UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Rodrigo de Castro Cardoso

Aplicativo móvel para registro e compartilhamento de mídias usando mapa interativo

Uberlândia, Brasil

2025

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Rodrigo de Castro Cardoso

Aplicativo móvel para registro e compartilhamento de mídias usando mapa interativo

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, como parte dos requisitos exigidos para a obtenção título de Bacharel em Sistemas de Informação.

Orientador: Maria Adriana Vidigal de Lima

Universidade Federal de Uberlândia – UFU

Faculdade de Computação

Bacharelado em Sistemas de Informação

Uberlândia, Brasil

2025

Rodrigo de Castro Cardoso

Aplicativo móvel para registro e compartilhamento de mídias usando mapa interativo

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, como parte dos requisitos exigidos para a obtenção título de Bacharel em Sistemas de Informação.

Trabalho aprovado. Uberlândia, Brasil, 15 de maio de 2025:

Maria Adriana Vidigal de Lima
Orientadora

Claudiney Ramos Tinoco

Luiz Gustavo Almeida Martins

Uberlândia, Brasil
2025

Resumo

Este trabalho apresenta o desenvolvimento de um aplicativo para dispositivos com o sistema operacional Android, focado no registro e organização de memórias pessoais associadas a locais geográficos. A aplicação permite aos usuários criar marcadores interativos no mapa, associando fotos e anotações aos locais escolhidos, além de compartilhar seletivamente essas memórias com amigos adicionados. Para o armazenamento de dados e mídias, foram utilizadas tecnologias como Firebase Realtime Database e Firebase Storage, garantindo escalabilidade e sincronização em tempo real. O trabalho seguiu a metodologia ágil Kanban e utilizou o padrão arquitetural MVVM para promover a modularidade e manutenção do código. Como resultado prático, foram implementadas 12 telas com funcionalidades como autenticação, integração com Google Maps, manipulação de marcadores, salvamento de mídias na nuvem e interação com outros usuários por meio de listas de amigos, todas sincronizadas em tempo real utilizando o Firebase.

Palavras-chave: Android. Firebase. Mapas Interativos. Desenvolvimento Mobile. Registro de Mídias.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

API	Application Programming Interface
IDE	Integrated Development Environment
ISO	International Organization for Standardization
MVVM	Model-View-ViewModel
SDK	Software Development Kit

Sumário

1 Introdução

O avanço tecnológico nas últimas décadas tem transformado a forma como as pessoas lidam com a coleta e armazenamento de memórias. Se antigamente era necessário uma câmera e rolos de filme para gerar fotos que seriam guardadas em álbuns, com o crescimento do uso de dispositivos móveis como *smartphones*, a prática de capturar momentos por meio de fotos e vídeos se tornou uma atividade quase diária. Esse contexto de facilidade na captura de momentos abre possibilidades para o desenvolvimento de soluções que possibilitam a organização dessas memórias, associando mídias ou textos a determinados locais visitados de forma prática e intuitiva, como a aplicação proposta neste trabalho.

Estima-se que existam cerca de 3 bilhões de dispositivos móveis com o sistema operacional Android ativos, o que representa cerca de 75% dos usuários totais de *smartphones* (TEAM, 2024). Nesse contexto, Rossi (2019) destaca a importância do uso de *smartphones* para turistas e viajantes, que formam o público-alvo do trabalho. Durante o levantamento de referências para esse trabalho, não encontramos uma ferramenta robusta e moderna e flexível para a solução do que foi proposto.

A implementação de mapas interativos e personalizados em aplicações vem se mostrando útil por sua facilidade de interpretação, pois respondem à atividade do usuário na tela do dispositivo. Propostas como as de Rane, John e Murthy (2020) demonstram que a adoção de aplicações com mapas interativos pode auxiliar no entendimento geográfico de alunos, graças à sua fácil visualização.

O objetivo geral deste trabalho é projetar e implementar um aplicativo nativo para dispositivos com o sistema operacional Android que:

1. Apresente um mapa-múndi interativo;
2. Permita ao usuário criar *marcadores* em qualquer coordenada geográfica na área correspondente a superfície de um país;
3. Associe fotos e anotações a esses pontos;
4. Armazene tais mídias, de forma escalonável, na nuvem por meio do *Firebase Storage*;
5. Viabilize o compartilhamento desses dados, controlado por um sistema de amizade com outros usuários.

Ao integrar essas funcionalidades, busca-se transformar o registro de memórias de lugares em uma experiência prática e visualmente rica. Para atingir esse objetivo, foram definidas as seguintes etapas:

1. **Revisão Bibliográfica**: levantamento de artigos, dissertações e aplicações correlatas.
2. **Análise de Mercado**: estudo crítico de soluções já existentes.
3. **Referencial Teórico**: estudo de padrões arquiteturais, metodologias ágeis e ferramentas (Kotlin, MVVM, Google Maps SDK, Firebase).
4. **Implementação do Protótipo Funcional** com as seguintes atividades:
   a) disponibilizar o mapa interativo com personalização cromática de camadas;
   b) permitir criação/edição de marcadores e associação de mídias;
   c) integrar autenticação segura de usuários;
   d) armazenar dados e arquivos em nuvem (Realtime Database & Storage);
   e) implementar o sistema de amizades (envio, aceite ou recusa de solicitações);
   f) habilitar o compartilhamento seletivo de mapas para amigos;
   g) otimizar o desempenho para diferentes versões e tamanhos de dispositivo.

1.1 Organização do trabalho

Este trabalho foi desenvolvido em 5 capítulos, conforme descrito a seguir:

1. **Capítulo 1 - Introdução**: apresenta a contextualização do problema, os objetivos geral e específicos, e a justificativa para o desenvolvimento do aplicativo proposto.
2. **Capítulo 2 - Fundamentação teórica**: descreve os conceitos e tecnologias utilizados no desenvolvimento da aplicação, incluindo o sistema operacional Android, a linguagem Kotlin, o SDK Google Maps e os serviços do Firebase.
3. **Capítulo 3 - Trabalhos Correlatos**: apresenta uma analise comparativa entre a aplicação desenvolvida neste trabalho e outras soluções existentes, destacando suas funcionalidades, semelhanças e diferenças.
4. **Capítulo 4 - Desenvolvimento**: detalha todas as etapas do desenvolvimento da aplicação, incluindo a definição de requisitos, modelagem do banco de dados, e interface das telas implementadas.
5. **Capítulo 5 - Conclusão**: apresenta os resultados alcançados, aponta as limitações da solução e sugere direções para trabalhos futuros.

2 Fundamentação teórica

No desenvolvimento do trabalho proposto, serão utilizadas diversas ferramentas, cada uma cobrindo requisitos centrais. Foi escolhida a linguagem de programação *Kotlin*, linguagem recomendada para o desenvolvimento de aplicações nativas na plataforma Android (KOLTINLANG, 2024). A execução ocorre sobre o *Android SDK*, utilizando o ambiente de desenvolvimento integrado *Android Studio*, IDE específica e recomendada para o desenvolvimento de aplicações Android.

Para a disponibilização do mapa interativo, foi escolhido o *SDK do Google Maps Android*, fornecido pelo Google e que permite a visualização e customização do mapa.

No que tange ao armazenamento dos dados, serão utilizados o *Firebase Realtime Database*, e para o armazenamento das mídias o *Firebase Storage*. Ambos oferecem escalabilidade, sincronização automática e APIs que podem ser facilmente integradas em projetos com Android. O *Firebase Authentication* fica responsável pela segurança e o gerenciamento de sessões.

A organização do código segue o padrão de arquitetura *MVVM (Model-View-ViewModel)* que permite boa separação de responsabilidade e desacoplamento do código.

As subseções a seguir apresentam, em maior detalhe cada uma dessas tecnologias e sua utilidade na solução.

2.1 Android

O Android é um sistema operacional de código aberto baseado no *Kernel* Linux, e projetado especificamente para a utilização em dispositivos móveis. Seu desenvolvimento foi iniciado pela empresa Android Inc. que posteriormente foi adquirida pelo Google para a continuidade do desenvolvimento e lançamento em 2007. Atualmente é mantido pela Open Handset Alliance (OHA), uma cooperação entre grandes empresas do ramo de tecnologia e comunicação lideradas pelo Google. Essa cooperação tem por objetivo a criação de uma plataforma flexível para o desenvolvimento de aplicações e ampla gama de recursos oferecidos aos desenvolvedores e usuários finais (Developers (2024b)). Uma vantagem da plataforma é possuir o código aberto, o que permite que cada fabricante possa fazer alterações para customizar seus produtos como quiser, além de permitir contribuições da comunidade de desenvolvedores, garantindo assim a constante evolução da plataforma, o que acelerou a adoção mundial do sistema operacional Android.

A arquitetura da plataforma Android é organizada em camadas Developers (2024a), o que facilita a modularização e personalização do sistema. As camadas da plataforma

retratadas na Figura 1 incluem:

- *Kernel* Linux: A base da plataforma Android, fornece funcionalidades essenciais como o gerenciamento de processos, controle e gerenciamento de memoria e *drivers* para interação com hardware. Além disso, a utilização do *Kernel* Linux permite que o Android utilize seus recursos de segurança;
- Camada de abstração de hardware (HAL): Essa camada fornece interfaces padrão para que o software possa interagir diretamente com o hardware do dispositivo, que permite que os fabricantes implementem funcionalidades especificas como câmeras ou módulos Bluetooth;
- Android Runtime (ART): É o gerenciador de execução utilizado pelo Android para executar aplicativos através de arquivos DEX;
- Bibliotecas C/C++ nativas: Diversos componentes do sistema Android como o ART e HAL são implementados em C/C++. Os aplicativos desenvolvidos podem acessar essas bibliotecas nativas no código através do Android NDK;
- Java/Kotlin API: Essa camada inclui todas as APIs necessárias para criar aplicativos Android. Essas APIs são programadas em Java e Kotlin e oferecem funcionalidades para acessar os recursos do sistema operacional Android. Os desenvolvedores então conseguem utilizar os recursos do SO de forma facilitada;
- Aplicativos do sistema: O Android possui uma série de aplicativos básicos, como câmera, navegador, envio de SMS, realização de chamadas telefônicas e outros. Os aplicativos do sistema fornecem funcionalidades principais e recursos que podem ser acessados por um desenvolvedor de um aplicativo.

O SO Android vem acompanhando as tendências tecnológicas recentes e continua a se adaptar às demandas do mercado. Com a crescente utilização de dispositivos inteligentes e a Internet das Coisas (IoT), o Android expandiu seu ecossistema para incluir o suporte a *wearables* (relógios e pulseiras inteligentes), *Smart* TVs e até mesmo multimídias de automóveis através do Android Auto.

2.2 Kotlin

O Kotlin é uma linguagem de programação estaticamente tipada, de código aberto desenvolvida pela JetBrains. Seu intuito é ser uma linguagem concisa, segura e mais expressiva do que o Java. O Kotlin permite aos desenvolvedores escrever menos código para realizar as mesmas tarefas escritas em Java, reduzindo tanto a complexidade quanto a possibilidade de erros. A linguagem foi projetada para ser totalmente interoperável com

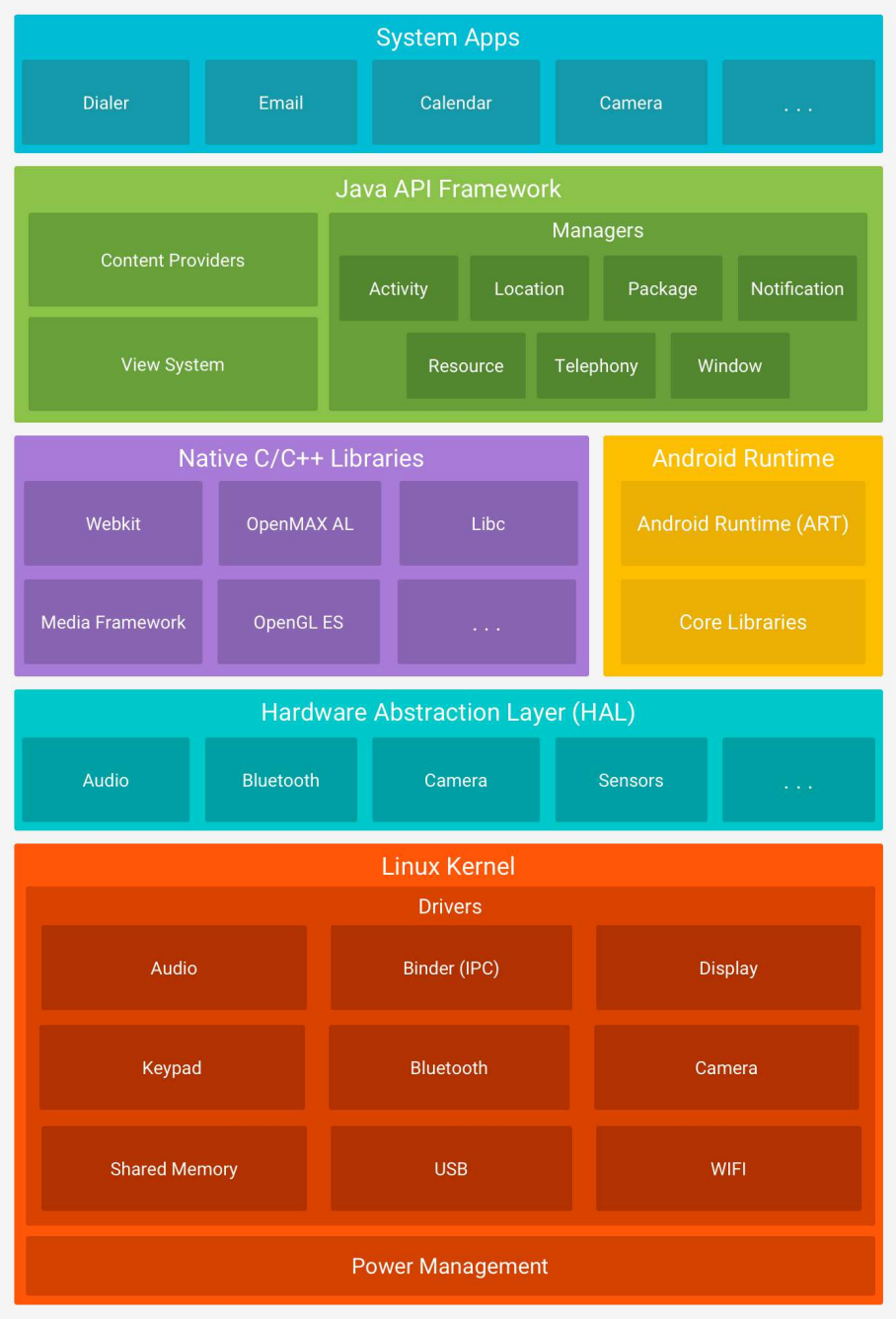


Figura 1 – Pilha de software do Android.
Fonte: Extraído de (DEVELOPERS, 2024a)

Java, facilitando assim a transição de aplicativos já desenvolvidos em Java para Kotlin, visto que desde 2019, o Kotlin foi adotado como linguagem padrão para o desenvolvimento de aplicativos na plataforma Android, o que trouxe grande popularidade à linguagem (KOLTINLANG, 2024). Alguns dos recursos mais importantes do Kotlin são:

- *Null Safety* (Segurança contra nulos): O Kotlin possui um tratamento de segurança contra valores nulos, evitando erros em tempo de execução;
- Interoperabilidade total com Java: Essa característica da linguagem permite a utilização de código escrito em Kotlin em projetos escritos em Java e vice-versa, o que permite uma adoção gradual e a utilização de bibliotecas já existentes em Java;
- Sintaxe clara e concisa: Permite que desenvolvedores escrevam menos código para realizar as mesmas tarefas que fariam em Java, o que torna o código mais legível e de fácil manutenção.

2.3 MVVM

O MVVM é um padrão de arquitetura de software que visa a separação de responsabilidades no desenvolvimento das aplicações, principalmente em plataformas como o Android. Foi projetado para desacoplar a lógica de negócios da interface do usuário, o que proporciona um código mais organizado, testável e de fácil manutenção (GAMMA, 2009). O MVVM é composto por 3 camadas com responsabilidades distintas, sendo elas:

- Model: Representa a camada de dados e lógica de negócios. Essa camada é responsável pela obtenção, manipulação e processamento dos dados utilizados na aplicação, podendo envolver requisições em APIs externas ou consultas em bancos de dados remotos ou locais.
- View: É a camada que se refere a interface do usuário. É responsável por exibir os dados ao usuário e registrar interações como toques ou cliques.
- View Model: A camada de View Model atua como um intermediário entre a View e o Model, recebendo os dados da Model e os preparando para serem exibidos na View. Além disso, fica responsável por reagir a eventos de interação do usuário, notificando a camada de Model caso necessário.

O trabalho de Lou et al. (2016) realiza uma análise abrangente para determinar se as arquiteturas MVP e MVVM superam a arquitetura MVC (Model-View-Controller) em termos de qualidade. Foi adotado o Método de Análise de *Trade-off* de Arquitetura com os critérios: testabilidade, modificabilidade e desempenho. A análise e os experimentos demonstraram que MVP e MVVM apresentam melhor testabilidade, modificabilidade

(com um nível baixo de acoplamento) e desempenho (consumindo menos memória). O padrão MVC propõe a separação da aplicação em três componentes principais: Model, responsável pela lógica de negócios e manipulação de dados; View, que exibe as informações ao usuário; e Controller, que atua como intermediário, processando as entradas do usuário e atualizando a Model ou a View conforme necessário. Contudo, na prática, especialmente em ambientes como o Android, o Controller acaba frequentemente se misturando com a View, resultando em um forte acoplamento entre interface e lógica, o que dificulta a manutenção e a testabilidade do sistema. O padrão MVVM aprimora essa separação ao introduzir a camada de ViewModel. O ViewModel é responsável por gerenciar o estado e a lógica de apresentação dos dados, preparando-os para exibição na interface e reagindo aos eventos disparados pelo usuário, enquanto a View se limita exclusivamente à exibição e encaminhamento dessas ações. Portanto, enquanto o MVC tende a ter uma separação de camadas mais teórica, mas menos efetiva na prática, o MVVM oferece uma estrutura mais clara e organizada, favorecendo atributos importantes como testabilidade, modificabilidade e manutenção do código, conforme será apresentado na Tabela 1.

Tabela 1 – Resumo comparativo entre MVC, MVP e MVVM

Aspecto	**MVC**	**MVP**	**MVVM**
Separação de camadas	Baixa	Boa (Presenter)	Muito boa (ViewModel + DataBinding)
Testabilidade	Limitada	Alta	Alta
Modificabilidade	Menor (acoplado)	Maior	Maior, mas *bindings* adicionam dependência
Uso de memória	Maior	Menor em comparação ao MVC	Similar ao MVP

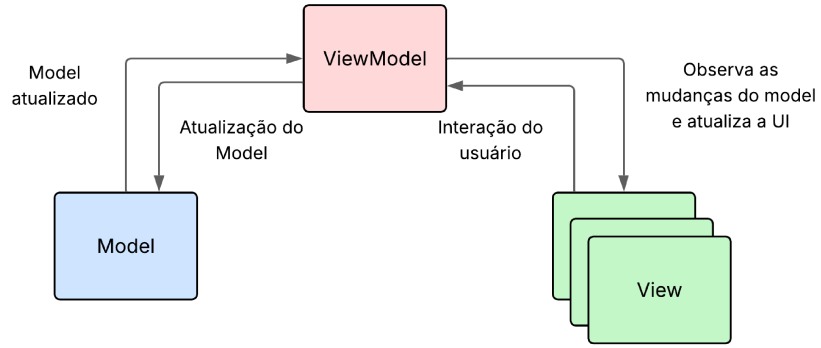


Figura 2 – Representação das camadas da arquitetura MVVM.
Fonte: Adaptado de Souza (2024).

2.4 SDK Google Maps Android

O SDK (*Software Development Kit*) do Google Maps para Android é a biblioteca que permite aos desenvolvedores integrar funcionalidades de mapas em seus aplicativos Android. Esse SDK fornece diversas funcionalidades e ferramentas, como:

- Visualização e personalização de mapas;
- Funcionalidades de navegação;
- Adição de marcadores e polígonos no mapa;
- Suporte para mapas e navegação offline;
- Funcionalidade para criar camadas a partir de GeoJSON.

O SDK é parte do Google Play Services, o que garante atualizações constantes para as novas funcionalidades e suporte a novas versões do SO Android (GOOGLE, 2024b). Com os avanços na indústria de dispositivos móveis, também está disponível a sua utilização em dispositivos *wearables* (vestíveis), como pulseiras ou relógios inteligentes, através do Wear OS, e também em automóveis com o Android Auto.

2.5 Firebase

O *Firebase* é uma plataforma *Backend as a Service (BaaS)* desenvolvida e mantida pelo Google que oferece uma série de ferramentas e serviços para o desenvolvimento de aplicações móveis e *web*, com *SDKs* de fácil integração em projetos Android Google (2024a). As três ferramentas que serão utilizadas neste trabalho são:

2.5.1 Firebase Authentication

Responsável pela autenticação de usuários, abstraindo toda a complexidade de armazenar credenciais e lidar com fluxos de segurança. Esse SDK fornece:

- Autenticação por e-mail/senha, número de telefone, ou provedores externos como Google, Facebook e Apple;
- Gerenciamento automático de sessão;
- Regras de segurança integradas ao banco de dados e ao *Storage*, permitindo a criação de condições de acesso ao banco como por exemplo somente o “dono” do nó pode ler ou escrever nele.

2.5.2 Firebase Realtime Database

É um banco de dados **NoSQL orientado a documentos JSON**, hospedado na nuvem e sincronizado em tempo real com todos os clientes conectados. Em seu funcionamento geral, os dados são armazenados em uma única árvore JSON. O SDK também gerencia *cache* local e fila de eventos *offline*, possibilitando operações em modo desconectado.

Dentre as vantagens sobre bancos não relacionais estão:

- **Baixa latência em tempo real**: mudanças propagam-se em milissegundos para todos os dispositivos.
- **Modelo flexível**: não exige esquema fixo; perfeito para iterações rápidas de produto.
- **Escalabilidade horizontal**: *shard* automático feito pelo provedor, eliminando manutenção de servidores.
- **Autorização declarativa**: regras de segurança escritas em *JSON-like* são avaliadas no próprio servidor, reduzindo código no cliente.

Por essas características, sua fácil integração e criação das operações, o Realtime Database foi escolhido para este trabalho, que demanda sincronização instantânea de países, marcadores e estados de amizade entre diversos usuários.

2.5.3 Firebase Storage

O *Firebase Storage* baseia-se no *Google Cloud Storage* e oferece um serviço de **armazenamento binário escalável**, acessível via URL assinado. Seus principais recursos utilizados são:

- ***Upload* resiliente**: o SDK retoma automaticamente transferências interrompidas (rede instável).
- **Metadados customizados**: tipo MIME, dimensões de imagem, *timestamp*, etc.
- **Regras de segurança**: podem referenciar o ID do usuário autenticado ou valores no Realtime Database para permitir, por exemplo, que apenas amigos vejam determinadas mídias.

A combinação *Authentication + Realtime Database + Storage* entrega o conjunto mínimo necessário de *backend* para o aplicativo: cadastro seguro, sincronização em tempo real dos dados estruturados e hospedagem de fotos/vídeos.

3 Trabalhos Correlatos

Este capítulo apresenta uma análise comparativa de soluções existentes voltadas para o registro de informações geográficas e memórias associadas a lugares, com o objetivo de contextualizar o desenvolvimento do aplicativo proposto neste trabalho. A comparação considera funcionalidades, propósitos e tecnologias utilizadas, destacando pontos de convergência e diferenciação em relação a este trabalho.

O primeiro exemplo analisado é o Trip Tracker, um aplicativo Android proposto em 2011 (REDDY, 2011). Ele permite aos usuários criar diários de viagem com base na localização obtida por GPS, associando imagens e anotações a pontos específicos. A visualização dos registros é feita por meio de marcadores em um mapa, com todos os dados armazenados localmente no dispositivo. O Trip Tracker se assemelha a este trabalho ao permitir o registro de memórias vinculadas a localizações geográficas e a visualização interativa em mapas. O aplicativo não esta disponível para o público se tratando apenas de um trabalho de pesquisa. Suas funcionalidades principais são:

- Criação de diários de viagem;
- Registro de localizações por GPS;
- Associação de imagens e notas a locais específicos;
- Exibição de viagens no mapa por meio de marcadores;
- Armazenamento local dos dados no dispositivo.

Entre os aspectos que se aproximam deste trabalho, destacam-se:

- Registro de memórias associadas a localizações geográficas;
- Exibição visual em mapas;
- Permite adicionar descrições e fotos aos pontos registrados.

Apesar das similaridades observadas, o Trip Tracker apresenta diferenças significativas em termos de arquitetura, recursos oferecidos e forma de armazenamento de dados, que o distinguem da solução proposta neste trabalho:

- Utiliza armazenamento local, enquanto este trabalho utiliza nuvem;
- Não possui funcionalidades sociais como sistema de amizade;

- Arquitetura de código mais antiga, sem padrões modernos como MVVM.

Outra solução relevante é o GeoMaps, um aplicativo com foco educacional, criado para auxiliar estudantes no desenvolvimento de habilidades de interpretação cartográfica, conforme descrito em (RANE; JOHN; MURTHY, 2020). Ele permite a visualização de múltiplas camadas de mapas simultaneamente, oferece *feedback* imediato ao usuário e está alinhado ao currículo escolar. Embora tenha um propósito diferente do trabalho aqui proposto, o GeoMaps compartilha a utilização de mapas interativos e incentiva a interação com o conteúdo geográfico. Contudo, sua aplicação não contempla o registro de imagens ou memórias pessoais, sendo voltada exclusivamente ao ensino de geografia. O aplicativo GeoMaps não esta disponível para o público se tratando apenas de um trabalho de pesquisa. Suas funcionalidades principais são:

- Uso de múltiplas camadas de mapas simultaneamente;
- Foco em resolver problemas geográficos reais;
- *Feedback* imediato e suporte a autoaprendizagem;
- Alinhamento ao currículo escolar.

Considerando os aspectos semelhantes a este trabalho, destacam-se:

- Utiliza mapas interativos;
- Permite interação direta com o mapa;
- Incentiva o enriquecimento da experiência do usuário.

Sobre as diferenças, podem-se listar:

- Foco educacional, enquanto este trabalho é voltado para o registro pessoal de memórias;
- Não realiza registro de fotos.

Já o Polarsteps representa uma abordagem mais moderna e comercial para anotações em viagens. Disponível gratuitamente para Android e iOS (POLARSTEPS, 2025) em suas respectivas lojas de aplicativos oficiais, o aplicativo registra automaticamente os trajetos percorridos pelo usuário, permitindo inclusão de fotos, vídeos e descrições aos locais visitados. Além disso, possibilita o compartilhamento de viagens com controle de privacidade e oferece recursos extras, como a geração de álbuns e livros de viagem impressos. Assim como este trabalho, o Polarsteps foca na experiência do usuário, utiliza mapas para visualização dos trajetos. Podem-se listar as seguintes funcionalidades principais:

- Rastreamento automático de viagens;
- Registro de fotos, vídeos e descrições;
- Criação de álbuns e livros de viagem;
- Compartilhamento com privacidade controlada.

Em relação a este trabalho, podem ser identificadas as seguintes semelhanças:

- Registro de viagens com mídias;
- Uso de mapas para visualização dos trajetos;
- Foco na experiencia do usuário.

Diferentemente deste trabalho, o PolarStep permite:

- Rastreamento baseado em trajeto percorrido utilizando a localização do dispositivo;
- A criação de marcadores pode ser feita apenas nos trajetos percorridos.

A Tabela 2 resume as características das soluções analisadas, facilitando a visualização comparativa em relação ao trabalho desenvolvido. A partir dessa análise, observa-se que este trabalho se inspira em elementos presentes em soluções consolidadas, como o uso de mapas interativos e o registro de memórias multimídia, ao mesmo tempo em que propõe diferenciais, como o uso de tecnologias modernas, arquitetura baseada em padrões de desenvolvimento atuais e integração com serviços em nuvem, oferecendo uma solução personalizada para o registro e compartilhamento de experiências pessoais de viagem.

Característica	**Trip Tracker (2011)**	**GeoMaps (2020)**	**Polarsteps (2016)**	**Este trabalho (2025)**
Plataforma	Android	Web-based	Android/iOS	Android
Armazenamento	Local	Nenhum	Nuvem própria	Firebase
Adiciona fotos e descrições	Sim	Não	Sim	Sim
Compartilhamento	Não	Não	Sim	Sim
Foco principal	Registro de viagens	Ensino de geografia	Registro de viagens	Registro de memórias pessoais

Tabela 2 – Resumo comparativo entre as soluções analisadas.
Fonte: Do autor.

4 Desenvolvimento

Este capítulo apresenta como o desenvolvimento da aplicação foi estruturado, desde a análise de requisitos, modelagem do banco de dados e a construção e fluxo das telas, seguindo a metodologia Kanban.

4.1 Requisitos

Com base na análise dos requisitos levantados e representados na Figura 3 , foi possível identificar as principais funcionalidades que deverão ser implementadas para atender aos objetivos propostos. A partir dessa definição, elaborou-se um *backlog* inicial contendo as tarefas essenciais para o desenvolvimento da aplicação, conforme descritas a seguir:

1. Criação do projeto
2. Integração com o Firebase
3. Criação do modelo de dados
4. Implementação da tela de login
5. Implementação da tela de cadastro
6. Teste da autenticação no Firebase Authentication
7. Integração com o SDK Google Maps
8. Implementação da tela de mapas
9. Implementação das funcionalidades de salvamento de país no banco
10. Implementação do Dialog de detalhes do país
11. Implementação das funcionalidades de criar marcadores no banco
12. Implementação do Dialog de detalhes do marcador
13. Implementação da funcionalidade de upload de mídias para o Firebase Storage
14. Teste da funcionalidade de upload
15. Implementação da tela de listagem de lugares
16. Implementação da tela de detalhes de país

17. Implementação da tela de detalhes do marcador
18. Implementação da tela de listagem de amigos
19. Implementação das funcionalidades relacionadas a solicitação de amizade
20. Implementação do acesso ao mapa de outros usuários adicionados
21. Implementação da tela de perfil

A definição dos requisitos foi um primeiro passo importante para estabelecer as funcionalidades principais da aplicação de registro de memórias de viagem, como o uso de mapas interativos, a associação de mídias a locais visitados e a possibilidade de compartilhamento com outros usuários. Esses requisitos orientaram a estrutura do trabalho, desde a modelagem dos dados até as escolhas tecnológicas de *frontend* e *backend*. Ao longo do trabalho, cada requisito foi abordado de forma integrada ao processo de desenvolvimento, garantindo que a aplicação pudesse atender às expectativas de usabilidade, desempenho e propósito social propostos desde sua concepção.

4.2 Metodologia Kanban

Para guiar o desenvolvimento deste trabalho e a obtenção dos objetivos propostos, será adotada uma metodologia ágil. Optou-se pelo uso do Kanban, devido à sua simplicidade e capacidade de organização visual das tarefas. As atividades previamente definidas no *backlog* são distribuídas entre as colunas do quadro: “TO DO”, “IN PROGRESS”, “VALIDATION” e “DONE”, representando as etapas do ciclo de desenvolvimento e testes das funcionalidades.

As tarefas têm início na coluna “TO DO” e, ao serem selecionadas para desenvolvimento, são movidas para “IN PROGRESS”. Após a implementação, a tarefa é transferida para “VALIDATION”, onde são realizados os testes e a verificação dos requisitos. Uma vez validada, a tarefa é considerada concluída e é finalmente movida para a coluna “DONE”. O quadro com as colunas e tarefas está representado na Figura 4.

4.3 Modelagem do banco

Como especificado na fundamentação teórica, foi escolhido o uso do Firebase Realtime Database. Essa solução fornece um banco de dados não relacional orientado a objetos do tipo JSON.

O banco de dados foi modelado de forma a garantir eficiência em operações de leitura e escrita, especialmente considerando a escalabilidade necessária para o crescimento

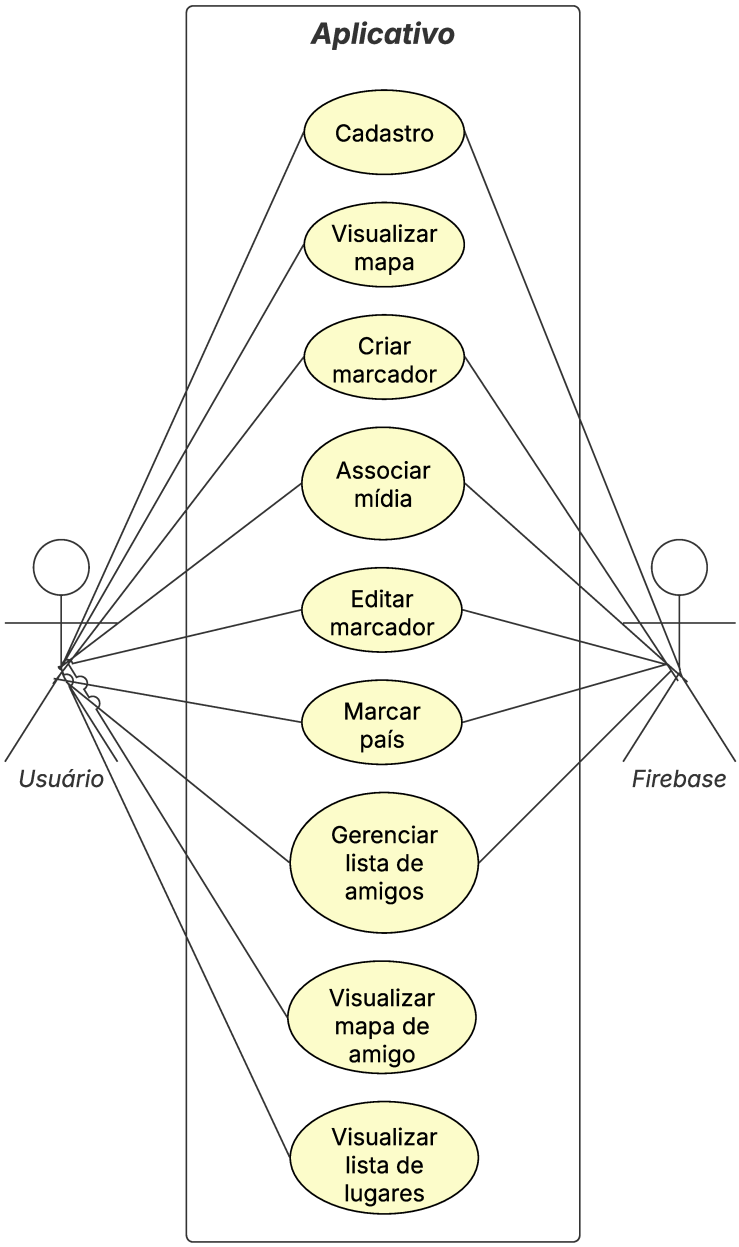


Figura 3 – Diagrama de casos de uso.
Fonte: Do autor.

do número de usuários. Dois nós principais foram definidos na raiz: o nó `/nicknames`, que permite buscas rápidas para garantir que não existam dois usuários com o mesmo apelido, e o nó `/users`, contendo dados de perfil, marcadores, países visitados e lista de amigos. Essa estrutura visa otimizar consultas e permitir expansões futuras sem necessidade de uma reestruturação significativa. Essa representação pode ser vista na Figura 5.

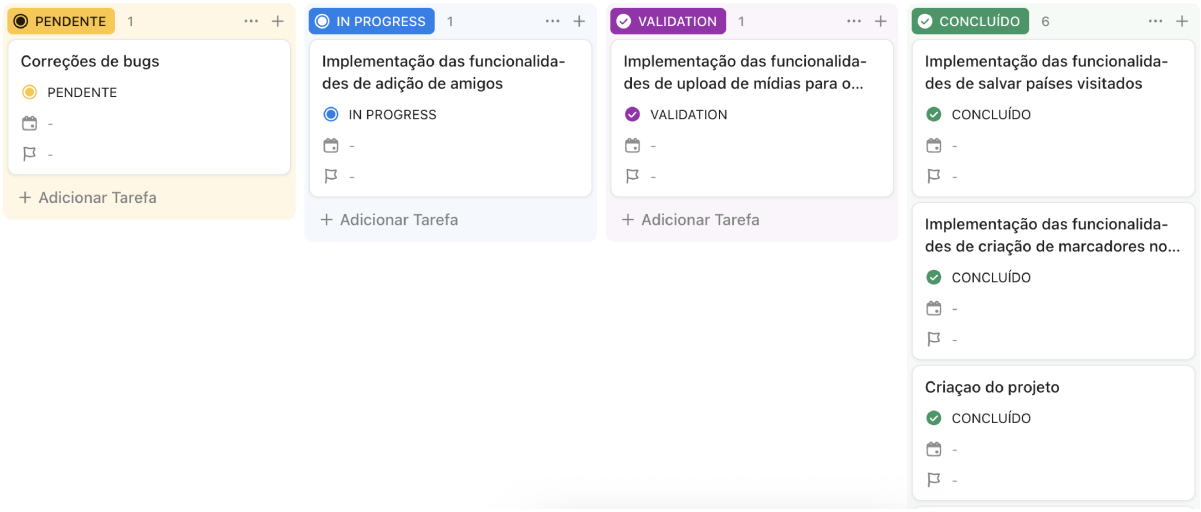


Figura 4 – Quadro Kanban.
Fonte: Do autor.

4.4 Fluxo de telas

Nesta seção, estão representadas todas as tela presentes no aplicativo, detalhando sua interface e ações possíveis.

4.4.1 Tela inicial/login

Na tela inicial do aplicativo, representada na Figura 6, é apresentada a logo do aplicativo, um título, subtítulo, campos para preencher e-mail e senha, e dois botões, um para efetuar o login e outro para redirecionamento para a tela de cadastro. Os campos de e-mail e senha possuem validações para que forcem o usuário a digitar corretamente. As validações incluem:

- Existência do caractere '@' no campo de e-mail;
- Se o tamanho da senha é de no mínimo oito caracteres;

4.4.2 Tela de cadastro

Na tela de cadastro vista na Figura 7, são exibidos campos para que o usuário preencha as informações, além de um botão para confirmar o cadastro. As informações pedidas nos campos dessa tela incluem:

- Nome
- Nickname do usuário
- E-mail

Figura 5 – Estrutura do banco não relacional ilustrando os nós principais.
Fonte: Do autor.

- Senha
- Confirmação de senha

As mesmas validações feitas na tela anterior se aplicam nesta também, além da validação para verificar se as senhas digitadas são iguais e se os campos de nome e nickname não estão vazios. Após o clique no botão e as validações feitas, as credenciais do usuário são adicionadas na base do Firebase Authentication e seus dados são adicionados ao banco de dados e, em seguida, o usuário é redirecionado para a tela de mapa.

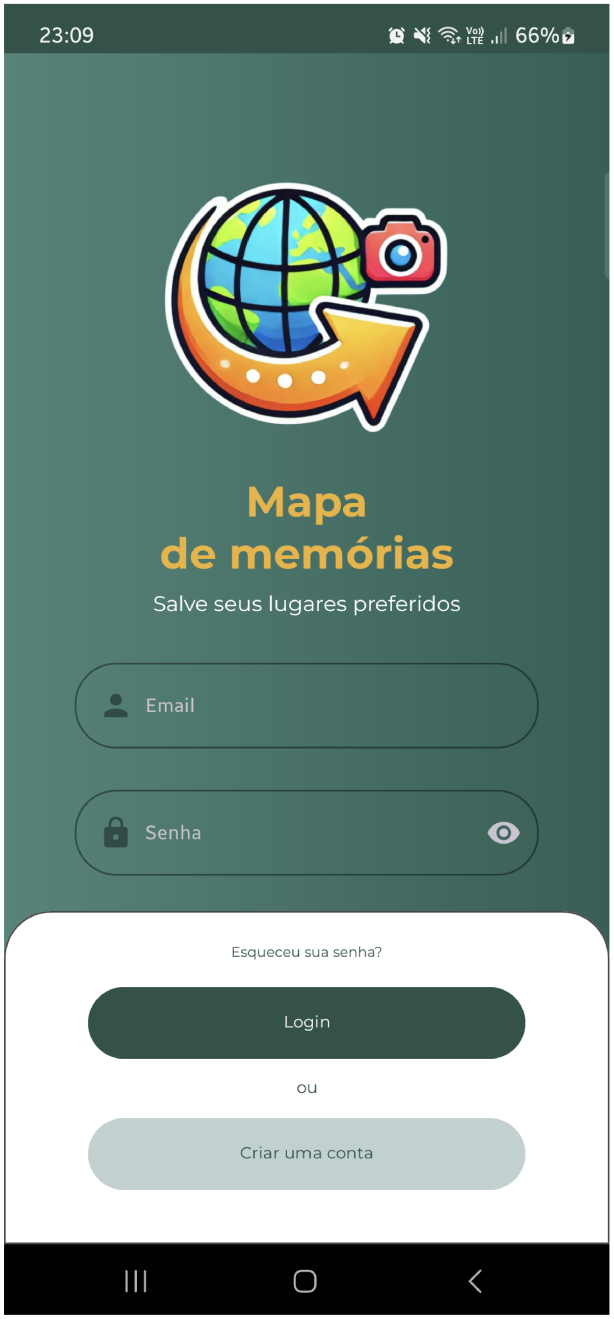


Figura 6 – Tela inicial/login.
Fonte: Do autor.

4.4.3 Tela de mapa

Tela principal do aplicativo, em que se exibe o mapa (Figura 8). No canto superior direito encontra-se o botão "Minha Localização", que quando acionado permite que o usuário centralize o mapa em sua posição atual mediante o aceite da permissão de acesso à localização do dispositivo. Na parte inferior da tela esta a barra de navegação principal do aplicativo, onde é possível navegar para outras três telas, sendo elas "Lugares", "Amigos" e "Perfil" respectivamente. Nessa tela de mapa, é possível realizar as seguintes interações com o mapa:

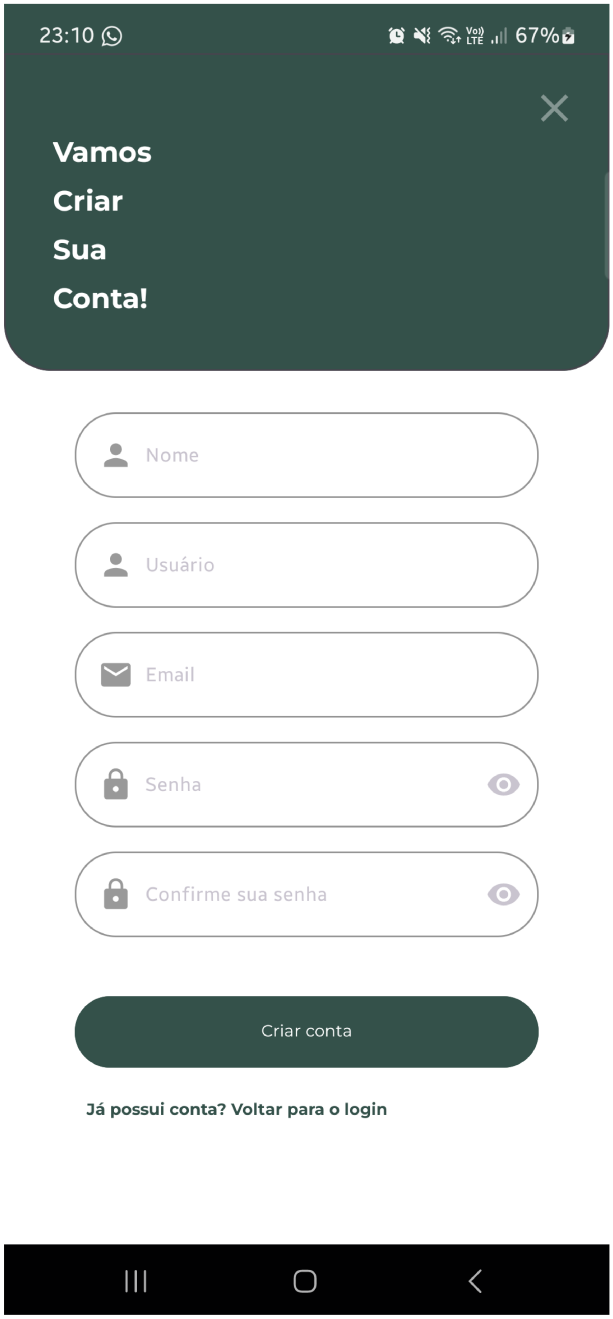


Figura 7 – Tela de cadastro.
Fonte: Do autor.

- Clique rápido na área de um país. Essa interação realiza a abertura de um Bottom Sheet Dialog do país, descrito na subseção 4.4.4.
- Clique longo na área de algum país. Essa interação realiza a abertura do Dialog para criação de um marcador, descrito no subseção 4.4.5.
- Clique em um marcador existente. Essa interação realiza a abertura de um Bottom Sheet Dialog de informações do marcador, descrito no subseção 4.4.6.

Figura 8 – Tela de mapa.
Fonte: Do autor.

4.4.4 Bottom Sheet Dialog de país

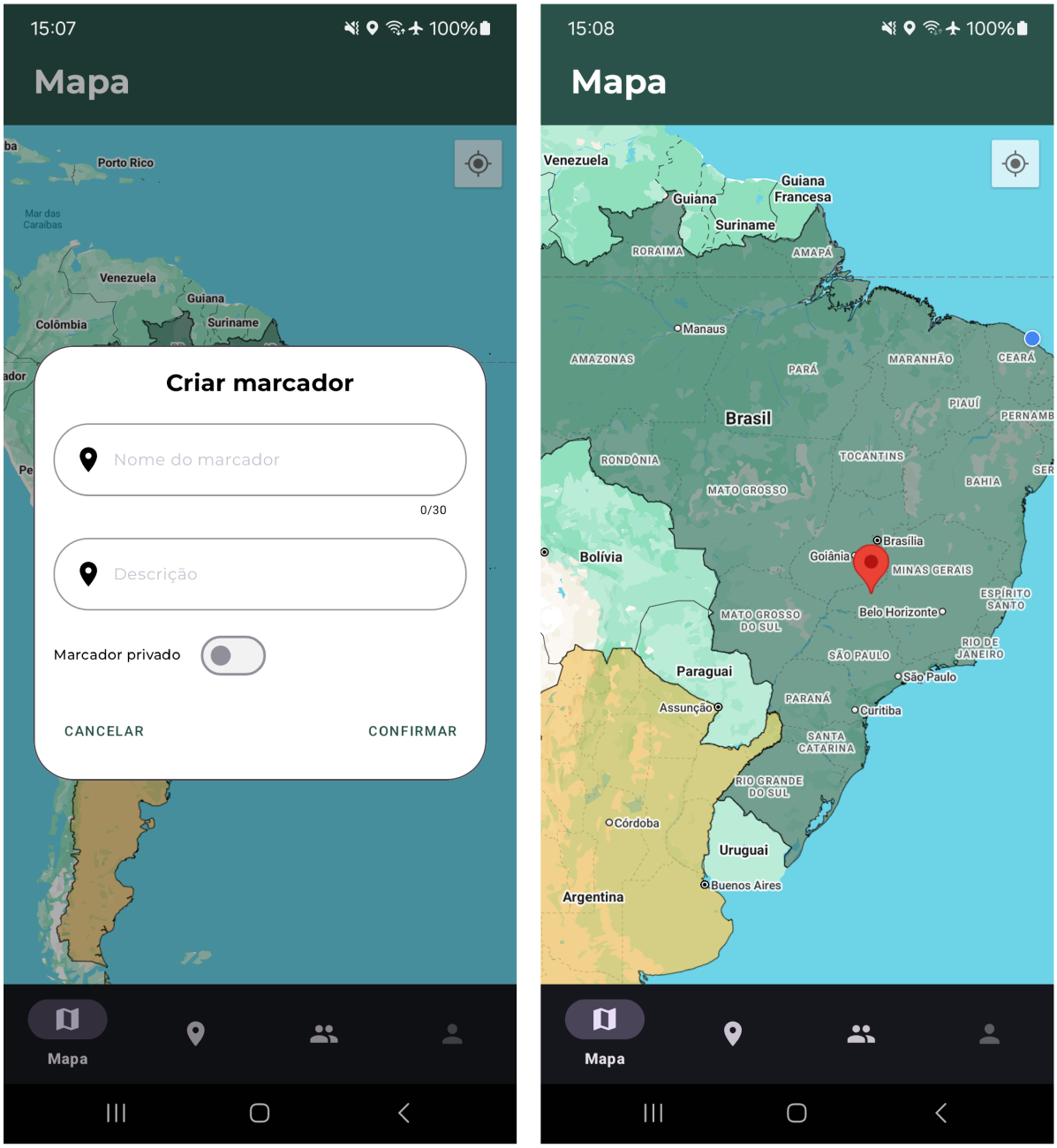
Essa janela exibida sobre o mapa, apresenta a bandeira do país clicado, seu nome e código ISO, além de duas caixas de seleção para que o usuário indique se já o visitou ou quer visitar, como representado na Figura 9. Após a seleção, a área do país no mapa é colorida, indicando que foi marcado. Caso o usuário selecione que já visitou, a área é colorida de verde e caso marque que quer visitar a área é colorida de amarelo.

(a) Exemplo selecionado “Já visitei”. (b) Exemplo selecionado “Quero ir”.

Figura 9 – Bottom Sheet Dialog de país.
Fonte: Do autor.

4.4.5 Dialog de adicionar marcador

Essa janela representada na Figura 10 é exibida sobre o mapa após um clique longo no mapa na área de algum país. Ela possui dois campos para serem preenchidos, o nome do marcador, que é obrigatório, e a descrição, que é opcional, possui também um *Switch* que indica se esse marcador será privado ou público, e por fim dois botões, um para cancelar e fechar a janela e outro para confirmar a criação do marcador. Após a confirmação, a janela é fechada e o marcador é exibido no mapa.



(a) Janela de criação de marcador.

(b) Mapa com o marcador criado.

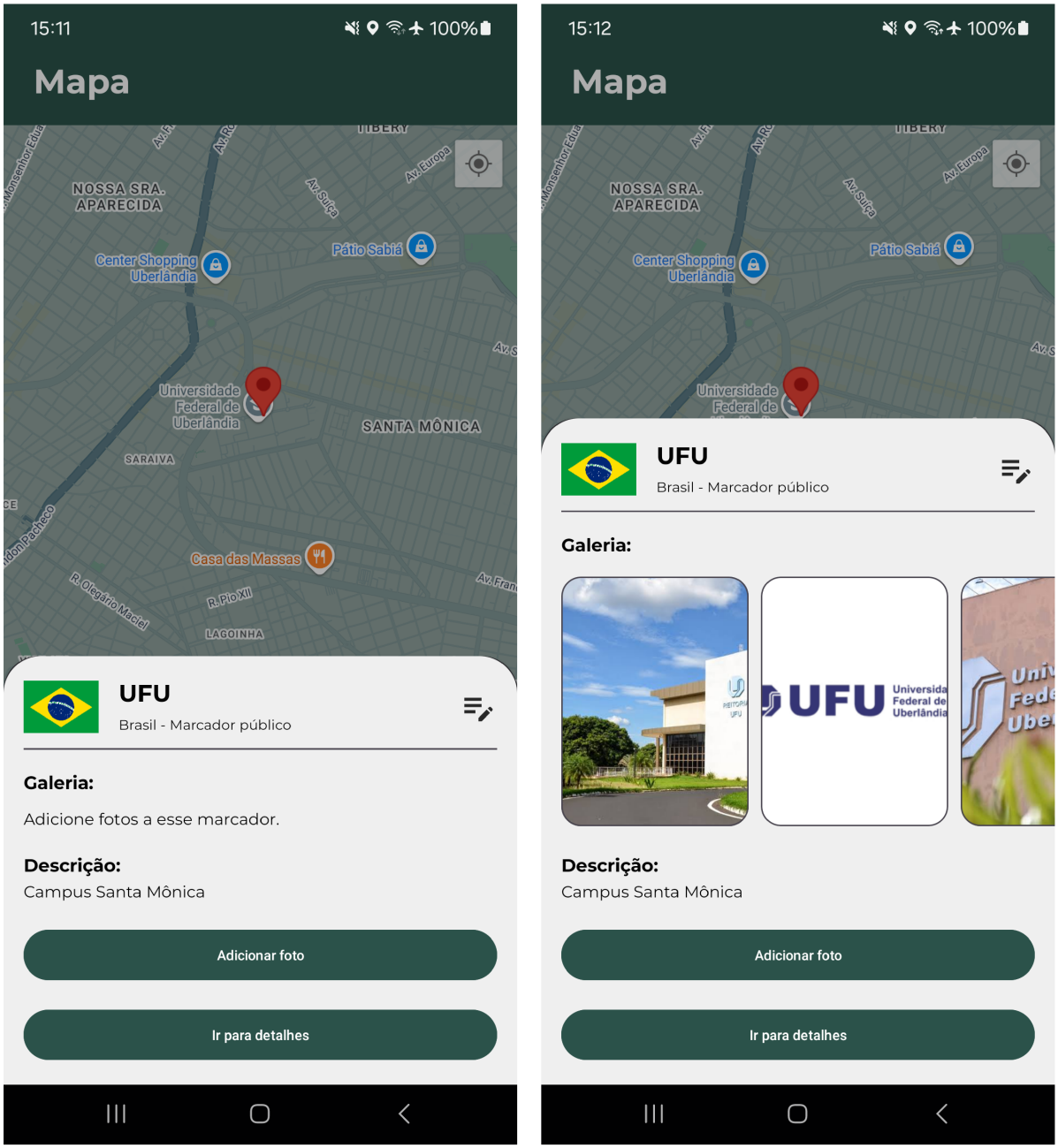
Figura 10 – Dialog de criação de marcador.
Fonte: Do autor.

4.4.6 Bottom Sheet Dialog de informações do marcador

Essa janela representada na Figura 11, é exibida sobre o mapa a partir da parte inferior após o clique em um marcador no mapa. São exibidas informações relacionadas ao marcador como:

- Nome e bandeira do país em que se encontra o marcador
- Nome do marcador

- Informação se o marcador é publico ou privado
- Um ícone clicável para edição das informações do marcador
- Descrição do marcador
- Botão para adicionar mídias associadas ao marcador
- Botão para redirecionar para a tela de detalhes do marcador descrito no subseção 4.4.9



(a) Informações do marcador sem imagem. (b) Informações do marcador com imagens.

Figura 11 – Bottom Sheet Dialog de detalhes do marcador.
Fonte: Do autor.

Figura 12 – Dialog de edição de marcador.
Fonte: Do autor.

Nesse contexto de informações do marcador, temos a possibilidade de edição do nome, descrição e privacidade do marcador através da janela de edição representada na Figura 12 e acessada através do clique no ícone do canto superior direito da janela de informações do marcador. Nessa janela é possível também realizar a exclusão desse marcador, apagando a referência do marcador e das imagens a ele associadas no banco de dados.

4.4.7 Tela de lugares

Essa tela é acessada quando ocorre o clique no segundo item da barra de navegação. Ela é composta por duas abas, uma com a listagem dos países marcados e outra com a listagem de todos os marcadores adicionados. As abas podem ser alternadas clicando sobre seus rótulos. Na listagem dos países vista na imagem (a) da Figura 13, cada item da lista apresenta a bandeira do país, seu nome, e um ícone que representa se foi marcado como "já visitado"ou "quero visitar". O clique em um item da lista redireciona para a tela de detalhes do país, descrita na subseção 4.4.8. Na listagem dos marcadores vista na imagem (b) da Figura 13, cada item da lista corresponde a um marcador criado pelo usuário. Em cada item pode ser observado uma foto associada ao marcador, seu nome e descrição. Se o marcador não possuir fotos associadas, uma imagem de placeholder será exibida no lugar. O clique em um item da lista de marcadores redireciona para a tela de detalhes do marcador, descrita na subseção 4.4.9.

4.4.8 Tela de detalhes do país

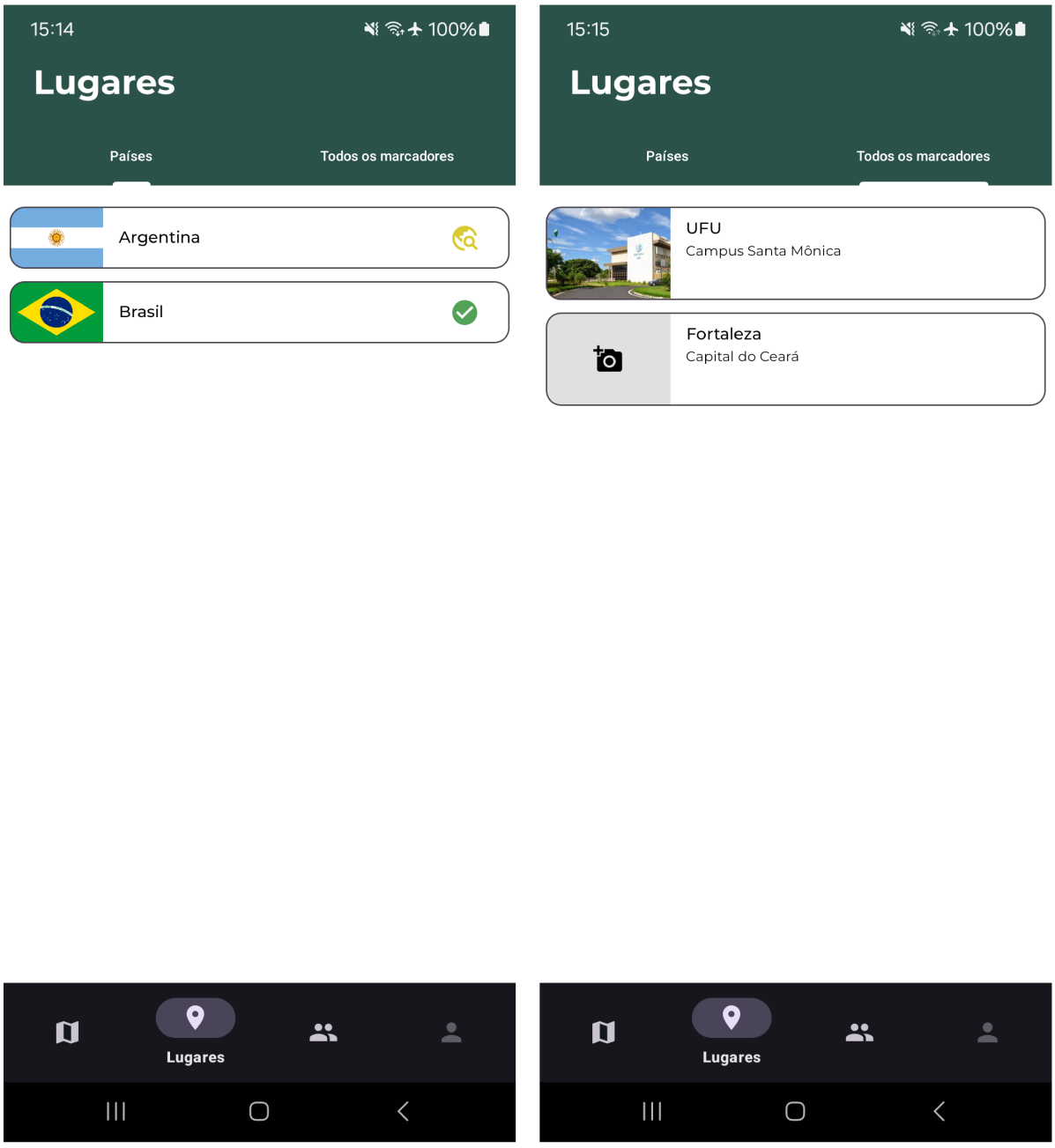
Nessa tela, representada na Figura 14 são exibidos a bandeira do país em um *header* que colapsa a medida que o usuário faz o *scroll* para baixo na tela, um *card* com o nome do país, o número de marcadores associados a esse país, e a indicação se já foi visitado ou se ainda deseja visitar. Além disso, é exibida a lista de marcadores associados a esse país. O clique em um marcador redireciona para a tela de detalhes do marcador, descrita na subseção 4.4.9.

4.4.9 Tela de detalhes do marcador

Essa tela, que pode ser vista na Figura 15, exibe uma imagem em um *header* colapsável, um *card* com o nome, país e descrição do marcador, além da lista de imagens associadas ao marcador em forma de galeria.

4.4.10 Tela de amigos

Essa tela, representada pela imagem (a) da Figura 16, exibe a lista de amigos e solicitações de amizade enviadas ou recebidas. Cada item da lista contem uma foto do amigo, seu *nickname*, e caso a solicitação tenha partido do próprio usuário, uma mensagem de “Solicitação enviada”. Caso a solicitação tenha sido enviada por outro usuário, são exibidos dois ícones, para aceite ou recusa da solicitação. Em caso de recusa, o item é removido da lista, já em caso de aceite os botoes de aceite e recusa dão lugar a um ícone de um mapa, que permite navegar ate o mapa do amigo para visualizar seus países marcados e marcadores adicionados descrito na subseção 4.4.11.

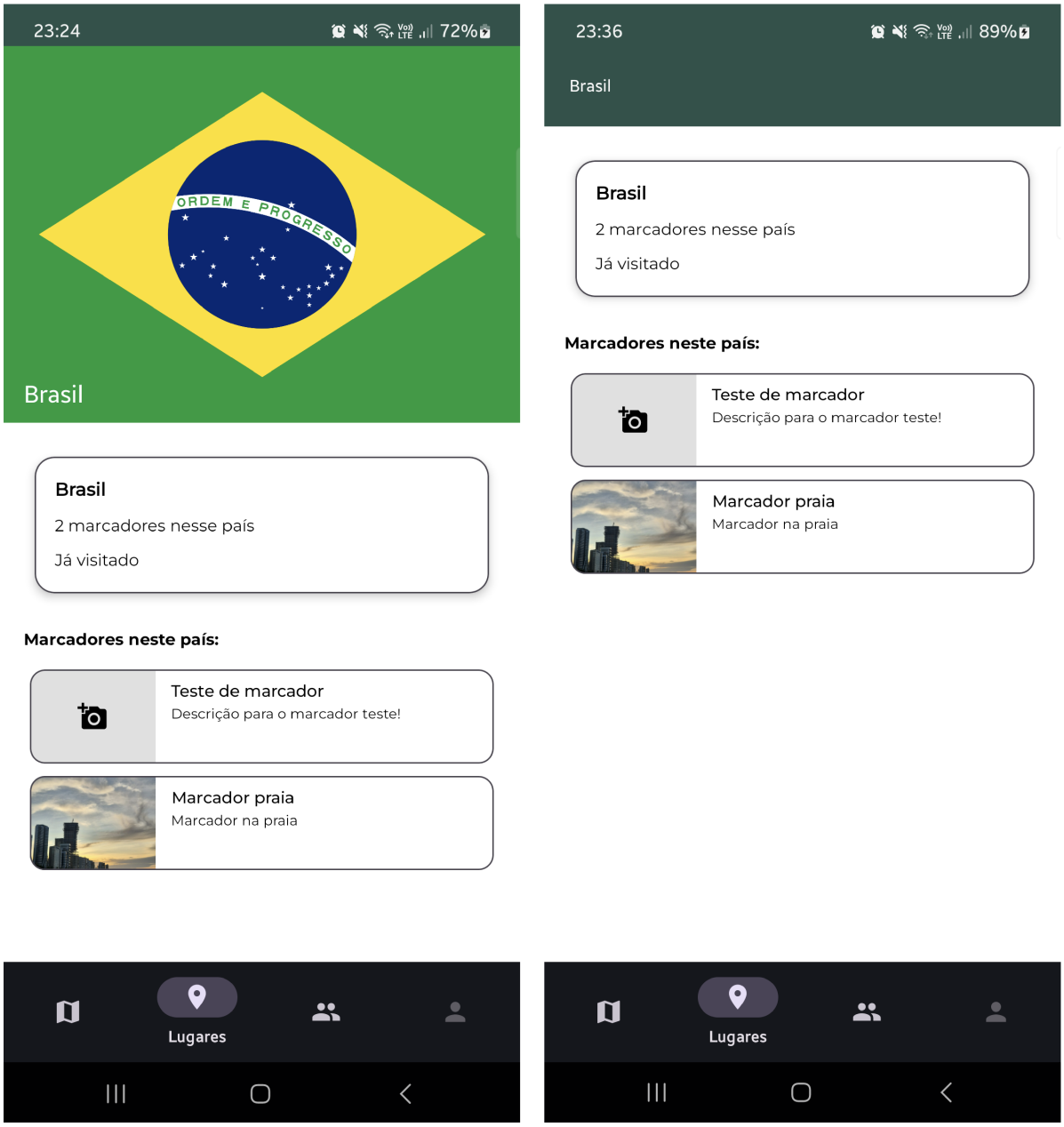


(a) Listagem de países.

(b) Listagem de marcadores.

Figura 13 – Tela de listagem de lugares.
Fonte: Do autor.

Nesse mesmo contexto da tela de amigos é exibido um botão no canto inferior direito que ao ser acionado exibe uma janela representada na imagem (b) da Figura 16. Nessa janela o usuário digita o nickname do amigo que quer adicionar. Ao clicar em confirmar, é realizada uma consulta na lista de nicknames do banco de dados para se obter o identificador único do usuário que se quer adicionar e então é feita a requisição para salvar essa solicitação para ambos os usuários.

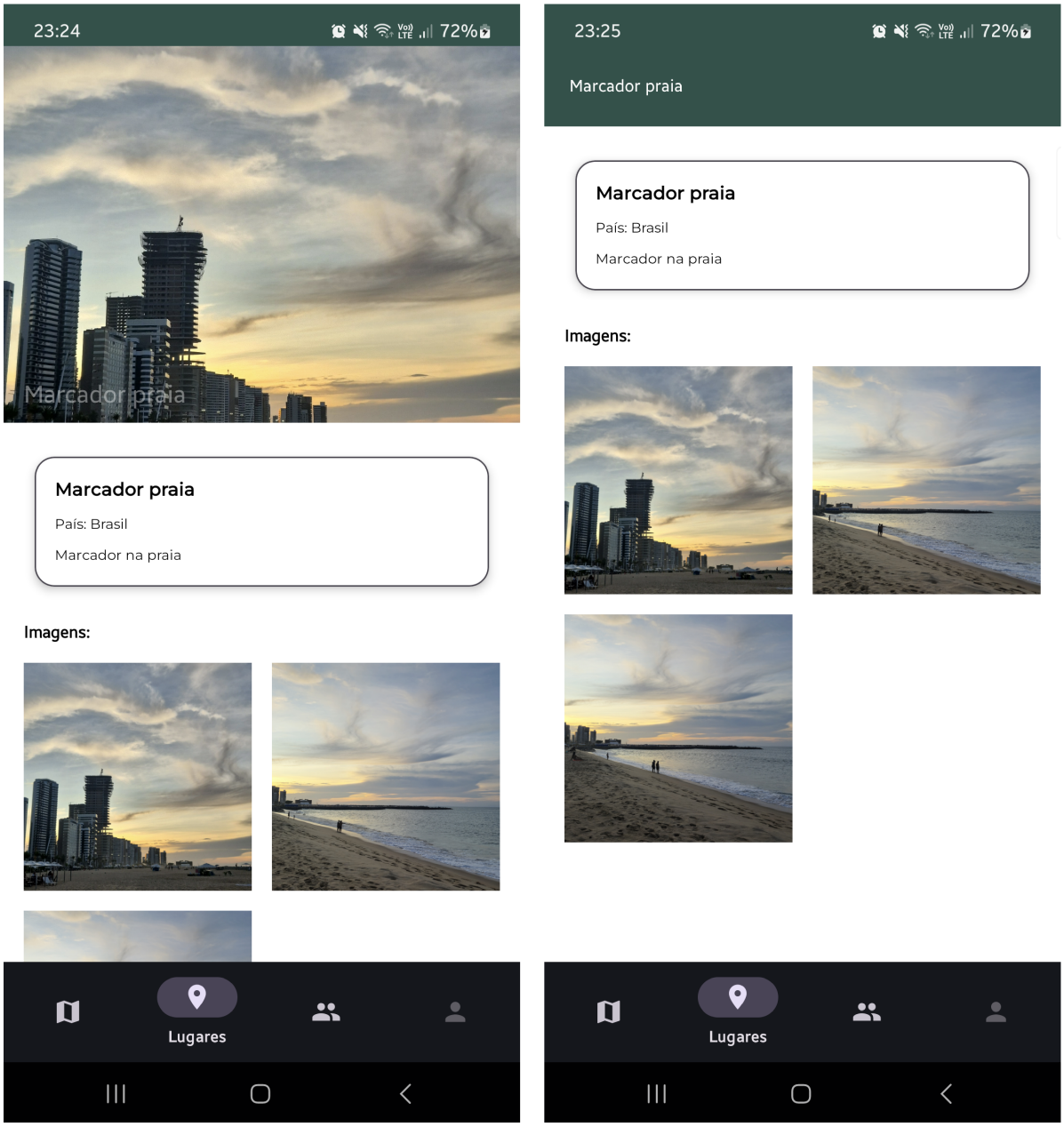


(a) Detalhes do país com *header* expandido. (b) Detalhes do país com *header* expandido.

Figura 14 – Tela de detalhes do país.
Fonte: Do autor.

4.4.11 Tela de mapa de amigo

Nessa tela que pode ser vista na imagem (a) da Figura 17, semelhante a tela de mapa do usuário, é exibido o mapa do amigo clicado na tela de listagem de amigos e seus países e marcadores públicos podem ser visualizados. Assim como no mapa do usuário, também é possível interagir com os marcadores visíveis, e com isso a janela de informações do marcador descrita na subseção 4.4.6 aparece, porem não são exibidos o ícone para edição, e os dois botoes de interação como visto na imagem (b) da Figura 17.



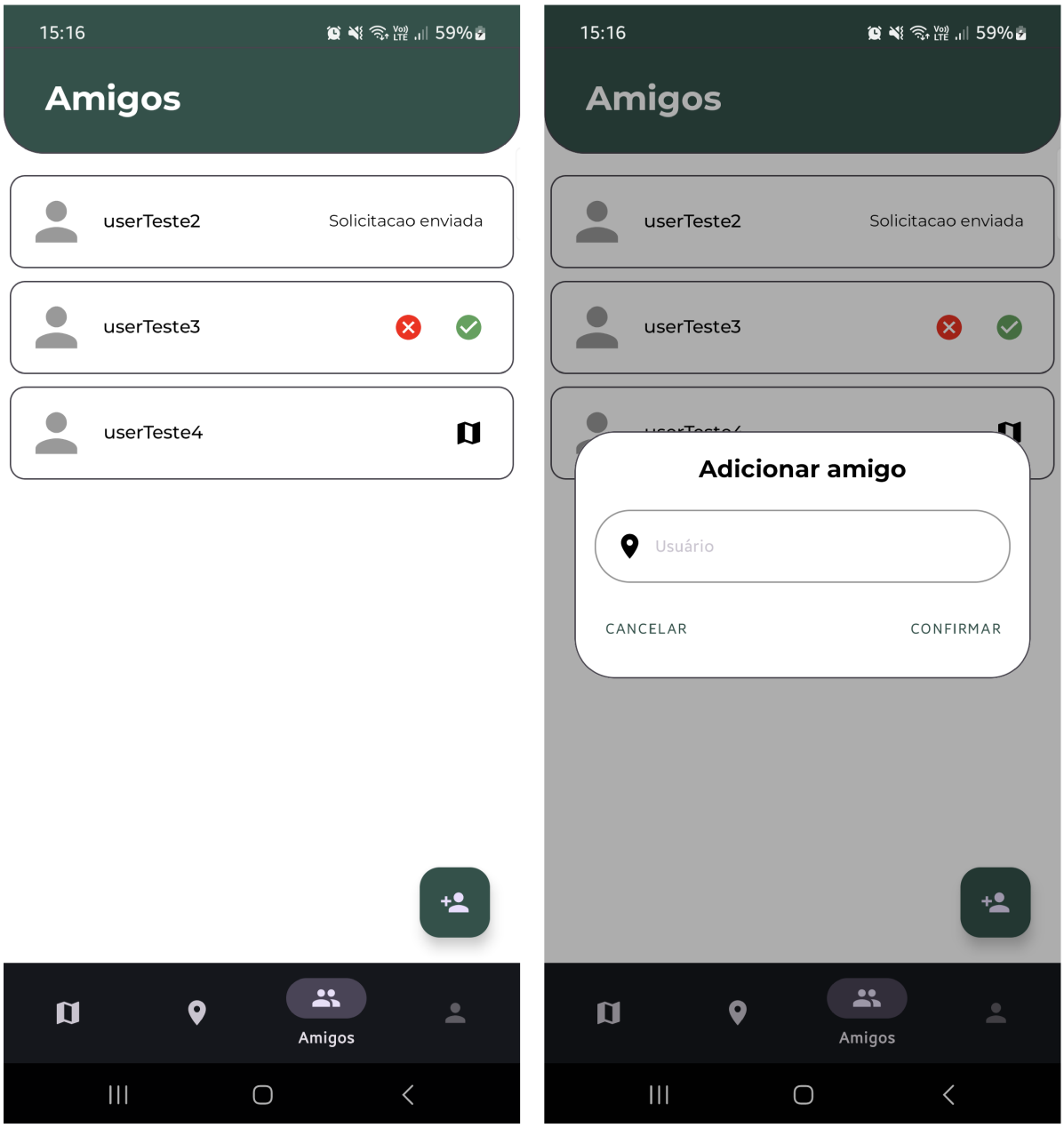
(a) Detalhes do marcador com *header* expandido. (b) Detalhes do marcador com *header* colapsado.

Figura 15 – Tela de detalhes do marcador.
Fonte: Do autor.

4.4.12 Tela de perfil

Nessa tela representada na Figura 18 o usuário pode ver seus dados como:

- Nome
- E-mail
- *Nickname*
- Número de países visitados



(a) Lista de amigos.

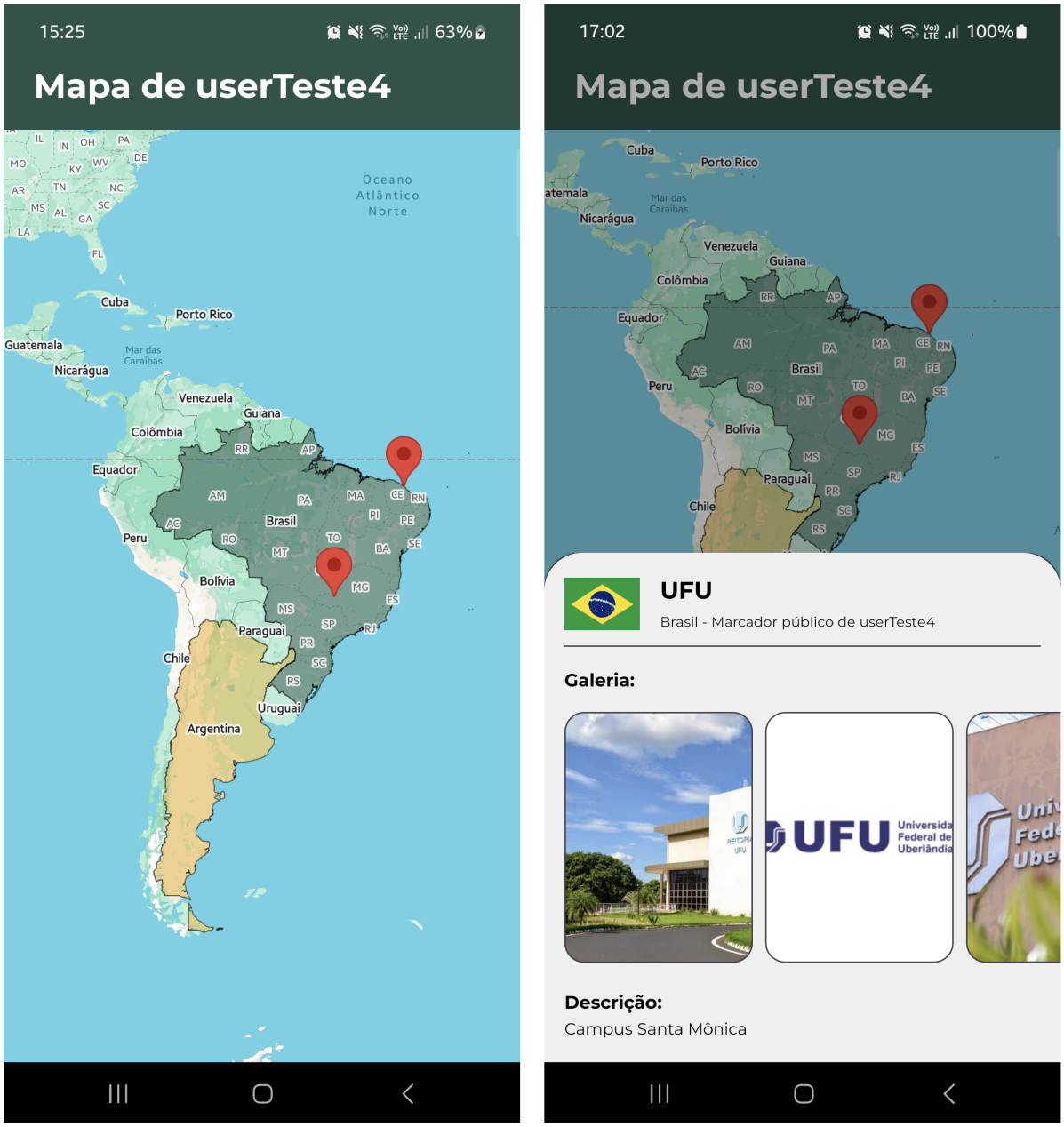
(b) Dialog de adicionar amigos.

Figura 16 – Tela de amigos
Fonte: Do autor.

- Número de marcadores adicionados
- Botão para o usuário realizar o logout e retornar para a tela de login.

Essas informações são atualizadas automaticamente conforme o usuário marca novos países ou adiciona novos marcadores, oferecendo assim uma visão sempre atualizada de suas atividades no aplicativo.

Com a execução das etapas descritas neste capítulo, foi possível implementar um protótipo funcional que atende aos requisitos definidos inicialmente. As funcionalidades integradas,



(a) Visualização do mapa do amigo. (b) Janela de informações do marcador do amigo.

Figura 17 – Tela de mapa do amigo.
Fonte: Do autor.

combinadas ao uso das tecnologias apresentadas, possibilitaram a construção de uma aplicação robusta e escalável para o registro de memórias do lugares desejados.

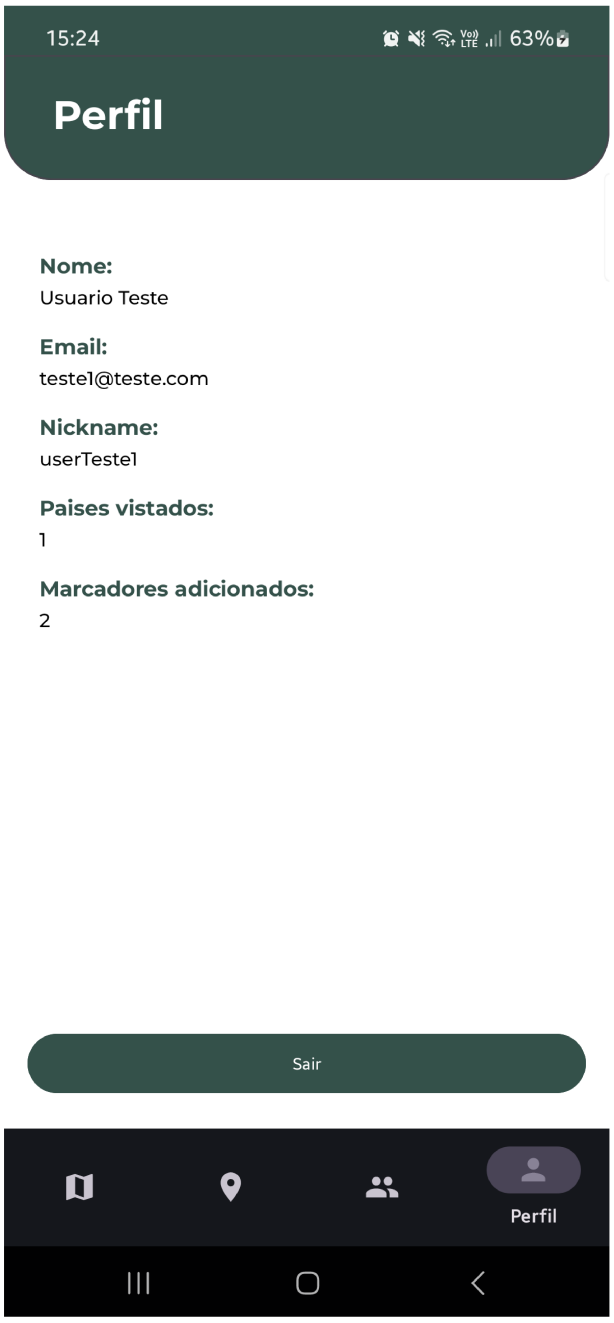


Figura 18 – Tela de perfil.
Fonte: Do autor.

5 Conclusão

O desenvolvimento do aplicativo de mapa interativo e registro de mídias atingiu os objetivos propostos no início deste trabalho. A integração das tecnologias estudadas, como Kotlin, Google Maps SDK e Firebase, aliada à metodologia ágil Kanban e à arquitetura MVVM, permitiu a construção de uma solução eficiente, intuitiva e escalável.

A aplicação entrega aos usuários a possibilidade de criar marcadores personalizados em qualquer área geográfica, associar fotos e descrições, além de compartilhar suas experiências com amigos de forma seletiva. Essas funcionalidades proporcionam uma maneira rica e dinâmica de registrar memórias de lugares que o usuário desejar.

Comparativamente às soluções previamente analisadas, a aplicação desenvolvida se destaca pela combinação única de flexibilidade na criação de marcadores personalizados, robustez no armazenamento escalável e segurança no compartilhamento seletivo das informações, tornando-se uma solução inovadora e adequada às necessidades dos usuários atuais.

Durante o processo de desenvolvimento, desafios técnicos como a integração segura de autenticação, integração de mapas e sincronização de dados em tempo real foram superados com sucesso. Esses aprendizados reforçaram a importância de práticas consolidadas de engenharia de software.

O trabalho desenvolvido não foi testado com usuários finais, portanto não possui métricas e pontos de melhoria apontados por usuários para correção. Essa limitação poderá ser explorada em estudos futuros.

Como trabalhos futuros, propõe-se a ampliação das funcionalidades como interações com marcadores de amigos, criação de um *feed* interativo com ações de usuários amigos, implementação de regras de segurança e acesso ao banco mais robustas, melhorias de interface e usabilidade, e por fim, testes com usuários reais a fim de obter pontos de melhoria.

Por fim, o trabalho não apenas cumpriu seus requisitos funcionais, como também proporcionou uma rica oportunidade de aprendizado prático sobre o ciclo completo de desenvolvimento de aplicações móveis modernas.

Referências

DEVELOPERS, A. **Visão geral da arquitetura do Android**. 2024. <https://source.android.com/docs/core/architecture?hl=pt-br>. [Online; acessado em: 27 out. 2024]. Citado 2 vezes nas páginas 10 e 12.

_____. **Visão geral do AOSP**. 2024. <https://source.android.com/docs/setup/about?hl=pt-br>. [Online; acessado em: 27 out. 2024]. Citado na página 10.

GAMMA, E. **Padrões de Projetos: Soluções Reutilizáveis**. Bookman, 2009. ISBN 9788577800469. Disponível em: <https://books.google.com.br/books?id=U91CYCqTCgkC>. Citado na página 13.

GOOGLE. 2024. <https://firebase.google.com/docs?hl=pt-br>. [Online; acessado em: 27 outubro 2024]. Citado na página 15.

GOOGLE, D. **Visão geral do SDK do Maps para Android**. 2024. <https://developers.google.com/maps/documentation/android-sdk/overview?hl=pt-br>. [Online; acessado em: 27 outubro 2024]. Citado na página 15.

KOLTINLANG. **Kotlin for Android**. 2024. <https://kotlinlang.org/docs/android-overview.html>. [Online; acessado em: 3 outubro 2024]. Citado 2 vezes nas páginas 10 e 13.

LOU, T. et al. A comparison of Android native app architecture MVC, MVP and MVVM. **Eindhoven University of Technology**, 2016. Citado na página 13.

POLARSTEPS. **Polarsteps - Travel Tracker**. 2025. <https://play.google.com/store/apps/details?id=com.polarsteps&hl=pt_BR>. [Online; acessado em: 10 fevereiro 2025]. Citado na página 18.

RANE, A.; JOHN, V. N.; MURTHY, S. Geomaps: An interactive application to enhance map comprehension skills of students. In: **2020 IEEE 20th International Conference on Advanced Learning Technologies (ICALT)**. [S.l.: s.n.], 2020. p. 254–258. Citado 2 vezes nas páginas 8 e 18.

REDDY, S. S. **Trip Tracker Application on Android**. Tese (Doutorado) — San Diego State University, 2011. Citado na página 17.

ROSSI, R. A relevãncia do uso de smartphones durante a experiência turística. **Turismo Visão e ação**, 2019. Citado na página 8.

SOUZA, A. L. D. **Android MVVM Design Pattern**. 2024. <https://cwi.com.br/blog/design-patterns-android/>. [Online; acessado em: 10 maio 2025]. Citado na página 14.

TEAM, S. B. **iPhone vs Android Statistics**. 2024. <https://backlinko.com/iphone-vs-android-statistics>. [Online; acessado em: 3 outubro 2024]. Citado na página 8.