

EXPLORAHABITAT: UM PROJETO PARA AUXILIAR AS SAÍDAS A CAMPO DOS CLUBES DE CIÊNCIAS

Matheus Soares Lima, Dalton Solano Reis – Orientador

Curso de Bacharel em Ciência da Computação
Departamento de Sistemas e Computação
Universidade Regional de Blumenau (FURB) – Blumenau, SC – Brasil

matheusl@furb.br, dalton@furb.br

Resumo: O presente artigo apresenta o processo de desenvolvimento da extensão de um aplicativo para auxiliar saídas a campo dos Clubes de Ciências utilizando o framework Flutter. Tem como objetivo proporcionar uma melhor experiência de uso. O aplicativo foi desenvolvido utilizando ferramentas específicas do Flutter para gerenciamento de estados e injeção de dependências. Também foi utilizada a plataforma Back4App em conjunto com o Parse para criação de um backend com gerenciamento de banco de dados e autenticação de usuários. Para avaliar a funcionalidade do aplicativo, foi elaborado um roteiro de utilização com onze usuários. O aplicativo atingiu seu objetivo proporcionando uma melhor experiência de uso.

Palavras-chave: Flutter. Saídas a Campo. Gerenciamento de estados. Injeção de dependências. Back4App.

1 INTRODUÇÃO

Segundo Freitas e Santos (2021), um Clube de Ciências é uma subcategoria de um Clube Escolar que possui o objetivo de reunir um grupo de pessoas para promover discussões e momentos de lazer sobre diversos temas no qual há um interesse mútuo. Um Clube Escolar se diferencia de outros clubes justamente pelo seu objetivo educacional entre professores e alunos. O Clube de Ciências se segmenta dos Clubes Escolares na especialização na comunicação da ciência entre os participantes do clube.

O Clube de Ciências é composto por professores que são os mediadores do conhecimento e estudantes comumente chamados de clubistas evitando serem referidos como alunos, pois de acordo com Freitas e Santos

Consideramos que no Clube de Ciências o termo aluno seria inapropriado, pois nesse espaço espera-se que os participantes sejam ativos, protagonistas e que suas vozes sejam consideradas nas decisões. (FREITAS; SANTOS; 2021, p. 24).

Dentro deste contexto, para o aprendizado científico os clubistas são expostos a uma grande gama de atividades em diversas áreas, sendo que o mediador individualmente ou através de um consenso comum entre todos os clubistas definirá a estratégia mais adequada ao objetivo pretendido (FREITAS; SANTOS; 2021, p. 28). De acordo com Córdoba (2012, p. 3), as atividades podem ser trabalhos em equipes em projetos e estudos científicos, atividades laboratoriais, saídas a campo em acampamentos ou passeios científicos, organização e implementação de campanhas, organização de atividades culturais e recreativas, organização e participação em atividades de divulgação, como feiras, conferências para clubistas e exposições e até atividades de colaboração com instituições comunitárias.

Korbes (2021) desenvolveu o aplicativo ExploraHabitat que tem como objetivo auxiliar as saídas a campo dos Clubes de Ciências, permitindo que o clubista a partir da criação de atividades propostas por um mediador, desperte um lado mais investigativo, possibilitando uma maior interação com a natureza e o desenvolvimento de sua própria autonomia. O aplicativo utiliza recursos dos dispositivos móveis como o Global Positioning System (GPS), câmera, vídeo e áudio para simular instrumentos de uso comum na realização das atividades de um clubista. Korbes (2021) afirma que, mesmo o ExploraHabitat cumprindo o esperado, existem problemas que dificultam a utilização do aplicativo, como por exemplo, a falta de um design mais atrativo e amigável, interfaces responsivas que se adaptam de acordo com o dispositivo, entre outros. Contudo outro ponto notado, foi a falta de exploração do framework Flutter, no qual não foi adotado nenhum padrão de desenvolvimento e a utilização de um gerenciador de estados, como o MobX do Flutter, por exemplo, o que auxilia na realização de melhorias na interface e na inclusão de novas funcionalidades facilitando a extensão em projetos futuros.

Desta forma, o objetivo deste trabalho é estender o aplicativo ExploraHabitat (KORBES, 2021) proporcionando uma melhor experiência de uso com a modernização da interface e uma melhor arquitetura no desenvolvimento do aplicativo. Os objetivos específicos são: avaliar a efetividade da utilização de um gerenciador de estados e um injetor de dependências, avaliar a utilização de uma plataforma *backend as a service* (BaaS) para sincronização das informações e avaliar a usabilidade do aplicativo com os usuários.

2 FUNDAMENTAÇÃO TEÓRICA

Esta seção tem como objetivo explorar as técnicas e ferramentas mais relevantes para a realização deste trabalho. A subseção 2.1 apresenta o conceito de gerenciamento de estados. A subseção 2.2 apresenta o conceito da injeção de dependências. A subseção 2.3 apresenta o conceito de Backend as a Service.

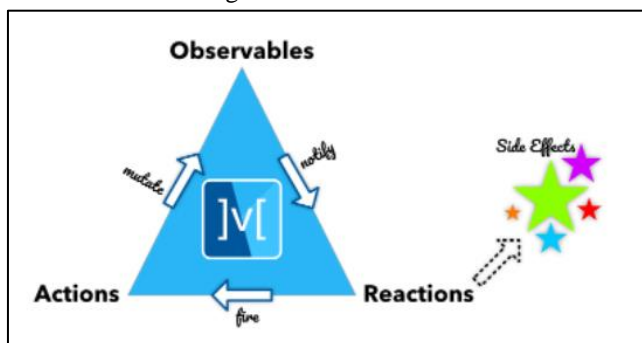
2.1 GERENCIAMENTO DE ESTADOS

No desenvolvimento de um aplicativo em Flutter é importante a definição de uma forma de gerenciamento de estados na interface do aplicativo. O estado é a informação dos componentes em tela que pode mudar com o tempo e é utilizado para definir como eles irão se comportar (FLUTTER; 2022a). O Flutter é considerado declarativo, ou seja, ele constrói a interface para refletir diretamente o estado atual do aplicativo. Quando o estado muda os componentes são reconstruídos na interface (FLUTTER; 2022b).

De acordo com a documentação do Flutter (2022c), existem dois tipos de estado o *ephemeral state* e o *app state*. O *ephemeral state* é o estado que pode ser contido em um único componente, quando não há necessidade de acessar esse estado em outras telas, neste caso não é necessário a utilização de um gerenciador de estados. O *app state* é o estado que precisa ser mantido e compartilhado enquanto o usuário navega pelas telas do aplicativo, como por exemplo, as preferências do usuário em uma loja de compras ou os artigos que ainda não foram lidos em um site de notícias. O gerenciamento de estados de uma aplicação é um assunto complexo, por isso existem diversas abordagens e ferramentas para facilitar este processo. É importante levar em consideração que nem todas as abordagens podem ser viáveis para qualquer cenário, é necessário entender o seu funcionamento e qual o cenário mais recomendado para ser utilizada (FLUTTER; 2022d).

O MobX é uma biblioteca de gerenciamento de estados baseada em uma estrutura composta de observáveis, ações e reações. Utiliza anotações e geradores de código para reduzir a quantidade de código repetitivo necessário para o gerenciamento de estados (FLUTTER; 2022e). No MobX os observáveis representam o estado da aplicação. As ações são usadas para alterar os observáveis. As reações observam o estado e são notificadas quando ele for alterado, desta forma reconstruindo o componente na interface com o novo estado. Na Figura 1 está a representação do fluxo do funcionamento do MobX.

Figura 1 – Fluxo MobX



Fonte: Flutter (2022d).

De acordo com Slepnev (2020, p. 80), após a utilização do MobX algumas das vantagens percebidas foi o mecanismo de atualização utilizando observáveis, reações e ações, a redução da quantidade de código necessário e a facilidade no aprendizado dos seus conceitos básicos. Slepnev (2020, p. 80) notou que uma das principais desvantagens percebidas foi a falta uma função *built-in* para injeção de dependências.

2.2 INJEÇÃO DE DEPENDÊNCIAS

A injeção de dependências é um padrão de projeto utilizado para diminuir o acoplamento entre os módulos de um software. Separando os componentes de ter dependências diretas faz com que o código seja mais organizado e fácil de realizar melhorias e correções (FLUTTER; 2022f).

Conforme o desenvolvimento de um aplicativo progride, é necessário separar em classes a lógica de negócio dos componentes da interface. Para acessar as classes com lógica de negócio por meio da interface do aplicativo existem bibliotecas que realizam a injeção das dependências, como o Provider e o GetIt. O Provider permite instanciar e expor uma classe para que possa ser acessada em qualquer lugar dentro de uma árvore de componentes na interface. Para acessar esse valor é necessário apenas que o componente conheça o contexto em que foi criado (FLUTTER; 2022g). No GetIt todas as dependências são criadas como um *singleton*, ou seja, existirá apenas uma instância da classe especificada, assim descartando a necessidade de conhecer o contexto para acessar o seu valor. Após realizar o registro é possível encontrar a classe em qualquer lugar da aplicação (FLUTTER; 2022f).

2.3 BACKEND AS A SERVICE

À medida que a concorrência no mercado de desenvolvimento de softwares aumenta, os desenvolvedores estão sempre à procura de maneiras rápidas e eficientes de disponibilizar novos aplicativos. Com a utilização de plataformas de Backend as a Service (BaaS) é possível gerenciar os serviços de backend permitindo que os desenvolvedores concentrem no lado do cliente de seus aplicativos. O BaaS automatiza e simplifica o processo de desenvolvimento lidando com toda a estrutura, desta forma não é necessário que o desenvolvedor comece do zero criando e aperfeiçoando os serviços de backend (BACK4APP; 2022).

O Back4App é uma plataforma BaaS baseada no Parse Server, que é uma estrutura de BaaS de código fonte aberto. O Parse Server tem como principal objetivo prover recursos para automatizar o processo de desenvolvimento de um backend para aplicativos. A plataforma Back4App é o principal serviço de hospedagem do Parse Server e oferece diversos serviços adicionais como: migração de aplicativos, ferramentas de gerenciamento baseadas na web, backup e recuperação, monitoramento e suporte especializado (BACK4APP; 2022). Também é possível realizar o gerenciamento de banco de dados, armazenamento de arquivos e o gerenciamento de usuários.

2.4 VERSÃO ANTERIOR DO APLICATIVO

Korbes (2021) desenvolveu o aplicativo ExploraHabitat para as plataformas Android e iOS. Foi desenvolvido com a linguagem Dart e o *framework* Flutter, utilizando o UI Toolkit para a interface. O aplicativo tem como principal objetivo apoiar atividades de saída a campo em Clubes de Ciências, com a utilização de recursos dos dispositivos móveis como câmera, vídeo, áudio e GPS para a realização das atividades dos Clubistas, facilitar a coleta de dados, compartilhar informações com outros usuários e a própria construção científica deles.

A utilização do aplicativo ExploraHabitat inicia com a escolha de perfil do usuário, onde ele escolhe entre Professor ou Clubista, após isso realiza a autenticação através de uma conta Google. Uma vez autenticado, o usuário poderá realizar o cadastro de um tema referente ao estudo em campo que será realizado. Dentro de um tema cadastrado poderá ser incluso objetivos e as atividades que irão compor o roteiro que será executado pelos Clubistas. O aplicativo permite que para cada objetivo possa ser criado um roteiro de atividades. As atividades são variadas, desde a tirar uma foto, realizar a gravação de um áudio ou vídeo, por exemplo. Os cadastros normalmente deverão ser realizados pelo professor coletivamente com os Clubistas, para que assim tenha uma mobilização entre todos os usuários. Por fim devido o aplicativo ter sido planejado para evitar a utilização de internet, pois em saídas a campo os clubistas podem estar em lugares onde não possui acesso a redes móveis ou wi-fi, desta forma o aplicativo permite a integração com outros usuários através da geração de um QR Code. Ele é gerado quando professor finaliza o cadastro de um tema, sendo assim, o clubista poderá realizar a leitura e obter todos os dados relacionado ao tema, incluindo o roteiro com as atividades para conclusão dos objetivos.

Na realização das atividades, o Clubista preenche as informações solicitadas, de acordo com a definição previamente realizada no cadastro do tema e realiza a finalização da tarefa. Após finalizar todas as atividades propostas no roteiro, o Clubista envia as respostas para o professor. As atividades enviadas são armazenadas em uma estrutura de pastas no Google Drive com a criação de um *folder*, permitindo que os dados possam ser acessados posteriormente.

Korbes (2021) apresenta resultados de um questionário sobre a usabilidade do aplicativo. Com os gráficos não foi possível identificar as principais dificuldades dos usuários, porém é possível identificar alguns problemas de alteração de informações de temas existentes e no envio de tema para o professor. Em geral, o aplicativo foi bem aceito e considerado autoexplicativo. Korbes (2021) relata que a maioria das ferramentas utilizadas foram efetivas na sua proposta, entretanto entende que o Flutter poderia ser mais bem estudado, principalmente em questões de redimensionamento de telas, designs mais atrativos, limites e posições de campo. Afirma também que uma limitação do aplicativo foi justamente no âmbito de desenhos em tela, que não cumpriu conforme o esperado. Korbes (2021) propôs sugestões de extensões futuras como a implementação de novas atividades, mais autonomia para o Clubista alterar e remover atividades, otimização no armazenamento de dados, melhorar a responsividade da tela, acessibilidade em outras línguas, inserir recursos do dispositivo como acelerômetro e bússola 3 entre outros.

Korbes (2021) concluiu que o trabalho é relevante para o estudo acadêmico em Clubes de Ciências, pois auxilia na automatização do processo de aplicação de perguntas, incentivando os Clubistas a trabalharem com mais autonomia. Também permite que possam dedicar mais tempo em outras atividades de ensino, ter uma maior interação com a natureza a partir do dispositivo móvel e na flexibilidade do professor ou o próprio Clubista proporem os roteiros das atividades.

2.5 TRABALHOS CORRELATOS

Esta subseção apresenta os trabalhos que possuem características semelhantes ao aplicativo desenvolvido. O primeiro trabalho é o Clube Virtual de Ciências (BET *et al.*; 2004) que busca trazer o ambiente do Clube de Ciências para o mundo virtual, apresentado no Quadro 1. O segundo se trata da proposta de criação um ambiente u-learning (MENDONÇA *et al.*; 2018) para contribuir no ensino e aprendizagem de botânica, apresentado no Quadro 2. O terceiro

é referente ao Ambcare (ROSA, 2015) um aplicativo desenvolvido com o objetivo de auxiliar a solução de incidentes ambientais, apresentado no Quadro 3.

Quadro 1 – Trabalho Correlato Clube Virtual de Ciências

Referência	Bet <i>et al.</i> (2004).
Objetivos	Simular o ambiente do Clube de Ciências vivenciado em sala de aula para a internet, permitindo a interação de professores e alunos sem a necessidade da presença física.
Principais funcionalidades	Possibilitar a divulgação de pesquisas feitas nas escolas. Simular experimentos relacionados com as disciplinas que fazem parte do clube. Responder testes para avaliação do conhecimento dos usuários.
Ferramentas de desenvolvimento	Web, não foi especificado outras ferramentas pelo autor.
Resultados e conclusões	Bet <i>et al.</i> (2004) concluiu que a utilização do computador nos estudos como um ótimo instrumento para o ensino, principalmente pelos recursos audiovisuais e a interatividade permitida entre os usuários, ainda mais com a internet facilitando a comunicação e sendo utilizada como fonte de pesquisa.

Fonte: elaborado pelo autor.

Quadro 2 – Trabalho Correlato Ambiente de aprendizagem ubíqua

Referência	Mendonça <i>et al.</i> (2018).
Objetivos	Contribuir no processo de ensino e aprendizagem de botânica, auxiliando na execução das aulas de campo através de um ambiente de aprendizagem ubíqua na utilização de alguns instrumentos dos dispositivos moveis.
Principais funcionalidades	Registrar a localização de plantas utilizando o GPS do celular. Criar conteúdo de aprendizado associados às plantas. Avaliar o conhecimento do aluno ao procurar as plantas associadas ao conteúdo. Identificar a planta correta com base no GPS do aluno. Apresentar dicas caso o aluno informe a planta incorreta.
Ferramentas de desenvolvimento	Framework Ionic e servidor web em PHP.
Resultados e conclusões	Mendonça <i>et al.</i> (2018) explica que como o trabalho está em fase de desenvolvimento apenas alguns resultados foram obtidos na fase inicial do projeto, como o levantamento e definição de requisitos, a caracterização do cenário de aplicação e a arquitetura do ambiente. Para extensão, pretende-se concluir a etapa de desenvolvimento e aplicar novos testes funcionais e de validação para verificar o comportamento em um ambiente real, para assim, constar se os objetivos da pesquisa foram de fato alcançados.

Fonte: elaborado pelo autor.

Quadro 3 – Trabalho Correlato Ambcare

Referência	Rosa (2015).
Objetivos	Disponibilizar recursos para auxiliar órgãos responsáveis pela monitoração e elaboração de planos de contingência para incidentes relacionados ao meio ambiente.
Principais funcionalidades	Registrar a ocorrência de incidentes ambientais. Apresentar os incidentes ambientais próximos em um mapa. Possibilitar que outros usuários apoiem e comentem nos incidentes. Permitir que uma entidade assuma um incidente e seja resolvido.
Ferramentas de desenvolvimento	Framework Phonegap para as plataformas Android, iOS e Windows Phone.
Resultados e conclusões	Rosa (2015) concluiu que é necessário um breve período de utilização após o lançamento efetivo do aplicativo, para que assim, seja verificado a efetividade da aplicação, mas que é possível assumir que poderá contar com o apoio popular e contribuir para o auxílio na solução de incidentes ambientais.

Fonte: elaborado pelo autor.

3 DESCRIÇÃO DO APLICATIVO

Nesta seção são apresentadas as etapas de especificação e desenvolvimento mais relevantes para o entendimento sobre o aplicativo desenvolvido.

3.1 ESPECIFICAÇÃO

O aplicativo permite a criação de temas para serem usados em saídas a campo para auxílio em pesquisas e coleta de dados. Os temas são utilizados como forma de abordagem para um determinado assunto de maneira mais abrangente

e genérica. Dentro de um tema são criados os objetivos que afunilam a ideia inicial em uma parte mais específica do assunto abordado. Para cada objetivo deve ser definido ao menos uma atividade que será um conjunto de ações que devem ser realizadas para atingir o propósito do objetivo.

Os Requisitos Funcionais (RF) do aplicativo estão apresentados no Quadro 4. Os Requisitos Não Funcionais (RNF) estão apresentados no Quadro 5:

Quadro 2 – Requisitos funcionais

RF01 - permitir a criação de um novo usuário
RF02 - permitir a autenticação do usuário
RF03 - permitir manter um tema (Create, Read, Update, Delete, List - CRUD)
RF04 - permitir o cadastro de objetivos específicos vinculados ao tema
RF05 - permitir o cadastro atividades vinculadas aos objetivos
RF06 - permitir a sincronização de um tema criado
RF07 - permitir a busca de um tema sincronizado
RF08 - permitir o envio de respostas para um tema sincronizado
RF09 - permitir a busca de respostas enviadas para um tema

Fonte: elaborado pelo autor.

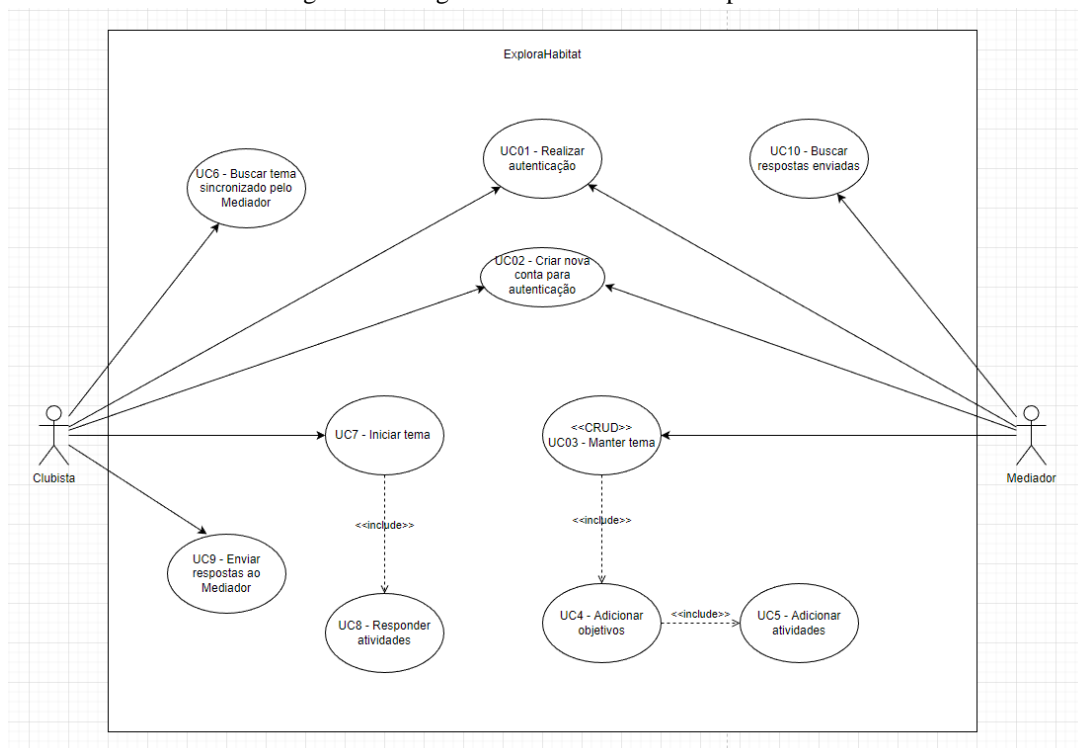
Quadro 5 – Requisito não funcionais

RNF01 - utilizar o <i>framework</i> Flutter e a linguagem de programação Dart
RNF02 - utilizar o Material Design do Flutter para desenvolvimento das interfaces
RNF03 - utilizar a biblioteca MobX para gerenciamento de estados
RNF04 - utilizar as bibliotecas Provider e GetIt para injeção de dependências
RNF05 - utilizar a biblioteca Hive para armazenamento de dados na memória do dispositivo
RNF06 - utilizar a biblioteca Parse Server para salvar os dados em um banco de dados no Back4App
RNF07 - deve ser desenvolvido no ambiente de programação Android Studio
RNF08 - deve permitir o funcionamento do aplicativo sem acesso à internet

Fonte: elaborado pelo autor.

Para melhor compreensão das funcionalidades do aplicativo a Figura 2 apresenta o diagrama de casos de uso dos atores Clubista e Mediador.

Figura 2 – Diagrama de casos de uso do aplicativo



Fonte: elaborado pelo autor.

No Quadro 6 é apresentada a matriz de rastreabilidade entre os requisitos funcionais e no Quadro 4 e os casos de uso da Figura 1.

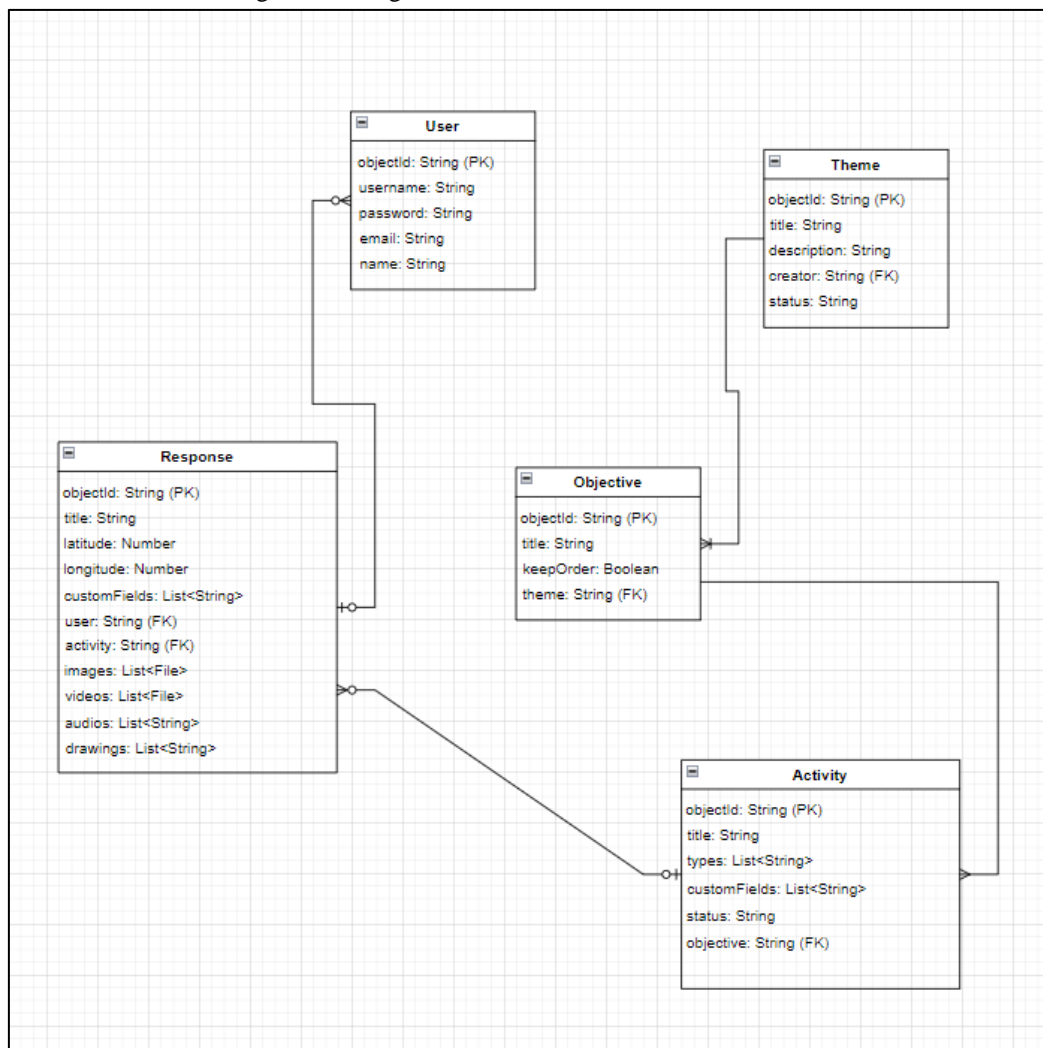
Quadro 6 – Matriz de rastreabilidade

Requisitos Funcionais (RF)	Casos de uso
possibilitar a autenticação do usuário	UC01
permitir a criação de um novo usuário	UC02
permitir manter um tema (CRUD)	UC03
permitir o cadastro de objetivos específicos vinculados ao tema	UC04
permitir o cadastro atividades vinculadas aos objetivos	UC05
permitir a busca de um tema sincronizado	UC06
permitir o envio de respostas para um tema sincronizado	UC07, UC08
permitir a busca de respostas enviadas para um tema	UC10

Fonte: elaborado pelo autor.

Para armazenar e gerenciar as informações foi utilizado o banco de dados hospedado pelo Back4App vinculado à aplicação. No Apêndice A apresenta a plataforma Back4App. A Figura 3 apresenta a modelagem utilizada na estruturação das tabelas. A classe *User* representa os dados do usuário autenticado. A classe *Theme* representa os temas que são criados pelos usuários. A classe *Objective* representa os objetivos vinculados a um tema. A classe *Activity* representa as atividades vinculadas a um objetivo. As atividades possuem uma lista de perguntas denominada *customFields* e uma lista dos recursos que são utilizados denominada *types*. A classe *Response* representa as respostas vinculadas a uma atividade que são enviadas pelos usuários. A resposta armazena as coordenadas da posição do usuário ao realizar o envio, as perguntas preenchidas e a lista de arquivos dos recursos utilizados como: imagens, áudios, vídeos e desenhos.

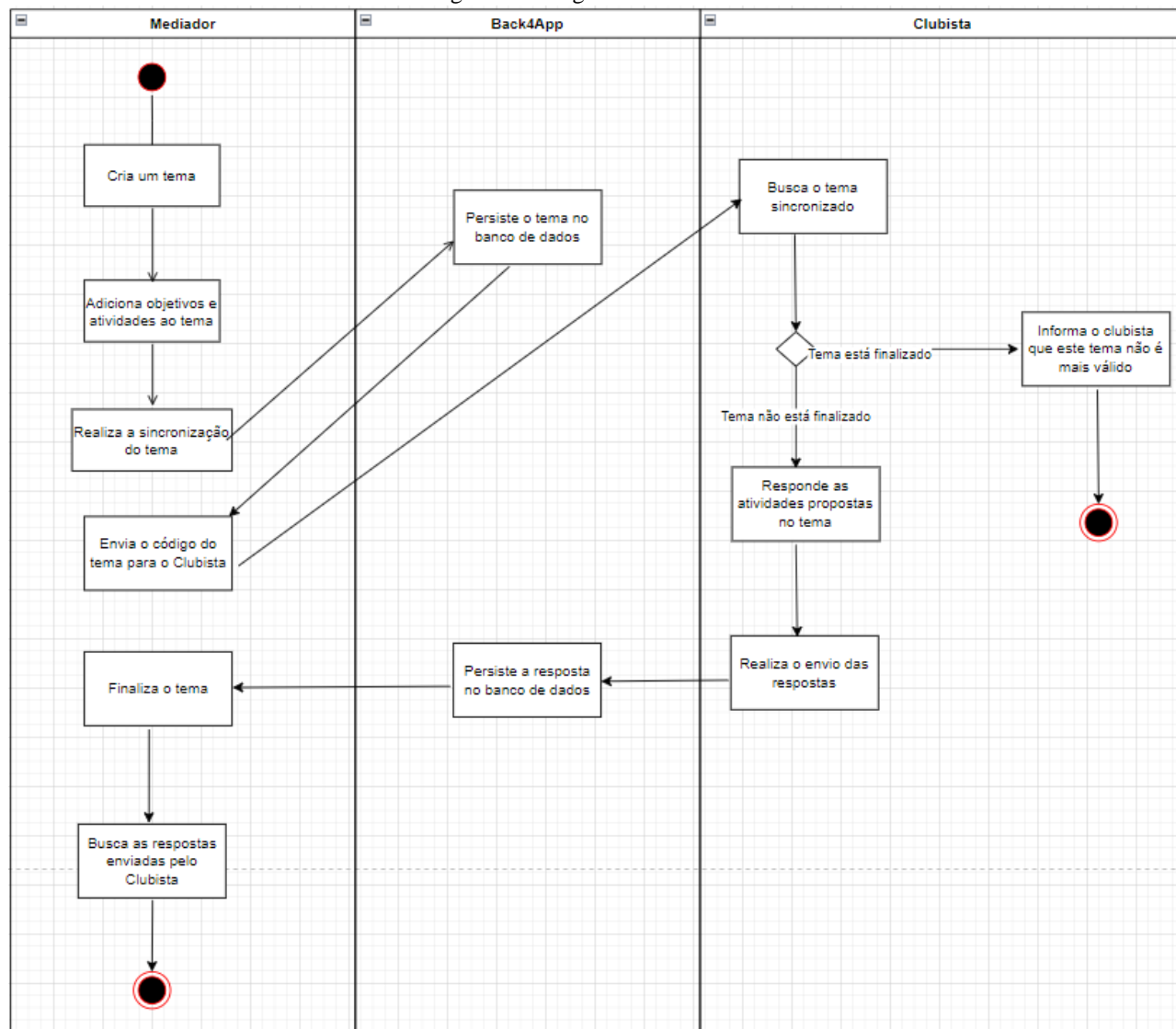
Figura 3 – Diagrama modelo entidade e relacionamento



Fonte: elaborado pelo autor.

No aplicativo os temas podem ser sincronizados ficando disponível para que qualquer usuário com o código do tema possa buscá-lo e executar as atividades propostas. Um tema pode ficar restrito a esse processo caso o usuário que o criou realize a sua finalização, desta forma é determinado que o tema está expirado e não aceita mais o envio de respostas. Após a finalização de um tema o usuário pode visualizar todas as respostas das atividades enviadas. A Figura 4 apresenta o diagrama de atividades demonstrando o fluxo da execução de um tema.

Figura 4 – Diagrama de atividades



Fonte: elaborado pelo autor.

3.2 IMPLEMENTAÇÃO

Nesta seção será apresentado os detalhes sobre o desenvolvimento do aplicativo descrevendo as partes mais importantes da implementação.

3.2.1 Estrutura do projeto

O projeto foi dividido em diretórios específicos para cada função no aplicativo. O diretório de UI é responsável por toda a parte da codificação da interface. Dentro dele são definidas as telas de navegação e os componentes comuns. O diretório services possui as Stores do MobX, as classes modelos e os repositórios para gerenciamento do banco de dados através do Parse Server.

A arquitetura do aplicativo foi estruturada para que cada tela possua uma Store do MobX para armazenar o estado e a partir dele determinar o comportamento dos componentes em tela. Como o aplicativo permite a navegação entre várias telas, é necessário que algumas Stores possam ser acessadas em qualquer momento, desta forma é utilizado o GetIt possibilitando que o estado armazenado em uma Store seja registrado como uma instância única. Também existem Stores que precisam apenas ser usadas em certos componentes de uma tela. Nesta situação é utilizado o Provider que possibilita uma Store ser descartada após não ser mais necessária e só ser acessada dentro de um contexto específico.

3.2.2 Desenvolvimento

O processo de autenticação do aplicativo é realizado através do Parse Server, que após informar as credenciais da aplicação criada no Back4App, permite que usuários possam criar contas, se autenticar e armazenar dados em um banco de dados. O Quadro 7 é apresenta o método responsável por inicializar a conexão do aplicativo com o servidor da aplicação no Back4App utilizando o Parse Server. Na linha 28 é realizada a chamada do método para inicializar a conexão. O método `initializeParse` declarado na linha 34 solicita três parâmetros que referenciam a aplicação criada no Back4App o `appId`, `serverUrl` e `clientKey`.

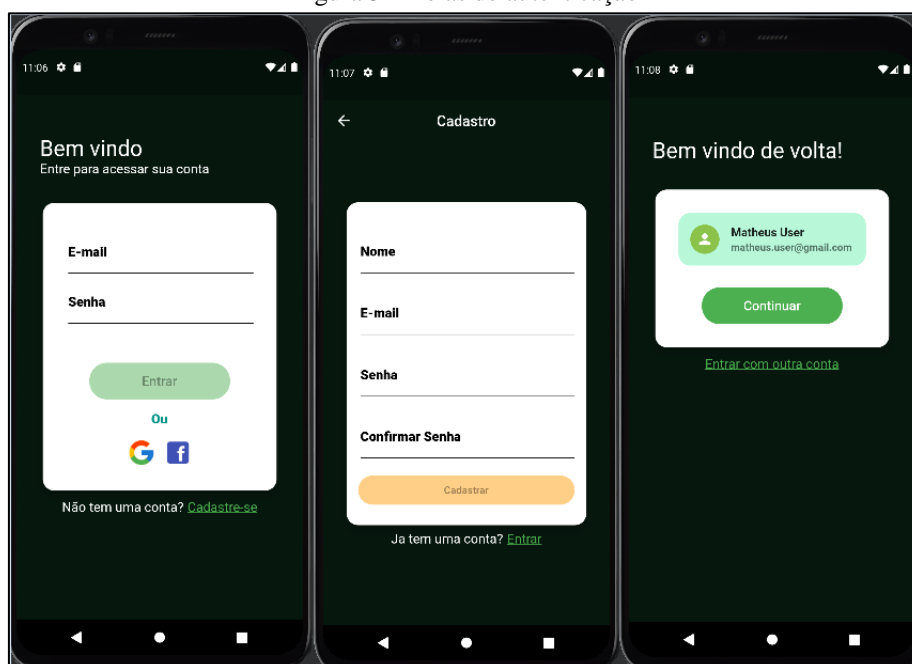
Quadro 7 – Método de inicialização do Parse Server

```
26 ▶ void main() async {  
27   WidgetsFlutterBinding.ensureInitialized();  
28   await initializeParse();  
29   await initializeHive();  
30   setupLocators();  
31   runApp(const MyApp());  
32 }  
33  
34 Future<void> initializeParse() async {  
35   await Parse().initialize(  
36     appId,  
37     serverUrl,  
38     clientKey: clientKey,  
39     autoSendSessionId: true,  
40     debug: true,  
41   );  
42 }
```

Fonte: elaborado pelo autor.

Ao abrir o aplicativo pela primeira vez a tela inicial apresentada será a tela de login, que permite ao usuário, caso já possua uma conta cadastrada, realizar a autenticação clicando no botão `Entrar`, conforme apresentada na Figura 5 (a). Se o usuário não possuir uma conta, poderá clicar no botão `Cadastre-se` e preencher os campos solicitados para criar uma conta. Os campos são apresentados conforme a Figura 5 (b). O aplicativo foi estruturado para exibir o último usuário autenticado. Este recurso poderá ser usado em momentos que não é possível acessar a internet para realizar uma nova autenticação, possibilitando que o usuário não seja interrompido de utilizar as funcionalidades do aplicativo, esta funcionalidade é apresentada conforme a Figura 5 (c).

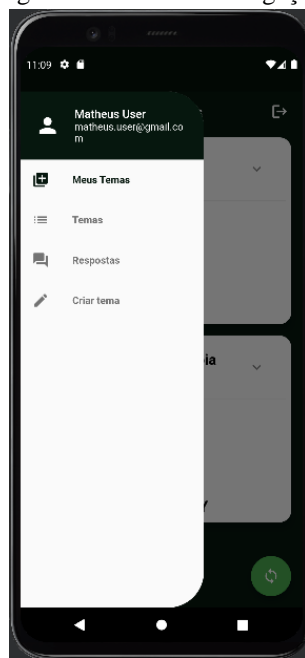
Figura 5 – Telas de autenticação



Fonte: elaborado pelo autor.

A tela de autenticação, assim como todas as demais telas, é controlada por uma *Store* do MobX. A *Store* desta tela é responsável por validar os campos que foram inseridos, efetuar a chamada no Back4App e validar a autenticação do usuário. Após autenticado o usuário poderá acessar a barra de navegação do aplicativo no canto superior esquerdo e escolher a tela que deseja acessar, conforme apresentado na Figura 6. A tela de login também tem a responsabilidade de realizar o registro das *Stores* de telas que o usuário poderá acessar a qualquer momento na utilização do aplicativo. No Quadro 8 é possível verificar o registro das *Stores* no GetIt. Na linha 68 é registrada a *Store ThemeStore* atrelada a tela Meus Temas, que é responsável pelo gerenciamento dos temas criados pelo usuário. Na linha 71 é registrada a *Store SyncedThemeStore* atrelada a tela Temas que possibilita a sincronização e realização das atividades de um tema. Na linha 74 é registrada a *Store ResponsesThemeStore* atrelada a tela Respostas, que é responsável por buscar as respostas dos temas criados pelo usuário enviadas pelos clubistas.

Figura 6 – Menu de navegação



Fonte: elaborado pelo autor.

Quadro 8 – Método de autenticação e registro de Stores

```

61      final user = await UserRepository().loginWithEmail(email!, password!);
62
63      final userMapped = User.fromParse(user);
64      GetIt.I<UserManagerStore>().setUser(userMapped);
65
66      _verifyRegistrations();
67
68      GetIt.I.registerSingleton(ThemeStore());
69      await GetIt.I<ThemeStore>().initThemesBox(userMapped.id!);
70
71      GetIt.I.registerSingleton(SyncedThemesStore());
72      await GetIt.I<SyncedThemesStore>().initThemesBox(userMapped.id!);
73
74      GetIt.I.registerSingleton(ResponsesThemeStore());
75      await GetIt.I<ResponsesThemeStore>().initThemesBox(userMapped.id!);

```

Fonte: elaborado pelo autor.

Ao acessar a tela Criar tema, será solicitado que o usuário informe um título e uma descrição para o tema. Após confirmar será direcionado para a tela Adicione objetivos, a qual permitirá a criação dos objetivos e atividades do tema. Como a criação de um tema está dentro de um contexto específico, ao direcionar para a tela de objetivos é utilizado o Provider para a criação de uma *Store* que poderá ser acessada dentro da árvore de componentes na inclusão dos objetivos e atividades do tema. No Quadro 9 na linha 75 é definido qual a *Store* será criada e na linha 76 é indicado para qual tela o usuário será direcionado.

Quadro 9 – Criação de Store através do Provider

```

71 Navigator.push(
72   context,
73   MaterialPageRoute(
74     builder: (_) => Provider(
75       create: (_) => CreateObjectiveStore(),
76       child: CreateObjetivoScreen(
77         createThemeStore: createThemeStore,

```

Fonte: elaborado pelo autor

A adição de atividades nos objetivos foi estruturada para dar mais liberdade ao usuário e evitar informações redundantes. A atividade é composta pela definição dos recursos e as perguntas que serão realizadas aos Clubistas. Os recursos podem ser fotos, vídeos, áudios ou desenhos. As perguntas podem ser adicionadas e customizadas livremente, é apenas necessário que seja definido o título da pergunta e o tipo da resposta esperada podendo ser um texto, um número inteiro, um número decimal, uma hora ou uma data. Na figura 7 é apresentado um exemplo de criação de uma atividade.

Figura 7 – Criação de uma atividade



Fonte: elaborado pelo autor.

Ao finalizar o cadastro do tema, o usuário poderá clicar no botão **Salvar** no canto superior direito e será direcionado para a tela **Meus Temas**. Ao realizar o processo é necessário que seja acessada a *Store* responsável por controlar os temas do usuário para que o tema recém-criado seja armazenado na memória interna do dispositivo. No Quadro 10 na linha 90 é demonstrado como é possível acessar uma *Store* que foi registrada como global utilizando o *GetIt*.

Quadro 10 – Acessando Store global

```

86 void saveTheme() {
87   themeContentError = null;
88
89   if (isThemeValid()) {
90     final ThemeStore themeStore = GetIt.I<ThemeStore>();
91
92     if (index != null) {
93       themeStore.update(index!, theme!);
94     } else {
95       theme!.creator = GetIt
96         .I<UserManagerStore>()
97         .user;
98       themeStore.add(theme!);

```

Fonte: elaborado pelo autor.

A tela `Meus Temas` lista todos os temas que o usuário criou. Com a utilização da biblioteca `Hive`, o aplicativo irá buscar utilizando um arquivo vinculado ao usuário os temas que estão armazenados no dispositivo. No Quadro 11 é apresentado o método que efetua a busca dos temas, na linha 26 é solicitada a permissão de acesso aos arquivos do dispositivo e na linha 29 é aberto o arquivo que armazena os temas. Caso o usuário esteja acessando de outro dispositivo e queira visualizar os temas que foram sincronizados anteriormente, é possível realizar a sincronização clicando no botão `Buscar temas sincronizados`. Os temas encontrados serão salvos no arquivo vinculado ao usuário.

Quadro 11 – Busca dos temas na memória do dispositivo

```
25 Future<void> initThemesBox(String userId) async {  
26     await Permission.storage.request().isGranted;  
27     final Directory result = await getApplicationSupportDirectory();  
28     Hive.init(result.path);  
29     myThemesBox = await Hive.openBox<ThemeExplora>('themes_explora.$userId');  
30 }
```

Fonte: elaborado pelo autor.

O tema após sua criação possuirá três status que determinam quais funções podem ser executadas. Um tema não sincronizado pode ser editado, excluído ou sincronizado. Ao realizar a sincronização de um tema é feita a persistência em uma tabela no banco de dados do `Back4App`. No Quadro 12 é apresentado o método responsável por esse processo, em que na linha 13 é criado um objeto do tipo `ParseObject`, responsável por armazenar toda a estrutura do tema, e na linha 26 é chamado o método `save` que efetua a persistência no `Back4App`. Este processo gera um código único vinculado ao tema. A partir do código gerado o usuário poderá compartilhar com um clubista permitindo que sincronize o tema para realizar atividades propostas. Um tema sincronizado pode ser apenas visualizado sem a possibilidade de edição. Também pode ser gerado um QR Code facilitando o compartilhamento do tema sem a necessidade de digitar o código do tema. Por fim, o tema pode ser finalizado, neste status o tema não poderá mais ser compartilhado e suas respostas estarão disponíveis para análise.

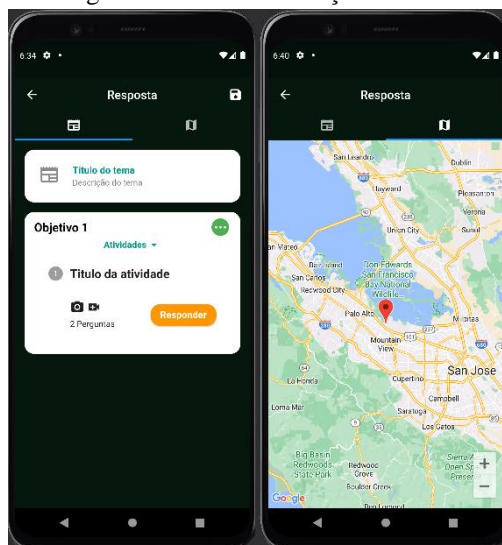
Quadro 12 – Método de persistência do tema no `Back4App`

```
13 final themeObject = ParseObject(keyThemeTable);  
14 themeObject.objectId = theme.id;  
15  
16 final parseAcl = ParseACL(owner: parseUser);  
17 parseAcl.setPublicReadAccess(allowed: true);  
18 parseAcl.setPublicWriteAccess(allowed: false);  
19 themeObject.setACL(parseAcl);  
20  
21 themeObject.set<String>(keyThemeTitle, theme.title);  
22 themeObject.set<String>(keyThemeDescription, theme.description);  
23 themeObject.set<ParseUser>(keyThemeCreator, parseUser);  
24 themeObject.set<String>(keyThemeStatus, theme.status.value);  
25  
26 var response = await themeObject.save();
```

Fonte: elaborado pelo autor.

A tela `Temas` permite executar as atividades propostas em um tema através da utilização do código único gerado ao sincronizar um tema. Ao buscar o tema, ele será listado e ficará pendente o envio das respostas. Ao iniciar o tema, o usuário será direcionado para uma tela que possui duas abas, conforme apresentado na Figura 8. A primeira aba será a tela de `Resposta` onde serão listados os objetivos e suas respectivas atividades. A segunda aba será a tela do `Mapa`, que mostra a posição do usuário ao responder cada atividade. Para responder uma atividade é necessário apenas escolher a atividade e clicar no botão `Responder`. Quando uma atividade é respondida, é realizada a coleta da geolocalização do usuário e adicionado um marcador com as coordenadas no mapa. Ao responder todas as atividades, o usuário poderá salvar as respostas clicando no botão `Salvar` e ao retornar para tela `Temas` poderá realizar a sincronização enviando as respostas para o `Back4App`.

Figura 8 – Tela de execução do tema



Fonte: elaborado pelo autor.

A tela Respostas, irá buscar todos os temas que foram finalizados pelo usuário, independentemente de haver ou não o envio de respostas. A premissa é controlar até quando um tema será válido e poderá ser executado por um clubista. Ao realizar a busca serão listados os temas finalizados permitindo que o usuário visualize todas as respostas enviadas pelos clubistas e identifique no mapa sua posição geográfica, podendo assim entender como cada clubista realizou as atividades propostas. Na Figura 9 é apresentado um tema que possui o envio da resposta para a atividade.

Figura 9 – Visualização das respostas do tema



Fonte: elaborado pelo autor.

4 RESULTADOS

Na seção a seguir são apresentados os testes de funcionalidade e compatibilidade, os testes com os usuários, a comparação com a versão anterior e a comparação com os trabalhos correlatos.

4.1 TESTES DE FUNCIONALIDADE E COMPATIBILIDADE

Para assegurar a funcionalidade e compatibilidade do aplicativo, os testes foram realizados utilizando o sistema operacional Android. O Flutter possibilita que os aplicativos desenvolvidos sejam multiplataforma, porém pela falta de

equipamentos não foram realizados testes no sistema operacional iOS. Os testes foram efetuados de duas formas. A primeira forma foi através do ambiente de programação Android Studio emulando o sistema Android. A segunda forma foi gerar uma versão instalável do aplicativo e instalar em três dispositivos: Motorola G6, Motorola G8 e Redmi 8A.

Ao iniciar o aplicativo após sua instalação, foi realizado o processo de cadastro do usuário e a autenticação ocorreu corretamente. Após autenticado, as telas de navegação foram apresentadas com sucesso. Foi acessada a tela *Criar tema* e a criação do tema ocorreu da forma esperada, foi possível criar o tema, adicionar objetivos, adicionar atividades e por fim salvar o tema. Após criar o tema ele foi listado corretamente na tela *Meus Temas* e apresentando as funções de gerenciamento do tema. Ao realizar a sincronização do tema e acessar a tela *Temas* foi possível buscar o tema com sucesso. Foi testada a execução de um tema respondendo as atividades e usando os recursos de foto e vídeo. Ao realizar a resposta das atividades a localização foi registrada no mapa conforme esperado. Após salvar as respostas e voltar para a tela *Temas* foi possível efetuar a sincronização das respostas com sucesso. Voltando para a tela de *Meus Temas* foi efetuada a finalização corretamente. Ao acessar a tela de *Respostas* e buscar os temas finalizados, foi listado o tema conforme esperado. Acessando o tema, as respostas nas atividades foram exibidas apresentando o nome do usuário que as realizou corretamente. Os testes de compatibilidade ocorreram com sucesso, com exceção do dispositivo Redmi 8A, em que não foi possível utilizar o recurso de câmera corretamente. Ao acessar uma atividade e tirar uma foto era perdido a referência da tela, impedindo que a foto fosse armazenada.

4.2 TESTES COM USUÁRIOS

Com o objetivo de verificar a usabilidade, navegabilidade e funcionalidade do aplicativo, foi elaborado um roteiro para identificar o perfil do usuário, um passo a passo apresentando o fluxo de utilização na visão do Mediador e na visão do Clubista e perguntas para avaliação do aplicativo. Com o roteiro elaborado, foi criado um questionário na plataforma Google Forms e enviado a dezenove usuários. O Quadro 13 apresenta as perguntas para identificar o perfil do usuário. Os usuários possuem em média 21 anos. Todos utilizam os dispositivos móveis com frequência. Apenas três usuários já participaram de um Clube de Ciências. Apenas cinco usuários já utilizaram um dispositivo móvel em uma saída a campo para coleta de dados.

Quadro 13 – Perfil do usuário

Ordem	Etapas
1	Idade
2	Você utiliza dispositivos móveis com qual frequência?
3	Você conhece ou já participou de algum clube de ciências?
4	Já utilizou um dispositivo móvel em uma saída a campo para coletar dados?

Fonte: elaborado pelo autor.

O Quadro 14 é apresenta o roteiro da visão do Mediador e no Quadro 15 a visão do Clubista. Ao realizar os passos apenas quatro usuários apresentaram problemas na sincronização do tema na visão do Mediador, o que os impossibilitaram de prosseguir com os passos seguintes, os demais usuários conseguiram concluir com sucesso. Na visão do Clubista, sete usuários relataram problemas na sincronização das respostas, alegando demora na realização do processo.

Quadro 14 – Roteiro do Mediador

Ordem	Etapas
1	Abra o aplicativo ExploraHabitat, crie um usuário usando o mesmo e-mail anteriormente informado e tente seguir os passos descritos abaixo.
2	Acessar a tela "Criar Tema" clicando no ícone de navegação no canto superior esquerdo. Definir um título e uma descrição para o tema e confirmar.
3	Para o tema criado anteriormente, adicionar objetivos e para cada objetivo adicionar ao menos uma atividade. Na atividade adicionar o título, os recursos que serão utilizados e as perguntas que serão respondidas definindo o tipo da resposta esperada.
4	Verificar se é possível modificar o tema, os objetivos e as atividades. Também é possível reordenar as atividades segurando a atividade e movendo para a posição desejada. Após realizar as alterações, salvar o tema clicando no ícone superior direito.
5	Acessar a tela "Meus Temas" e localizar o tema criado. Expandir a flecha e verificar as opções apresentadas. Será possível editar, excluir, sincronizar ou copiar o tema.
6	Realizar a sincronização de um tema, verificar se o código de compartilhamento do tema está sendo exibido na parte inferior do card.
7	Após a sincronização, expandir a flecha e verificar as opções apresentadas. Será possível visualizar, copiar, exibir o QR Code, finalizar ou fechar o tema.

Fonte: elaborado pelo autor.

Quadro 15 – Roteiro do Clubista

Ordem	Etapas
1	Abra o aplicativo ExploraHabitat, crie um usuário ou utilize o mesmo usuário do guia de Mediador e tente seguir os passos descritos abaixo.
2	Acessar a tela "Temas" e clicar no botão na parte inferior direita da tela. Inserir o código ou ler o QR Code do tema criado anteriormente. Aguardar até que o tema seja sincronizado.
3	Após a sincronização, expandir a flecha e verificar as opções apresentadas. Será possível executar e fechar o tema.
4	Escolher um tema e clicar na opção de executar. Será direcionado para a tela de "Resposta", na qual possui duas abas, sendo a primeira a listagem de objetivos e suas atividades e a segunda o mapa da localização atual.
5	Responder todas as atividades de todos os objetivos definidos. Acessar a segunda aba do mapa e verificar se foi adicionado o marcador de localização para as atividades respondidas.
6	Após responder, salvar as respostas clicando no ícone no canto superior direito. Na tela "Temas", realize a sincronização das respostas. As respostas poderão ser visualizadas pelo Mediador, na tela de "Respostas" assim que o tema for finalizado.

Fonte: elaborado pelo autor.

O Quadro 16 apresenta as perguntas realizadas para a avaliação do aplicativo. As perguntas de classificação foram aplicadas aos usuários dentro de uma escala de 1 a 5, em que 1 representa “Péssimo” e 5 representa “Excelente”. A partir dessa escala é possível definir que classificações entre 1 e 2 significam que o item não foi atendido corretamente e valores entre 3 e 5 determinam que o item foi atendido. No Apêndice B são apresentados os resultados da pesquisa por pergunta.

Quadro 16 – Perguntas de avaliação do aplicativo

Ordem	Pergunta
1	Você conseguiu concluir os objetivos dessa pesquisa com facilidade?
2	Quantas tarefas você concluiu sem nenhum auxílio externo?
3	Como você classifica a usabilidade do ExploraHabitat em Geral?
4	Como você classifica a navegabilidade das telas no ExploraHabitat?
5	Você acha que as interfaces do ExploraHabitat são amigáveis?
6	Você acha que o ExploraHabitat cumpriu seu objetivo de auxiliar em atividades de saídas a campo?
7	Você recomendaria o ExploraHabitat para alguém que deseja realizar pesquisas em saídas a campo?
8	Você possui algum comentário geral, crítica ou sugestão?

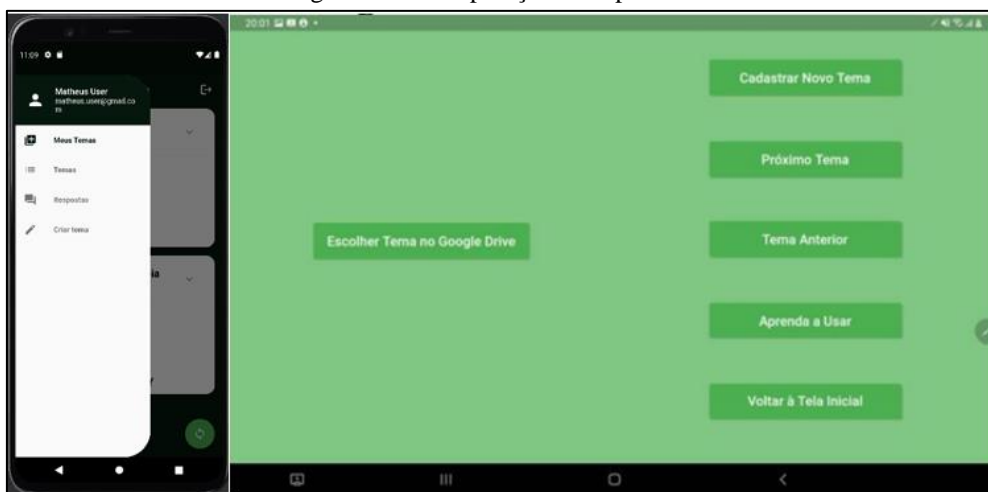
Fonte: elaborado pelo autor.

Em geral todos conseguiram concluir os objetivos da pesquisa com facilidade, sendo que 84% dos usuários classificaram o item como atendido. Um ponto notado foi a dificuldade em realizar as tarefas sem auxílio externo, pois apenas três usuários conseguiram sem nenhum auxílio. Em relação a usabilidade 89% dos usuários classificaram como atendido. A navegabilidade das telas do aplicativo foram classificadas como atendida por 84% dos usuários. As interfaces do ExploraHabitat obtiveram uma boa classificação em geral, pois todos os usuários classificaram este item como atendido. No cumprimento do objetivo de auxiliar atividades de saídas a campo, 89% dos usuários classificaram como atendido. Por fim, 89% dos usuários recomendariam o aplicativo para alguém que desejasse realizar pesquisas em saídas a campo.

4.3 COMPARAÇÃO COM A VERSÃO ANTERIOR

O aplicativo desenvolvido define um fluxo de utilização permitindo uma maior flexibilidade no gerenciamento dos temas e na execução das atividades. A criação dos componentes da interface com o framework Material Design do Flutter permitiu uma melhor padronização da estrutura do aplicativo. A estrutura é definida a partir do menu de navegação, no qual o usuário poderá criar temas, acessar os temas sincronizados e visualizar as respostas recebidas, conforme pode ser verificado na Figura 10 (a). No aplicativo desenvolvido por Korbes (2021), o fluxo se inicia a partir da escolha de perfil. Após a escolha o usuário é direcionado para uma tela com botões apresentando todas as funcionalidades atreladas aquele perfil não havendo um fluxo de navegação direcionando o usuário, conforme pode ser verificado na Figura 10 (b).

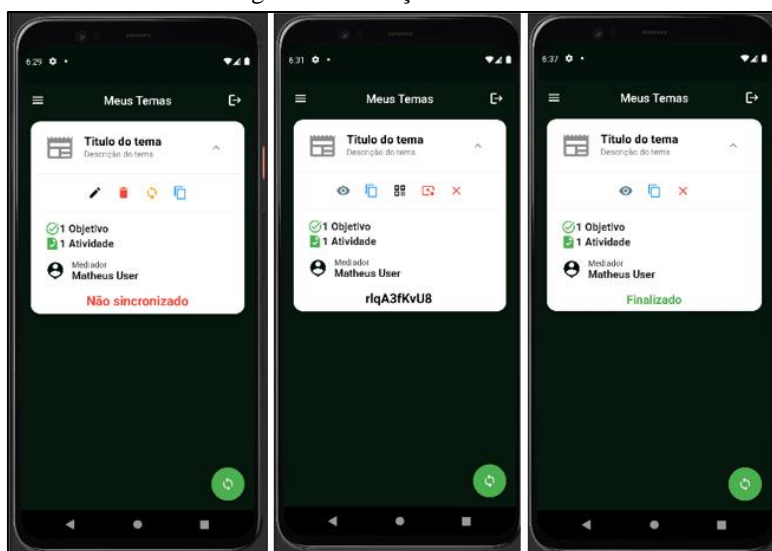
Figura 10 – Comparação dos aplicativos



Fonte: elaborado pelo autor.

O aplicativo desenvolvido possibilita que o usuário gerencie os seus temas com maior controle e liberdade. Os temas possuem funcionalidades que são exibidas de acordo com o seu status. Por exemplo, temas não sincronizados possuem quatro funções: editar, excluir, sincronizar e copiar, conforme pode ser verificado na Figura 11 (a). A função de editar permite que o usuário altere informações definidas no momento da criação do tema como título, descrição, objetivos e atividades. A função de excluir remove o tema permanentemente. A função sincronizar persiste o tema no banco de dados. A função copiar permite duplicar o tema, possibilitando que o usuário possa utilizar as mesmas informações para outras atividades. Temas sincronizados estão aptos para serem respondidos, sendo assim, apresentam novas funções que são: visualizar, gerar QR Code e finalizar, conforme pode ser verificado na Figura 11(b). A função de visualizar permite verificar as informações de um tema sem possibilidade de edição. A função gerar QR Code permite compartilhar o código do tema de maneira mais prática e fácil. A função finalizar encerra o tema e impede o envio de novas respostas.

Figura 11 – Funções dos temas



Fonte: elaborado pelo autor.

A criação de atividades no aplicativo desenvolvido foi planejada para possibilitar uma customização completa, definindo exatamente o que é necessário para o que está sendo proposto. As atividades são compostas em duas partes: recursos e perguntas, conforme pode ser verificado na Figura 12 (a). Os recursos são ferramentas dos dispositivos móveis que podem ser utilizados em uma atividade à campo, como gravar um vídeo ou tirar uma foto. As perguntas são as informações que se deseja obter sobre a atividade em questão. Cada pergunta possui um texto explicativo e um tipo esperado para melhor definição da resposta. No aplicativo de Korbés (2021) todas as atividades são pré-definidas e não permitem ser modificadas para atender algum cenário específico, conforme pode ser verificado na Figura 12 (b).

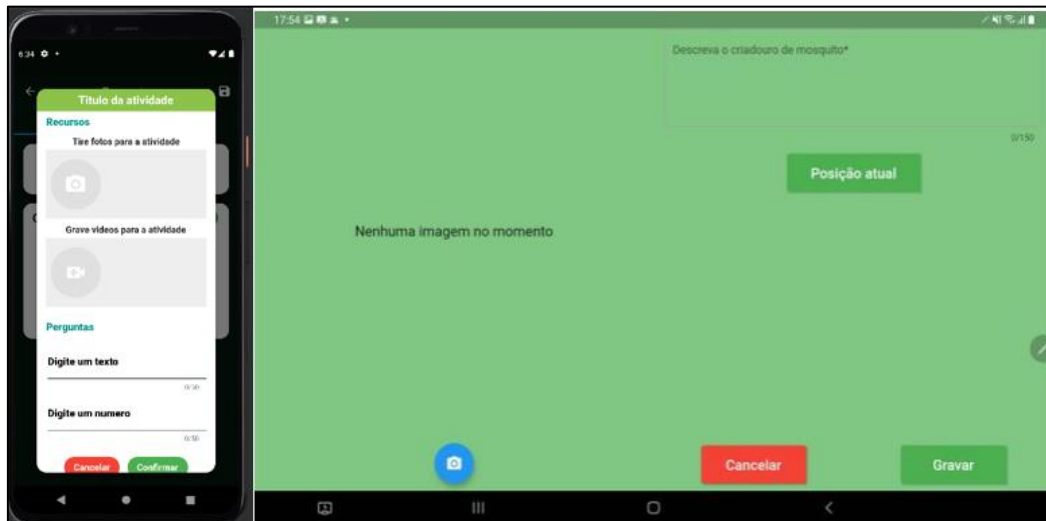
Figura 12 – Comparação dos aplicativos



Fonte: elaborado pelo autor.

A execução das atividades no aplicativo desenvolvido foi estruturada como uma listagem, para permitir que o usuário realize todo o processo em uma sequência direta. Primeiro são apresentados os recursos escolhidos e após as perguntas definidas para atividade, conforme pode ser verificado na Figura 13 (a). No aplicativo de Korbes (2021) a atividade é apresentada de forma dispersa na tela, conforme pode ser verificado na Figura 13 (b). O aplicativo atual também permite que as respostas possam ser visualizadas no próprio aplicativo, permitindo que o usuário possa identificar o clubista e analisar as atividades executadas, conforme foi verificado na Figura 9. Uma limitação do aplicativo de Korbes (2021) é que as respostas só podem ser visualizadas a partir do Google Drive, sendo necessário que o usuário saia do aplicativo.

Figura 13 – Comparação dos aplicativos



Fonte: elaborado pelo autor.

4.4 COMPARAÇÃO COM OS TRABALHOS CORRELATOS

O Quadro 17 apresenta uma comparação do trabalho desenvolvido com os trabalhos correlatos. Conforme pode ser observado, apenas o trabalho de Rosa (2015) não possui propósito voltado para educação. O trabalho de Bet *et al.* (2004) e o ExploraHabitat tem como principal objetivo estimular o aprendizado em clubes de ciências. O trabalho de Mendonça *et al.* (2018) possui foco no aprendizado em botânica.

Os trabalhos apresentados foram disponibilizados em diferentes plataformas. O trabalho de Bet *et al.* (2004) foi desenvolvido para Web. O trabalho de Mendonça *et al.* (2018) e o ExploraHabitat foram desenvolvidos para Android e iOS, porém no ExploraHabitat o sistema operacional iOS não foi validado. O trabalho de Rosa (2015) foi desenvolvido para Android, Windows Phone e iOS.

Todos os três trabalhos possuem uma forma de compartilhar informações entre os usuários. Em relação a produção de conteúdo para avaliar os usuários, apenas o trabalho de Rosa (2015) não se encaixa devido ter como objetivo

a geração de registros de incidentes ambientais. Já em saídas a campo apenas o trabalho de Bet *et al.* não é necessário, pois tem como objetivo a simulação de um clube de ciências no meio virtual. O trabalho de Mendonça *et al.* (2018) utiliza a geolocalização do usuário para avaliar se o usuário identificou a planta correta. O trabalho de Rosa (2015) utiliza a geolocalização em conjunto com o mapa para registrar a posição do incidente e exibir para os usuários próximos ao incidente. Já no ExploraHabitat a geolocalização e o mapa são utilizados para registrar a posição do clubista ao responder uma atividade e exibir para o Mediador.

Quadro 17 – Comparativo trabalhos correlatos

	Bet <i>et al.</i> (2004)	Mendonça <i>et al.</i> (2018)	Rosa (2015)	ExploraHabitat
Propósito educacional	Sim	Sim	Não	Sim
Saídas a campo	Não	Sim	Sim	Sim
Produção de conteúdo avaliativo	Sim	Sim	Não	Sim
Compartilhar informações entre os usuários	Sim	Sim	Sim	Sim
Abordagem	Aprendizado em clube de ciências	Aprendizado em botânica	Incidentes ambientais	Aprendizado em clube de ciências
Plataforma	Web	Android e iOS	Android, Windows Phone e iOS	Android e iOS, mas validado apenas no Android
Utilização da geolocalização do usuário	Não	Sim	Sim	Sim
Utilização de mapa	Não	Não	Sim	Sim

Fonte: elaborado pelo autor.

Desta forma, o trabalho ExploraHabitat é relevante por ser uma alternativa para fomentar o aprendizado em clubes de ciências. O aplicativo estimula a realização de atividades com maior autonomia e interação com os membros do clube, proporcionando uma melhor visualização das atividades, tanto pelo Clubista quanto pelo Mediador, através do acompanhamento pelo mapa.

5 CONCLUSÕES

Em consideração aos objetivos deste trabalho, o aplicativo mostrou atender seu propósito em proporcionar uma melhor experiência de uso. A partir da análise dos resultados da pesquisa os usuários avaliaram o aplicativo com uma boa usabilidade e navegabilidade entre as telas. Os usuários também conseguiram realizar a criação e execução dos temas podendo utilizar os recursos dos dispositivos móveis e acompanhar seu progresso através de um mapa. A maioria dos usuários tiveram dificuldades na compreensão do fluxo de funcionamento do aplicativo e tiveram dúvidas em como realizar alguns passos, assim entende-se que é necessário melhorar a apresentação do fluxo entre as telas e a criação de um guia de utilização. Alguns usuários também apontaram lentidão e falha na sincronização das respostas.

A utilização do MobX com a estrutura de ações, reações e observáveis facilitou o controle do comportamento dos componentes em tela através do gerenciamento dos estados. Por ser baseado em geração de código é necessário pouco código para as implementações. O Provider e o GetIt como injetores de dependência possibilitaram uma melhor organização do código e permitiu a definição de um melhor controle do que pode ser acessado entre os componentes. Com a utilização de um gerenciador de estados e injetores de dependência é possível adicionar novas funcionalidades facilmente. Para a modernização da interface o framework Material Design possui componentes visualmente agradáveis e permitiu a estilização para padronização com a temática do aplicativo. O Android Studio se mostrou eficaz devido a sua compatibilidade com o *framework* Flutter e a possibilidade de emular um sistema operacional do Android. A utilização do Back4App com o Parse Server como uma plataforma BaaS, possibilitou criar facilmente um *backend* que suprisse as necessidades do aplicativo com gerenciamento do banco de dados, armazenamento de arquivos e o gerenciamento de usuários. A utilização da biblioteca Hive se mostrou eficaz em salvar dados na memória interna do aplicativo, permitindo que o usuário possa utilizar sem acesso à internet.

A limitação do trabalho é em relação a plataforma Back4App, pois possui uma limitação mensal de requisições, armazenamento de dados e arquivos no uso do plano gratuito. As limitações da plataforma também impactaram na demora e falha da sincronização de algumas respostas, pois caso o tema possuísse um arquivo com tamanho superior a 25 megabytes não seria salvo, impedindo que os usuários prosseguissem com a utilização do aplicativo. Assim para que não haja impactos, será necessário a aquisição de um plano pago ou buscar outras plataformas para a persistência dos temas e armazenamento dos arquivos. Outra limitação é devido ao aplicativo funcionar sem acesso à internet, os dados são armazenados constantemente na memória interna do dispositivo, sendo necessário que o usuário sempre esteja sincronizando os dados com o Back4App. As sugestões de possíveis extensões são:

- validar a utilização no sistema operacional iOS;

- b) disponibilizar nas lojas de aplicativos para maior acessibilidade;
- c) adicionar mais recursos do dispositivo móvel nas atividades do tema;
- d) avaliar e otimizar a sincronização dos temas e das respostas;
- e) avaliar a possibilidade de salvar arquivos em outra plataforma;
- f) melhorar o mapa para exibir mais detalhes da atividade realizada;
- g) melhorar o mapa para que ao clicar em um marcador possua a opção de navegar até a atividade e vice-versa;
- h) permitir alterar a senha da conta do usuário;
- i) permitir que o mediador possa avaliar as atividades realizadas;
- j) permitir a criação de *templates* prontos para as atividades;
- k) criar uma interface web para a criação e compartilhamento dos *templates*, fomentando a interação na comunidade dos clubes de ciências;
- l) permitir o gerenciamento dos dados do perfil do usuário;
- m) permitir autenticação do usuário através da conta Google e Facebook;
- n) explorar os conceitos da aprendizagem ubíqua no aplicativo.

REFERÊNCIAS

BACK4APP. **O que é o Backend como serviço?**. 2022. Disponível em: <https://blog.back4app.com/pt/o-que-e-o-backend-como-servico/#Back4App>. Acesso em: 22 jun. 2022.

BET, Sabrina et al. Um ambiente colaborativo para auxílio ao aprendizado: Clube Virtual de Ciências. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO. 2004, Lages. **Anais do Simpósio Brasileiro de Informática na Educação**. Lages: Sbie, 2004. p. 84-87. Disponível em: <http://ojs.sector3.com.br/index.php/sbie/article/view/383>. Acesso em: 20 jun. 2022.

CÓRDOBA. **Club Escolar de Ciencias y Tecnologías. Ministerio de Educación**; Ministerio de Ciencia y Tecnología, 2012. Disponível em: <https://www.igualdadycalidadcba.gov.ar/SIPEC-CBA/publicaciones/documentos/Club%20de%20ciencias%2025-7-12.pdf>. Acesso em: 20 set. 2021.

FLUTTER. **The official package repository for Dart and Flutter apps**. 2022a. Disponível em: <https://api.flutter.dev/flutter/widgets/State-class.html>. Acesso em: 22 jun. 2022.

FLUTTER. **The official package repository for Dart and Flutter apps**. 2022b. Disponível em: <https://docs.flutter.dev/development/data-and-backend/state-mgmt/declarative>. Acesso em: 22 jun. 2022.

FLUTTER. **The official package repository for Dart and Flutter apps**. 2022c. Disponível em: <https://docs.flutter.dev/development/data-and-backend/state-mgmt/ephemeral-vs-app>. Acesso em: 22 jun. 2022.

FLUTTER. **The official package repository for Dart and Flutter apps**. 2022d. Disponível em: <https://docs.flutter.dev/development/data-and-backend/state-mgmt/options>. Acesso em: 22 jun. 2022.

FLUTTER. **The official package repository for Dart and Flutter apps**. 2022e. Disponível em: <https://pub.dev/packages/mobx>. Acesso em: 22 jun. 2022.

FLUTTER. **The official package repository for Dart and Flutter apps**. 2022f. Disponível em: https://pub.dev/packages/get_it. Acesso em: 22 jun. 2022.

FLUTTER. **The official package repository for Dart and Flutter apps**. 2022g. Disponível em: <https://pub.dev/packages/provider>. Acesso em: 22 jun. 2022.

FREITAS, Thais C. de Oliveira, SANTOS, Carlos A. M. dos. **Clubes de ciências na Escola: um guia para professores, gestores e pesquisadores**. Curitiba: Associação Brasileira de Editores Científicos, 2021. 166p.

KORBES, Gustavo H. **ExploraHabitat**: Um aplicativo para apoiar as saídas a campo em Clubes de Ciências. 2021. 69 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Centro de Ciências Exatas e Naturais, Universidade Regional de Blumenau. Disponível em: <http://dsc.inf.furb.br/tcc/index.php?cd=6&tcc=2080>. Acesso em: 20 set. 2021.

MENDONÇA, Karoene Dirlene et al. Ambiente de Aprendizagem Ubíqua para Auxiliar o Estudo de Botânica em Atividades de Aula de Campo. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 29. 2018, Parnaíba. **Anais do Simpósio Brasileiro de Informática na Educação**. Parnaíba: Sbie, 2018. p. 1738. Disponível em: <http://ojs.sector3.com.br/index.php/sbie/article/view/8141>. Acesso em: 20 jun. 2022.

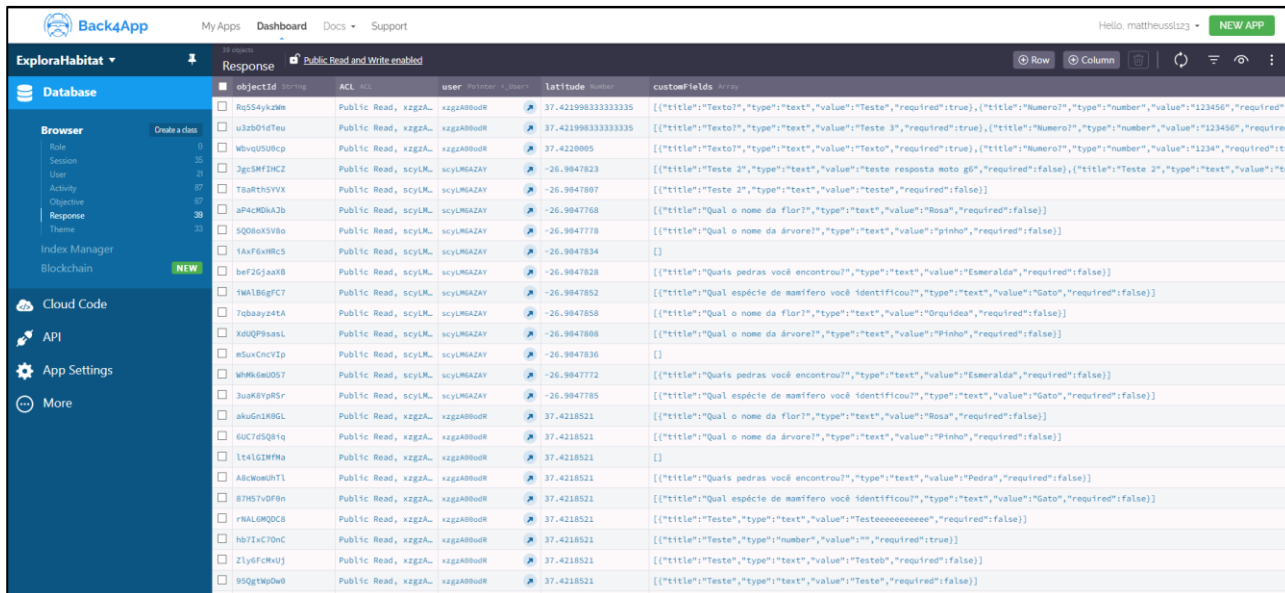
ROSA, Vagner Santos da. Ambicare: monitoramento ambiental usando dispositivos móveis. **Revista de Empreendedorismo, Inovação e Tecnologia**, Passo Fundo, v. 1, n. 2. 2015. Disponível em: <https://seer.imed.edu.br/index.php/revistas/article/view/776>. Acesso em: 20 jun. 2022.

SLEPNEV, Dmitrii. **State management approaches in Flutter**. 2020. 98 f. Bachelor's thesis (Bachelor of Engineering) – South-Eastern Finland University of Applied Sciences.

APÊNDICE A – BACK4APP

A plataforma Back4App apresenta um dashboard para a aplicação criada. Permite visualizar as tabelas criadas, logs de autenticação e dados gerais da conta. A Figura 10 apresenta o dashboard da plataforma.

Figura 14 – API Plataforma Back4App



objectId	ACL	user	latitude	longitude	customFields
RqS54yKzm	Public Read, xzgzA...	xzgzAR0u0R	37.42199833333335		[{"title": "Teste 1", "type": "text", "value": "Teste", "required": true}, {"title": "Numero 1", "type": "number", "value": "123456", "required": true}]
u3z010Teu	Public Read, xzgzA...	xzgzAR0u0R	37.42199833333335		[{"title": "Teste 2", "type": "text", "value": "Teste 2", "required": true}, {"title": "Numero 2", "type": "number", "value": "123456", "required": true}]
WbVqU5URcp	Public Read, xzgzA...	xzgzAR0u0R	37.4220005		[{"title": "Teste 3", "type": "text", "value": "Teste 3", "required": true}, {"title": "Numero 3", "type": "number", "value": "123456", "required": true}]
3gc5FIHCZ	Public Read, scyLM...	scyLMAZAY	-26.9047823		[{"title": "Teste 4", "type": "text", "value": "teste resposta moto g6", "required": false}, {"title": "Teste 5", "type": "text", "value": "teste", "required": false}]
T8a8tH9VYX	Public Read, scyLM...	scyLMAZAY	-26.9047807		[{"title": "Teste 6", "type": "text", "value": "teste", "required": false}]
aP4cMDKA3b	Public Read, scyLM...	scyLMAZAY	-26.9047768		[{"title": "Qual o nome da flor?", "type": "text", "value": "Rosa", "required": false}]
SQ08oXSV8o	Public Read, scyLM...	scyLMAZAY	-26.9047778		[{"title": "Qual o nome da árvore?", "type": "text", "value": "Pinho", "required": false}]
1A4FexHRC5	Public Read, scyLM...	scyLMAZAY	-26.9047834		[{"title": "Qual o nome da árvore?", "type": "text", "value": "Pinho", "required": false}]
beF2GjaaxB	Public Read, scyLM...	scyLMAZAY	-26.9047828		[{"title": "Quais pedras você encontrou?", "type": "text", "value": "Esmeralda", "required": false}]
1WALB6gFC7	Public Read, scyLM...	scyLMAZAY	-26.9047852		[{"title": "Qual espécie de mamífero você identificou?", "type": "text", "value": "Gato", "required": false}]
7qb3ayz4tA	Public Read, scyLM...	scyLMAZAY	-26.9047858		[{"title": "Qual o nome da flor?", "type": "text", "value": "Orquídea", "required": false}]
XdUQP9sasL	Public Read, scyLM...	scyLMAZAY	-26.9047808		[{"title": "Qual o nome da árvore?", "type": "text", "value": "Pinho", "required": false}]
mSuxCncV1p	Public Read, scyLM...	scyLMAZAY	-26.9047836		[{"title": "Qual o nome da árvore?", "type": "text", "value": "Pinho", "required": false}]
WMK6u0U57	Public Read, scyLM...	scyLMAZAY	-26.9047772		[{"title": "Quais pedras você encontrou?", "type": "text", "value": "Esmeralda", "required": false}]
3uaK8YpRSr	Public Read, scyLM...	scyLMAZAY	-26.9047785		[{"title": "Qual espécie de mamífero você identificou?", "type": "text", "value": "Gato", "required": false}]
akuGn1M9GL	Public Read, xzgzA...	xzgzAR0u0R	37.4218521		[{"title": "Qual o nome da flor?", "type": "text", "value": "Rosa", "required": false}]
6UC7d5Q81q	Public Read, xzgzA...	xzgzAR0u0R	37.4218521		[{"title": "Qual o nome da árvore?", "type": "text", "value": "Pinho", "required": false}]
1t41G2HfMa	Public Read, xzgzA...	xzgzAR0u0R	37.4218521		[{"title": "Qual o nome da árvore?", "type": "text", "value": "Pinho", "required": false}]
ABcWomUHL	Public Read, xzgzA...	xzgzAR0u0R	37.4218521		[{"title": "Quais pedras você encontrou?", "type": "text", "value": "Pedra", "required": false}]
87H57vDF0n	Public Read, xzgzA...	xzgzAR0u0R	37.4218521		[{"title": "Qual espécie de mamífero você identificou?", "type": "text", "value": "Gato", "required": false}]
rNALQMOC8	Public Read, xzgzA...	xzgzAR0u0R	37.4218521		[{"title": "Teste", "type": "text", "value": "Testeoooooooooooo", "required": false}]
hb7Ic70hC	Public Read, xzgzA...	xzgzAR0u0R	37.4218521		[{"title": "Teste", "type": "number", "value": "1", "required": true}]
Z1y6fCkuUj	Public Read, xzgzA...	xzgzAR0u0R	37.4218521		[{"title": "Teste", "type": "text", "value": "Teste", "required": false}]
95qgtw0w9	Public Read, xzgzA...	xzgzAR0u0R	37.4218521		[{"title": "Teste", "type": "text", "value": "Teste", "required": false}]

Fonte: elaborado pelo autor.

APÊNDICE B – RESPOSTAS DOS USUÁRIOS

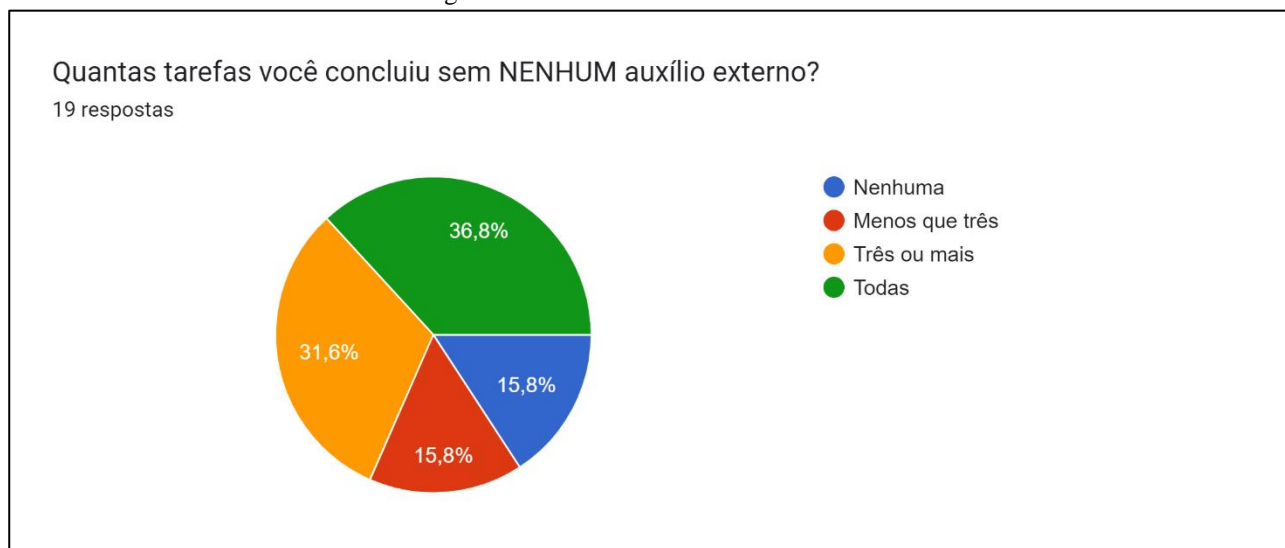
Neste apêndice são apresentados os resultados das respostas dos usuários em relação ao questionário sobre a utilização do aplicativo.

Figura 15 – Classificação sobre a facilidade da pesquisa



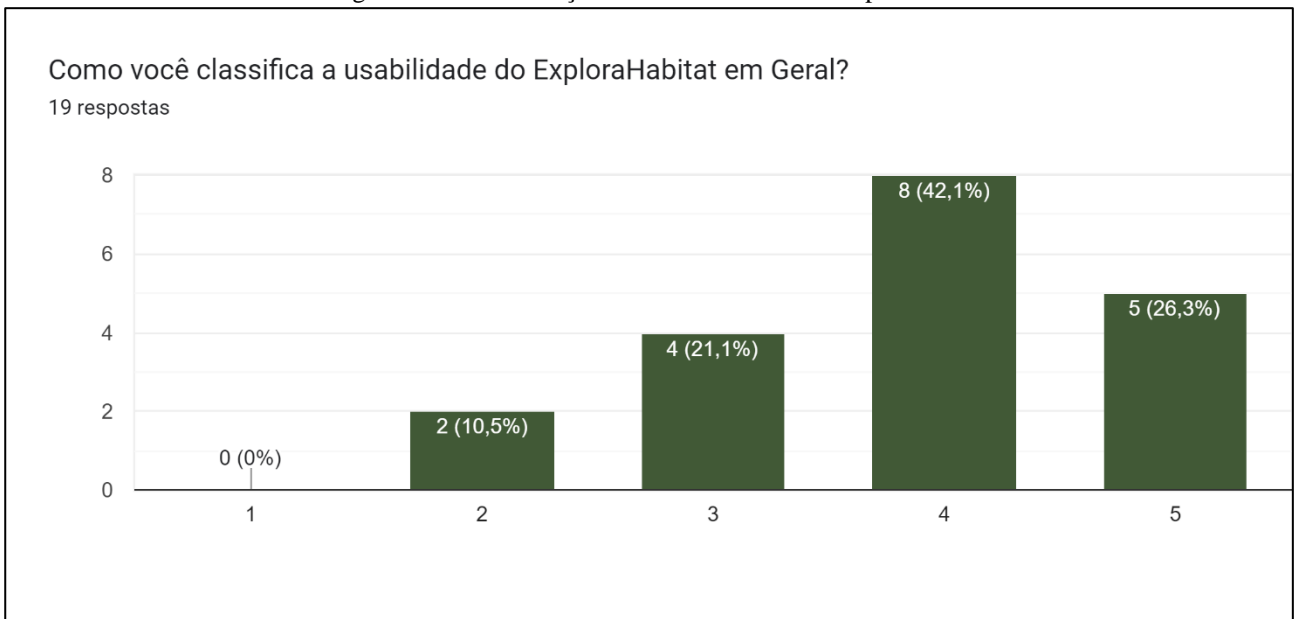
Fonte: elaborado pelo autor.

Figura 16 – Tarefas realizadas sem auxílio



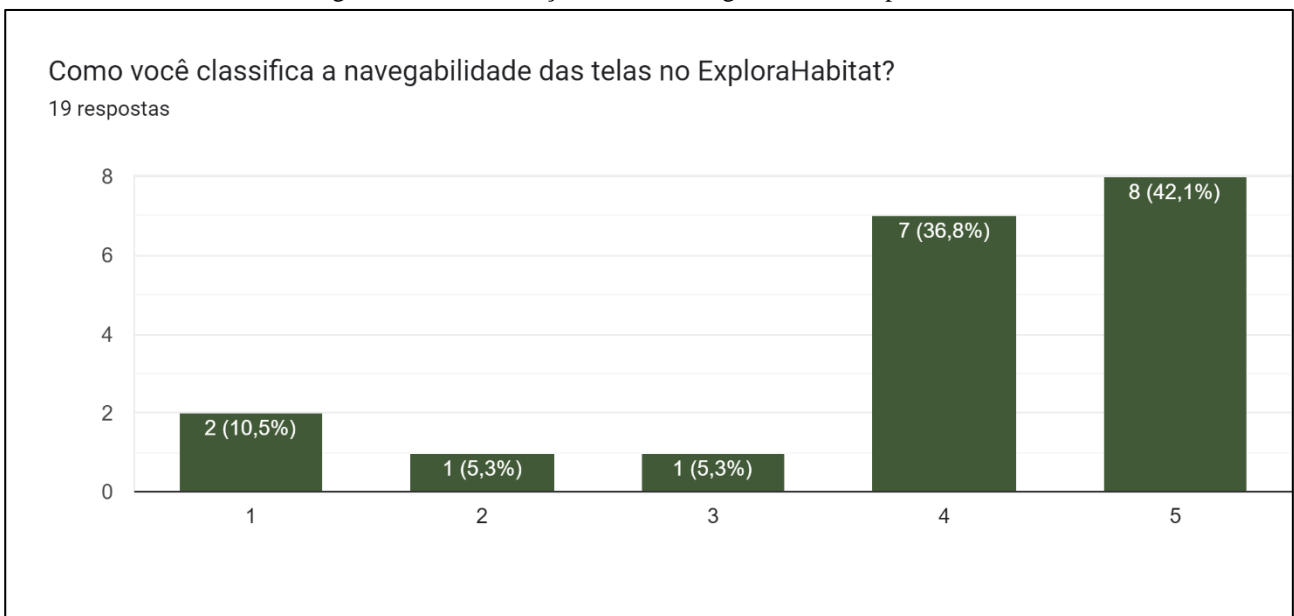
Fonte: elaborado pelo autor.

Figura 17 – Classificação sobre a usabilidade do aplicativo



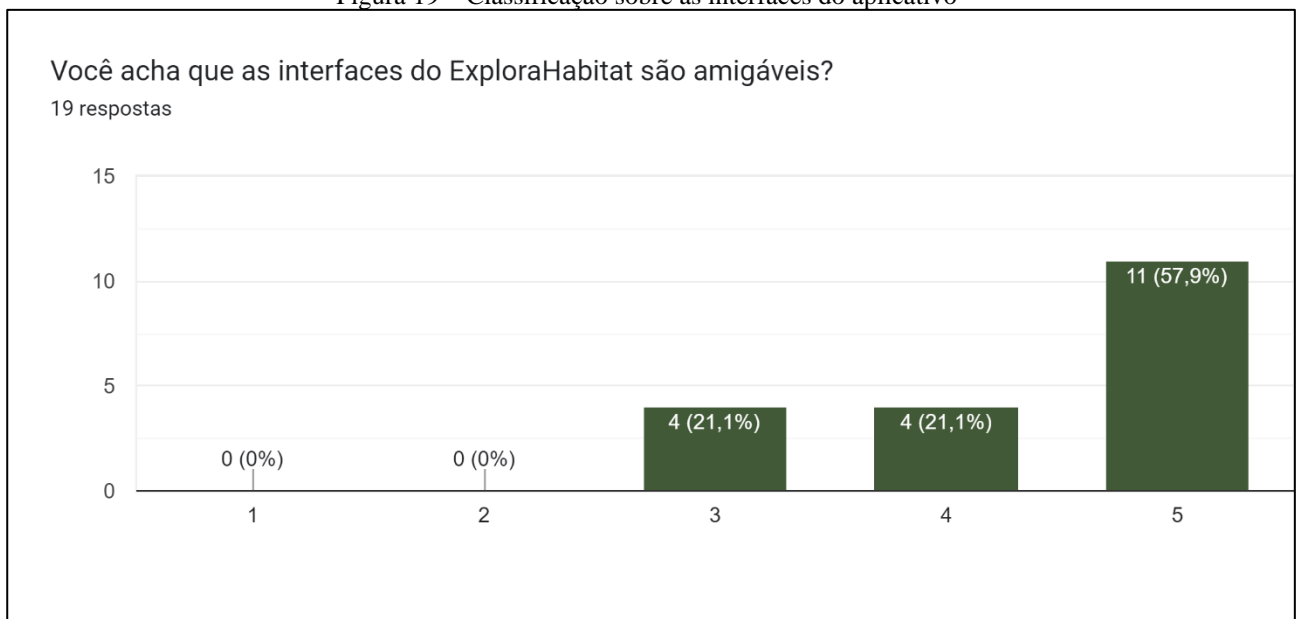
Fonte: elaborado pelo autor.

Figura 18 – Classificação sobre a navegabilidade do aplicativo



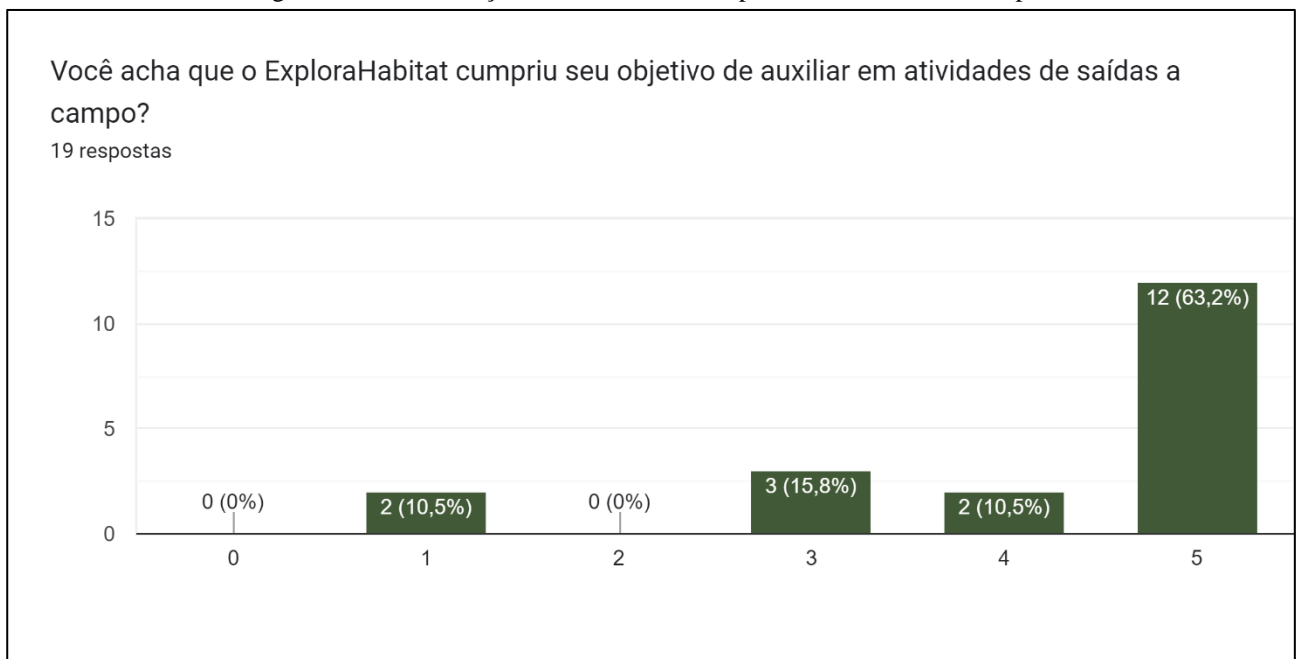
Fonte: elaborado pelo autor.

Figura 19 – Classificação sobre as interfaces do aplicativo



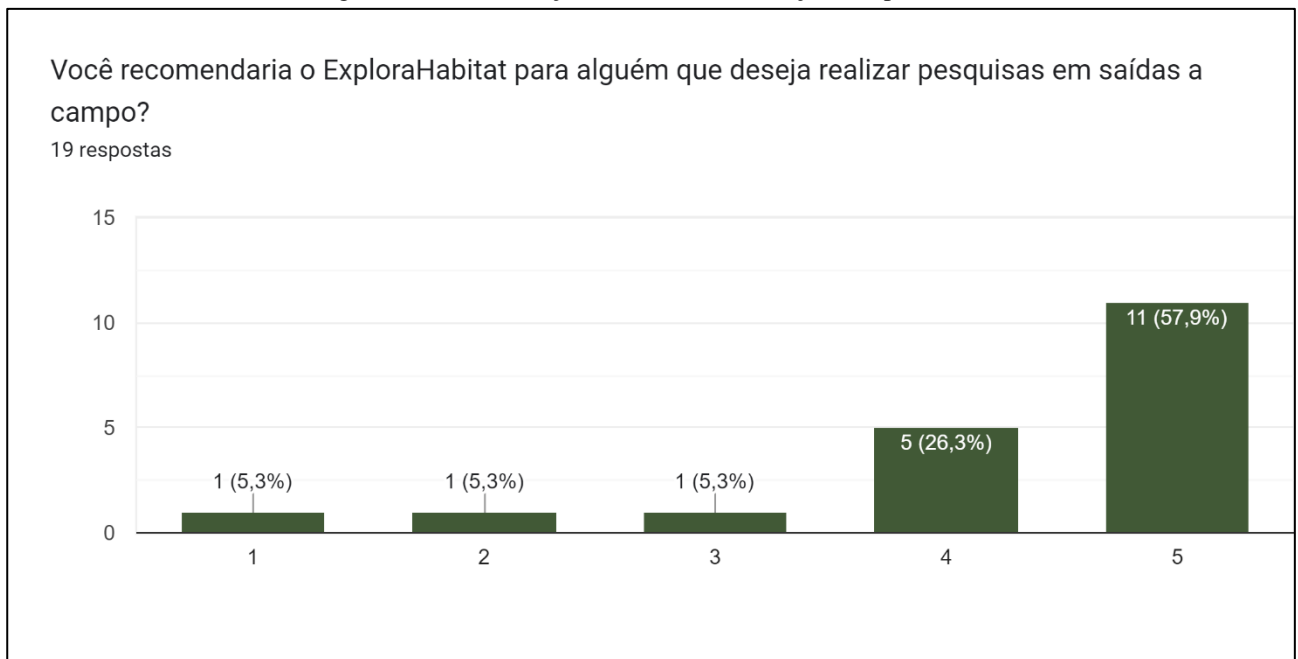
Fonte: elaborado pelo autor.

Figura 20 – Classificação sobre o auxílio do aplicativo nas saídas a campo



Fonte: elaborado pelo autor.

Figura 21 – Classificação sobre a recomendação do aplicativo



Fonte: elaborado pelo autor.