Encontre na UFMS: facilitando a navegação pelo campus com cooperatividade

Filipe dos Santos Pires[1], **Wilson Eduardo Fantucci Diniz de Almeida**[1], **Ana Karina Dourado Salina de Oliveira**[1]

[1]Faculdade de Computação - Universidade Federal do Mato Grosso do Sul (UFMS)
Campo Grande - MS - Brasil

filipe_pires@ufms.br, wilson.eduardo@ufms.br, ana.salina@ufms.br

Resumo. *Este artigo descreve o desenvolvimento do aplicativo móvel "Encontre na UFMS", que surgiu com o intuito de suprir a falta de acesso às informações dos locais presentes no campus da UFMS de Campo Grande, para alunos, visitantes e docentes. O principal objetivo do aplicativo Encontre na UFMS é oferecer uma ferramenta simples e acessível para ajudar os seus usuários a encontrarem pontos de interesse, além de possibilitar a cooperatividade da comunidade de usuários que podem sugerir a criação ou alteração de locais presentes no aplicativo, possibilitando que o aplicativo mantenha-se sempre atualizado. O aplicativo foi desenvolvido utilizando: o framework Flutter no frontend, o framework Fastify no backend e o banco de dados MySQL para armazenamento de dados gerais do aplicativo. Este artigo detalha a estrutura e o funcionamento do aplicativo.*

Abstract. *This article describes the development of the mobile application "Encontre na UFMS", which emerged with the aim of filling the lack of access to information about the locations on the UFMS campus in Campo Grande, for students, visitors and teachers. The main objective of the Encontre na UFMS application is to offer a simple and accessible tool to help its users find points of interest, as well as to enable the cooperativity of the user community that can suggest the creation or alteration of locations present in the application, enabling the application to always stay up to date. The application was developed using: the Flutter framework on the frontend, the Fastify framework on the backend and the MySQL database for storing general application data. This article details the structure and operation of the application.*

1. Introdução

O campus da UFMS de Campo Grande possui uma vasta área ocupada por diversas faculdades, institutos e blocos em geral. Esse fato dificulta que visitantes, alunos e até mesmo docentes conheçam ou encontrem locais presentes na UFMS. Dessa forma, foi criado o Encontre na UFMS, um aplicativo que visa centralizar locais e pontos de interesse no campus da UFMS de Campo Grande.

O principal objetivo do aplicativo é permitir que os usuários busquem e localizem pontos de interesse dentro do campus, de modo que obtenham informações sobre como chegar ao local, além de outras informações que possam ser pertinentes para conhecimento geral e individual. O aplicativo tem o intuito de ser colaborativo para a comunidade acadêmica, possibilitando que todos contribuam para sua constante atualização, além de proporcionar facilidade de acesso e utilização aos usuários.

Este artigo está estruturado da seguinte maneira: Seção 2: Nesta seção são abordados os trabalhos relacionados. Seção 3: Nesta seção são abordados os requisitos e a arquitetura. Seção 4: Nesta seção é abordada a implementação. Seção 5: Esta seção apresenta as considerações finais. Seção 6: Nesta seção são abordados os trabalhos futuros. Referências: Nesta seção são listadas as referências utilizadas no desenvolvimento deste trabalho.

2. Trabalhos Relacionados

Nesta seção abordamos aplicativos existentes que têm objetivos semelhantes ao do aplicativo Encontre na UFMS e que podem fornecer insights úteis para seu desenvolvimento e implementação. Dois aplicativos relevantes nesse contexto são *Google Maps* [20] e *Localização UFMS* [24].

O Google Maps, da Google LLC, é altamente utilizado no mundo todo diariamente como ferramenta de geolocalização. Ele possui diversas funcionalidades, sendo uma das principais a de encontrar um local desejado pelo usuário e, também, fornecer uma rota viável para que o usuário possa chegar ao destino. O aplicativo da Google também é capaz de dar diversas informações sobre os locais disponíveis para busca, como horário de funcionamento e informações de contato.

O Localização UFMS, desenvolvido pela própria UFMS, é uma aplicação web que disponibiliza um mapa interativo de todos os campi da UFMS, incluindo diversos pontos de interesse. Nessa ferramenta, o usuário consegue realizar buscas utilizando diversos filtros e visualizar informações detalhadas sobre os pontos de interesse, como lotação, horários, disponibilidade e agenda. Dependendo da classificação de um certo local pode mostrar informações diferentes, alguns possuem até mesmo fotos enquanto outros só possuem informações de tamanho do local.

2.1. Análise Crítica e contribuição do Encontre na UFMS

Para podermos analisar o diferencial do aplicativo Encontre na UFMS podemos compará-lo com os aplicativos Google Maps e Localização UFMS. O Google Maps é um aplicativo de geolocalização muito utilizado no mundo todo, porém, ele não é focado em um local específico, como o campus da UFMS, e sim em todo o mundo, sendo assim ele não possui registrado diversos locais de menor escala mas que podem ser de grande importância para

algum visitante ou alunos. Já o Localização UFMS é um aplicativo web que disponibiliza um mapa interativo da UFMS que possui mais de 2900 pontos de interesse registrados, porém, ele não possui alguns pontos de interesse que podem ser de grande importância para a comunidade acadêmica, como por exemplo, o Restaurante Universitário e também não informa ao usuário como chegar até o local desejado já que ele é focado em apenas mostrar informações sobre um certo local, ele também não tem fotos de todos os locais e não é colaborativo uma vez que apenas a administração da UFMS pode adicionar novos locais.

2.2. Comparação entre o Encontre na UFMS e outros aplicativos

A Tabela 1 apresenta uma comparação entre os aplicativos Google Maps, Localização UFMS e Encontre na UFMS, destacando as principais características de cada aplicativo.

Características	**Google Maps**	**Localização UFMS**	**Encontre na UFMS**
Objetivo Principal	Geolocalização global	Mapa interativo da UFMS	Navegação pelo campus da UFMS
Público-Alvo	Usuários em geral	Comunidade acadêmica da UFMS	Comunidade acadêmica da UFMS
Principais Funcionalidades	Busca de locais, rotas, informações de contato, horários de funcionamento	Busca de locais no campus, informações sobre pontos de interesse	Busca de locais, rotas, cadastro colaborativo de locais, avaliações de locais

Tabela 1. Comparação entre aplicativos de localização

3. Requisitos e Arquitetura

3.1. Análise de contexto

3.1.1. Visão geral

O sistema proposto é um aplicativo para dispositivos móveis que visa facilitar a navegação pelo campus da UFMS, em Campo Grande. A principal responsabilidade do aplicativo é fornecer informações sobre a localização de prédios, serviços e pontos de interesse no campus, bem como permitir que usuários sugiram adições de novos pontos de interesse e atualizações de informações.

3.1.2. Condições Restritivas

O aplicativo é projetado exclusivamente para dispositivos móveis com sistema operacional Android, tendo o Android 6.0 (API 23) como versão mínima suportada. O aplicativo também exige conexão com a internet para funcionar corretamente.

3.1.3. Benefícios

O aplicativo auxiliará estudantes, professores, visitantes e demais usuários do campus a se localizarem e a encontrarem informações sobre os prédios e serviços disponíveis. Além disso, o aplicativo permitirá que usuários sugiram adições de novos pontos de interesse e atualizações de informações, contribuindo para a melhoria contínua do aplicativo.

3.2. Analise de requisitos

3.2.1. Metodologia

Para o desenvolvimento do aplicativo, foi realizado o levantamento de requisitos através de reuniões com a equipe de desenvolvimento. Novos requisitos se mostraram importantes conforme o avanço do desenvolvimento dos requisitos anteriormente propostos.

3.2.2. Lista de Atores

Foram identificados três tipos de atores:

- **Usuário comum não logado**: Tem acesso aos locais e suas respectivas informações, além dos filtros de buscas. Pode se cadastrar, logar ou recuperar o seu perfil de usuário no sistema.
- **Usuário comum logado**: Tem acesso aos mesmos privilégios do usuário comum não logado, com a adição de: Permissão para alterar informações de seu perfil de usuário, favoritar e avaliar locais, visualizar o registro dos últimos locais acessados, sugerir a adição ou alteração de locais.
- **Administrador**: Possui permissões de administrador do sistema, pode adicionar ou alterar locais.

3.2.3. Lista de Funcionalidades

As funcionalidades desenvolvidas para a aplicação e os atores responsáveis estão registrados na Tabela 2:

Funcionalidade	Ator(es)
Visualizar informações de locais	Qualquer usuário
Buscar ou filtrar por locais	Qualquer usuário
Cadastrar conta	Usuário não logado
Fazer login	Usuário não logado
Recuperar conta	Usuário não logado
Editar perfil de usuário	Usuário comum logado e Administrador
Favoritar locais	Usuário comum logado e Administrador
Avaliar locais	Usuário comum logado e Administrador
Visualizar registros de locais acessados	Usuário comum logado e Administrador
Sugerir adição ou alteração de locais	Usuário comum logado
Adicionar ou alterar locais	Administrador

Tabela 2. Funcionalidades e Atores do aplicativo Encontre na UFMS

3.3. Especificação dos Requisitos de Software

Os requisitos são divididos entre os Requisitos Funcionais e Não-Funcionais. Abaixo, estão listados alguns dos requisitos funcionais descritos no *Documento de Requisitos Aplicativo Encontre na UFMS* [17]:

- **Listagem de pontos importantes**: O sistema deve listar pontos importantes do campus de Campo Grande da UFMS. Exemplo: Blocos Acadêmicos, Bancos, Pontos Turísticos, Restaurantes, etc.
- **Listagem de informações detalhadas**: O sistema deve fornecer informações detalhadas a respeito de um determinado local, tais como: Nome, endereço, telefone, horário de funcionamento, fotos, entre outras informações.
- **Redirecionamento para o Google Maps**: O sistema deve oferecer um link que redirecione o usuário para o aplicativo ou página na internet do Google Maps [20] com o endereço do local inserido.
- **Busca de pontos específicos**: O sistema deve permitir que o usuário busque um ponto específico utilizando o nome do local e/ou filtros de busca.
- **Login**: O sistema deve permitir que o usuário faça login, além de possibilitar o cadastro e a recuperação de conta.
- **Favoritar**: O sistema deve permitir que o usuário favorite os locais de interesse.
- **Avaliar**: O sistema deve permitir que o usuário avalie os locais, utilizando uma escala de 1 a 5 estrelas.
- **Criar e alterar locais**: O sistema deve permitir que usuários façam sugestões para inclusão ou alteração de locais.

O sistema visa oferecer ao usuário uma experiência de busca e localização eficiente, com uma interface amigável e intuitiva que permite navegar e encontrar informações úteis sobre cada local. Além disso, possui funcionalidades como redirecionamento para o Google Maps [20], a fim de traçar rotas para que o usuário consiga chegar ao local. Também inclui funcionalidades de cadastro, login e recuperação de conta, permitindo ao usuário usufruir de opções como avaliação de locais, favoritá-los e sugerir novos locais ou editar locais existentes no aplicativo.

4. Solução Implementada

O aplicativo Encontre na UFMS foi desenvolvido com o intuito de facilitar a navegação pelo campus da UFMS em Campo Grande, permitindo que usuários encontrem e localizem pontos de interesse dentro do campus, estes sendo cadastrados por outros usuários, possibilitando que locais que não estão presentes em outros aplicativos de localização sejam documentados e compartilhados com a comunidade acadêmica. Os usuários também terão poder de avaliar os locais cadastrados, podendo assim, ajudar outros usuários a escolherem o melhor local para suas necessidades.

4.1. Implementação

O desenvolvimento do aplicativo Encontre na UFMS foi dividido em duas partes principais: o backend e o frontend. Trataremos a seguir de cada uma dessas partes, detalhando as tecnologias utilizadas e as funcionalidades implementadas.

4.1.1. Frontend

O frontend foi desenvolvido utilizando a ferramenta *Flutter* [19], um framework para a linguagem *Dart* [18] e que pertence ao Google. Tanto o Dart quanto o Flutter foram criados principalmente para possibilitar o desenvolvimento de aplicativos móveis. Por se tratar de uma ferramenta relativamente nova, possui apenas 7 anos desde o seu lançamento, ela se aproveita de vários padrões já bem estabelecidos em outras linguagens e frameworks para criar um ambiente moderno e de fácil adequação para os desenvolvedores.

A arquitetura do aplicativo foi dividida em três camadas principais: a camada de apresentação, a camada de domínio e a camada de dados. A camada de apresentação é responsável por exibir os dados para o usuário e capturar as interações do usuário com o sistema. A camada de domínio contém as regras de negócio do sistema, enquanto a camada de dados é responsável por acessar os dados do sistema, seja de um banco de dados local ou de uma API remota. Essa divisão de camadas é ilustrada na Figura 1.

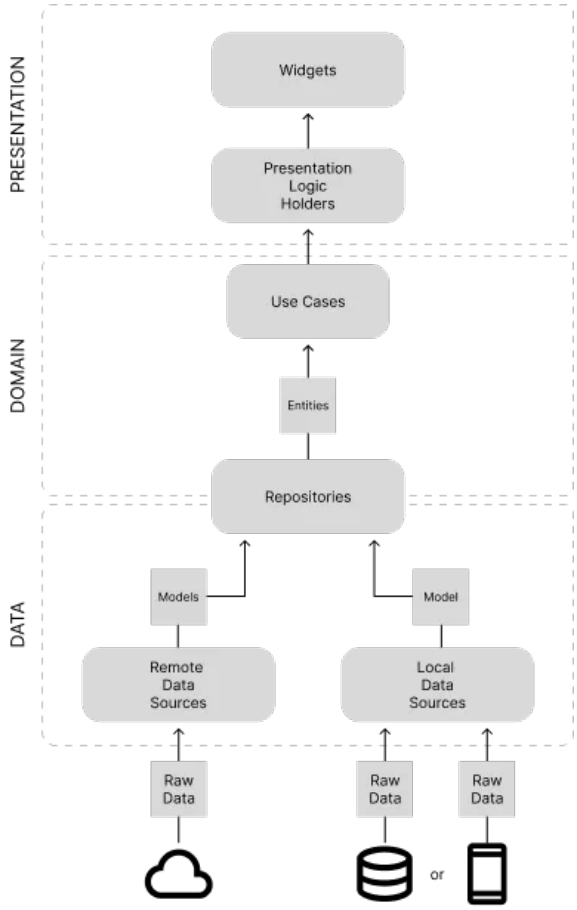

Figura 1. **Exemplificação da arquitetura em camadas**
Fonte: Semih Altin[22]

Embora o Flutter seja capaz de gerar aplicativos nativos para Android e iOS, o aplicativo Encontre na UFMS foi desenvolvido apenas para Android, visto que a Apple, dona do sistema operacional iOS, restringe o desenvolvimento para seu sistema caso o desenvolvedor não possua um dispositivo da marca. Mas, em teoria, o aplicativo poderia ser facilmente adaptado para iOS, bastando apenas compilar o código fonte em um ambiente de desenvolvimento da Apple. O código-fonte do frontend está disponível no repositório em: *Frontend Encontre Na Ufms* [16].

Uma das grandes preocupações ao se desenvolver um aplicativo móvel é a gerência de possíveis estados enquanto o usuário navega e interage com este aplicativo, como fazer telas conversarem entre si e como manter a experiência o mais contínua e suave possível. Para isso é muito importante separar a apresentação dos dados para o usuário da lógica de negócios que faz todo o tratamento desses dados. A fim de exercer esse papel foi utilizdo o padrão BLoC (Business Logic Component), com a biblioteca *flutter_bloc* [6], que permite acessar todo o estado da aplicação a qualquer momento e também dispara eventos quando certos estados mudam para que a apresentação dos dados esteja sempre atualizada.

Para armazenar localmente o token do usuário logado foi utilizado o *Hive* [23], um banco de dados não relacional que tem como principais características a leveza e a

rapidez. O mini mapa que é utilizado para mostrar a localização e para cadastrar um novo local facilitando o processo para o usuário foi implementado utilizando a biblioteca *google_maps_flutter* [7] desenvolvido pela própria equipe do Flutter.

4.1.2. Backend

O backend foi desenvolvido utilizando o framework *Fastify* [5] com *Node.js* [11], proporcionando alto desempenho e grande velocidade nas requisições em comparação a outros frameworks, além da facilidade de uso. Foi utilizado *TypeScript* [15] para aumentar a segurança contra erros de tipagem durante o processo de desenvolvimento.

Durante o desenvolvimento, foi utilizado a linguagem de programação Typescript para tipagem de variáveis, classes e objetos, evitando eventuais erros que poderiam ser causados por incompatibilidade de tipos. Para a execução do código utilizando o Node.js, foi utilizado uma ferramenta do TypeScript: ”tsc”ou TypeScript Compiler, para transpilação do código-fonte em JavaScript durante o desenvolvimento, enquanto que para ser utilizado em produção, foi utilizado o tsup, uma ferramenta de empacotamento e transpilação de código TypeScript para JavaScript, rápida e eficiente.

O código foi desenvolvido em um padrão de camadas, similar ao padrão Controller Service Repository (CSR), ou Controlador-Serviço-Repositório, onde a camada do Controlador recebe as requisições HTTP e as passa para a camada de Serviço, onde a lógica de negócio está presente, depois, acessa a camada Repositorio que é responsável por acessar os dados. A arquitetura implementada no backend possui uma camada de Model ou Modelo ao invés da Repositório, que contêm a descrição da estruturação do banco de dados e é responsável pela manipulação dos dados, sendo a representação da arquitetura utilizada simular a Figura 2.

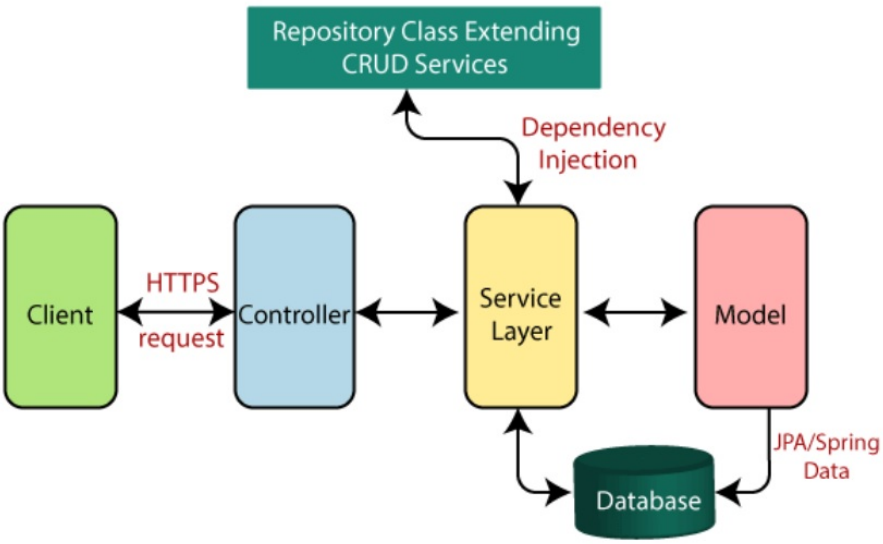

Figura 2. **Exemplificação de arquitetura CSR com Model**
Fonte: Services [13]

Para a comunicação entre o backend e o frontend, foi utilizado uma API REST. O

código-fonte do backend está disponível no repositório em: *Backend Encontre Na Ufms* [25]

4.1.3. Banco de dados

Para o armazenamento de dados no backend, foi utilizado o *MySQL* [10], um sistema de gerenciamento de banco de dados relacional open-source que oferece confiabilidade e desempenho. A conexão entre o Node.js e o MySQL foi feita utilizando o *Drizzle* [4], um mapeador de objetos (ORM) que foca na velocidade e eficiência na entrega de dados entre as duas ferramentas.

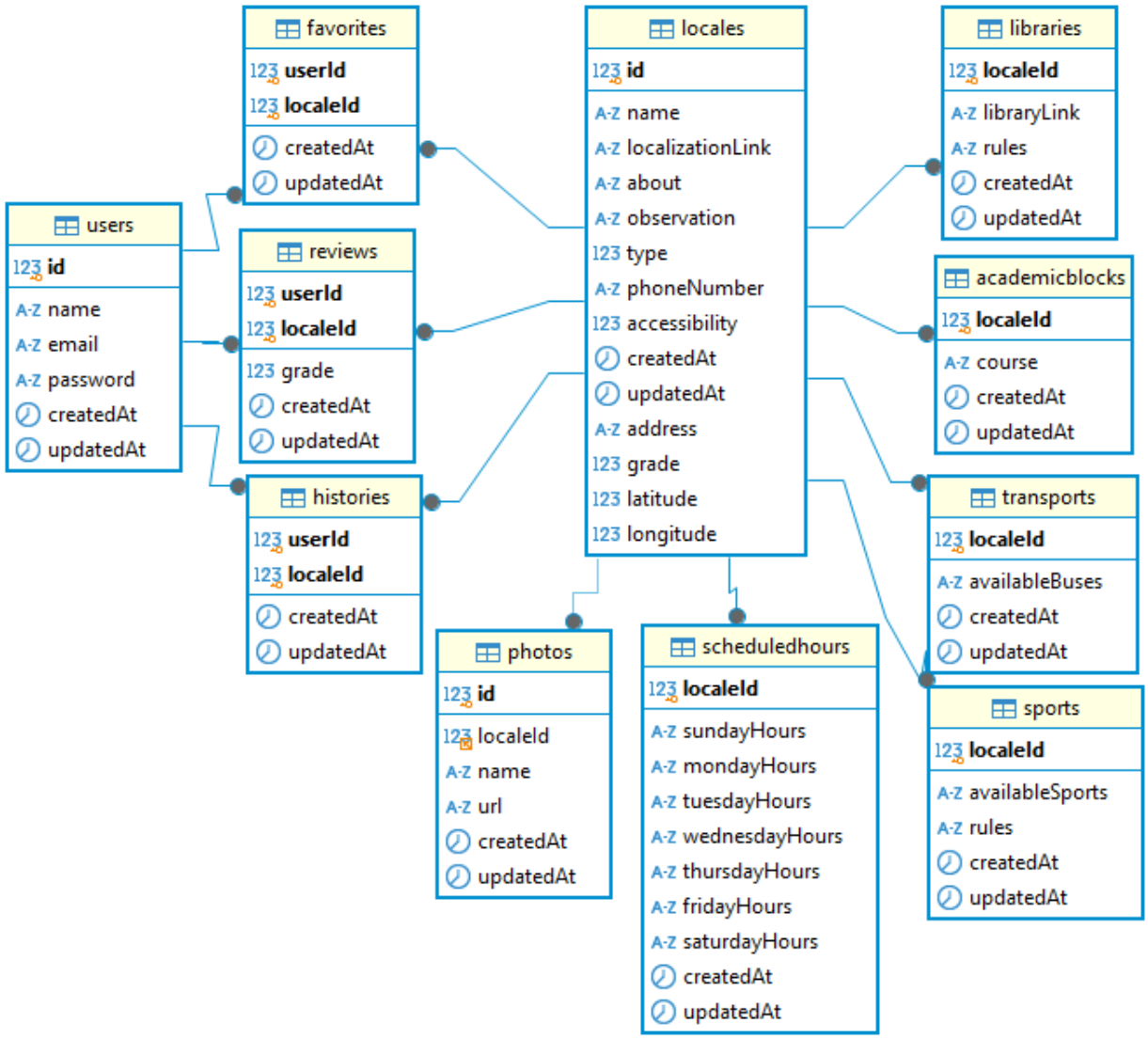

Figura 3. **Interface do DBeaver exibindo a estrutura do banco de dados [2].**

Como descrito na Figura 3, o centro da aplicação e do banco de dados são locais. A estrutura do banco foi modelada para permitir uma organização eficiente das

informações, com tabelas que representam tanto os locais quanto os atributos e relacionamentos associados:

- **Locales**: Armazena os dados gerais de cada local, como nome, descrição, e localização.
- **Schedules**: Relacionamento de um para um com a tabela Locales, responsável por armazenar os horários de funcionamento dos locais ao longo da semana.
- **Photos**: Relacionamento de um para muitos com a tabela Locales, onde cada local pode ter diversas fotos associadas.
- **Users**: Armazena os dados do usuário, como nome, email, senha.
- **Histories**: Relacionamento de muitos para muitos com a tabela Locales e a tabela Users, onde cada usuário pode ter um histórico de locais visualizados recentemente.
- **Reviews**: Relacionamento de muitos para muitos com a tabela Locales e a tabela Users, armazena as avaliações de cada usuário feitas aos locais.
- **Favorites**: Relacionamento de muitos para muitos com a tabela Locales e a tabela Users, onde cada usuário pode ter locais favoritos.
- **AcademicBlocks**: Tabela exclusiva para armazenar os nomes dos cursos pertencentes ao respectivo bloco acadêmico.
- **Libraries**: Tabela exclusiva para armazenar as as regras e o link para site externo da biblioteca.
- **Sports**: Tabela exclusiva para armazenar as regras e os esportes disponíveis para serem praticados no centro de esportes.
- **Transportes**: Tabela exlusiva para armazenenar os ônibus que passam no local.

A tabela “Locales” armazena os dados gerais que compõem um determinado local. Sendo que cada local pode possuir horários de funcionamento durante a semana, sendo armazenado na tabela “Schedules”. Os locais também podem ter fotos, sendo armazenados na tabela “Photos”.

Os locais foram divididos em tipos específicos para melhor organização dos dados e apresentação ao usuário final, sendo eles: Blocos Acadêmicos, Pontos Turísticos, Bancos, Restaurantes, Serviços de Saúde, Bibliotecas, Centros de Esportes, Transportes, Estacionamentos e Prédios Gerais. Alguns desses tipos posssuem tabelas próprias com informações específicas, como é o caso dos tipos: Blocos Acadêmicos, Bibliotecas, Centros de Esportes e Transportes.

Além disso, também foi criado uma tabela “Users” para armazenar os dados básicos do usuário para cadastro e login do mesmo dentro do sistema. Os usuários possuem 3 relações com a tabela “Locales”, formando 3 tabelas, sendo elas: “Histories”, “Reviews” e “Favorites”, que armazenam respectivamente:

- O histórico dos 5 últimos locais visualizados recentemente
- As avaliações que o usuário fez aos locais
- Os locais favoritos.

Para a conexão entre o Node.js e o MySQL, foi utilizado o Drizzle que é uma ferramenta de mapeamento de objetos, cujo o foco é a velocidade de entrega de dados entre as duas ferramentas.

4.2. Login

Os usuários tem a possibilidade de se cadastrarem no aplicativo, sendo o cadastro um requisito não obrigatório. Para usuários logados no sistema, são utilizados tokens *JWT (JSON Web Token)* [8] para autenticação, garantindo acesso seguro e eficaz ao sistema.

Para o envio de e-mails de recuperação de senha, foi utilizado o *MailJet* [9], uma ferramenta que fornece acesso a APIs para envio de e-mails e mensagens de texto (SMS).

4.3. Ambiente de desenvolvimento: Visual Studio Code

O ambiente de desenvolvimento escolhido foi o *Visual Studio Code* [21], uma das IDEs mas utilizadas no mundo graças ao grande número de ferramentas que podem ser integradas nela para aumentar as funcionalidades de desenvolvimento, teste e depuração.

Foi utilizado também o *Android SDK (Software Development Kit)* [1], que é um conjunto de ferramentas que permite o desenvolvimento de aplicativos para a plataforma Android. Ele inclui um depurador, bibliotecas, um emulador de dispositivo baseado em *QEMU* [12], documentação, amostras de código e tutoriais.

4.4. Bibiliotecas e Dependências

Diversas bibliotecas e dependências foram utilizadas no desenvolvimento do aplicativo Encontre na UFMS, dentre elas destacam-se:

- **Fastify**: Framework para desenvolvimento de aplicações web com Node.js [5].
- **TypeScript**: Linguagem de programação que extende JavaScript, adicionando tipagem à linguagem [15].
- **tsup**: Empacotador de arquivos TypeScript e JavaScript, utilizado para otimizar e gerar builds [14].
- **MySQL**: MySQL é um sistema de gerenciamento de banco de dados relacional de código aberto baseado em SQL [10].
- **Drizzle**: Mapeador de objetos para conexão entre Node.js e MySQL [4].
- **MailJet**: Ferramenta para envio de emails e sms [9].
- **Flutter**: Framework para desenvolvimento de aplicativos móveis [19].
- **Dart**: Linguagem de programação utilizada no Flutter [18].
- **hive**: Banco de dados não relacional utilizado para armazenamento local [23].
- **google_maps_flutter**: Biblioteca para implementação de mapas no Flutter [7].
- **flutter_bloc**: Biblioteca para gerenciamento de estados no Flutter [6].
- **dio**: Biblioteca para requisições HTTP no Flutter [3].

Todas essas bibliotecas e dependências, bem como outras que foram especificadas nos projetos do frontend e do backend, foram fundamentais para o desenvolvimento do aplicativo Encontre na UFMS, permitindo a implementação de funcionalidades essenciais e a integração entre o backend e o frontend.

4.5. Testes e Execução

O aplicativo Encontre na UFMS foi testado apenas por nós em diversos dispositivos Android, sendo alguns deles dispositivos físicos aos quais possuímos e outros emulados, a fim de garantir a compatibilidade e o bom funcionamento em diferentes tamanhos de tela e versões do sistema operacional. Foi utilizado o emulador do Android Studio para testar

o aplicativo no Pixel 6a na API 33 do Android mas foi principalmente testado em dispositivos físicos, como o Galaxy S21 fe e Xiaomi Note 8 Pro, pela facilidade de uso e para poder verificar a usabilidade real do aplicativo em todo momento.

Os testes foram feitos utilizando as principais funcionalidades do aplicativo, na sequência dos items descritos na tabela 1, sendo que a maior parte dos testes foram feitos realizando repetidas ações que de carregamento de informações, que resultam em novas chamadas de API para o backend, garantindo a consistência do backend em receber a requisição e tratá-la corretamente para retornar os dados ao frontend, além de garantir a consistência do frontend em fazer as requisições para a API corretamente e ao apresentar os dados recebidos ao usuário de forma visual.

4.5.1. Diagrama de Navegação

A Figura 4 ilustra o diagrama de navegação do aplicativo Encontre na UFMS:

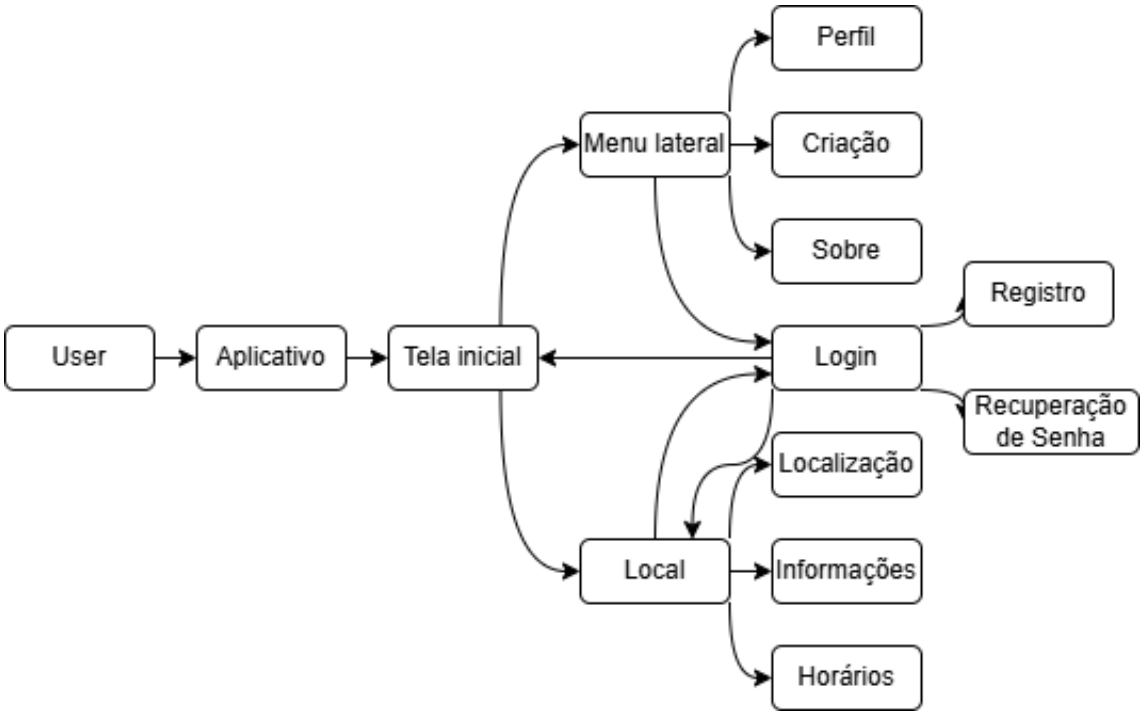

Figura 4. **Diagrama de Navegação**

O diagrama de navegação do aplicativo apresentado acima demonstra os fluxos básicos de navegação que o usuário pode experenciar durante o uso do mesmo, vale destacar que algumas telas como Perfil e Criação só são acessíveis caso o usuário esteja logado.

4.5.2. Tela 1: Listagem de Locais

A tela de listagem de locais é a tela inicial do aplicativo, a primeira e principal tela apresentada para o usuário. Nela, o usuário pode visualizar todos os locais cadastrados no

aplicativo, podendo filtrar os locais por categoria e buscar locais através de um campo de pesquisa por texto.

Cada local é representado por um segmento contendo a foto, o nome, o endereço, a categoria, se o local está provavelmente fechado naquele horário e um indicador que mostra se o local está marcado como favorito ou não. Cada segmento é clicável, redirecionando o usuário para a tela de detalhes do local. O usuário pode também clicar no indicador de favorito para marcar ou desmarcar o local como favorito caso ele esteja logado, caso não esteja o usuário é redirecionando para a tela de Login que será abordada mais a frente. A listagem de locais é paginada, exibindo 10 locais por vez, uma nova requisição é feita toda vez que o usuário chega ao final da lista, carregando a próxima página, se houver.

Na parte superior da tela há uma seção com um botão, que abre o menu lateral, e um campo de texto para pesquisa. Logo abaixo há um filtro de categorias, uma série de botões que podem ser arrastados horizontalmente, permitindo que o usuário filtre os locais por uma ou mais categorias, de acordo com suas necessidades. Na busca por texto foi aplicado um *debounce* de 1s para evitar que a busca seja feita a cada caractere digitado, o que poderia causar um consumo excessivo de recursos da API. A Figura 5 mostra a tela de listagem de locais.

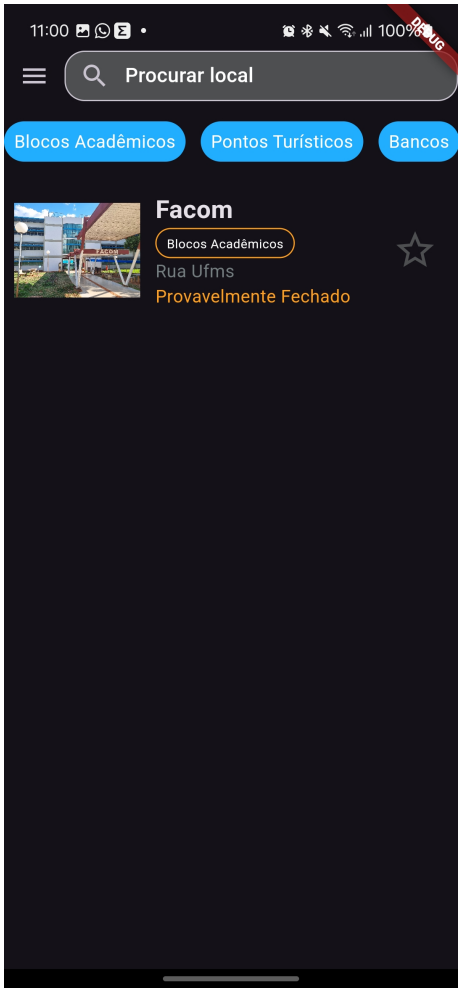

Figura 5. **Tela 1: Listagem de Locais**

4.5.3. Tela 2: Local

A tela de local apresenta todos os detalhes do local que foi selecionado pelo usuário. Nela, o usuário pode visualizar uma ou mais fotos do local, o nome, um indicador que é mostrado caso o local tenha opções de acessibilidade e a avaliação média do local, que é a média das avaliações por outros usuários. Além disso existem 3 abas que o usuário pode navegar sem sair da tela. A aba de localização é a aba inicial e mostra o endereço do local, um mapa com a localização do local indicada e uma seção para o usuário dar sua própria avaliação do local, caso esteja logado. A aba de informações exibe alguns detalhes do local como observações e telefone de contato. Por fim, a aba de Horários exibe os horários de funcionamento do local, caso estejam cadastrados. A Figura 6 mostra a tela de local com a aba de localização, já a Figura 7 apresenta as demais abas.

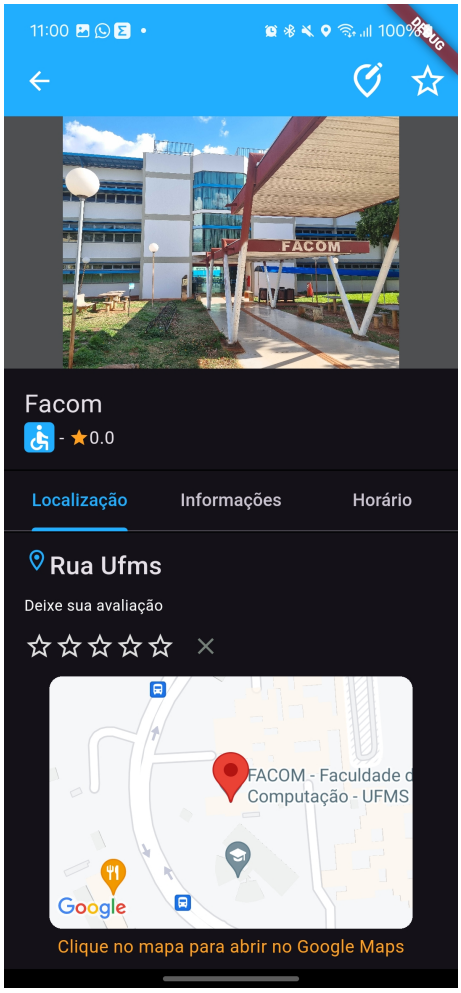

Figura 6. Tela 2: Local

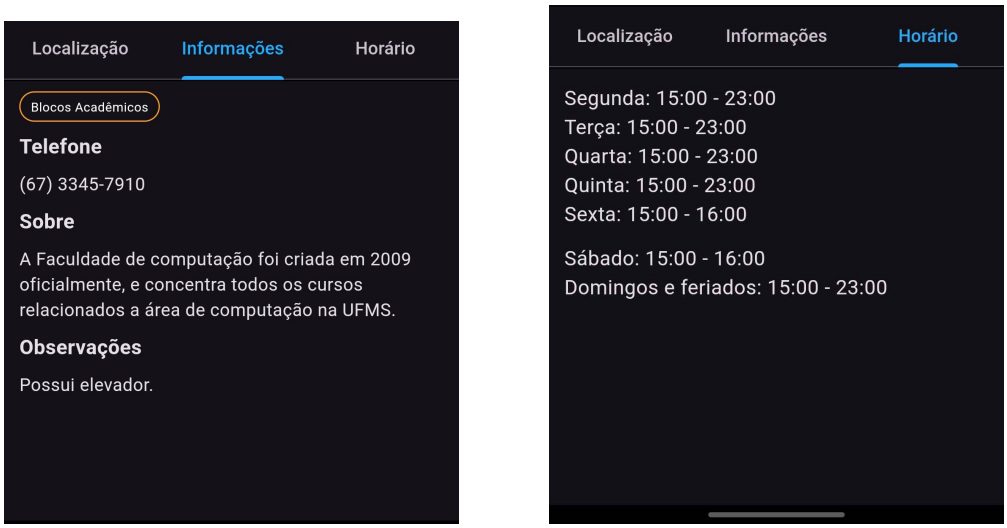

Figura 7. Abas: Informações e Horários

4.5.4. Tela 3: Menu Lateral

O menu lateral é acessível através de um botão na parte superior da tela de listagem de locais. Ele é composto por uma série de botões que redirecionam o usuário para diferentes telas do aplicativo. O menu lateral é uma forma de organizar as funcionalidades do aplicativo de forma intuitiva e acessível, permitindo que o usuário navegue facilmente entre as diferentes telas do aplicativo.

Essa tela muda de acordo com o estado de autenticação do usuário, caso ele esteja logado, aparecerá o nome do usuário logo abaixo o nome do aplicativo e os botões de Perfil e Criação assim como um botão de Sair, caso contrário, todas essas informações são omitidas e apenas o botão de Sobre é exibido, e no lugar do botão de Sair é exibido o botão de Login. Abaixo as duas figuras exemplificam ambos os estados. A Figura 8 mostra o menu lateral tanto no estado logado quanto no estado deslogado.

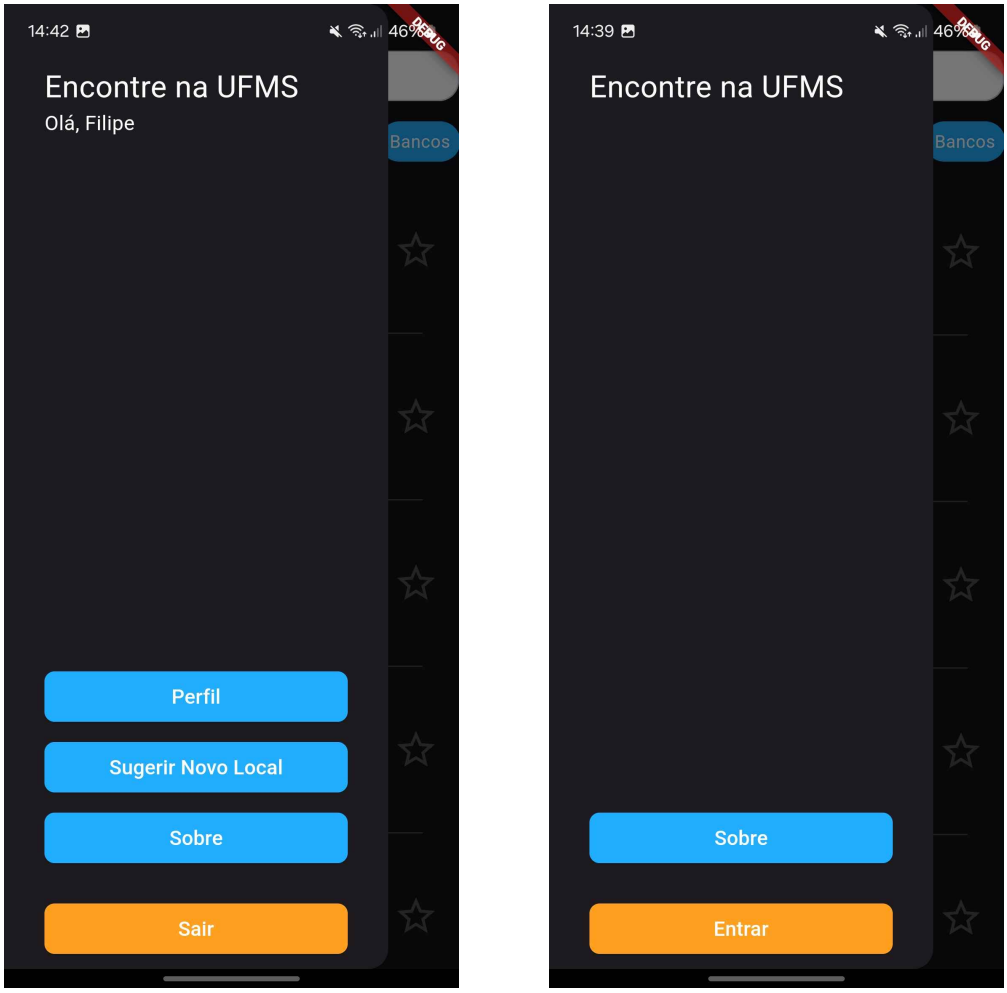

Figura 8. Tela 2: Menu Lateral (Logado e Deslogado)

4.5.5. Tela 4: Perfil

A tela de perfil exibe as informações do usuário logado, nome e e-mail. Além disso, a tela também permite que o usuário edite seu nome e altere sua senha. A tela de perfil é uma forma de o usuário gerenciar suas informações pessoais e garantir que seus dados estejam sempre atualizados. A Figura 9 mostra a tela de perfil.

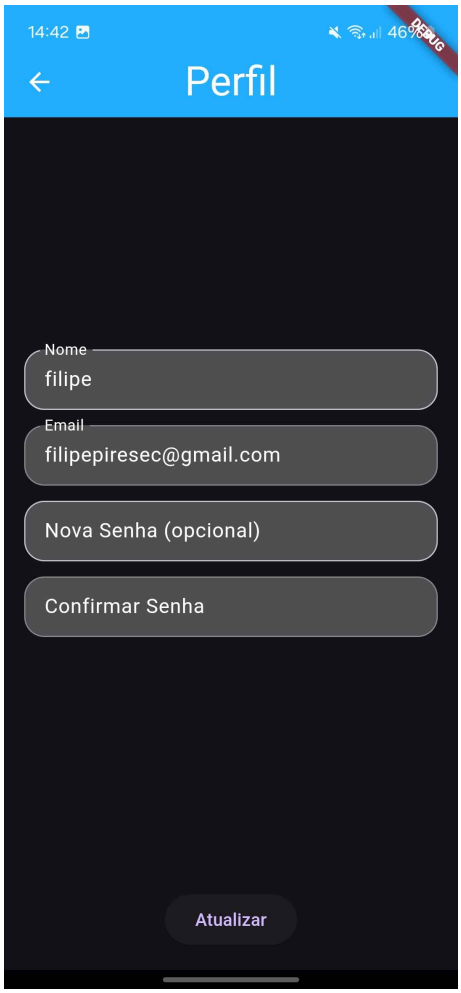

Figura 9. **Tela 4: Perfil**

4.5.6. Tela 5: Criação

A tela de criação possibilita que os usuários preencham uma série de informações sobre o local que desejam cadastrar, como nome, endereço, categoria, horários de funcionamento, telefone de contato, observações e fotos, e enviem para uma avaliação. A tela de criação é uma forma de os usuários contribuírem com o aplicativo, adicionando novos locais e enriquecendo a base de dados do sistema. A Figura 10 mostra as telas de criação.

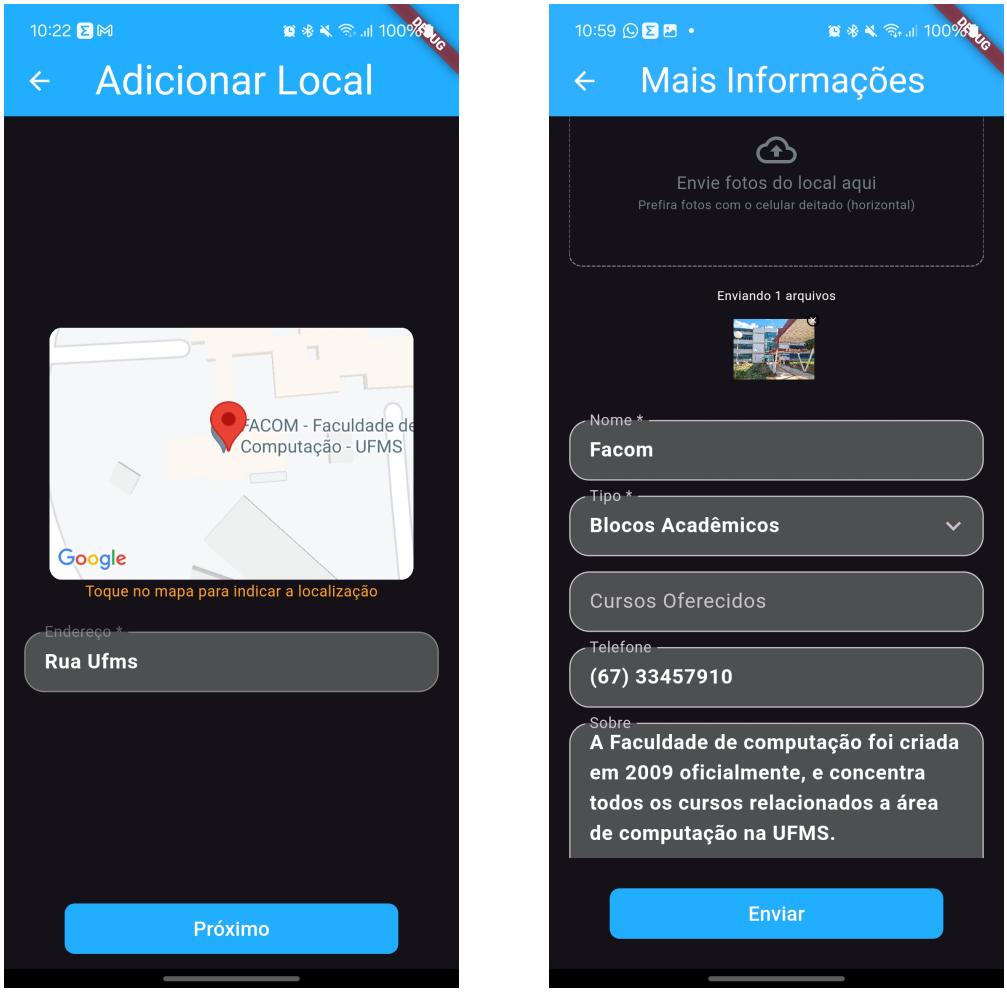

Figura 10. Tela 5: Criação

4.5.7. Tela 6: Sobre

A tela de sobre exibe informações sobre o aplicativo, como a versão, os desenvolvedores e o código fonte. A tela de sobre é uma forma de os usuários conhecerem mais sobre o aplicativo e os desenvolvedores por trás dele. A Figura 11 mostra a tela de sobre.

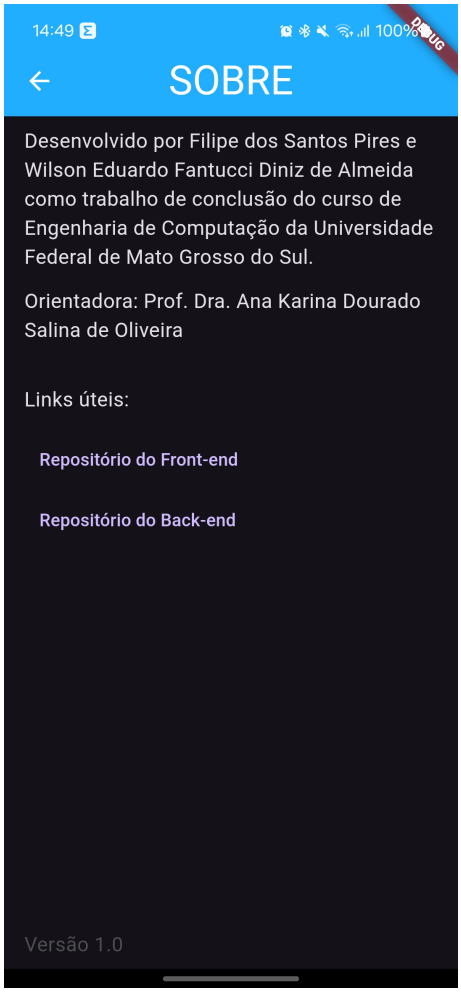

Figura 11. **Tela 6: Sobre**

4.5.8. Tela 7: Login

A tela de login pode ser alcançada através de 2 fluxos, o primeiro é através do botão de Login na tela de Menu Lateral e o segundo é quando o usuário tentar marcar um local como favorito sem estar logado. A tela de login é composta por um campo de e-mail e um campo de senha, além de botões para iniciar o processo de registro de uma nova conta ou para recuperar a senha, caso o usuário tenha esquecido. A Figura 12 mostra a tela de login e de cadastro.

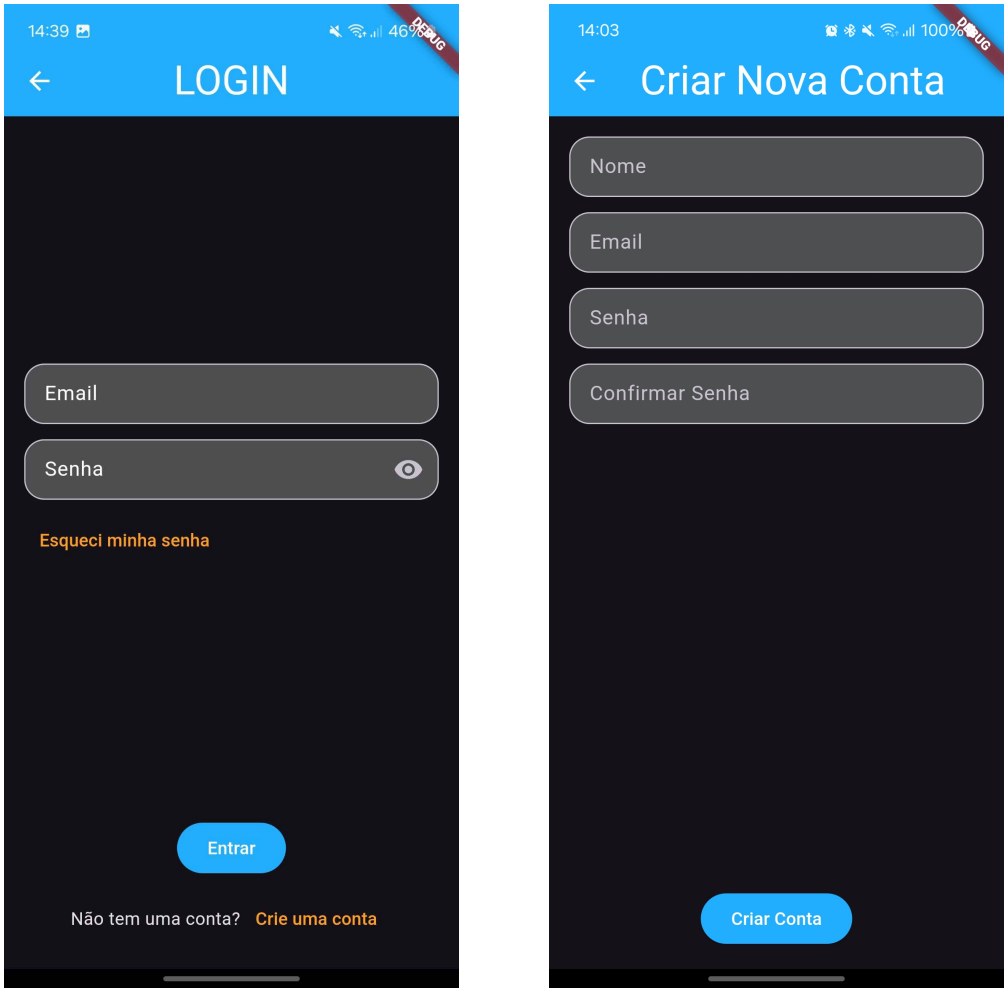

Figura 12. **Tela 7: Login**

Caso o usuário entre no processo de recuperação de senha ele deverá informar o email cadastrado e um email será enviado com um código de recuperação, que deverá ser informado numa tela subsequente, caso a verificação seja bem sucedida o usuário poderá definir uma nova senha. A Figura 13 mostra as telas de recuperação de senha.

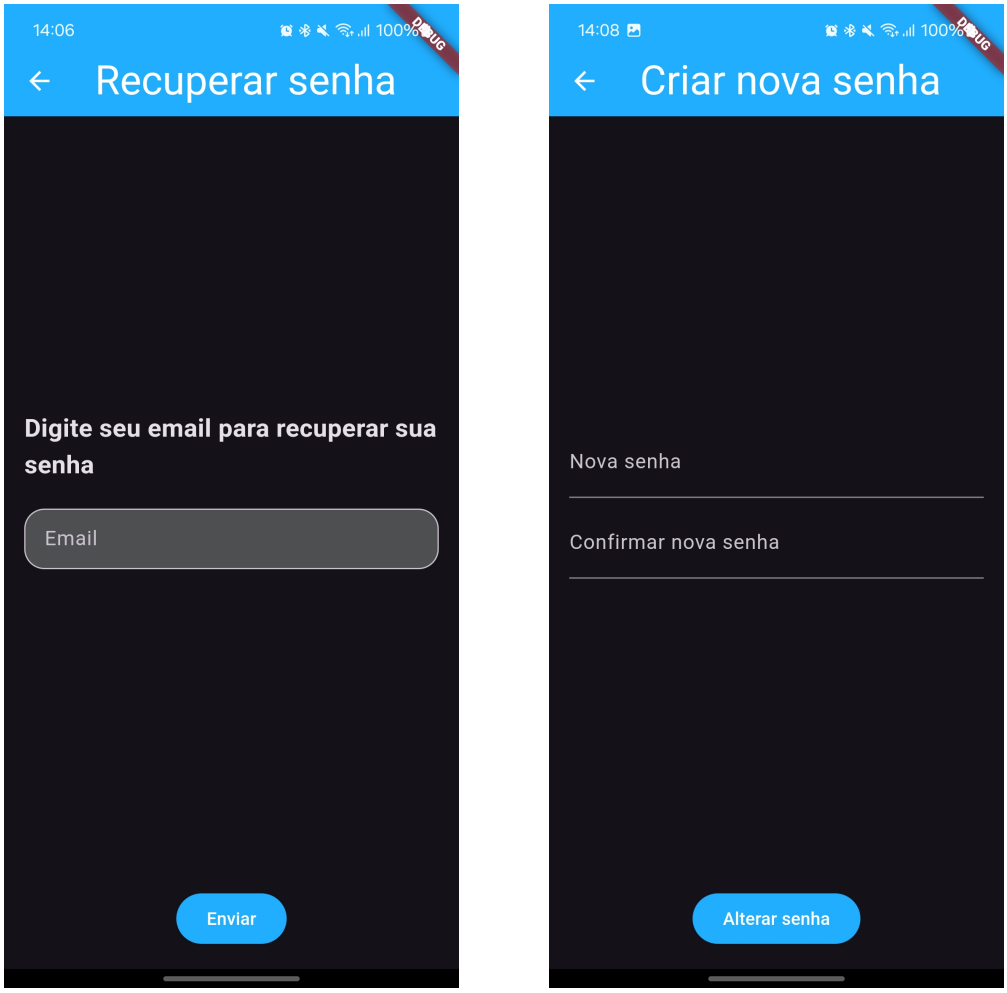

Figura 13. **Tela 7: Login**

5. Considerações Finais

O objetivo do aplicativo Encontre na UFMS foi desde o ínicio ser uma ferramenta simples e acessível para ajudar os frequentadores do campus a se localizarem no dia a dia. Conseguimos implementar isso graças a integração com a API do Google Maps em conjunto com as funcionalidades de colaboração como a criação de novos locais e edição de informações que possam estar defasadas. Acreditamos que a aplicação pode ser muito útil para os estudantes, professores e visitantes da UFMS, facilitando a navegação pelo campus e tornando a experiência mais agradável.

A partir do uso do Flutter para desenvolvimento do aplicativo, foi possível desenvolver um aplicativo com uma interface amigável e intuitiva e que no futuro pode ser expandida para mais plataformas além do Android. A integração com o backend foi simples e eficiente graças a adoção do Fastify com Node.js como servidor. No banco de dados o MySQL foi uma escolha acertada tendo em vista a natureza relacional dos dados presentes no aplicativo, além de ser uma ferramenta amplamente utilizada e com muita documentação disponível.

A motivação para o desenvolvimento do aplicativo foram as muitas vezes em que um visitante ou calouro veio nos perguntar onde ficava um determinado prédio ou sala e

na maior parte das vezes não sabíamos responder e não sabíamos onde encontrar aquela informação. Com o Encontre na UFMS esperamos que essas situações sejam menos frequentes e que a comunidade acadêmica possa se beneficiar da ferramenta. No começo a quantidade de locais cadastrados não será grande, mas após um tempo e com o apoio da comunidade acadêmica do campus acreditamos que o aplicativo pode se tornar uma ferramenta muito útil para todos.

As aplicações já mencionadas, Google Maps e Localização UFMS, são ótimas ferramentas mas que não abrangem todas as necessidades da comunidade acadêmica da UFMS. O Google Maps é uma ferramenta muito ampla e que não é focada em um único local, é difícil encontrar diversas localidades importantes nela. Já o Localização UFMS possui quase todas as localidades da UFMS, mas não possui uma interface amigável e intuitiva como o Google Maps e não possui funcionalidades de traçar rotas e navegação, e por não ser colaborativo pode ter informações desatualizadas ou faltantes.

O aplicativo Encontre na UFMS é uma ferramenta que tenta unir o melhor dos dois mundos, com uma interface amigável e intuitiva como o Google Maps e com a colaboratividade e foco na UFMS do Localização UFMS. Acreditamos que o aplicativo pode ser muito útil para a comunidade acadêmica da UFMS e que com o tempo e o apoio da comunidade ele pode trazer uma qualidade de vida maior aos transeuntes do campus.

6. Trabalhos Futuros

Por mais que o aplicativo já seja funcional, vemos que ainda há muito espaço para melhorias e novas funcionalidades. A seguir, listamos algumas ideias para trabalhos futuros:

Requisitos que foram levantados mas não implementados seriam um bom ponto de partida para futuras implementações. Alguns exemplos são a possibilidade de se registrar e entrar no aplicativo através de serviços externos como o Google e o Facebook, o histórico de pesquisa para facilitar a experiência do usuário, a introdução de um modo claro, a adição de opções de acessibilidade mais avançadas e também um recurso para que o usuário possa reportar problemas com o aplicativo.

Seria interessante adicionar uma seção de comentários nos locais, onde os usuários pudessem deixar suas opiniões e avaliações mais detalhadas sobre o funcionamento do local. Isso poderia para serviços como o de restaurantes e lanchonetes, por exemplo.

Por mais que o aplicativo possa ser considerado como pronto para uso, ele ainda não foi lançado oficialmente. Seria importante que fosse disponibilizado para download em lojas de aplicativos, como a Google Play Store, isso também exigiria que o backend fosse hospedado em um servidor para que o aplicativo pudesse ser acessado a qualquer momento.

Atualmente as sugestões de novos locais, assim como as de alterações, são enviadas por email para análise do administrador. Criar um serviço web simples que permita a visualização e aprovação dessas sugestões com maior facilidade e clareza ajudaria muito o processo, o que aceleraria a adição de novos locais e a correção de informações erradas.

Essas melhorias fariam o Encontre na UFMS mais completo e agradável de se usar, o que consolidaria seu uso e popularidade entre os estudantes e visitantes da UFMS.

Referências

[1] Android SDK (Software Development Kit). https://developer.android.com/studio?gad_source=1&gclid=Cj0KCQiAouG5BhDBARIsAOc08RSOXPHcZzYbSZXC0qrXhxH9bZxszJJ6B7LEsW_HMXSKV3uhz3qPYL4aAkQMEALw_wcB&gclsrc=aw.ds&hl=pt-br. Acesso em: 15 nov. 2024.

[2] Dbeaver: Universal database tool. https://dbeaver.io/. Acesso em: 15 nov. 2024.

[3] dio: A powerful HTTP networking package for Dart/Flutter. https://pub.dev/packages/dio. Acesso em: 15 nov. 2024.

[4] Drizzle ORM: A lightweight and performant TypeScript ORM with developer experience in mind. https://orm.drizzle.team/docs/kit-overview. Acesso em: 15 nov. 2024.

[5] Fastify: Fast and low overhead web framework, for Node.js. https://fastify.dev/. Acesso em: 15 nov. 2024.

[6] flutter_bloc: Flutter Widgets that make it easy to implement the BLoC (Business Logic Component) design pattern. https://pub.dev/packages/flutter_bloc. Acesso em: 15 nov. 2024.

[7] google_maps_flutter: A Flutter plugin for integrating Google Maps in iOS and Android applications. https://pub.dev/packages/google_maps_flutter. Acesso em: 15 nov. 2024.

[8] JSON Web Tokens: Are an open, industry standard RFC 7519 method for representing claims securely between two parties. https://jwt.io/. Acesso em: 15 nov. 2024.

[9] Mailjet: Email delivery service for marketing & developer teams. https://www.mailjet.com/. Acesso em: 15 nov. 2024.

[10] MySQL. https://www.mysql.com/. Acesso em: 15 nov. 2024.

[11] Node.js: A JavaScript runtime built on Chrome's V8 JavaScript engine. https://nodejs.org/pt. Acesso em: 15 nov. 2024.

[12] QEMU: A generic and open source machine emulator and virtualizer. https://www.qemu.org/. Acesso em: 15 nov. 2024.

[13] Services. http://www.datadisk.co.uk/html_docs/spring_boot/spring_boot_3.html. Acesso em: 06 dezem. 2024.

[14] tsup: Bundle your TypeScript library with no config, powered by esbuild. https://github.com/egoist/tsup. Acesso em: 15 nov. 2024.

[15] TypeScript: JavaScript With Syntax For Types. https://www.typescriptlang.org/. Acesso em: 15 nov. 2024.

[16] Filipe dos Santos Pires. Frontend encontre na ufms. https://github.com/FilipePires17/Encontre-na-UFMS-front. Acesso em: 15 nov. 2024.

[17] Filipe dos Santos Pires e Wilson Eduardo F. D. de Almeida. Documento de Requisitos Aplicativo Encontre na UFMS. https://docs.google.com/document/

d/1-Idc2ILVyVy7u2HpjPZLY4AKSEJ2tRMohcmn3i-kMpc/edit?usp=sharing. Acesso em: 15 nov. 2024.

[18] Google LLC. Dart: An approachable, portable, and productive language for high-quality apps on any platform. https://dart.dev/. Acesso em: 15 nov. 2024.

[19] Google LLC. Flutter: Transforms the entire app development process. Build, test, and deploy beautiful mobile, web, desktop, and embedded apps from a single codebase. https://flutter.dev/. Acesso em: 15 nov. 2024.

[20] Google LLC. Google Maps. https://play.google.com/store/apps/details?id=com.google.android.apps.maps&pcampaignid=web_share, 2005. Acesso em: 15 nov. 2024. Avaliação na Play Store: 4.3 +10bi de Downloads.

[21] Microsoft. Visual Studio Code: Code editor redefined and optimized for building and debugging modern web and cloud applications. https://code.visualstudio.com/. Acesso em: 15 nov. 2024.

[22] Semih Altin. Flutter-clean architecture. https://medium.com/@semihaltin99/flutter-clean-architecture-8759ad0213dd, 2024. Acesso em: 15 nov. 2024.

[23] Simon Choi. hive. https://github.com/isar/hive. Acesso em: 15 nov. 2024.

[24] Universidade Federal de Mato Grosso do Sul. Localização UFMS. https://agendamentos.ufms.br/localizacao. Acesso em: 15 nov. 2024.

[25] Wilson Eduardo F. D. de Almeida. Backend encontre na ufms. https://github.com/wildevn/EncontreNaUfmsAPI. Acesso em: 15 nov. 2024.