.getInstance()
            .getReference()
            .child(me.getCaminhoFoto());
      storageRef.delete().addOnSuccessListener(new OnSuccessListener<Void>() {
         @Override
         public void onSuccess(Void unused) {
            Toast.makeText(TelaFotoElemento.this, "Item removido com sucesso!",
Toast.LENGTH_SHORT).show();
            finish();
         }
      }).addOnFailureListener(new OnFailureListener() {
         @Override
```

```
        public void onFailure(@NonNull Exception e) {
            Toast.makeText(TelaFotoElemento.this, "Falha ao remover Item!",
Toast.LENGTH_SHORT).show();
        }
    });
  }

  public void telaParque() {
    Intent i = new Intent(TelaFotoElemento.this, TelaParque.class);
    startActivity(i);
    TelaParque.mp = mp;
  }

  public void atualizaParque() {
    Parque p = new Parque(mp.getElementos(), mp.getNome(), mp.getEstado(),
mp.getCidade(), mp.getBairro(), userId, mp.getDescricao(), 5);
    p.salvar();
  }

  public void addInfo(View view) {
    adb = new AlertDialog.Builder(this);
    adb.setTitle("Editar Informações");
    final EditText input = new EditText(this);
    input.setText(me.getInfos());
    input.setInputType(InputType.TYPE_CLASS_TEXT |
InputType.TYPE_TEXT_FLAG_MULTI_LINE);
    adb.setView(input);
    adb.setPositiveButton("OK", new DialogInterface.OnClickListener() {
```

```
        @Override
        public void onClick(DialogInterface dialog, int which) {
            me.setInfos(input.getText().toString());
            for (Elemento e : mp.getElementos()) {
                if(e.getCaminhoFoto().equals(me.getCaminhoFoto())){
                    e.setInfos(me.getInfos());
                    break;
                }
            }
            atualizaParque();
            telaParque();
        }
    });
    adb.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            dialog.cancel();
        }
    });

    adb.show();

}

public void exibeInfo(View view) {
    adb = new AlertDialog.Builder(this);
    adb.setTitle("Informações");
```

```
        adb.setMessage(me.getInfos());
        adb.show();
    }
    public void carregaFoto(){
        final File localFile;
        try {
            localFile = File.createTempFile("foto","jpg");
            reference.getFile(localFile).addOnSuccessListener(new
OnSuccessListener<FileDownloadTask.TaskSnapshot>() {
                @Override
                public void onSuccess(FileDownloadTask.TaskSnapshot taskSnapshot) {
                    img = BitmapFactory.decodeFile(localFile.getAbsolutePath());
                    iv.setImageBitmap(img);
                }
            }).addOnFailureListener(new OnFailureListener() {
                @Override
                public void onFailure(@NonNull Exception e) {
                    Toast.makeText(TelaFotoElemento.this,"Imagem não pôde ser
carregada",Toast.LENGTH_SHORT).show();
                }
            });
        } catch (
                IOException e) {
            e.printStackTrace();
        }
    }
    public void wiki(View view){
        Intent i = new Intent(TelaFotoElemento.this, TelaWiki.class);
```

```
        startActivity(i);
        TelaWiki.url = "https://pt.wikipedia.org/wiki/"+me.getNome();
    }

    @Override
    public void onBackPressed() {
        Intent i = new Intent(TelaFotoElemento.this, TelaListaElementos.class);
        startActivity(i);
        super.onBackPressed();
    }
}
package com.example.prototipo.telas.parque;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import com.example.prototipo.R;

public class TelaWiki extends AppCompatActivity {
    static String url;
    WebView webview;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_tela_wiki);
```

```
        getSupportActionBar().hide();
        webview = (WebView) findViewById(R.id.webview);
        webview.setWebViewClient(new WebViewClient());
        webview.loadUrl(url);
    }
    @Override
    public void onBackPressed() {
        Intent i = new Intent(TelaWiki.this, TelaFotoElemento.class);
        startActivity(i);
        super.onBackPressed();
    }
}
```

APÊNDICE B - Código XML

```
<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"

   xmlns:app="http://schemas.android.com/apk/res-auto"

   xmlns:tools="http://schemas.android.com/tools"

   android:layout_width="match_parent"

   android:layout_height="match_parent"

   android:background="@drawable/cac"


   tools:context=".telas.principais.MainActivity">


   <com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton

      android:id="@+id/botaoSobre"

      android:layout_width="196dp"

      android:layout_height="63dp"

      android:layout_marginStart="8dp"

      android:layout_marginTop="8dp"

      android:layout_marginEnd="8dp"

      android:text="Sobre"

      android:onClick="sobre"

      android:textAlignment="center"

      android:textSize="24sp"

      android:textStyle="normal|bold"

      app:backgroundTint="#156861"

      app:layout_constraintEnd_toEndOf="parent"
```

```
    app:layout_constraintHorizontal_bias="0.497"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

  <com.google.android.gms.common.SignInButton
    android:id="@+id/botao"
    android:layout_width="360dp"
    android:layout_height="50dp"
    android:layout_marginStart="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginBottom="232dp"
    android:contextClickable="true"
    android:textAlignment="center"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:background="@drawable/cac"
  tools:context=".telas.principais.TelaInicial">
```

```
<com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton
    android:id="@+id/botaoSair"
    android:layout_width="196dp"
    android:layout_height="63dp"
    android:layout_marginStart="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginBottom="28dp"
    android:text="Deslogar"
    android:textAlignment="center"
    android:textSize="24sp"
    android:textStyle="normal|bold"
    app:backgroundTint="#4A7C11"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent" />

<com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton
    android:id="@+id/botaoNovoParque"
    android:layout_width="368dp"
    android:layout_height="74dp"
    android:layout_marginStart="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginBottom="35dp"
    android:text="Novo Parque"
    android:textAlignment="center"
```

```
        android:textSize="24sp"
        android:textStyle="normal|bold"
        app:layout_constraintBottom_toTopOf="@+id/botaoSair"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent" />

    <com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton
        android:id="@+id/botaoListarParques"
        android:layout_width="368dp"
        android:layout_height="74dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="100dp"
        android:layout_marginEnd="8dp"
        android:text="Listar Parques"
        android:textAlignment="center"
        android:textSize="24sp"
        android:textStyle="normal|bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
```

```
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/cac_sem"
    tools:context=".telas.parque.TelaCadastroParques">

    <com.google.android.material.textfield.TextInputEditText
        android:id="@+id/nomeP"
        android:layout_width="0dp"
        android:layout_height="72dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="24dp"
        android:layout_marginEnd="8dp"

android:background="@drawable/common_google_signin_btn_icon_dark_normal_backgro
und"
        android:hint="Nome do parque"
        android:textAlignment="center"
        android:textSize="24sp"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <com.google.android.material.textfield.TextInputEditText
        android:id="@+id/estadoP"
        android:layout_width="0dp"
        android:layout_height="72dp"
```

```
        android:layout_marginStart="8dp"
        android:layout_marginTop="4dp"
        android:layout_marginEnd="8dp"

android:background="@drawable/common_google_signin_btn_icon_dark_normal_backgro
und"
        android:hint="Estado"
        android:textAlignment="center"
        android:textSize="24sp"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/nomeP" />


    <com.google.android.material.textfield.TextInputEditText
        android:id="@+id/cidadeP"
        android:layout_width="0dp"
        android:layout_height="72dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="4dp"
        android:layout_marginEnd="8dp"

android:background="@drawable/common_google_signin_btn_icon_dark_normal_backgro
und"
        android:hint="Cidade"
        android:textAlignment="center"
        android:textSize="24sp"
        android:textStyle="bold"
```

```
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/estadoP" />

  <com.google.android.material.textfield.TextInputEditText
    android:id="@+id/bairroP"
    android:layout_width="0dp"
    android:layout_height="72dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="4dp"
    android:layout_marginEnd="8dp"
android:background="@drawable/common_google_signin_btn_icon_dark_normal_backgro
und"
    android:hint="Bairro"
    android:textAlignment="center"
    android:textSize="24sp"
    android:textStyle="bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/cidadeP" />

  <com.google.android.material.textfield.TextInputEditText
    android:id="@+id/descricaoP"
    android:layout_width="0dp"
    android:layout_height="170dp"
```

```
        android:layout_marginStart="8dp"
        android:layout_marginTop="4dp"
        android:layout_marginEnd="8dp"

android:background="@drawable/common_google_signin_btn_icon_dark_normal_backgro
und"
        android:hint="Descrição"
        android:textAlignment="center"
        android:textSize="24sp"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/bairroP" />

    <com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton
        android:id="@+id/botaoCadastraParque"
        android:layout_width="196dp"
        android:layout_height="63dp"
        android:layout_marginStart="8dp"
        android:layout_marginEnd="8dp"
        android:layout_marginBottom="24dp"
        android:text="Cadastrar"
        android:textAlignment="center"
        android:textSize="24sp"
        android:textStyle="normal|bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
```

```
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:background="@drawable/cac_sem"
  tools:context=".telas.parque.TelaListaParques">

  <EditText
    android:id="@+id/pesquisaElemento"
    android:layout_width="0dp"
    android:layout_height="40dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="3dp"
    android:layout_marginEnd="8dp"
    android:hint="ELEMENTO"
    android:textAlignment="center"
    android:textSize="20sp"
    android:textStyle="bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/pesquisaCidade" />
```

```
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/rv"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/pesquisaElemento"
        app:layout_constraintVertical_bias="0.0" />

    <EditText
        android:id="@+id/pesquisaCidade"
        android:layout_width="0dp"
        android:layout_height="40dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="3dp"
        android:layout_marginEnd="8dp"
        android:hint="CIDADE"
        android:textAlignment="center"
        android:textSize="20sp"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/cac_sem"
    tools:context=".telas.parque.TelaParque">

    <com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton
        android:id="@+id/botaoCadastraParque2"
        android:layout_width="0dp"
        android:layout_height="63dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="60dp"
        android:layout_marginEnd="8dp"
        android:onClick="local"
        android:text="Onde Fica"
        android:textAlignment="center"
        android:textSize="24sp"
        android:textStyle="normal|bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.497"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/nomeParque" />
```

```
<com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton
    android:id="@+id/botaoCadastraParque3"
    android:layout_width="0dp"
    android:layout_height="63dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="20dp"
    android:layout_marginEnd="8dp"
    android:onClick="elementos"
    android:text="Itens"
    android:textAlignment="center"
    android:textSize="24sp"
    android:textStyle="normal|bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/botaoCadastraParque2" />

<com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton
    android:id="@+id/botaoCadastraParque4"
    android:layout_width="0dp"
    android:layout_height="63dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="20dp"
    android:layout_marginEnd="8dp"
    android:onClick="descricao"
    android:text="Descrição"
    android:textAlignment="center"
    android:textSize="24sp"
```

```
    android:textStyle="normal|bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.498"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/botaoCadastraParque3" />

<com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton
    android:id="@+id/botaoCadastraParque5"
    android:layout_width="wrap_content"
    android:layout_height="63dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="56dp"
    android:layout_marginEnd="8dp"
    android:onClick="colaborar"
    android:text="Add Item"
    android:textAlignment="center"
    android:textSize="24sp"
    android:textStyle="normal|bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.497"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/botaoCadastraParque4" />

<com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton
    android:id="@+id/botaoRemoveParque"
    android:layout_width="wrap_content"
    android:layout_height="63dp"
```

```
    android:layout_marginStart="8dp"
    android:layout_marginTop="60dp"
    android:layout_marginEnd="8dp"
    android:onClick="deletaParque"
    android:text="Remover Parque"
    android:textAlignment="center"
    android:textSize="24sp"
    android:textStyle="normal|bold"
    app:backgroundTint="#F30A0A"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.495"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/botaoCadastraParque5" />

  <TextView
    android:id="@+id/nomeParque"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="86dp"
    android:layout_marginTop="45dp"
    android:layout_marginEnd="84dp"
    android:textAlignment="center"
    android:textSize="34sp"
    android:textStyle="bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

```
</androidx.constraintlayout.widget.ConstraintLayout>

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"

   xmlns:app="http://schemas.android.com/apk/res-auto"

   xmlns:tools="http://schemas.android.com/tools"

   android:layout_width="match_parent"

   android:layout_height="match_parent"

   android:background="@drawable/cac_sem_opacidade"

   tools:context=".telas.parque.TelaCadastroElemento">


   <de.hdodenhof.circleimageview.CircleImageView

      android:id="@+id/foto"

      android:layout_width="340dp"

      android:layout_height="340dp"

      android:layout_centerHorizontal="true"

      android:layout_centerVertical="true"

      android:layout_marginStart="8dp"

      android:layout_marginEnd="8dp"

      android:layout_marginBottom="24dp"

      android:onClick="foto"

      android:src="@drawable/tirar_foto"

      app:civ_border_color="#FF000000"

      app:civ_border_width="2dp"

      app:layout_constraintBottom_toTopOf="@+id/botaoCadastraElemento"

      app:layout_constraintEnd_toEndOf="parent"

      app:layout_constraintStart_toStartOf="parent" />
```

```
<com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton
    android:id="@+id/botaoCadastraElemento"
    android:layout_width="196dp"
    android:layout_height="63dp"
    android:layout_marginStart="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginBottom="16dp"
    android:onClick="cadastra"
    android:text="Cadastrar"
    android:textAlignment="center"
    android:textSize="24sp"
    android:textStyle="normal|bold"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent" />

<com.google.android.material.textfield.TextInputEditText
    android:id="@+id/nomeE"
    android:layout_width="0dp"
    android:layout_height="72dp"
    android:layout_marginStart="8dp"
    android:layout_marginEnd="8dp"
android:background="@drawable/common_google_signin_btn_icon_dark_normal_backgro
und"
    android:hint="Nome do Item"
```

```
    android:textAlignment="center"
    android:textSize="24sp"
    android:textStyle="bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<RadioButton
    android:id="@+id/radioAnimal"
    android:layout_width="175dp"
    android:layout_height="50dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:text="ANIMAL"
    android:textSize="24sp"
    android:textStyle="bold"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/nomeE" />

<RadioButton
    android:id="@+id/radioVegetal"
    android:layout_width="175dp"
    android:layout_height="50dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:text="VEGETAL"
    android:textSize="24sp"
```

```
    android:textStyle="bold"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/radioAnimal" />

<RadioButton
    android:id="@+id/radioFacilidade"
    android:layout_width="175dp"
    android:layout_height="50dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:text="FACILIDADE"
    android:textSize="24sp"
    android:textStyle="bold"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/radioVegetal" />

<RadioButton
    android:id="@+id/radioOutroElementoNatural"
    android:layout_width="170dp"
    android:layout_height="100dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="8dp"
    android:text="OUTRO ELEMENTO NATURAL"
    android:textSize="24sp"
    android:textStyle="bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/nomeE" />
```

```
    <RadioButton
        android:id="@+id/radioOutro"
        android:layout_width="wrap_content"
        android:layout_height="50dp"
        android:layout_marginEnd="67dp"
        android:text="OUTRO"
        android:textSize="24sp"
        android:textStyle="bold"
        app:layout_constraintBaseline_toBaselineOf="@+id/radioFacilidade"
        app:layout_constraintEnd_toEndOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".telas.parque.TelaListaElementos">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/rve"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:background="@drawable/cac_sem"
```

```
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.5"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:background="@drawable/cac_sem"
  tools:context=".telas.parque.TelaFotoElemento">

  <com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton
    android:id="@+id/botaoRemoveElemento"
    android:layout_width="0dp"
    android:layout_height="63dp"
    android:layout_marginStart="12dp"
    android:layout_marginEnd="12dp"
    android:layout_marginBottom="15dp"
    android:onClick="deleta"
    android:text="Remover Elemento"
    android:textAlignment="center"
    android:textSize="24sp"
```

```
    android:textStyle="normal|bold"
    app:backgroundTint="#F30A0A"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.494"
    app:layout_constraintStart_toStartOf="parent" />

<com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton
    android:id="@+id/botaoAddInfos"
    android:layout_width="0dp"
    android:layout_height="63dp"
    android:layout_marginStart="12dp"
    android:layout_marginEnd="12dp"
    android:layout_marginBottom="15dp"
    android:onClick="addInfo"
    android:text="Editar Informações"
    android:textAlignment="center"
    android:textSize="24sp"
    android:textStyle="normal|bold"
    app:backgroundTint="#50F30A"
    app:layout_constraintBottom_toTopOf="@+id/botaoWiki"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.495"
    app:layout_constraintStart_toStartOf="parent" />

<com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton
    android:id="@+id/botaoExibeInfos"
```

```
    android:layout_width="0dp"
    android:layout_height="63dp"
    android:layout_marginStart="12dp"
    android:layout_marginEnd="12dp"
    android:layout_marginBottom="15dp"
    android:onClick="exibeInfo"
    android:text="Exibir Informações"
    android:textAlignment="center"
    android:textSize="24sp"
    android:textStyle="normal|bold"
    app:backgroundTint="#009688"
    app:layout_constraintBottom_toTopOf="@+id/botaoAddInfos"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.494"
    app:layout_constraintStart_toStartOf="parent" />

<com.google.android.material.floatingactionbutton.ExtendedFloatingActionButton
    android:id="@+id/botaoWiki"
    android:layout_width="0dp"
    android:layout_height="63dp"
    android:layout_marginStart="12dp"
    android:layout_marginEnd="12dp"
    android:layout_marginBottom="15dp"
    android:onClick="wiki"
    android:text="wikipedia"
    android:textAlignment="center"
    android:textSize="24sp"
```

```
        android:textStyle="normal|bold"
        app:backgroundTint="#444747"
        app:layout_constraintBottom_toTopOf="@+id/botaoRemoveElemento"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent" />

    <ImageView
        android:id="@+id/fotoElemento"
        android:layout_width="370dp"
        android:layout_height="400dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="12dp"
        android:layout_marginEnd="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:srcCompat="@tools:sample/avatars" />
</androidx.constraintlayout.widget.ConstraintLayout>
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/cac_sem"
```

```
    tools:context=".telas.parque.TelaWiki">

    <WebView
        android:id="@+id/webview"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.0" />
</androidx.constraintlayout.widget.ConstraintLayout>
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:padding="10dp"
    app:cardCornerRadius="10dp"
    app:cardElevation="8dp">

    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
android:orientation="vertical">


<LinearLayout
   android:layout_width="wrap_content"
   android:layout_height="60dp"
   android:orientation="horizontal">


   <TextView
      android:id="@+id/tvNome"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Nome:"
      android:textSize="24sp"
      android:textStyle="bold" />


   <TextView
      android:id="@+id/valorNomeE"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_marginStart="19dp"
      android:text="nome do parque"
      android:textSize="17sp" />
</LinearLayout>


<LinearLayout
   android:layout_width="wrap_content"
   android:layout_height="60dp"
```

```
    android:orientation="horizontal">

    <TextView
        android:id="@+id/tvCidade"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Cidade:"
        android:textSize="24sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/valorTipoE"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="9dp"
        android:text="nome da cidade"
        android:textSize="17sp" />
</LinearLayout>

<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

    <TextView
        android:id="@+id/tvEstado"
        android:layout_width="wrap_content"
```

```
            android:layout_height="wrap_content"
            android:text="Estado:"
            android:textSize="24sp"
            android:textStyle="bold" />

        <TextView
            android:id="@+id/valorEstado"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginStart="12dp"
            android:text="nome do estado"
            android:textSize="17sp" />
    </LinearLayout>
  </LinearLayout>


</androidx.cardview.widget.CardView>
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView
xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  app:cardCornerRadius="5dp"
  android:elevation="5dp"
  app:cardUseCompatPadding="true">
```

```
<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:padding="15dp">

    <de.hdodenhof.circleimageview.CircleImageView
        android:layout_width="80dp"
        android:layout_height="80dp"
        android:src="@mipmap/ic_launcher"
        android:layout_centerVertical="true"
        android:id="@+id/foto"
        app:civ_border_width="2dp"
        app:civ_border_color="#FF000000"
        />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/valorNomeE"
        android:text="Aqui vai o nome do item"
        android:textStyle="bold"
        android:textSize="25sp"
        android:textColor="#000"
        android:layout_toRightOf="@id/foto"
        android:layout_marginLeft="10dp"/>

    <TextView
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/valorTipoE"
        android:text="Aqui vai o tipo do item"
        android:textSize="25sp"
        android:textColor="#000"
        android:layout_toRightOf="@id/foto"
        android:layout_below="@id/valorNomeE"
        android:layout_marginLeft="10dp"/>

    </RelativeLayout>

</androidx.cardview.widget.CardView>
```

APÊNDICE C - Artigo

CIÊNCIA AMBIENTAL CIDADÃ: UM APLICATIVO PARA ENGAJAR A PARTICIPAÇÃO DAS PESSOAS NOS PARQUES AMBIENTAIS.

Jackson Dener Wrublak

Departamento de Informática e Estatística - Universidade Federal de Santa Catarina (UFSC)

jwrublak@gmail.com

Abstract. *Technological changes in recent decades have made it possible to share goods, services and information between people in a more dynamic way, this statement appears in the article "Effectiveness of Crowdsourcing as Support to Public Security", by João Moisés Brito Mota and Afonso Carneiro Lima Combining technological advances with crowdsourcing techniques and citizen science, the course conclusion work aims to solve the problem of people's disengagement in relation to environmental parks through the feasibility and implementation of a computational tool in a mobile application format.*

Resumo. *As mudanças tecnológicas ocorridas nas últimas décadas têm possibilitado o compartilhamento de bens, serviços e informações entre as pessoas de maneira mais dinâmica, esta afirmação consta no artigo "Efetividade do Crowdsourcing como Apoio à Segurança Pública", de João Moisés Brito Mota e Afonso Carneiro Lima. Unindo o avanço tecnológico às técnicas de crowdsourcing e ciência cidadã, o trabalho de conclusão de curso visa resolver o problema de desengajamento das pessoas em relação aos parques ambientais através da viabilização e implementação de uma ferramenta computacional em formato de aplicativo de celular.*

1. Introdução

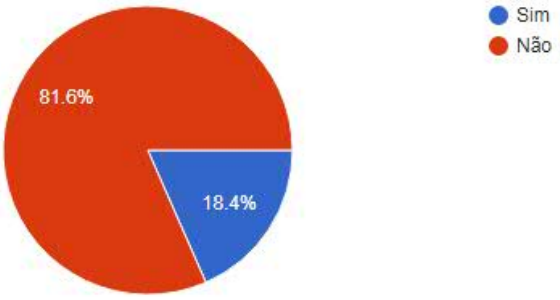
De acordo com Patrícia Alexandrini Menao 2019, as áreas verdes urbanas possuem diversos benefícios, que por sua vez vão muito além da valorização visual e ornamental de um espaço. Elas possuem a importante função de reduzir efeitos da poluição e dos ruídos, agem diretamente na redução da temperatura e na velocidade dos ventos, além de influenciarem no balanço hídrico e ainda podem servir de abrigo a diversos animais silvestres que vivem nas cidades, como pássaros, insetos e até macacos.

Portanto, o objetivo geral do trabalho visa identificar o nível de desengajamento das pessoas em relação aos parques ambientais, modelar e oferecer uma ferramenta computacional para smartphones que estimule e apóie o interesse neste tema, trazendo informações e propondo a participação da comunidade, ao mesmo tempo coletando dados para auxílio à tomada de decisões para os gestores.

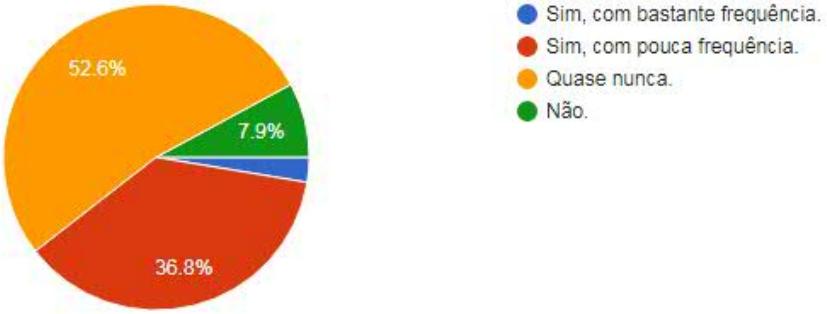
O nível de desengajamento foi medido através de uma pesquisa no formato de enquete com moradores de Florianópolis e região, visando coletar algumas informações relativas ao nível de informação. As respostas foram coletadas através de um formulário

virtual divulgado nas redes sociais em dois momentos, primeiramente entre os meses de maio e junho de 2019, depois entre janeiro e maio de 2021. Em ambas as coletas o padrão de respostas manteve-se o mesmo, mostrando que a pandemia do novo coronavírus não impactou no conhecimento e utilização dos parques:

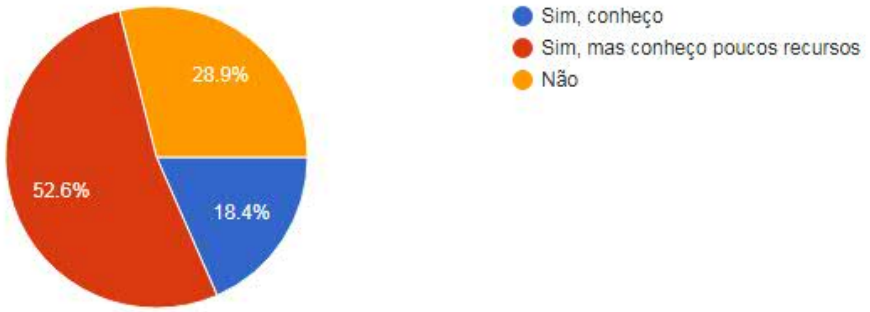
Você sabia que Florianópolis possui mais de 20 parques Ecológicos? (Parque de Coqueiros, Parque Ecológico do Córrego Grande, Jardim Botânico, Parque Municipal da Lagoa do Per, Parque da Luz, etc)



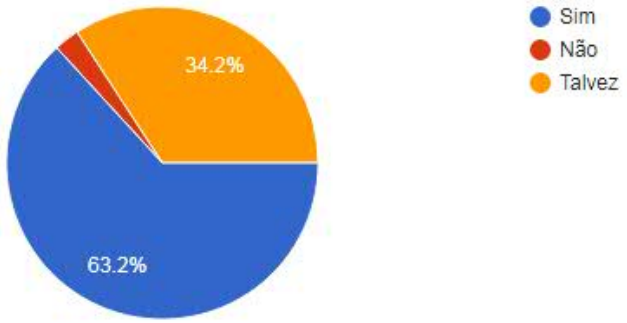
Você costuma visitar parques ecológicos?



Você conhece todos os recursos do parque que você visita? (Lixeiras, locais para crianças, piqueniques, trilhas, locais esportivos, mesas, bancos, onde encontrar determinado animal ou vegetal)



Se existisse alguma ferramenta que coletasse informações sobre o parque e te auxiliasse na tomada de decisão sobre os recursos, você utilizaria essa ferramenta?



Com base nas respostas dos cidadãos, percebe-se pouco conhecimento entre os entrevistados em saber se existe uma variedade de parques em Florianópolis, apesar de apenas 8,6% dos entrevistados não terem o hábito de visitá-los. Nota-se também que apenas 2,6% dos entrevistados visitam os parques ecológicos com bastante frequência e 52,6% quase nunca faz a visitação. Esse alto índice de não visitação pode ter relação com a barreira encontrada no artigo PERCEPÇÃO PARENTAL DAS BARREIRAS PARA O CONTATO DA CRIANÇA COM A NATUREZA, de Peres, Felippe e Kuhnen, 2019, barreira esta que é a falta de disponibilidade dos pais para acompanhar os filhos a áreas verdes urbanas. Assim como as pessoas que participaram do questionário no estudo citado (pais de crianças de 6 a 9 anos de idade), há possibilidade de que outras pessoas também possuam essa mesma barreira.

Desenvolvimento

A solução proposta visa tratar os problemas encontrados nos dois primeiros questionamentos, sobre o conhecimento da diversidade de parques e a frequência nas visitações, respectivamente. O estudo consiste em desenvolver um aplicativo móvel prático, objetivo e simplificado para que a comunidade como um todo tenha condições de utilizá-lo.

O lado front-end da aplicação foi totalmente implementado utilizando a IDE Android Studio, optou-se pela plataforma android pois, segundo o site canaltech, em publicação feita por Bruna Rasmussen, 70,1% dos smartphones vendidos no mundo possuem esse sistema operacional. A linguagem escolhida para implementar a parte lógica do aplicativo foi Java juntamente com o XML para a parte de GUI, por se tratar da linguagem mais trabalhada durante o período de graduação, além de ser nativa em programação para Android. Após escolher a plataforma, IDE e linguagem (previstas nos requisitos não funcionais), foram analisadas algumas possibilidades de Android SDK, que trata-se do Kit de Desenvolvimento de Software para Android. Esse kit permite aos desenvolvedores criarem aplicativos para a plataforma Android de forma nativa. O Android SDK inclui projetos de exemplo com código-fonte, ferramentas de desenvolvimento, emuladores e bibliotecas necessárias para criar os aplicativos Android.

Para o lado back-end, foi utilizado em sua totalidade o Google Firebase, que é a plataforma de desenvolvimento de aplicativos móveis do Google. Essa plataforma possui versatilidade e os tipos de aplicativos que podem ser desenvolvidos com Firebase são: Android, iOS e Web.

Toda sua base é construída na infraestrutura do Google, sendo categorizado como um programa de banco de dados NoSQL, que armazena dados em documentos do tipo JSON.

A codificação do projeto pode ser acessada no seguinte repositório: https://github.com/jackdnr/TCC

Conclusão

Através da enquete, um nível de desengajamento consideravelmente alto em relação aos parques ambientais foi detectado. Portanto, foi desenvolvida uma ferramenta computacional para smartphones que estimula e apóia o interesse das pessoas em áreas verdes, trazendo informações e propondo a participação, ao mesmo tempo que coleta dados para auxílio em tomadas de decisões.

Referências

SILVA, Eduardo. Firebase: o que é e quando usar no desenvolvimento mobile?. [S. l.], 6 maio 2021. Disponível em: https://blog.geekhunter.com.br/firebase-o-que-e-e-quando-usar-no-desenvolvimento-mobile/. Acesso em: 27 maio 2021.

Peres, Felippe e Kuhnen: PERCEPÇÃO PARENTAL DAS BARREIRAS PARA O CONTATO DA CRIANÇA COM A NATUREZA, 2019.

MOTA, João Moisés Brito; LIMA, Afonso Carneiro. Efetividade do Crowdsourcing como Apoio à Segurança Pública. Universidade de Fortaleza, Programa de Pós-graduação em Administração, Fortaleza, CE, Brasil, [s. l.], 19 ago. 2018. Disponível em: https://www.scielo.br/j/rac/a/G8yYVspTFRNGVZGBHj388hy/?lang=pt&format=pdf. Acesso em: 19 mar. 2021.

A IMPORTÂNCIA DAS ÁREAS VERDES URBANAS. Portal de Educação Ambiental, 11 mar. 2019. Disponível em: https://www.infraestruturameioambiente.sp.gov.br/educacaoambiental/vida-sustentavel/a-importancia-das-areas-verdes-urbanas/. Acesso em: 23 ago. 2021.