INSTITUTO FEDERAL GOIANO
CAMPUS URUTAÍ
NÚCLEO DE INFORMÁTICA
CURSO DE SISTEMAS DE INFORMAÇÃO

PAULO VICTOR BUENO FLORES

UMA APLICAÇÃO PARA LOCALIZAÇÃO E AUXILIO NA MANUTENÇÃO DE TRILHAS E PISTAS PARA CICLISMO DE MONTANHA

Urutaí, 6 de janeiro de 2025

INSTITUTO FEDERAL GOIANO
NÚCLEO DE INFORMÁTICA
SISTEMAS DE INFORMAÇÃO

PAULO VICTOR BUENO FLORES

UMA APLICAÇÃO PARA LOCALIZAÇÃO E AUXILIO NA MANUTENÇÃO DE TRILHAS E PISTAS PARA CICLISMO DE MONTANHA

Trabalho tecnológico apresentado ao Núcleo de Informática, curso de Sistemas de Informação, do Instituto Federal Goiano, como parte das exigências para obtenção do título de Bacharel em Sistemas de Informação.

Orientador(a):
Rachel Lopes Carcute

Urutaí, 6 de janeiro de 2025

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema Integrado de Bibliotecas (SIBI) – Instituto Federal Goiano

F634a

Flores, Paulo Victor Bueno.

Uma aplicação para localização e auxilio na manutenção de trilhas e pistas para ciclismo de montanha [manuscrito] / Paulo Victor Bueno Flores. – Urutaí, GO: IF Goiano, 2024.
35 fls. : il., tabs.

Orientador: Prof. Me. Rachel Lopes Carcute.

Trabalho de Conclusão de Curso (Bacharelado em Sistemas de Informação) – Instituto Federal Goiano, Campus Urutaí, 2024.

1. Cicloturismo. 2. Bike Park. 3. Arquitetura Limpa. 4. Arquitetura Hexagonal. I. Carcute, Rachel Lopes. II. Título. III. Instituto Federal Goiano.

CDU 004.85:796.6

Ficha elaborada por Johnathan Pereira Alves Diniz – Bibliotecário/CRB 1 nº 2376

INSTITUTO FEDERAL
Goiano

Repositório Institucional do IF Goiano - RIIF Goiano
Sistema Integrado de Bibliotecas

TERMO DE CIÊNCIA E DE AUTORIZAÇÃO

PARA DISPONIBILIZAR PRODUÇÕES TÉCNICO-CIENTÍFICAS NO REPOSITÓRIO INSTITUCIONAL DO IF GOIANO

Com base no disposto na Lei Federal nº 9.610, de 19 de fevereiro de 1998, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia Goiano a disponibilizar gratuitamente o documento em formato digital no Repositório Institucional do IF Goiano (RIIF Goiano), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IF Goiano.

IDENTIFICAÇÃO DA PRODUÇÃO TÉCNICO-CIENTÍFICA

- ☐ Tese (doutorado)
- ☐ Dissertação (mestrado)
- ☐ Monografia (especialização)
- ☑ TCC (graduação)
- ☐ Artigo científico
- ☐ Capítulo de livro
- ☐ Livro
- ☐ Trabalho apresentado em evento
- ☐ Produto técnico e educacional - Tipo:

Nome completo do autor:
Paulo Victor Bueno Flores

Matrícula:
2017101202010140

Título do trabalho:
UMA APLICAÇÃO PARA LOCALIZAÇÃO E AUXILIO NA MANUTENÇÃO DE TRILHAS E PISTAS PARA CICLISMO DE MONTANHA

RESTRIÇÕES DE ACESSO AO DOCUMENTO

Documento confidencial: ☑ Não ☐ Sim, justifique:

Informe a data que poderá ser disponibilizado no RIIF Goiano: ☐☐/☐☐/☐☐☐☐

O documento está sujeito a registro de patente? ☐ Sim ☑ Não

O documento pode vir a ser publicado como livro? ☐ Sim ☑ Não

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O(a) referido(a) autor(a) declara:

• Que o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;

• Que obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autoria, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia Goiano os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;

• Que cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia Goiano.

Urutaí (Local) — 23/01/2025 (Data)

Paulo Victor Bueno Flores
Assinatura do autor e/ou detentor dos direitos autorais

Ciente e de acordo:

Assinatura do(a) orientador(a)

SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA GOIANO

Formulário 965/2024 - DE-UR/CMPURT/IFGOIANO

Paulo Victor Bueno Flores

UMA APLICAÇÃO PARA LOCALIZAÇÃO E AUXILIO NA MANUTENÇÃO DE TRILHAS E PISTAS PARA CICLISMO DE MONTANHA

Monografia, defendida por Paulo Victor Bueno Flores, apresentado ao Instituto Federal de Educação, Ciência e Tecnologia Goiano, como parte das exigências para a obtenção do título de Bacharel em Sistemas de Informação, aprovados pela banca examinadora.

COMISSÃO EXAMINADORA

_____

Profa.Me. Rachel Lopes Carcute

Orientadora

_____

Prof. Me. Jorcivan Silva Ramos

Avaliador

_____

Profa. Dra.. Vivian Cirino de Lima

Avaliadora

Urutaí (GO), 17 de dezembro de 2024.

Documento assinado eletronicamente por:

- **Rachel Lopes Carcute**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 17/12/2024 19:48:24.
- **Vivian Cirino de Lima**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 17/12/2024 19:50:50.
- **Jorcivan Silva Ramos**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 17/12/2024 19:50:53.

Este documento foi emitido pelo SUAP em 17/12/2024. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifgoiano.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 663177
Código de Autenticação: d7d8cf9e25

INSTITUTO FEDERAL GOIANO
Campus Urutaí
Rodovia Geraldo Silva Nascimento, Km 2.5, SN, Zona Rural, URUTAÍ / GO, CEP 75790-000
(64) 3465-1900

Dedico este trabalho aos meus amigos ciclistas.

AGRADECIMENTOS

Agradeço a todos que contribuíram para a realização deste trabalho.

RESUMO

Este trabalho apresenta o processo de desenvolvimento de um software voltado para a divulgação de locais de prática de esportes com bicicleta, aplicando boas práticas de engenharia de software. O desenvolvimento da aplicação segue modelos que garantem manutenibilidade, com base nos princípios do SOLID e na arquitetura Limpa. O *frontend* foi implementado utilizando um *framework cross-platform*, garantindo compatibilidade com múltiplos dispositivos, enquanto o *backend* foi desenvolvido com base na Arquitetura Hexagonal. Ambas as partes foram construídas com a linguagem Dart. O resultado é um aplicativo móvel que permite a divulgação de locais para a prática de esportes, como pistas e eventos, e onde os usuários podem compartilhar *feedback* sobre esses locais. Por fim, o aplicativo foi publicado em uma loja de aplicativos.

Palavras-chave: Cicloturismo. Bike Park. Arquitetura Limpa. Arquitetura Hexagonal

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AOT Ahead-of-time. Pag.19

API Application Programming Interface. Pag.16

ARM Advanced RISC Machine. Pag.20

CCP O Princípio do Fechamento Comum. Pag.22

CRP O Princípio do Reuso Comum. Pag.22

DDD Domain-Driven Design. Pag.84

DIP Dependency Inversion Principle. Pag.22

IEEE Institute of Electrical and Electronic Engineers. Pag.21

ISP Interface Segregation Principle. Pag.22

JIT Just-in-time. Pag.19

JWT JSON Web Token. Pag.29

LSP Liskov Substitutio Principle. Pag.22

OCP Open/Closed Principli. Pag.22

REP O Princípio da Equivalência do Reuso. Pag.22

RF Requisitos Funcionais. Pag.27

RNF Requisitos Não Funcionais. Pag.27

SRP Single Responsibility Principli. Pag.21

UI Interface de Usuário. Pag.15

SUMÁRIO

CAPÍTULO 1

INTRODUÇÃO

A bicicleta vem cada vez mais ganhando o mercado, de 2019 a 2020 teve um aumento de 50% (ABRACICLO, 2020). Praticar esportes com a bicicleta faz bem para saúde do corpo e para a mente, em especifico atividades radicais proporciona uma maior presença de estímulos ao prazer (FEIXA, 1995) (TERUYA, 2000). No Brasil os parques e trilhas voltados para o ciclista vem ganhando espaço, pois é responsável por trazer um maior número de turistas, beneficiando o comercio regional (DELONG, 2012). Existe aplicações que são voltadas para a prática do esporte, como o Strava (Strava Inc., 2024) que é um aplicativo que permite salvar a trajetória feita por um ciclista ou pedestre que também é uma rede social. O aplicativo conta com diversas funcionalidades, permitindo avaliar a performance dos usuários, em destaque temos a funcionalidade "mapa", onde tem as opções de segmento e rotas, essa funcionalidade mostra os percursos já mapeados por usuários anteriores. A opção de Rotas, tem a possibilidade de navegar por trilhas recomendadas pela comunidade. Além do Strava, já mencionado anteriormente, o Trailforks (TRAILFORKS, 2024) traz uma função similar de roteirizar os percursos feitos por seu usuário, mostra a localização no mapa em forma de segmentos, mostrar o estado das pistas e seu nível de dificuldade, e o aplicativo também faz o mapeamento das trilhas. A manutenção das pistas é um assunto de extrema importância, uma vez que as mesmas por estarem expostas a natureza sofrem constantes desgastes causados por fenômenos naturais.

Objetivo deste trabalho é propor uma aplicação que exiba as localizações das trilhas para ciclistas de montanha e auxilie na manutenção dessas trilhas. Além disso, a aplicação tem como objetivos secundários notificar os usuários sempre que uma nova localização de trilha for adicionada, exibir as trilhas com seus respectivos estados de conservação e permitir que os

usuários se cadastrem e adicionem novas trilhas e novos locais destinados à prática de esportes.

1.1 Metodologia

Este trabalho tem por objetivo o desenvolver uma aplicação para auxiliar ciclistas na identificação de locais para prática de esportes. O processo de desenvolvimento do software proposto foi estruturado nas seguintes etapas:

- Planejamento: Nesta etapa, foram identificados os requisitos funcionais e não funcionais, além da definição da topologia da aplicação.
- Design: Na fase de design, foram elaboradas as interfaces de usuário.
- Desenvolvimento: Durante o desenvolvimento, foram realizadas a configuração do ambiente e a codificação do software.
- Testes: Nesta etapa, foram aplicados testes nas principais camadas do software, garantindo a qualidade e estabilidade do sistema.
- Deploy: Nesta fase, foram feitas as configurações necessárias para a implantação do aplicativo nas lojas.
- Implementação: Por fim, o aplicativo foi configurado e publicado na Google Play, ficando disponível para download.

1.2 Justificativa

Os aplicativos atuais são voltados para a performance do ciclista. Existem aplicativos que fazem mapeamento de rotas, cronometra o tempo gasto ao realizar uma trilha ou percurso, mas até agora não temos aplicativos que auxilia na manutenção das pistas e trilhas. Na maioria das aplicações é preciso pagar para ter algumas funcionalidades, neste trabalho é proposto um aplicativo 100% gratuito que tem como foco pistas e trilhas voltadas para o esporte de ciclismo de montanha. Assim agregando mais qualidade na realização das atividades, mais valor no ambiente onde é feito a pratica do esporte, além de possibilitar a localização das pistas ou trilhas com mais facilidade

CAPÍTULO 2

FUNDAMENTOS TEÓRICOS

2.1 Aplicações hibridas e nativas

Um dos papéis do desenvolvedor não é apenas gerar riqueza, mas também promover economia no processo de produção de software, conforme descrito por(MARTIN, 2012) em Clean Architecture. No desenvolvimento de software, existem duas abordagens principais: nativa e híbrida. De acordo com (DIGITAL, 2024), o desenvolvimento nativo oferece maior desempenho, mas apresenta um custo mais elevado, pois é projetado para uma única plataforma. Caso a empresa precise disponibilizar o software em outras plataformas, será necessário investir novamente na sua produção. Por outro lado, no desenvolvimento híbrido, um único código é utilizado para diversas plataformas, permitindo que a empresa invista apenas uma vez na criação do software, reduzindo custos e esforço de implementação.

2.2 Flutter

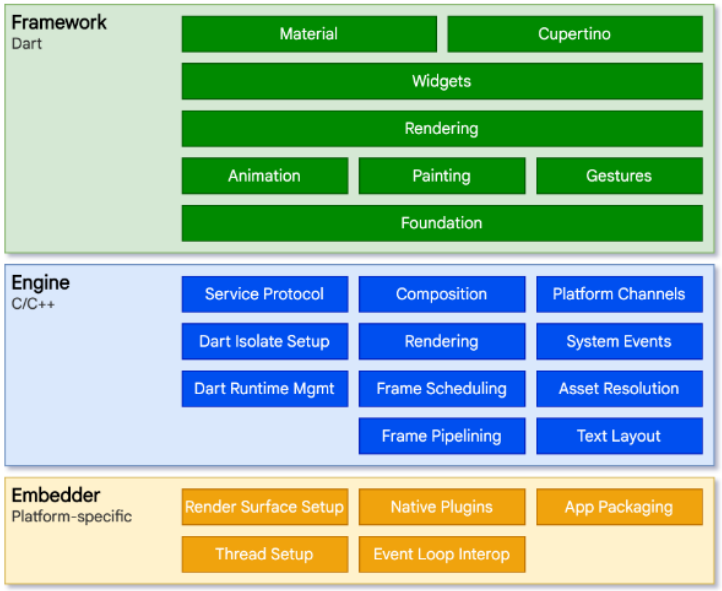
O Flutter é um framework de código aberto, multiplataforma, usado para desenvolver aplicações em qualquer sistema operacional. Foi criado em 2015 e lançado em 2018 pela Google. Apesar de ser relativamente novo, várias empresas já aderiram à ideia, tais como Nubank, BMW, Toyota, eBay, iFood, Alibaba, Google, Bosch, Carteira Digital de Trânsito e muitas outras (FLUTTER, 2024). No Flutter, as Interface de Usuário (UI) são widgets, que funcionam como se cada widget fosse uma peça de Lego. O framework foi desenvolvido na linguagem de programação Dart, criada pela Google. Uma das características do flutter é o desempenho, isso é devido o código ser compilado para a linguagem de máquina, isso ajuda ser executado

diretamente.

2.3 Arquitetura do Flutter

Conforme a documentação do flutter, o framework foi desenvolvido em camadas, utilizando uma arquitetura flexível e modular, o que permite adicionar facilmente novos recursos ou personalizar o comportamento existente. Essas camadas do Flutter são separadas por diferentes níveis de abstração, o que possibilita uma organização eficiente. Tudo isso proporciona escalabilidade, facilitando o desenvolvimento e a manutenção de aplicativos de forma modular, onde cada módulo possui suas responsabilidades específicas. O Flutter foi criado principalmente em Dart, mas também integra componentes de linguagens como C, C++ e utiliza a biblioteca gráfica Skia.

Figura 2.1: Arquitetura do Flutter.



Fonte: Documentação Flutter

Analisando o diagrama de arquitetura do Flutter, temos na camada mais baixa o Embedder, que é responsável por integrar o Flutter com uma plataforma específica. Para essa finalidade, cada plataforma tem o seu próprio Embedder. Conforme a plataforma, teremos um Embedder escrito em uma linguagem diferente. Para o Android, temos o Embedder escrito em C++ e Java; para o Mac e iOS, temos o Embedder em Objective-C/Objective-C++; e para o Linux e Windows, temos o Embedder escrito em C++. As responsabilidades do Embedder incluem possibilitar um ponto de entrada para a aplicação, gerenciar o acesso aos serviços do sistema operacional, como

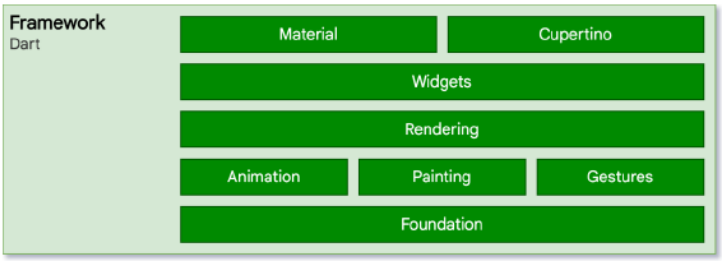
acesso de dados e entrada de dados, gerenciar o laço de eventos (Evento Loop) e a renderização.

No meio das camadas encontra-se a Engine, desenvolvida em C/C++. Em outros termos, podemos dizer que o Flutter Engine é o núcleo, o Kernel, responsável por fornecer suporte às funcionalidades básicas e fundamentais necessárias para executar todos os aplicativos em Flutter. Essas funcionalidades são primitivas fundamentais, podendo ser recursos como renderização de gráficos, gerenciamento de eventos, recursos do sistema operacional, gerenciamento de estado do aplicativo e comunicação com Application Programming Interface (API) nativa.

A Engine é responsável por rasterizar os elementos da tela sempre que for necessário atualizar a interface de usuário. Esse processo garante uma renderização fluida e de alta qualidade. Além disso, ela também é responsável por fornecer a implementação de baixo nível da API do Flutter, envolvendo gráficos por meio do Impeller no IOS. No Android e em outras plataformas, temos o SKIA, que é uma biblioteca de código aberto desenvolvida pelo Google.

Na camada mais alta da arquitetura, encontra-se o Framework Flutter, desenvolvido em Dart. Esta camada é a principal interface para os desenvolvedores que utilizam o Flutter, sendo, portanto, a mais frequente em seu uso. Na estrutura do framework, trabalhamos diretamente com os widgets, os quais são os elementos fundamentais na criação das interfaces de usuário dos aplicativos. Os widgets são estruturados hierarquicamente e são responsáveis por organizar e desenhar a interface do usuário na tela.

Figura 2.2: Flutter.



Fonte: Documentação Flutter

Analisando a Figura 2.2 temos:

- Foundation: Essa camada é a base de suporte, ela fornece funções básicas, para criar aplicativos, são funções essenciais, como gestão de tempo, detecção de gestos, manipulação de eventos e acesso à plataforma subjacente.

- Rendering: Esta é a camada de renderização, que oferece uma abstração para lidar com o layout, possibilitando a manipulação dinâmica dos widgets através de árvores. Sua função

principal é desenhar e pintar os elementos gráficos da interface do usuário.

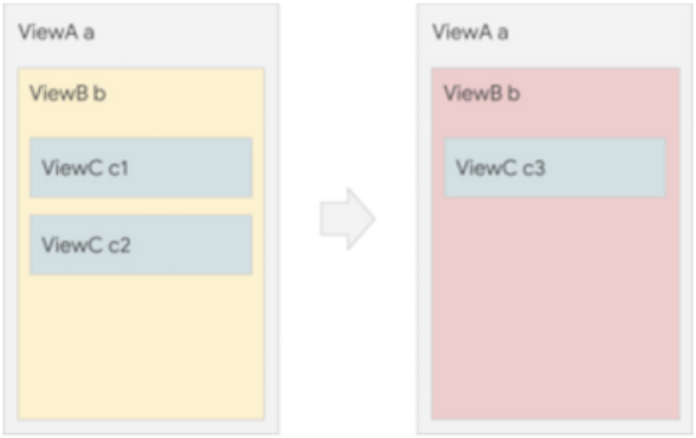
- Widgets: É responsável pela organização dos elementos visuais. Essa camada nos permite criar abstrações e composições, o que possibilita o desenvolvimento de blocos de construção para a criação de layouts robustos e reativos.
- Material e Cupertino: São conjuntos de diretrizes de design. Nesta camada faz uso das primitivas de composição da camada de Widgets, o que possibilita utilizar as bibliotecas de Widgets material e Cupertino para implementar as linguagens de design criadas pela Google e Apple.

2.4 Interface de usuários declarativas

Para compreendermos melhor a abordagem declarativa, é importante entender o seu oposto. No estilo imperativo, a implementação aborda o 'o que fazer' e 'como fazer'. Embora essa abordagem possibilite uma maior flexibilidade e eficiência no desenvolvimento, ela também adiciona uma maior complexidade ao trabalho do desenvolvedor. No entanto, o paradigma declarativo simplifica o processo, apesar de diminui a flexibilidade, concentrando-se exclusivamente em descrever o estado desejado. Assim no paradigma declarativo, a complexidade da implementação é assumida pelo framework Flutter.

Considere o exemplo da figura 2.3

Figura 2.3: Estrutura de interface.



Fonte: Documentação Flutter

Conforme ilustrado na figura 2.3, o paradigma imperativo demandaria uma abordagem complexa para uma operação simples, como a mudança da cor de um View de amarelo para

vermelho. Seria necessário ir ao ViewB, recuperar uma instância de 'b' utilizando seletores e, em seguida, invocar mutações nela. Estas mutações realizariam as modificações na tela de forma implícita. O código abaixo, apresentado na Figura 2.4, é um exemplo de como seria o estilo imperativa.

Figura 2.4: Código imperativo.

```
b.setColor(red)
b.clearChildren()
ViewC c3 = new ViewC(...)
b.add(c3)
```

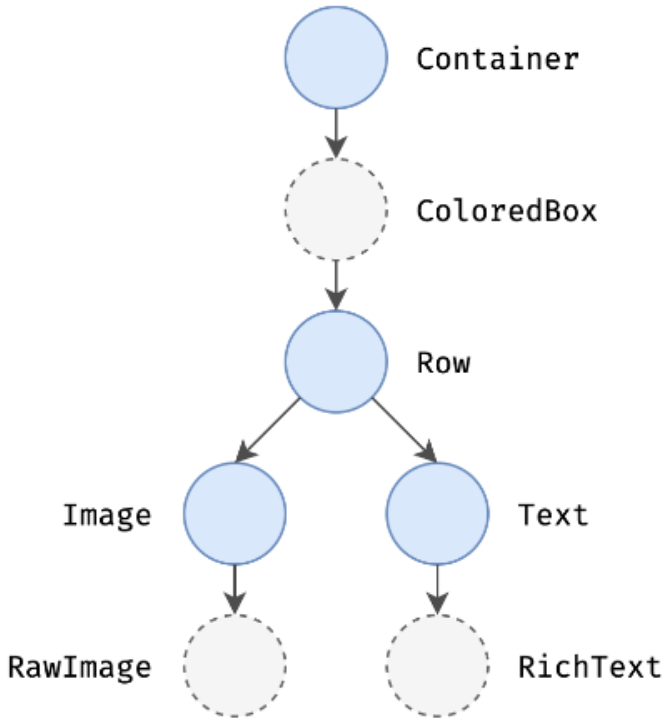
Fonte: Documentação Flutter

No entanto, no estilo declarativo, as views são widgets no Flutter que assumem estruturas imutáveis. A interface é construída por meio de arvores de componentes. Para alterar a interface, um widget invoca a sua própria reconstrução, que gera uma nova subárvore.

2.5 Widgets

No Flutter, os widgets são os meios essenciais para construir nossas interfaces, onde usamos o agrupamento de widgets para compor a UI de uma aplicação em Flutter. Essa estrutura de widgets resulta em uma árvore de componentes, na qual cada nó representa um widget. Na figura 2.5, podemos visualizar essa árvore de componentes.

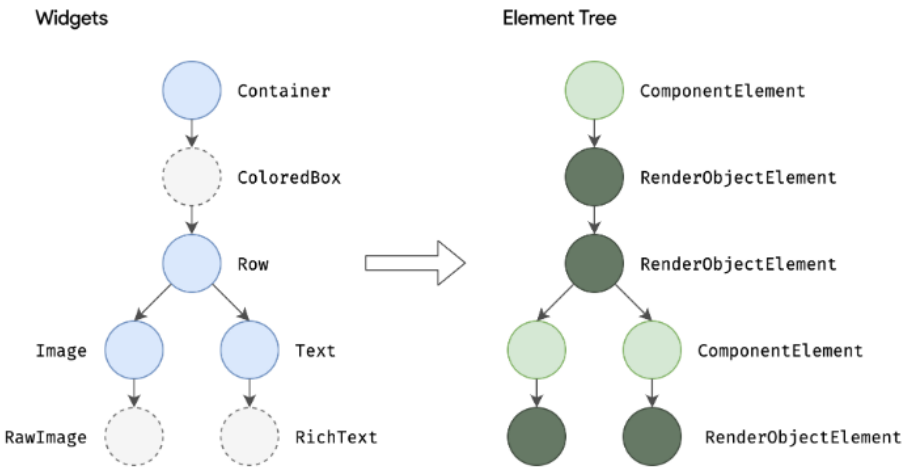
Figura 2.5: Widgets.



Fonte: Documentação Flutter

Porém, não estamos apenas lidando com a árvore de Widgets. Para compreendermos como as atualizações na interface são realizadas, é necessário entender o funcionamento da árvore de Elementos. Durante a construção da interface, o Flutter traduz os widgets expressos em uma árvore de elementos, onde cada widget corresponde a um elemento da árvore. Quando um widget sofre alterações, uma nova referência é passada para a árvore de elementos, a qual, por sua vez, é responsável por manter a estrutura lógica intacta e utilizar a referência passada para renderizar os elementos na arvore de renderização.

Figura 2.6: Arvores do Flutter.



Fonte: Documentação Flutter

2.6 DART

O Dart é uma linguagem orientada a objetos, multiparadigma, multiplataforma, fortemente tipada, com suporte à segurança nula de alta performance, apresentada pela Google, em 2011, com objetivo de substituir o javascript como principal linguagem embutida nos navegadores. Não teve tanto sucesso na substituição, mas por outro lado se destacou pelas suas características no framework Flutter. O que torna o Dart especial para ter sido escolhido como a linguagem oficial do Flutter? Uma das suas principais características é a velocidade, devido seus recursos Just-in-time (JIT) e Ahead-of-time (AOT). Com esses recursos o código é compilado diretamente para Advanced RISC Machine (ARM) nativo, que possibilita performance como uma aplicação nativa (DART, 2024). Suas principais características incluem:

- Sintaxe clara e de fácil compreensão: Para os programadores familiarizados com a linguagem C e de suas derivadas, o Dart possui uma sintaxe semelhante (C-like), ou seja, é compacta, expressiva e de fácil compreensão.

- Tipagem estática e dinâmica: Dart permite aos desenvolvedores escolher entre declarar os tipos ou permitir que o sistema faça a inferência de tipo.

- Robustez e Eficiência: Dart é rápido e eficiente pois é compilado diretamente para ARM, oferece suporte ao Flutter e a presenta programação assíncrona com foco na produtividade (JIT) e compilação (AOT), características que ajuda a manter o código de forma eficiente.

Além disso, é orientado a objetos e multiplataforma. Esses são aspectos que proporciona robustez ao Dart.

CAPÍTULO 3

ARQUITETURA DE SOFTWARE

Em meados de 1956, os desenvolvedores de software perceberam que, à medida que a complexidade do software aumentava, era necessário planejar e projetar o software antes de iniciar sua construção (Herbert D. Benington, 1956) (BENINGTON, 1956). Nessa época surgiram várias abordagens, mas foi apenas nos anos 1990 que tivemos o primeiro livro "Software Architecture: Perspectives on an Emerging Discipline" por (Shaw e Garlan) (SHAW; GARLAN, 1996). Mais tarde, ocorreu a criação da primeira norma padrão, a ISSO/IEEE 1471-2000, também conhecida como Pratica Recomendada Institute of Electrical and Electronic Engineers (IEEE) (HILLIARD, 2000) para Descrição Arquitetural de Sistemas Intensivos em Software. Segundo o que foi estabelecido pela IEE, a arquitetura de software é a estrutura fundamental que define a organização dos componentes do sistema, orienta o design de software e conecta todas as partes interessadas. Ela garante interoperabilidade, escalabilidade e manutenibilidade, possibilitando a evolução contínua do sistema.

3.1 SOLID

Segundo o Robert C. Martin (MARTIN, 2012), quando desenvolvemos sistemas, é imprescindível fazer uso dos princípios de design. Esses princípios foram cunhados por Michael Feathers, que criou o acrônimo conhecido como SOLID. Os princípios proporcionam a criação de sistemas mais fáceis de manter, escalar, entender e refatorar, permitindo metodologias de projeto ágil. Os princípios SOLID são:

- 1-Single Responsibility Principle (SRP): O princípio da responsabilidade única, uma classe deve ter um, e apenas um, motivo para mudar.

- 2-Open/Closed Principle (OCP): Principio do aberto fechado, o software deve ser aberto para extensão, mas fechado para modificação.

- 3-Liskov Substitutio Principle (LSP):Princípio da substituição de Liskov, os subtipos devem ser substituíveis pelo seu tipo base.

- 4-Interface Segregation Principle (ISP): Princípio da segregação de interface, interfaces especificas são melhores que uma interface geral.

- 5-Dependency Inversion Principle (DIP):Princípio da inversão do controle de dependência, depender de abstrações e não de classes concretas.

3.2 Acoplamento e coesão

Segundo Evans (EVANS, 2004), criador do Livro Domain-Driven Design, o software deve ser maleável, permitindo mudanças e adaptação continua. O acoplamento é o grau de dependência entre os objetos (Robert C. Martin) (MARTIN, 2012). Na criação de classes, componentes e camadas em um software, devemos buscar baixo acoplamento entre eles, o que é essencial para realizar testes, manutenção e escalabilidade (Robert C. Martin) (MARTIN, 2012). Já a coesão refere-se ao quão bem refinado e focado é um objeto, sendo o primeiro princípio do SOLID (SRP). O desejável é termos alta coesão e baixo acoplamento, conforme o princípio da responsabilidade única de Robert C. Martin. Quando falamos de coesão e acoplamento em design de software temos três princípios fundamentais, desenvolvidos por Robert C. Martin também conhecido como Uncle Bob. Os princípios são:

- O Princípio da Equivalência do Reuso (REP): Este princípio diz que os componentes que tem acoplamento entre si devem ser reutilizados juntos e lançados juntos, em conjunto como se fosse uma unidade.

- O Princípio do Fechamento Comum (CCP): Este princípio diz que as classes que mudam juntas devem ser unidas no mesmo componente. Portanto, as classes que mudam pelos mesmos motivos e ao mesmo tempo devem ser encapsuladas juntas. Assim como no princípio SRP, onde a responsabilidade única garante alta coesão nas classes, este princípio garante coesão nos componentes.

- O Princípio do Reuso Comum (CRP): Este princípio afirma que classes que são reutilizadas

juntas devem ser mantidas unidas no mesmo componente. O objetivo é facilitar o reuso, agrupando todas em um único componente.

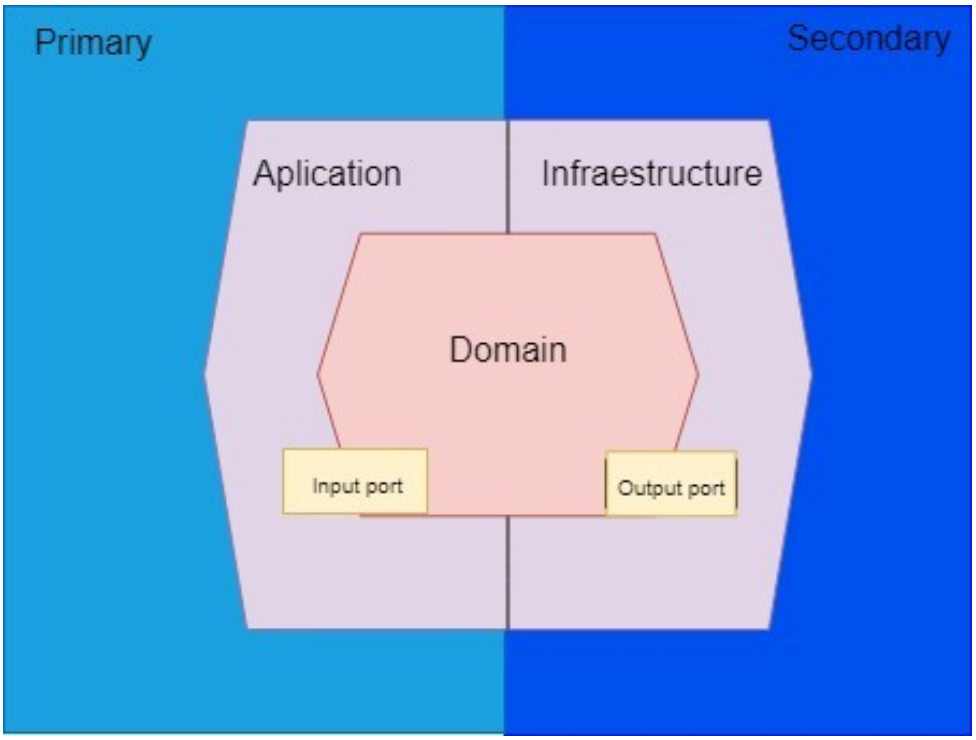
3.3 Backend & Frontend

No desenvolvimento de software, temos a separação em duas partes: o frontend e o backend. Segundo Martin Fowler (FOWLER, 2012), esta divisão é uma boa pratica recomendada para a construção de sistemas mais seguros, robustos e escaláveis. Para Eric Elliott (ELLIOTT, 2014), a separação do software em frontend e backend é primordial para a criação de aplicações escaláveis e fáceis de manter. Portanto, o frontend da aplicação será desenvolvido usando arquitetura limpa e o backend usando arquitetura hexagonal.

3.4 Arquitetura Hexagonal

A arquitetura hexagonal foi introduzida por Alistair Cockburn (COCKBURN, 2005) em 2005 em um artigo intitulado “Hexagonal Architecture” ou "Ports and Adapters”. Com base em sua experiencia de trabalho, Alistair Cockburn identificou alguns problemas recorrentes em arquiteturas tradicionais, como alto acoplamento, dificuldade para realizar testes, dependência de tecnologia e baixa flexibilidade. O principal objetivo dessa arquitetura é separar a regra de negócio da aplicação na camada de domínio. Portanto a camada de domínio é independente de tecnologia, não conhece outra camada além dela própria, não pode depender de nada, além dela mesmo. Na camada de domínio, para ter acesso, é necessário implementar suas interfaces, portas de entrada e saída, para realizar a manipulação de serviços. Não há um número fixo de camadas nessa arquitetura, mas, para simplificar, podemos resumir em três camadas principais. A camada de aplicação é responsável por se comunicar com o mundo externo, utilizando diversas tecnologias conforme a necessidade. Quando precisamos aplicar uma regra de negócio, implementamos uma porta de entrada na camada de domínio, que contém os serviços necessários para a manipulação de dados. A camada de infraestrutura, por sua vez, é responsável pela manipulação dos dados e implementa uma porta de saída da camada de domínio. Segundo Cockburn (2005), em seu modelo de arquitetura, temos dois atores, o primário é aquele que executa uma ação na aplicação, que vai provocar uma reação que por sua vez induz o ator secundário para obter uma resposta do sistema. O ator secundário não dispara a ação, mas é responsável por responder as chamadas do sistema.

Figura 3.1: Arquitetura Hexagonal.



Fonte: Adaptado de Vernon (2013).

Analisando a figura 3.1, temos as portas, que são interfaces que funcionam como um contrato entre as camadas. Elas são a forma como as camadas interagem para entrada e saída de dados. As portas de entrada representam tudo o que vem do lado primário, enquanto as portas de saída representam tudo o que vem do lado secundário para atender o sistema.

3.5 Arquitetura Limpa

Cunhada por Robert C. Martin em meados de 2012, no livro "O Guia do Artesão", esta arquitetura propõe um modelo em camadas com objetivo de separar as preocupações. Isso resulta em sistemas manuteníveis, altamente testáveis e de baixo acoplamento. Oriunda da arquitetura hexagonal (Alistair Cockburn, 2005) e da arquitetura em cebola (Jeffrey Palermo, 2008), que têm por característica comum o isolamento das preocupações, ambas compartilham de conceitos como o isolamento da regra de negócio e uma camada para interface de usuário UI. Segundo Robert C. Martin arquiteturas que compartilha dessas ideias, produz software de alta qualidade, com as seguintes propriedades:

- Independência de frameworks: A arquitetura deve ser independente de qualquer tecnologia. Em outras palavras, a camada de um software responsável pelas regras de negócio não

deve conhecer nenhuma tecnologia, pois esta muda constantemente, enquanto as regras de negócio raramente mudam. O framework deve ser usado como uma ferramenta que pode ser facilmente substituída.

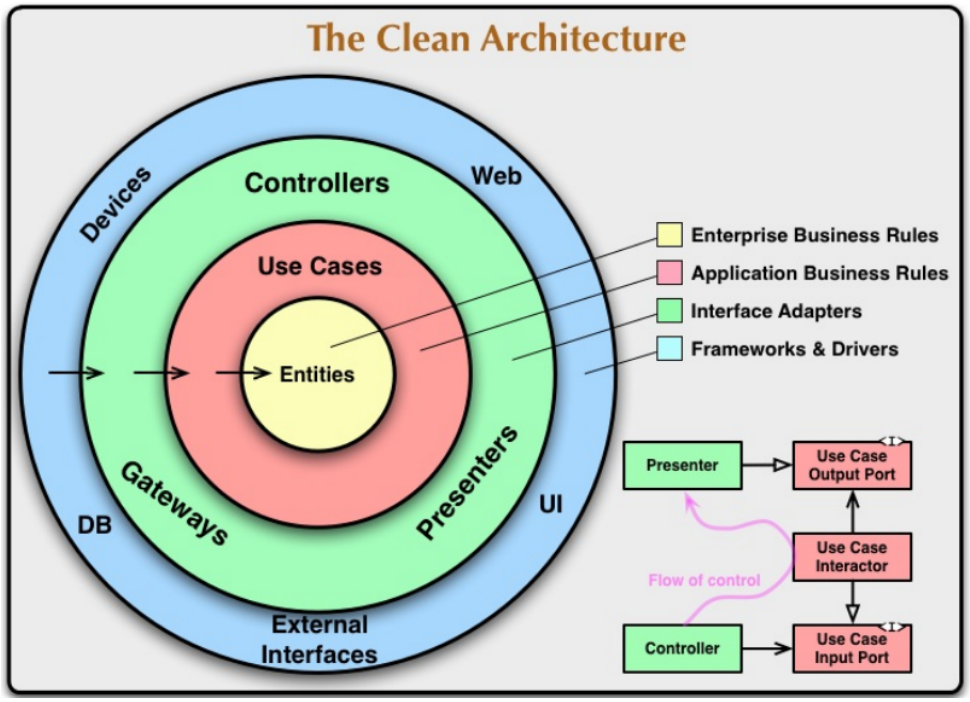
- Testabilidade: O código das regras de negócio pode ser testado facilmente, sem depender de tecnologia ou interface de usuário (UI).
- Independência da UI: A interface do usuário pode mudar facilmente sem impactar as regras de negócio da aplicação.
- Independência de banco de dados: Assim como na camada de interface do usuário, as alterações na camada de banco de dados não devem afetar as regras de negócio da aplicação.
- Independência de qualquer agência externa: A camada de casos de uso e entidades é responsável pelas regras de negócio da aplicação. Esta camada não conhece o mundo externo. Esse isolamento permite que as mudanças externas não impactem na lógica de negócio do software.

A Figura 3.2 representa o modelo da arquitetura limpa e suas camadas, onde cada anel do círculo concêntrico corresponde a uma camada com sua responsabilidade especifica. Essa arquitetura busca separar as preocupações de forma que cada camada tenha apenas uma única responsabilidade.

Ao analisar a Figura 3.2 da arquitetura limpa, começaremos pelo princípio fundamental que rege essa arquitetura. Segundo Robert C. Martin, "as dependências de código-fonte devem ir da camada mais externa para a mais interna, em direção às regras de negócio". Portanto, as camadas internas não devem ter conhecimento das camadas externas; apenas devem conhecer as camadas mais internas. Dessa forma, qualquer alteração feita nas camadas mais externas não terá impacto nas camadas mais internas. Não há um número definido de camadas nessa arquitetura, a seguir temos as quatro camadas da arquitetura limpa.

- Entidades (Entities): As entidades são os objetos cruciais das regras de negócio. São compostas de dados e métodos que encapsulam a lógica de negócio de uma aplicação. Esta camada não conhece nenhuma tecnologia, sendo composta apenas por código puro, portanto, mudanças nas camadas mais externas não devem causar nenhum impacto nesta camada.

Figura 3.2: Arquitetura Limpa.



Fonte: Blog Robert C. Martin.

- Casos De Uso (Use Cases): Os casos de uso fazem parte das regras de negócio e são os orquestradores da lógica para a manipulação das entidades.
- Adaptadores De Interface (Interface Adapters): Os adaptadores são responsáveis por converter os dados para objetos do nosso sistema, tanto para o uso interno, quanto para envio externo.
- Frameworks E Drivers (Frameworks & Drivers): Os Frameworks e Drivers são detalhes e são constantemente alterados, para não causar nenhum impacto deve ficar sempre mantidos nas extremidades das camadas.

CAPÍTULO 4

ANÁLISE DE REQUISITOS

Os requisitos no desenvolvimento de um sistema são fundamentais para garantir a compreensão e a clareza entre todas as partes interessadas. A seguir, apresento o levantamento dos Requisitos Funcionais (RF) e Requisitos Não Funcionais (RNF). Primeiramente, apresento o nome da aplicação que proponho desenvolver neste trabalho: PlaceRiders.

4.1 Requisitos funcionais

Na tabela 4.1 temos os requisitos funcionais do sistema. Estes requisitos descrevem o comportamento do sistema e suas funcionalidades.

Tabela 4.1: Requisitos Funcionais (RF)

Id	Requisitos
RF-01	O Sistema deve permitir adicionar locais para a prática de esportes.
RF-02	O sistema deve permitir adicionar eventos.
RF-03	O sistema deve exibir os locais onde é possível praticar esportes.
RF-04	O sistema deve exibir as pistas.
RF-05	O sistema deve exibir os eventos.
RF-06	O sistema deve permitir alertar sobre problemas na pista.
RF-07	O sistema deve permitir o envio de solicitações de participação para equipes de manutenção.
RF-08	O sistema deve permitir o cadastro de usuários.
RF-09	O sistema deve permitir a recuperação de senha.
RF-10	O sistema deve notificar os usuários.
RF-11	O sistema deve funcionar em várias plataformas.
RF-12	Autenticação de usuários.
RF-13	O sistema deve permitir adicionar vídeos e fotos ao perfil de usuário.
RF-14	O sistema deve permitir buscar eventos por nome ou data.
RF-15	O sistema deve permitir buscar o local de esporte por nome ou localização.
RF-16	O sistema deve permitir consultar o local de esporte.
RF-17	O sistema deve permitir curtir o local de esporte.
RF-18	O sistema deve permitir curtir o evento.
RF-19	O sistema deve permitir comentar no local de esporte.
RF-20	O sistema deve permitir comentar sobre o evento.
RF-21	O sistema deve permitir controle de solicitação de participação de equipe.
RF-22	O sistema deve permitir consultar a localização do local de esporte.
RF-23	O sistema deve permitir que o usuário exclua a conta.
RF-24	O sistema deve permitir editar a pista.
RF-25	O sistema deve permitir adicionar fotos à pista.
RF-26	O sistema deve permitir cadastrar pistas.
RF-27	O sistema deve permitir cadastrar grupos.
RF-28	O sistema deve permitir adicionar vídeos da pista.
RF-29	O sistema deve permitir excluir o local de esporte.
RF-30	O sistema deve permitir adicionar fotos do local de esporte.
RF-31	O sistema deve permitir adicionar a localização do local de esporte.
RF-32	O sistema deve permitir excluir o evento.
RF-33	O sistema deve permitir cadastrar equipes de manutenção.
RF-34	O sistema deve permitir editar o local de esporte.
RF-35	O sistema deve permitir editar o evento.
RF-36	O sistema deve permitir excluir a pista.
RF-37	O sistema deve permitir adicionar vídeos dos locais de esporte.

4.2 Requisitos não funcionais

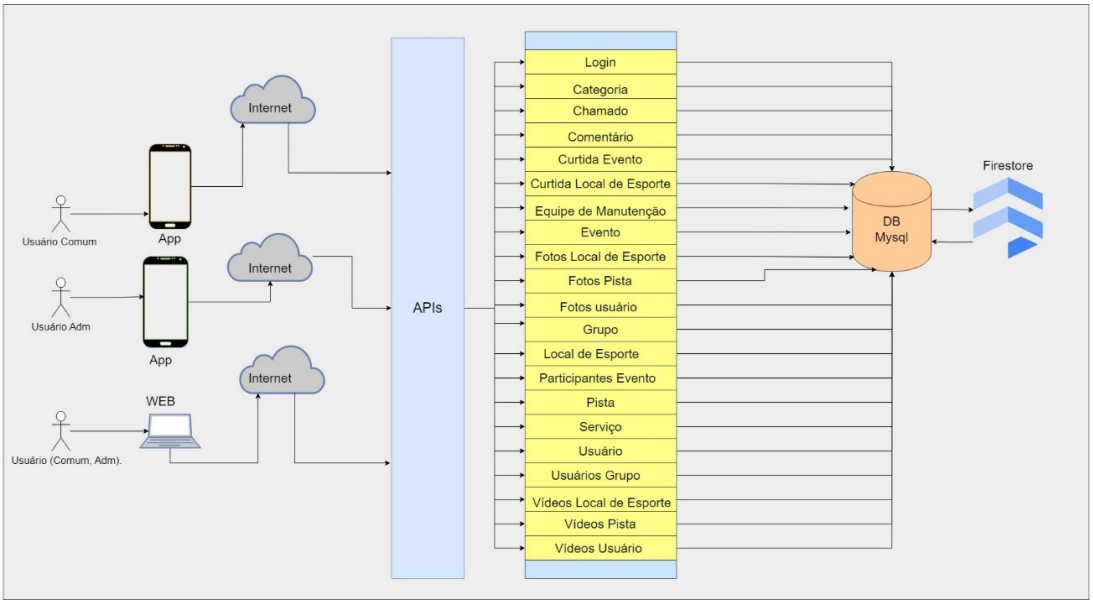
A tabela 4.2 descreve alguns atributos do sistema. Essas características representam as funcionalidades não funcionais.

Tabela 4.2: Requisitos Não Funcionais (RF)

Id	Requisitos
RNF-01	O sistema deve ser seguro
RNF-02	O sistema deve incluir verificação de email
RNF-03	Os usuários precisam se autenticar para acessar o sistema
RNF-04	JSON Web Token (JWT) com role de perfil
RNF-05	Segregação de perfis por usuário

4.3 Topologia da aplicação

Figura 4.1: Topologia da aplicação.



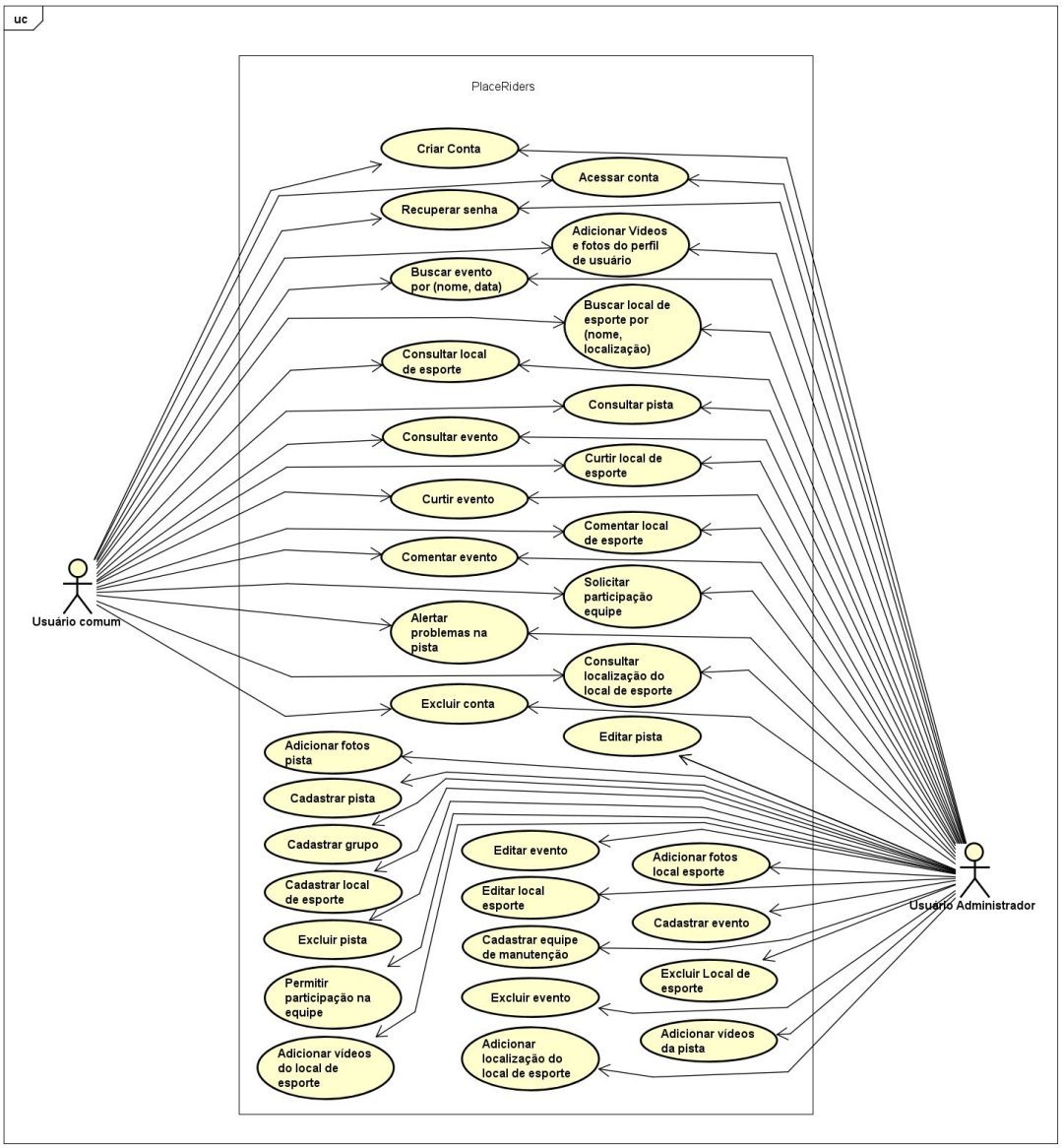
Fonte: Autor.

A Figura 4.1 apresenta a topologia do software proposto, ilustrando os diferentes dispositivos compatíveis com o aplicativo. Os usuários acessam o aplicativo em seus dispositivos, que realizam uma requisição via internet para uma API no backend. O servidor verifica se o usuário está autenticado e, caso positivo, processa a solicitação. O sistema utiliza dois bancos de dados. O banco relacional armazenar informações inter-relacionadas, enquanto o banco não relacional é destina ao armazenamento de mídias, como imagens, vídeos, mensagens e comentários. Após processar a solicitação, o servidor retorna uma resposta ao usuário.

4.4 Casos de uso

Logo abaixo, na figura 4.2, exploramos os requisitos do sistema com o diagrama de caso de uso. No aplicativo, temos dois tipos de usuários. O primeiro tipo é o usuário comum, que acessa o aplicativo com o objetivo de pesquisar locais para praticar esportes com a bicicleta e se informar sobre os eventos nesses locais. O segundo tipo de usuário é o administrador, que é responsável por um ou mais locais para a prática de esportes. Este pode adicionar eventos, pistas e equipes nos locais que gerencia, e essas informações serão compartilhadas com os usuários do aplicativo. Para que o usuário comum se torne um administrador, basta adicionar um local para praticar esporte. Portanto, o usuário que adicionou o local fica responsável pela administração desse local.

Figura 4.2: Caso de uso.

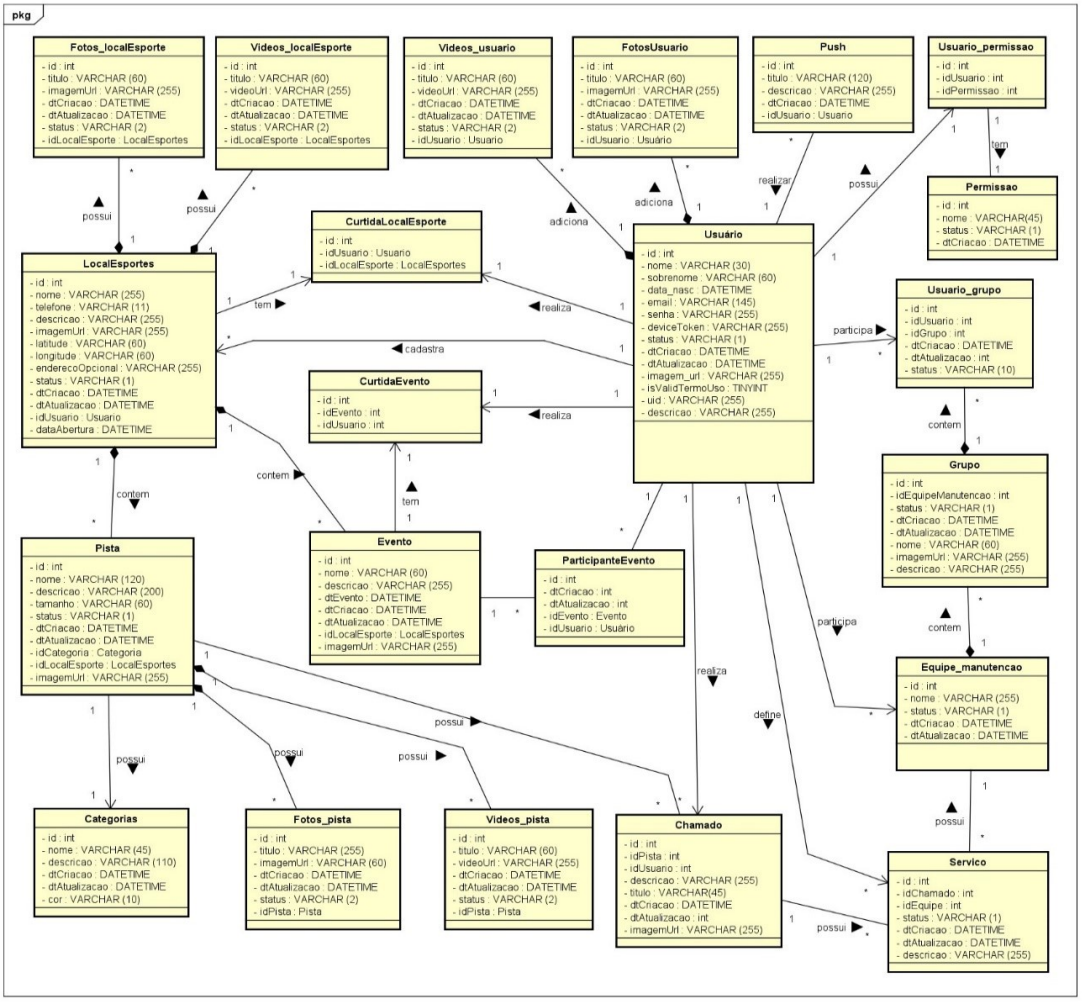


Fonte: Autor.

4.5 Modelo de domínio

A seguir, na Figura 4.3, temos o diagrama de classes que reflete o modelo de domínio. Podemos analisar de forma clara a estrutura lógica da aplicação. Dessa forma, temos uma visão de alto nível da abstração do sistema.

Figura 4.3: Modelo de domínio .



Fonte: Autor.

4.6 Requisitos atendidos

Caso de uso - Criar conta	
Requisito Atendido	RF-08
Atores	Usuário comum, Usuário Administrador
Pré-condição	Usuário comum ou administrador com o aplicativo, com acesso à internet e sem conta. O e-mail cadastrado deve ser válido.
Cenário principal	• O usuário acessa a tela de cadastro para criar uma conta nova. • O usuário informa todos os dados requisitados no formulário: nome, sobrenome, data de nascimento, e-mail, senha, confirmação de senha, e concorda com os termos de uso e políticas do aplicativo. • O sistema registra o usuário. • O usuário deve acessar o e-mail e validar a conta criada no aplicativo. • O aplicativo informa que a validação foi bem-sucedida.
Cenário alternativo	• O usuário não preencheu todos os campos e confirma o cadastro. – O sistema informa os campos que não foram preenchidos. • O usuário preencheu todos os campos, mas informou uma senha diferente. – O sistema informa que as senhas são diferentes. • O usuário preencheu todos os campos, mas não aceitou as políticas do aplicativo. – O sistema exibe um alerta que é necessário aceitar os termos de uso e as políticas. • O usuário preencheu todos os requisitos. – O sistema informa que o cadastro foi bem-sucedido, mas para concluir é necessário confirmar o e-mail enviado pelo sistema para validar a conta.
Pós-condição	A conta do usuário é criada no sistema.

Caso de uso – Acessar conta	
Requisito Atendido	RF-12
Atores	Usuário comum, Usuário administrador.
Pré-condição	Possuir uma conta já verificada no aplicativo. O aparelho deve ter acesso à internet.
Cenário principal	• O usuário acessa o aplicativo e vai para a tela de login. • O usuário preenche os campos de e-mail e senha. • O usuário clica em entrar para acessar o aplicativo. • O aplicativo redireciona o usuário para a tela inicial.
Cenário alternativo	• O usuário clica em "Entrar"no sistema, mas não confirmou o e-mail de verificação enviado pelo aplicativo. – O aplicativo mostra uma mensagem de erro, informando que foi enviado um e-mail um link para validar a conta. • O usuário informou e-mail ou senha incorretos. – O sistema exibe uma mensagem de erro, informando que a senha ou e-mail não confere. • O usuário esqueceu a senha de acesso. – O usuário deve clicar em "Esqueci minha senha"para iniciar o processo de recuperação de senha. • O usuário não possui conta. – O usuário deve clicar na opção "Criar nova conta"para iniciar o processe de criação de uma nova conta.
Pós-condição	O usuário tem acesso às funcionalidades do aplicativo.

Caso de uso – Recuperar senha	
Requisito Atendido	RF-09
Atores	Usuário comum, Usuário administrador.
Pré-condição	O usuário deve possuir cadastro no aplicativo. O usuário deve informar o mesmo e-mail usado para criar a conta. O aparelho deve ter acesso à internet.
Cenário principal	• O usuário clica em "Esqueci senha"e é direcionado para a tela de recuperação de senha. • O usuário informa o e-mail usado para acessar o aplicativo. • O usuário confirma a solicitação de uma nova senha. • O sistema envia um e-mail com um código necessário para a recuperação da conta. • O usuário informa o código de recuperação da conta e confirma. • O usuário é direcionado para a tela de nova senha.
Cenário alternativo	• O usuário não informou um e-mail cadastrado no sistema. – O sistema informa que o e-mail não existe no aplicativo.
Pós-condição	O usuário tem permissão para alterna a senha.

Caso de uso – Adicionar vídeos e fotos do perfil de usuário	
Requisito Atendido	RF-13
Atores	Usuário comum, Usuário administrador
Pré-condição	O usuário deve estar logado no sistema. O usuário deve permitir que o aplicativo acesse os arquivos do dispositivo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa a tela de perfil. • O usuário clica em "Configurações"e abre a opção de editar para adicionar a foto de perfil. • Na tela de perfil, o usuário clica no ícone de adicionar foto para incluir uma foto em seu perfil. • Na tela de perfil, o usuário clica no ícone de adicionar vídeos para incluir vídeo. • O aplicativo exibe a foto de perfil, as fotos adicionadas e os vídeos adicionados no perfil.
Cenário alternativo	• O usuário não deu permissão para acessar os arquivos do dispositivo. – O sistema mostra um pedido de permissão para acessar os arquivos do dispositivo. • O usuário selecionou um vídeo maior de 30MB. – O sistema mostra um alerta que o vídeo deve ser menor.
Pós-condição	As fotos e os vídeos são adicionados com sucesso ao perfil.

Caso de uso – Buscar evento	
Requisito Atendido	RF-14
Atores	Usuário comum, Usuário administrador
Pré-condição	O usuário deve estar logado no sistema. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa a tela inicial do aplicativo e seleciona a opção “Eventos”. • O usuário informa o nome ou a data do evento desejado na opção "Buscar". • O aplicativo busca os eventos.
Cenário alternativo	• O usuário informou um nome ou data inexistente. – O aplicativo não exibe nenhum evento.
Pós-condição	O aplicativo exibirá os eventos buscados pelo usuário.

Caso de uso – Buscar local de esporte por (nome, localização)	
Requisito Atendido	RF-15
Atores	Usuário comum, Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa a tela inicial e seleciona a opção “Locais”. • O usuário informa o nome ou o endereço do local que deseja buscar. • O aplicativo busca os locais.
Cenário alternativo	• O usuário informou um nome ou endereço inexistente. – O aplicativo não exibe nenhum local para praticar esportes. • Nenhum local foi encontrado. – Para realizar uma nova busca, basta apagar as informações no campo de pesquisa.
Pós-condição	O aplicativo exibe os locais encontrados.

Caso de uso – Consultar local de esporte	
Requisito Atendido	RF-16, RF-03
Atores	Usuário comum, Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa a tela inicial e seleciona a opção ‘Locais’. • O usuário clica no local que deseja consultar. • O aplicativo exibe o local para esportes.
Cenário alternativo	• O aplicativo não tem acesso à internet. – O usuário deve permitir o acesso à internet.
Pós-condição	O aplicativo abre a tela do local de esporte, exibindo todas as informações sobre o local, como a localização, descrição, eventos e as pistas disponíveis.

Caso de uso – Consultar pista	
Requisito Atendido	RF-04
Atores	Usuário comum, Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa a tela inicial na opção "Pistas". • O sistema exibe uma lista de categorias de pistas. • O usuário seleciona a categoria de pista desejada. • O sistema exibe as pistas de acordo com a categoria selecionada pelo usuário. • O usuário clica em uma pista que deseja ver. • O sistema exibe todas as informações da pista e do local ao qual ela pertence.
Cenário alternativo	• O usuário pode acessar as pistas pelo local de esporte. – O usuário clica no local de esporte para exibir suas informações e ver as pistas disponíveis.
Pós-condição	O aplicativo abre a tela com as informações da pista.

Caso de uso – Consultar evento	
Requisito Atendido	RF-05
Atores	Usuário comum, usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa a tela inicial do aplicativo e seleciona a opção “Eventos”. • O sistema exibirá os eventos disponíveis. • O usuário seleciona o evento desejado.
Cenário alternativo	• O usuário clica no local de esporte. – O local de esporte exibe os eventos disponíveis.
Pós-condição	O aplicativo permite ao usuário consultar as informações sobre o evento selecionado.

Caso de uso – Curtir local de esporte	
Requisito Atendido	RF-17
Atores	Usuário comum, Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa a tela inicial e seleciona a opção 'Locais'. • O sistema exibe todos os locais cadastrados no aplicativo. • O usuário clica no ícone de "Curtir".
Cenário alternativo	• O dispositivo não tem acesso à internet. – O aplicativo não exibirá os locais e será impossível curtir o local.
Pós-condição	O usuário pode conferir a sua reação na lista de curtidas do local.

Caso de uso – Curtir evento	
Requisito Atendido	RF-18
Atores	Usuário comum, Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa a tela inicial e seleciona a opção "Eventos". • O sistema exibe os eventos cadastrados. • O usuário clica no ícone de "Curtir".
Cenário alternativo	• O dispositivo não tem acesso à internet. – Nenhuma ação é registrada no sistema.
Pós-condição	O aplicativo exibe a curtida do usuário na lista de curtidas.

Caso de uso – Comentar local de esporte	
Requisito Atendido	RF-19
Atores	Usuário comum, Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa a tela inicial e seleciona a opção "Locais". • O sistema exibe os locais para prática de esportes. • O usuário clica na opção "Comentar"para adicionar um comentário. • Após fazer o comentário, o usuário clica no ícone para enviá-lo.
Cenário alternativo	• O usuário não tem acesso à internet. – O sistema não permite realizar o comentário enquanto o dispositivo não estiver conectado à internet.
Pós-condição	Um comentário é realizado no local de esporte.

Caso de uso – Comentar evento	
Requisito Atendido	RF-20
Atores	Usuário comum, Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa a tela inicial e seleciona a opção "Eventos". • O sistema exibe os eventos. • O usuário clica na opção "Comentar"para adicionar um comentário. • Após fazer o comentário, o usuário clica no ícone para enviá-lo.
Cenário alternativo	• O usuário não tem acesso à internet. – O sistema não permite realizar o comentário enquanto o dispositivo não estiver conectado à internet.
Pós-condição	O comentário é realizado no evento.

Caso de uso – Solicitar participação	
Requisito Atendido	RF-07
Atores	Usuário comum, Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o aplicativo na tela inicial em "Locais". • O aplicativo exibe os locais para praticar esportes. O usuário clica no local em que deseja solicitar a participação. • O aplicativo exibe as informações sobre o local. Logo abaixo, no local selecionado, há a opção 'Equipes'. • O usuário clica na equipe em que deseja participar e o sistema exibe os grupos que a equipe possui. • O usuário clica no ícone para enviar uma solicitação para participar da equipe.
Cenário alternativo	• O usuário é reprovado. – O usuário administrador do local de esporte não aprova a solicitação do usuário que deseja fazer parte da sua equipe de manutenção.
Pós-condição	O usuário é aceito pela equipe.

Caso de uso – Alertar problemas na pista	
Requisito Atendido	RF-06
Atores	Usuário comum, Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o aplicativo na tela inicial e seleciona a opção "Locais". • O usuário clica em um local de esporte, e o sistema exibe as informações do local e as pistas disponíveis. • O usuário clica na pista para relatar um problema. • O aplicativo exibe a tela da pista. O usuário clica nos três pontos na barra superior. • O aplicativo exibe a tela onde o usuário informa o problema. • O usuário preenche os campos de título e descrição do problema e clica em "Enviar". • O aplicativo notifica que o envio foi realizado com sucesso.
Cenário alternativo	• O aplicativo não tem acesso à internet. – O usuário deve permitir o acesso à internet no dispositivo.
Pós-condição	O sistema exibe um alerta de problema na pista na tela do local de esporte do administrador. O administrador encaminha a solução para uma equipe de manutenção.

Caso de uso – Consultar localização do local de esporte	
Requisito Atendido	RF-22
Atores	Usuário comum, usuário administrador.
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o aplicativo na tela inicial e seleciona a opção "Locais". • O usuário clica no local que deseja consultar. • O aplicativo exibe as informações sobre o local e sua localização.
Cenário alternativo	• O dispositivo não possui acesso à internet. – O usuário deve fornece o acesso à internet.
Pós-condição	O sistema exibe as informações do local e sua localização no mapa.

Caso de uso – Excluir conta	
Requisito Atendido	RF-23
Atores	Usuário comum, Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa a tela de perfil. • O aplicativo exibe a tela de perfil, onde em "Configurações"há a opção de excluir a conta. • O usuário clica em "Excluir conta"e confirma a solicitação.
Cenário alternativo	• O dispositivo não possui acesso à internet. – O usuário deve conectar o dispositivo à internet.
Pós-condição	O sistema exclui a conta do usuário.

Caso de uso – Editar pista	
Requisito Atendido	RF-24
Atores	Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o menu lateral e clica na opção "Meus Locais". • O usuário seleciona o local que deseja editar a pista. • O aplicativo exibe as pistas disponíveis nesse local. O usuário clica em "Editar pista". • Após realizar as alterações, o usuário clica em "Salvar"para confirmar as edições no local.
Cenário alternativo	• O dispositivo não tem acesso à internet. – O usuário deve fornecer acesso à internet para o dispositivo.
Pós-condição	Após realizar as alterações, o aplicativo salva os dados editados.

Caso de uso – Adicionar fotos pista	
Requisito Atendido	RF-25
Atores	Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o menu lateral e clica na opção "Meus Locais. • O usuário seleciona o local onde deseja editar a pista. • O aplicativo exibe as pistas disponíveis nesse local. O usuário clica na pista à qual deseja adicionar fotos. • O aplicativo exibe as informações da pista e a opção de adicionar fotos. • O usuário clica em "Adicionar Foto". • O aplicativo exibe a tela de adiciona fotos. • O usuário insere um título para a foto que deseja adicionar. • O usuário clica em "Próximo"para selecionar a foto. • O aplicativo abre os arquivos, e o usuário seleciona a foto que deseja enviar.
Cenário alternativo	• O usuário não permitiu o acesso aos arquivos do dispositivo. – O usuário deve permitir que o aplicativo acesse os arquivos do dispositivo.
Pós-condição	O aplicativo adiciona a foto à pista.

Caso de uso – Cadastrar pista	
Requisito Atendido	RF-26
Atores	Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet. Antes de adicionar uma pista, é necessário cadastrar um local de esporte.
Cenário principal	• O usuário acessa o menu lateral e clica na opção "Meus Locais". • O usuário seleciona o local onde deseja adicionar uma nova pista. • O aplicativo exibe os locais para praticar esportes. Logo abaixo do local selecionado pelo usuário, há a opção de adicionar pistas. • O usuário clica na opção "Cadastrar Pista"para cadastrar uma nova pista. • O usuário preenche os campos requisitados no formulário: nome, categoria, descrição e tamanho da pista. • Após preencher todos os campos, o usuário clica em "Salvar"para concluir o cadastro.
Cenário alternativo	• O usuário não cadastrou um local de esporte. – Para cadastrar uma pista, primeiro é necessário cadastrar um local de esporte.
Pós-condição	A pista é cadastrada no local de esportes.

Caso de uso – Cadastrar grupo	
Requisito Atendido	RF-27
Atores	Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet. Antes de adicionar um grupo é necessário cadastrar um local de esporte e ter uma equipe de manutenção cadastrada nesse local.
Cenário principal	• O usuário acessa o menu lateral e clica na opção "Meus Locais". • O usuário seleciona o local onde deseja adicionar um novo grupo. • O aplicativo exibe os locais para praticar esportes. Logo abaixo do local selecionado pelo usuário, há a opção de adicionar equipe. • O usuário clica em uma equipe onde deseja adicionar um grupo. • O aplicativo exibe a tela da equipe de manutenção, onde logo abaixo há a opção de criação de grupos. • O usuário clica em “criar grupo”, o aplicativo exibe a tela de criação de grupo. • O usuário preenche os requisitos de criação: nome e descrição do grupo. • O usuário clica em "Salvar"para finalizar o cadastro do grupo.
Cenário alternativo	• O usuário não criou uma equipe. – É necessário realizar o cadastro de uma equipe.
Pós-condição	O usuário cadastra um novo grupo na equipe.

Caso de uso – Cadastrar local de esporte	
Requisito Atendido	RF-01
Atores	Usuário administrador, Usuário comum.
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o menu lateral e clica na opção "Meus Locais". • O usuário clica em "Cadastrar Local de Esporte". • O usuário preenche os dados requisitados: nome, telefone, descrição, data de criação e localização do local. • O usuário clica no botão "Salvar"para concluir o cadastro.
Cenário alternativo	• O aplicativo não realizou o cadastro. – É preciso ter acesso à internet – O usuário deve preencher todos os campos do formulário.
Pós-condição	O local de esporte foi cadastrado com sucesso.

Caso de uso – Excluir pista	
Requisito Atendido	RF-36
Atores	Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o menu lateral e clica na opção "Meus Locais". • O aplicativo exibe a tela de "Meus Locais"; o usuário clica no local onde deseja excluir uma pista. • O aplicativo exibe as informações do local cadastrado; logo abaixo, estão listadas as pistas cadastradas no local. • O usuário clica no ícone lixeira em vermelho para realizar a exclusão da pista • O aplicativo exibe uma mensagem solicitando que o usuário confirme se realmente deseja excluir a pista. • O usuário confirma a exclusão.
Cenário alternativo	• O usuário não tem acesso à internet. – Para realizar a exclusão da pista é necessário que o usuário conecte o dispositivo à internet.
Pós-condição	A pista foi excluída com sucesso.

Caso de uso – Permitir participação na equipe	
Requisito Atendido	RF-21
Atores	Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o menu lateral e clica na opção "Meus Locais". • O aplicativo exibe a tela dos locais cadastrados. • O usuário clica em um local para ver as solicitações enviadas. • O aplicativo exibe as informações do local; logo abaixo, estão as equipes cadastradas e seus grupos. • O usuário clica em um grupo da equipe. O aplicativo exibe as informações do grupo e as solicitações de participação. • O usuário clica na opção "Solicitações de Grupos". • O aplicativo exibe a tela com os usuários que enviaram solicitações. • O usuário administrador decide se permite a participação do usuário no grupo.
Cenário alternativo	• O dispositivo não tem acesso à internet. – O usuário deve garantir que o dispositivo esteja conectado à internet.
Pós-condição	O usuário é adicionado ao grupo da equipe de manutenção do local de esportes.

Caso de uso – Adicionar vídeos do local de esporte	
Requisito Atendido	RF-37
Atores	Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet. O vídeo deve ter menos de 30MB.
Cenário principal	• O usuário acessa o menu lateral e clica na opção "Meus Locais". • O usuário seleciona o local onde deseja adicionar os vídeos. • O aplicativo exibe as informações do local; logo abaixo, há a opção de adicionar vídeos ao local de esportes. • O usuário clica na opção "Adicionar Vídeo". O aplicativo solicita um título para o vídeo. • O usuário clica na opção "Próximo", e o aplicativo encaminha para a tela onde é feito o upload do vídeo. • O usuário clica na opção "Salvar"para concluir o processo.
Cenário alternativo	• O usuário enviou um vídeo maior que 30 MB. – O aplicativo exibe uma mensagem de erro informando que o vídeo deve ter menos de 30MB.
Pós-condição	O vídeo foi adicionado ao local de esportes com sucesso.

Caso de uso – Editar evento	
Requisito Atendido	RF-35
Atores	Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o menu lateral e clica na opção "Meus Locais". • O usuário seleciona o local onde deseja editar um evento. • O sistema exibe os eventos cadastrados no local. • O usuário clica na opção de editar no evento. • Após realizar as alterações no evento, o usuário clica na opção "Salvar".
Cenário alternativo	• O dispositivo não tem acesso à internet. – O usuário deve fornecer acesso à internet para que seja possível realizar as alterações.
Pós-condição	O evento é atualizado como sucesso.

Caso de uso – Editar local de esporte	
Requisito Atendido	RF-34
Atores	Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o menu lateral e clica na opção "Meus Locais". • O aplicativo exibe os locais cadastrados. • O usuário clica na opção editar. • O aplicativo abre a tela de edição do local de esporte. • O usuário clica em "Salvar"para confirmar as atualizações no sistema.
Cenário alternativo	• O dispositivo não tem acesso à internet – O usuário deve fornecer acesso à internet para que seja possível realizar as alterações.
Pós-condição	O local de esporte foi atualizado com sucesso.

Caso de uso – Cadastrar equipe de manutenção	
Requisito Atendido	RF-33
Atores	Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o menu lateral e clica na opção "Meus Locais". • O usuário clica no local onde deseja adicionar a equipe de manutenção. • O aplicativo exibe o local selecionado. Logo abaixo, o usuário clica na opção "Cadastrar Equipe de Manutenção". • O aplicativo exibe a tela de cadastro de equipe. • O usuário preenche o capo requisitado: nome da equipe. • O usuário clica em "Salvar"para realizar o cadastro da equipe.
Cenário alternativo	• O aplicativo não tem acesso à internet. – O usuário deve permitir o acesso à internet.
Pós-condição	A equipe de manutenção é cadastrada com sucesso no aplicativo.

Caso de uso – Excluir evento	
Requisito Atendido	RF-32
Atores	Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o menu lateral e clica na opção 'Meus Locais'. • O aplicativo exibe os locais cadastrados. O usuário clica no local onde deseja excluir um evento. • O aplicativo exibe os eventos cadastrados no local. • O usuário clica no ícone de lixeira em vermelho para excluir o evento. • O usuário confirma a exclusão do evento.
Cenário alternativo	• O aplicativo não tem acesso à internet. – É necessário permitir o casse à internet para realizar a exclusão.
Pós-condição	O evento é excluído com sucesso.

Caso de uso – Adicionar localização do local de esporte	
Requisito Atendido	RF-31
Atores	Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o menu lateral e clica na opção "Meus Locais". • O usuário clica em "Cadastrar Local de Esporte". • O aplicativo abre a tela de cadastro do local. • O usuário seleciona uma das opções: localização atual ou selecionar no mapa. • O aplicativo exibe uma imagem do mapa com a localização selecionada.
Cenário alternativo	• O dispositivo não tem acesso à internet. – O usuário deve permitir o acesso à internet. • O usuário não aceitou as permissões para acessar a localização. – O usuário deve permitir que o aplicativo acesse a localização do dispositivo.
Pós-condição	A localização foi adicionada no local de esportes.

Caso de uso – Adicionar fotos do local de esporte	
Requisito Atendido	RF-30
Atores	Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o menu lateral e clica na opção "Meus Locais". • O usuário seleciona o local onde deseja adicionar a foto. • O aplicativo exibe as informações do local; logo abaixo, há a opção de adicionar uma foto no local de esportes. • O usuário clica na opção "Adicionar foto". O aplicativo solicita um título para a foto. • O usuário clica na opção "Próximo", e o aplicativo encaminha para a tela onde é feito o upload da imagem. • O usuário clica na opção "Salvar"para concluir o processo.
Cenário alternativo	• O dispositivo não tem acesso à internet. – O usuário deve permitir o acesso à internet.
Pós-condição	A foto foi adicionada no local de esportes.

Caso de uso – Cadastrar evento	
Requisito Atendido	RF-02
Atores	Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o menu lateral e clica na opção "Meus Locais". • O aplicativo exibe os locais cadastrados; o usuário seleciona um local onde deseja cadastrar um evento. • O usuário clica na opção "Cadastrar Evento". • O aplicativo exibe a tela de cadastro com os seguintes requisitos: nome, descrição e data em que o evento ocorrerá. • O usuário clica em "Salvar"para realizar o cadastro do evento.
Cenário alternativo	• O usuário não preencheu todos os campos requisitados. – O aplicativo exibe uma mensagem de erro informando que todos os campos devem ser preenchidos. • O dispositivo não tem acesso à internet. – O usuário deve permitir o acesso à internet.
Pós-condição	O evento é cadastrado

Caso de uso – Excluir local de esporte	
Requisito Atendido	RF-29
Atores	Usuário administrador
Pré-condição	O usuário deve estar logado no aplicativo. O dispositivo precisa ter acesso à internet.
Cenário principal	• O usuário acessa o menu lateral e clica na opção "Meus Locais". • O aplicativo exibe os locais de esporte cadastrados. • O usuário clica no ícone de lixeira vermelho para realizar a exclusão do local de esporte. • O aplicativo solicita que o usuário confirme se deseja realmente excluir o local de esporte.
Cenário alternativo	• O dispositivo não tem acesso à internet. – O usuário deve garantir que o dispositivo esteja conectado à internet.
Pós-condição	O local de esporte foi excluído com sucesso.

CAPÍTULO 5

DESENVOLVIMENTO

Logo abaixo estão as telas desenvolvidas na aplicação, que visa atender aos requisitos listados anteriormente.

5.1 Telas

Logo abaixo estão as telas desenvolvidas na aplicação, que visa atender aos requisitos listados anteriormente.

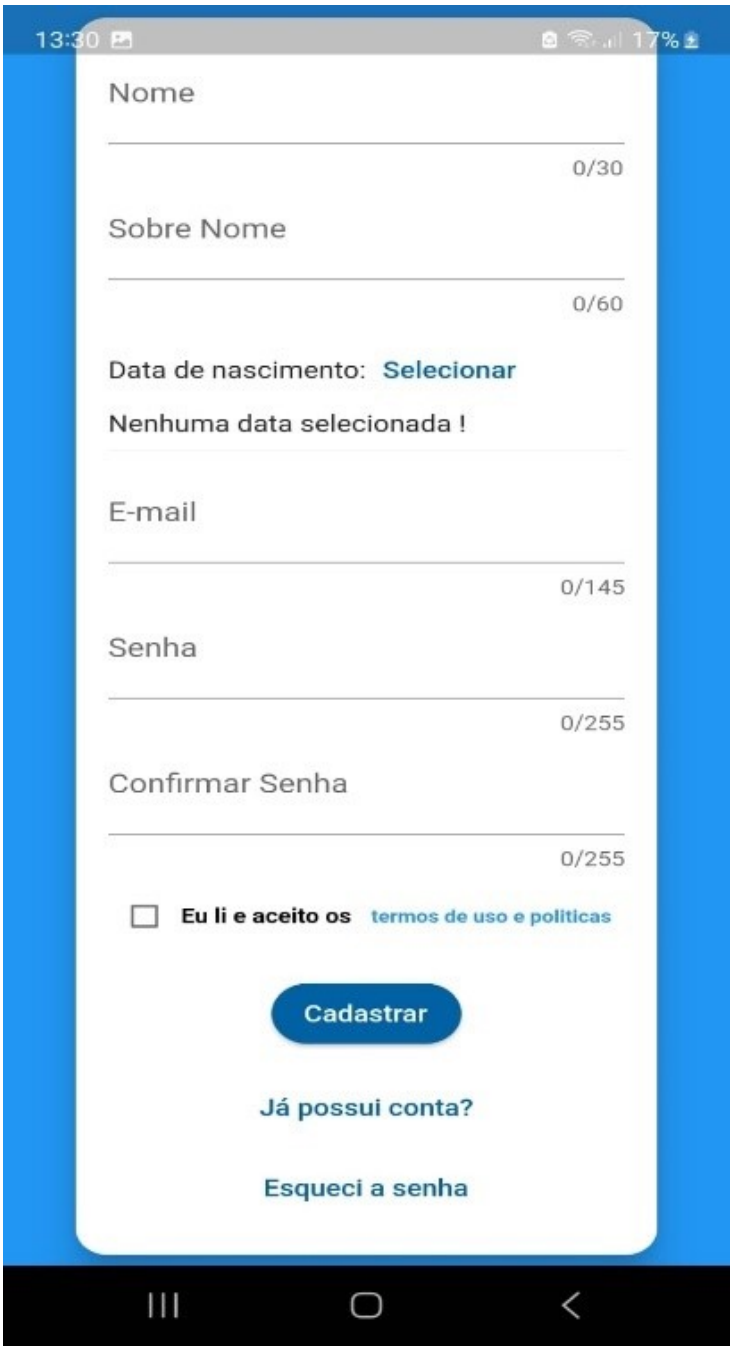
A Figura 5.1 mostra a tela de acesso para as demais funcionalidades do aplicativo exige que o usuário entre com suas credenciais. Após fornecer os dados de acesso, o aplicativo direciona o usuário para a tela inicial, onde estão disponíveis toas as funcionalidades. Caso o usuário não se lembre da senha, ele pode criar uma nova clicando na opção 'Esqueci a senha', onde o aplicativo guiará o usuário para a tela de recuperação de senha.

Figura 5.1: Acesso ao aplicativo .

Fonte: Autor.

A figura 5.2, é apresentada a tela de cadastro de usuários. O aplicativo possui dois tipos de usuários, e a tela de cadastro serve para registrar ambos. A principal diferença entre eles é que o usuário que possui um local de esporte é responsável por administrar esse local. Sendo assim, esse usuário é denominado administrador.

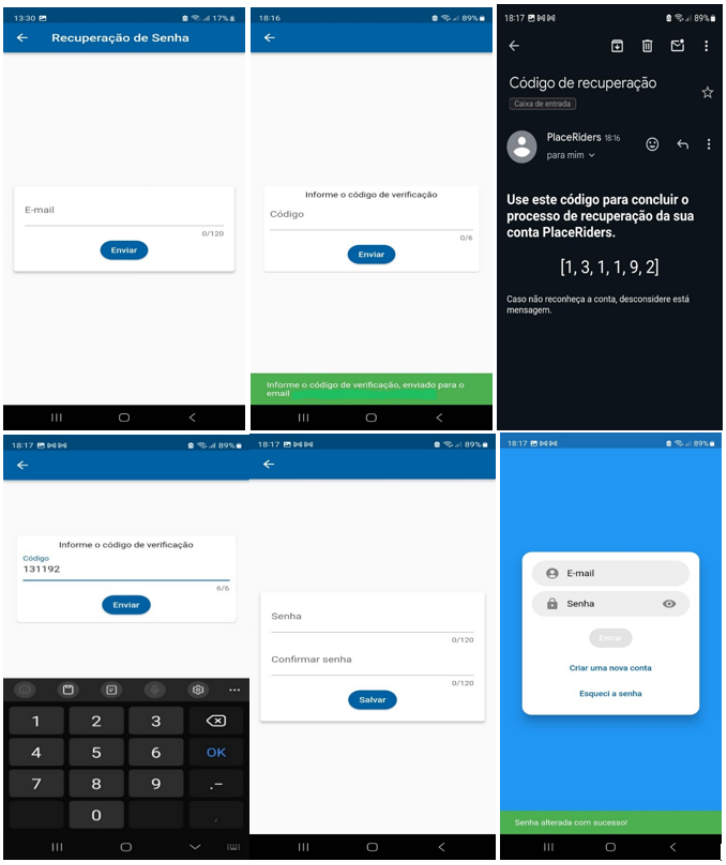
Figura 5.2: Cadastro de usuário.



Fonte: Autor.

Na figura 5.3, temos as telas responsáveis por iniciar o processo de recuperação de senha. Para isso, é necessário que o usuário informe o e-mail cadastrado. O sistema enviará um e-mail com um código de recuperação de senha, que deve ser inserido no aplicativo para confirmar a troca de senha. Se o código informado for válido, o aplicativo direcionará o usuário para a troca de troca de senha.

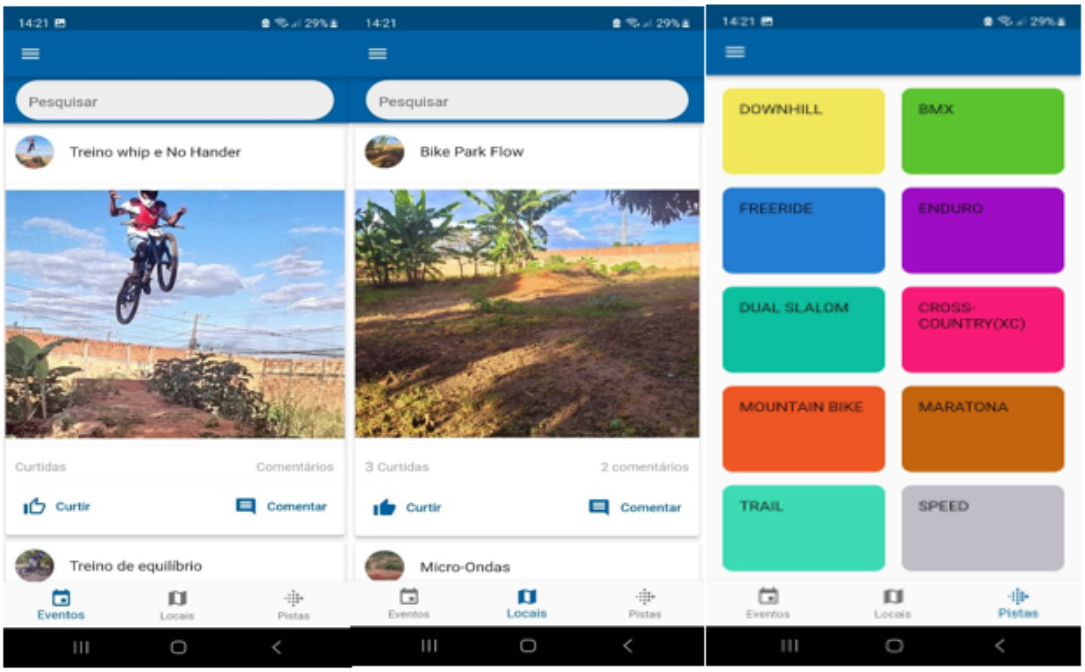
Figura 5.3: Recuperação de senha.



Fonte: Autor.

Na figura 5.4, é apresentada a tela inicial do aplicativo, composta por três seções: eventos, locais e pistas. Na primeira seção, são exibidos os eventos relacionados a bicicletas. O aplicativo permite ao usuário realizar comentários, curtir os eventos desejados e buscar eventos pelo nome. A seção de locais exibe os lugares para praticar esportes com bicicletas. Assim como na seção de eventos, é possível realizar buscas pelo nome e, especialmente, também pelo endereço. Na última seção, é exibida uma tela com pistas organizadas por categoria. Ao clicar em uma categoria desejada, o aplicativo mostra todas as pistas dessa categoria.

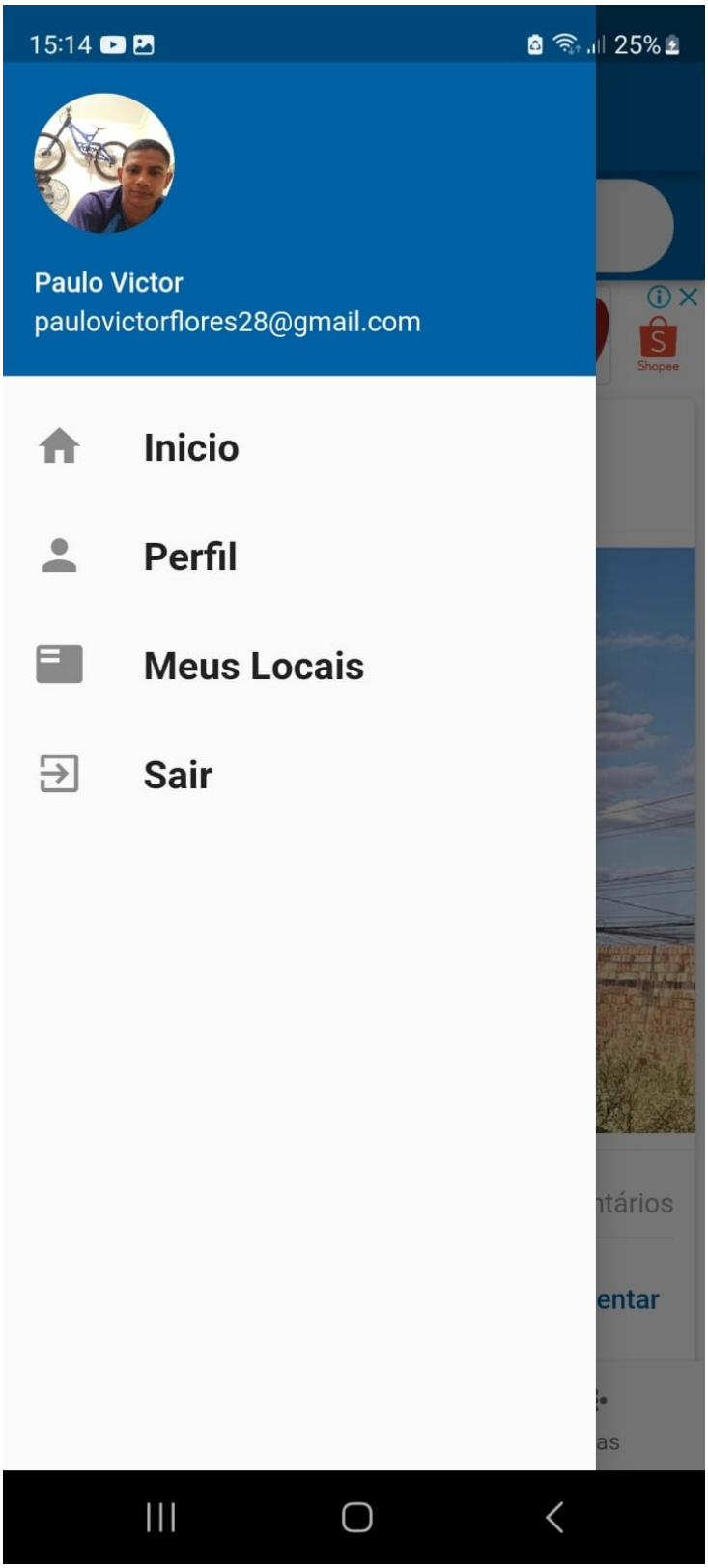
Figura 5.4: Tela inicial do aplicativo.



Fonte: Autor.

A figura 5.5 mostra a tela de menu com suas opções. O usuário pode clicar na opção "Inicio"para retornar às telas iniciais. Se o menu estiver sobreposto à tela inicial, basta arrastar o dedo para o lado para voltar. No menu, há a opção "Perfil", que exibe a tela onde o usuário pode adicionar fotos e vídeos, além de mostrar os grupos e equipes. Clicando na opção "Meus locais", o aplicativo exibe os locais cadastrados pelo usuário, que são compartilhados com os demais usuários do aplicativo. Por fim, ao clicar na opção "Sair", o aplicativo encerra a sessão do usuário.

Figura 5.5: Menu.

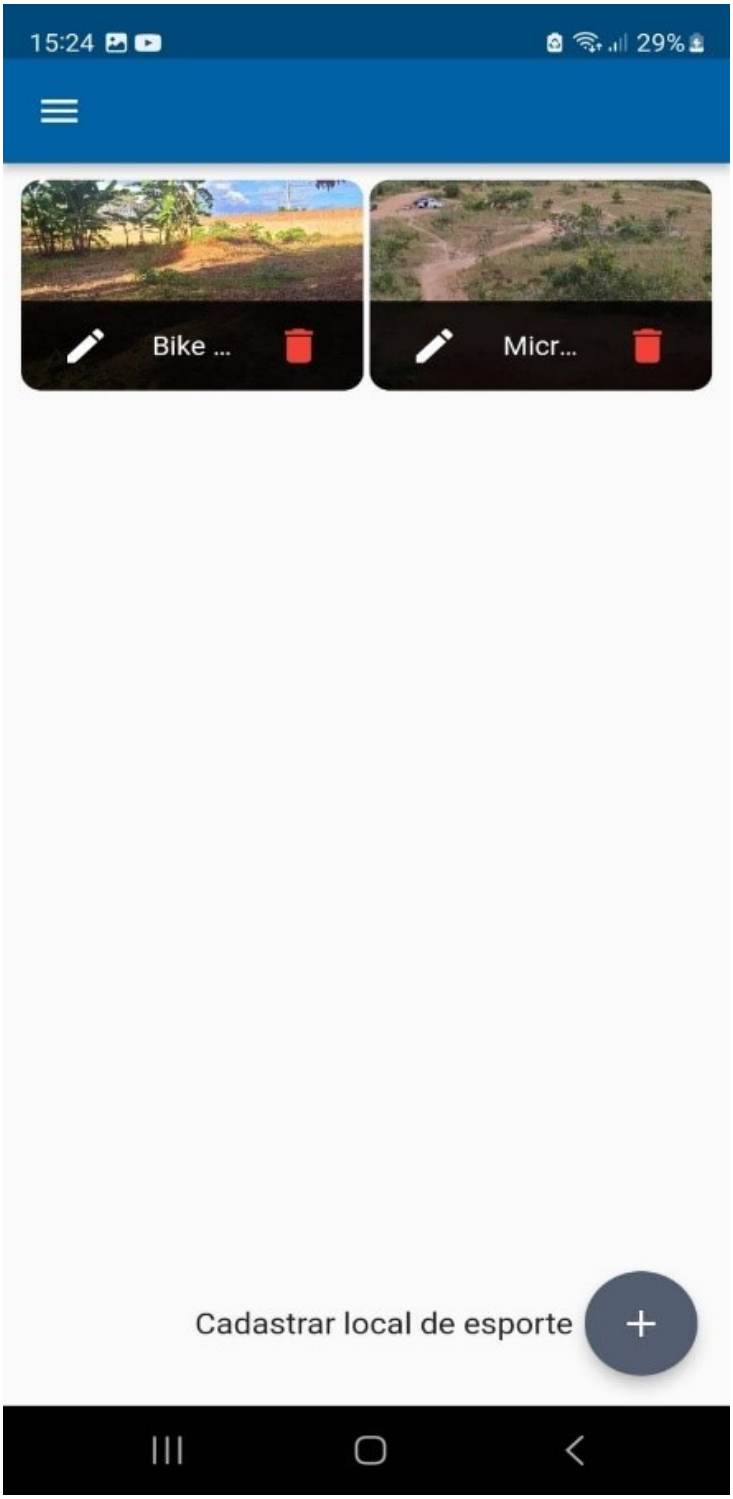


Fonte: Autor.

A figura 5.6 mostra a tela de "Meus locais". Nessa tela, é possível gerenciar um local de

esporte, permitindo cadastrar, excluir, editar e visualizar os locais já cadastrados.

Figura 5.6: Meus locais.

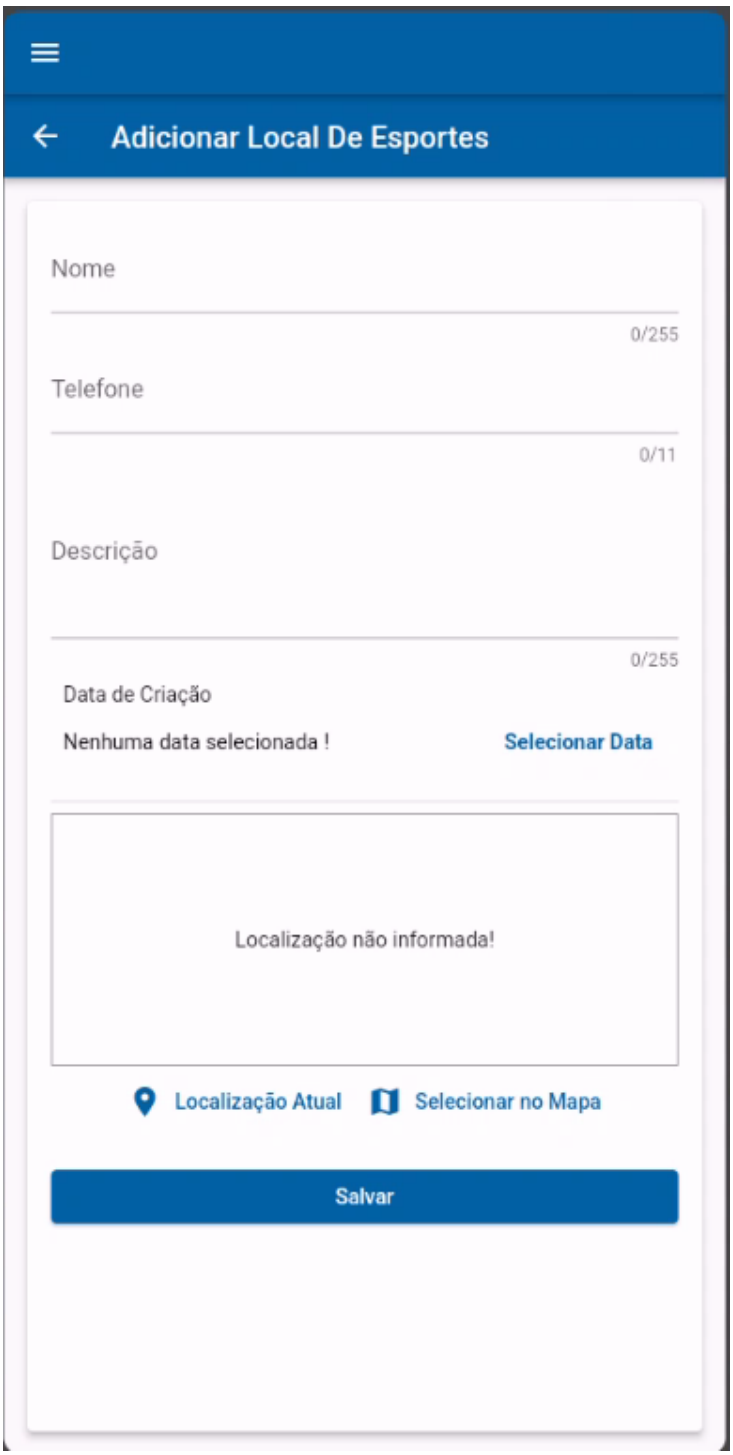


Fonte: Autor.

A Figura 5.7 mostra a tela de cadastro do local de esporte. O usuário deve preencher os campos requisitados: nome, telefone, descrição, data de criação e a selecionar a localização.

Após preencher todos os campos, basta clicar em "Salvar"para finalizar o cadastro do local.

Figura 5.7: Cadastro do local de esporte.

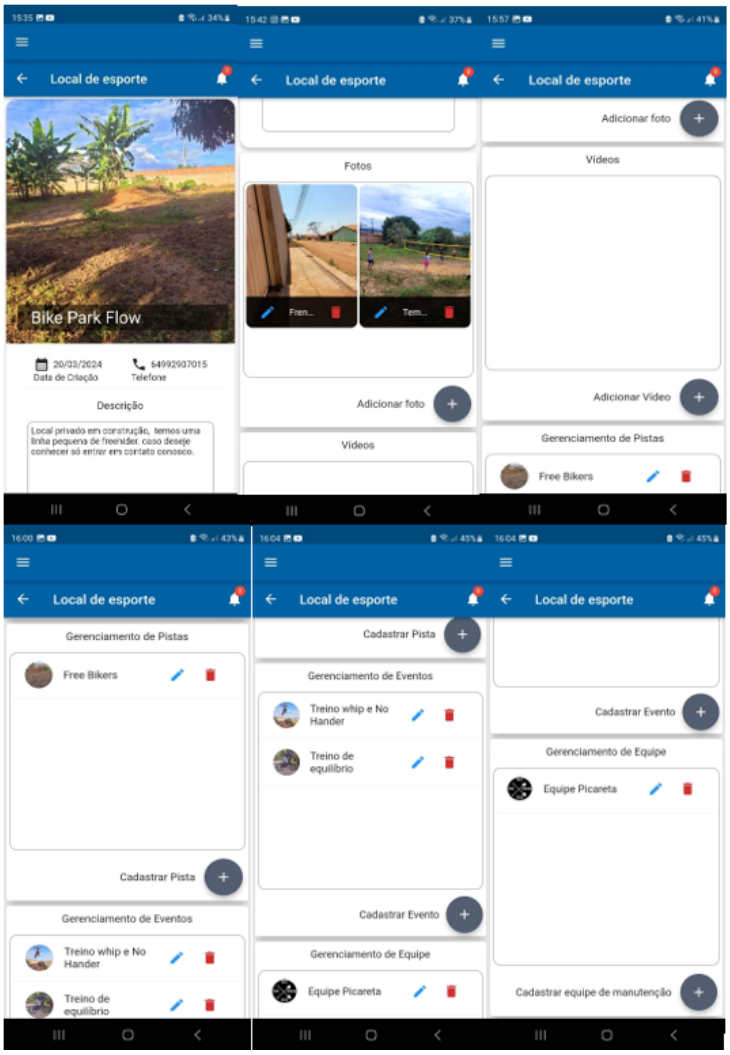


Fonte: Autor.

A Figura 5.8 ilustra a tela do local de esporte do administrador. Com o local de esporte já cadastrado, o aplicativo permite adicionar pistas, eventos, equipes de manutenção, grupos, fotos e vídeos relacionados ao local de esporte. Essa tela é especifica para o usuário que adicionou o

local de esporte, atuando como administrador desse local.

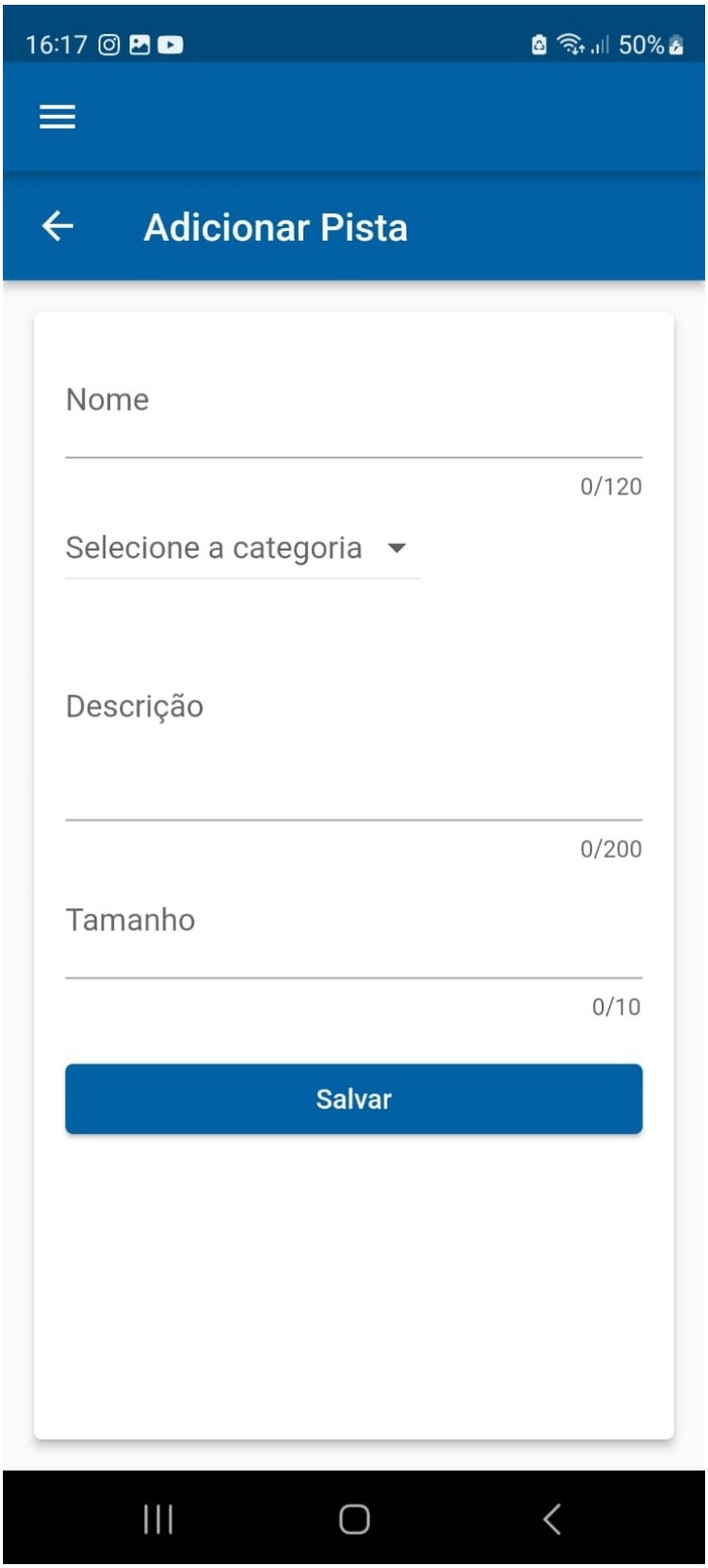
Figura 5.8: Administração do local de esporte.



Fonte: Autor.

A Figura 5.9 mostra a tela de cadastro de pista. Para cadastrar uma pista é necessário fornecer os dados requisitados: nome, categoria, descrição e tamanho. Após preencher todos os campos do formulário, o usuário deve clicar no botão "Salvar"para finalizar o cadastro da pista.

Figura 5.9: Cadastro da pista.

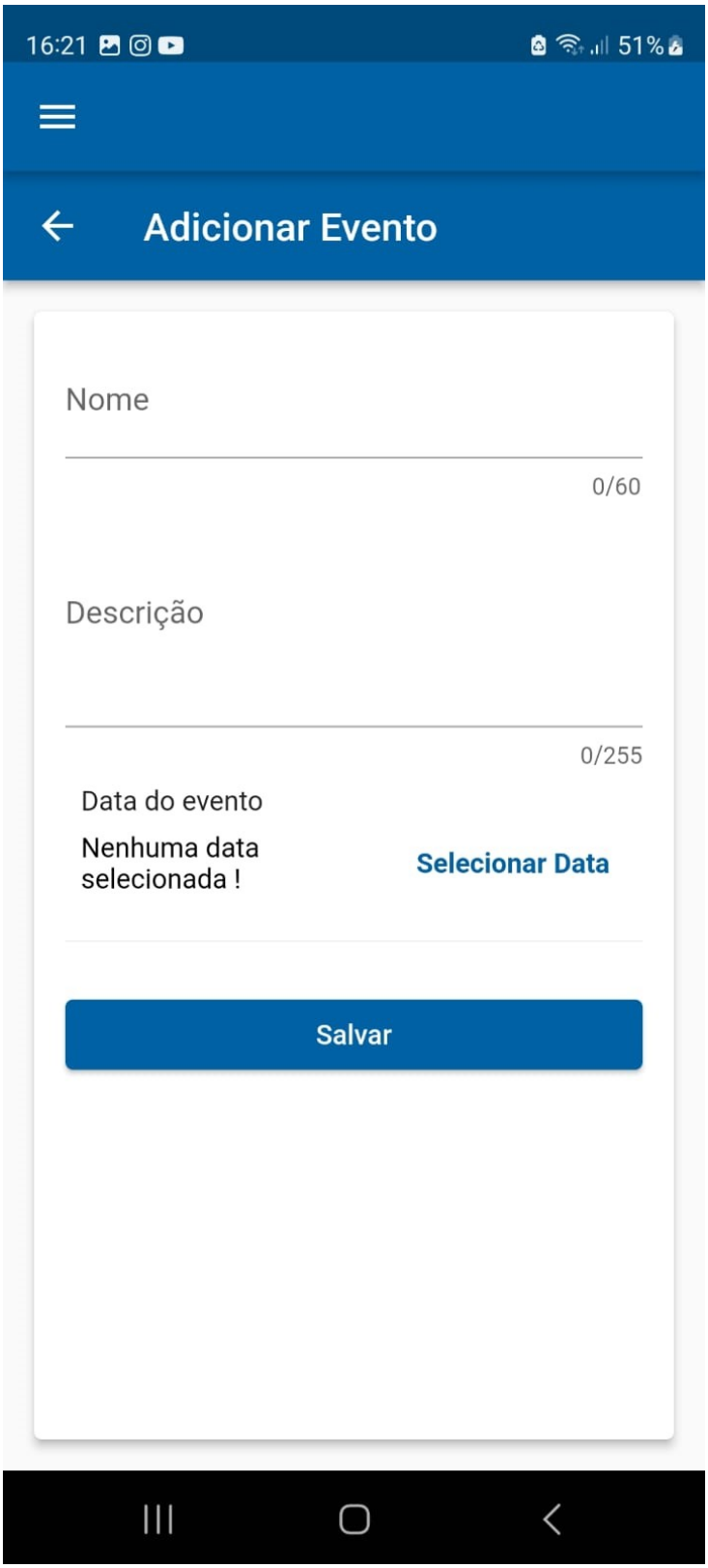


Fonte: Autor.

A Figura 5.10 ilustra a tela de cadastro de evento. Para realizar o cadastro, basta preencher

os seguintes campos: nome, descrição e data do evento. Após preencher todos os requisitos, o usuário deve clica na opção "Salvar"para finalizar o cadastro.

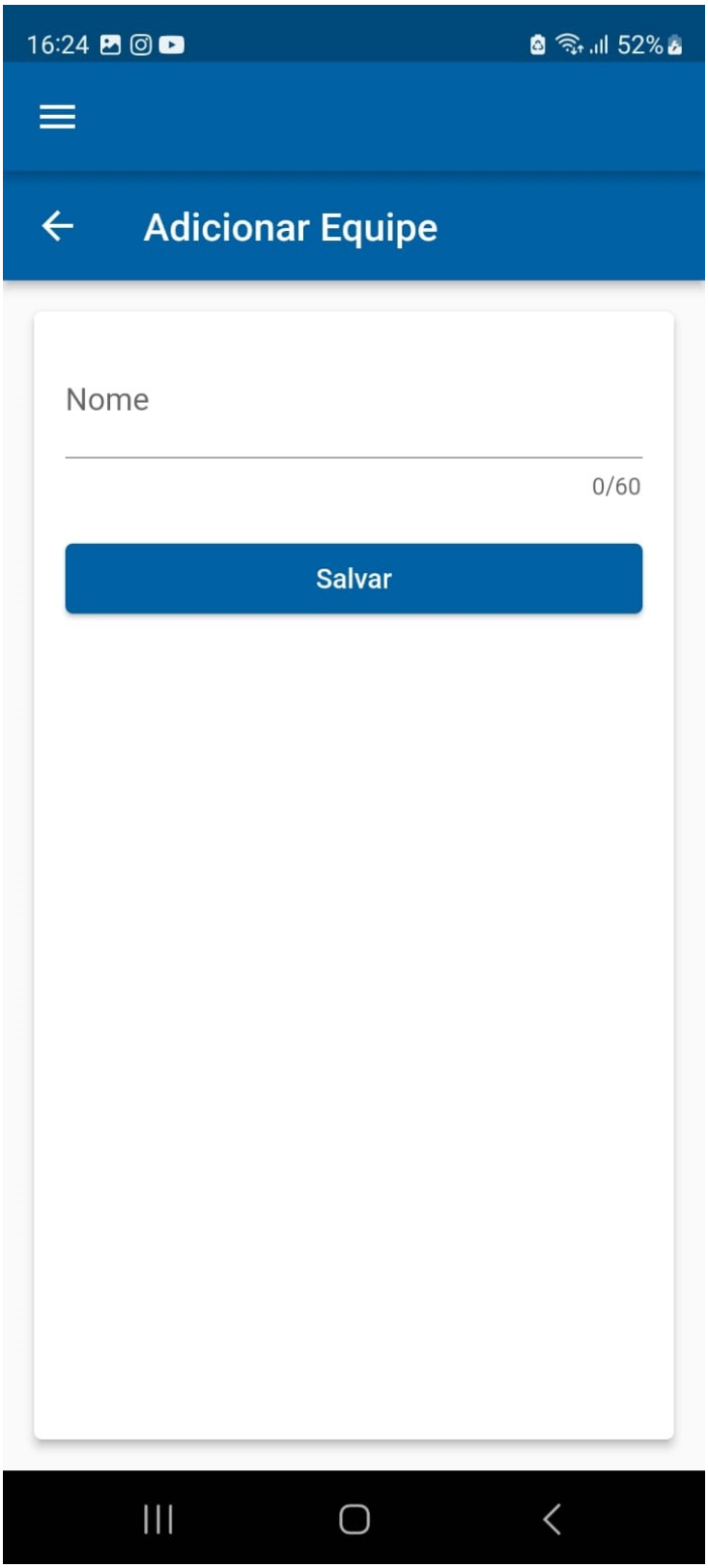
Figura 5.10: Cadastro evento.



Fonte: Autor.

A Figura 5.11 mostra a tela de cadastro de uma equipe de manutenção. Para realizar o cadastro, é necessário informar o nome da equipe e clicar no botão "Salvar"para concluir o processo.

Figura 5.11: Cadastro equipe de manutenção.

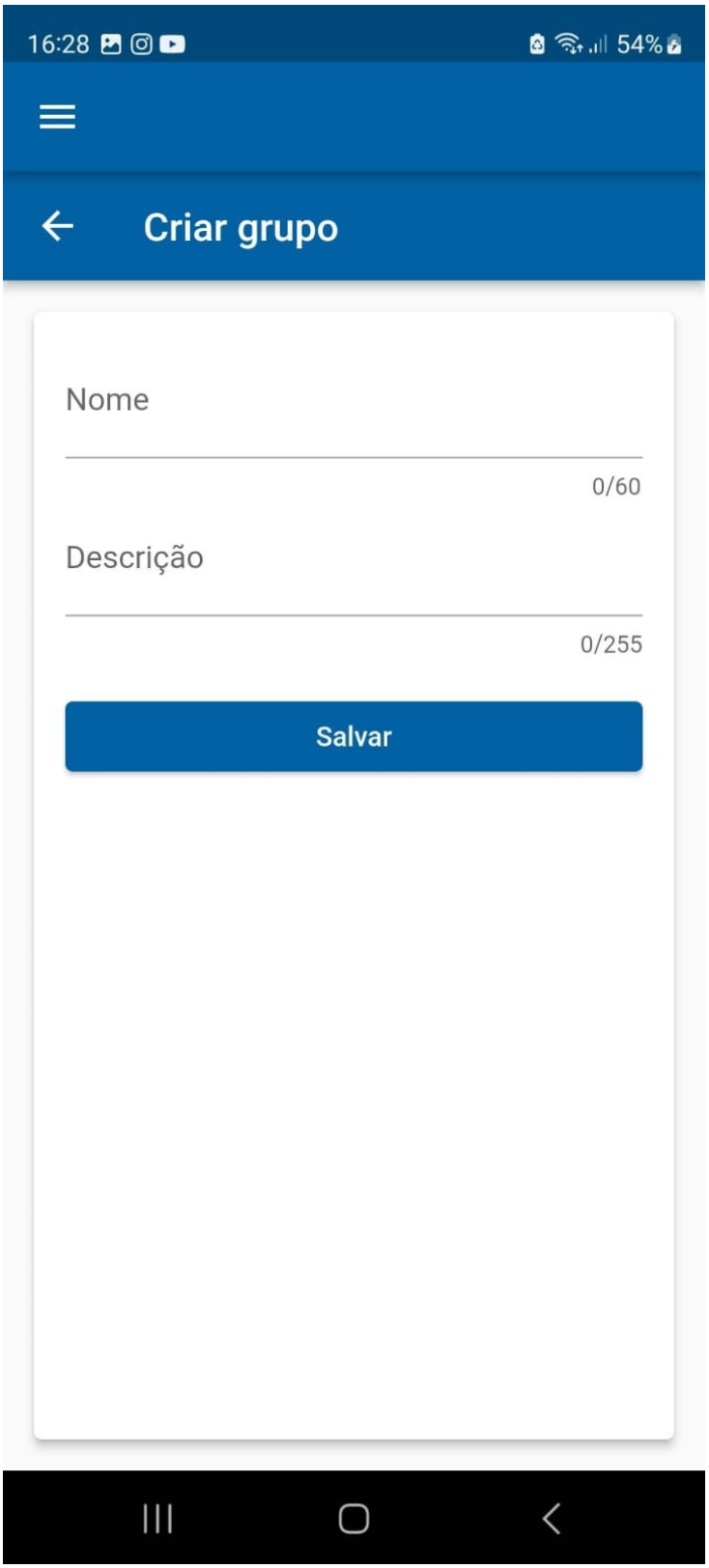


Fonte: Autor.

A Figura 5.12 mostra a tela de cadastro de grupo. Para cadastrar um grupo é necessário

preencher o formulário com os seguintes requisitos: nome e descrição. Após preencher esses requisitos, para concluir a criação do grupo, basta clicar no botão "Salvar".

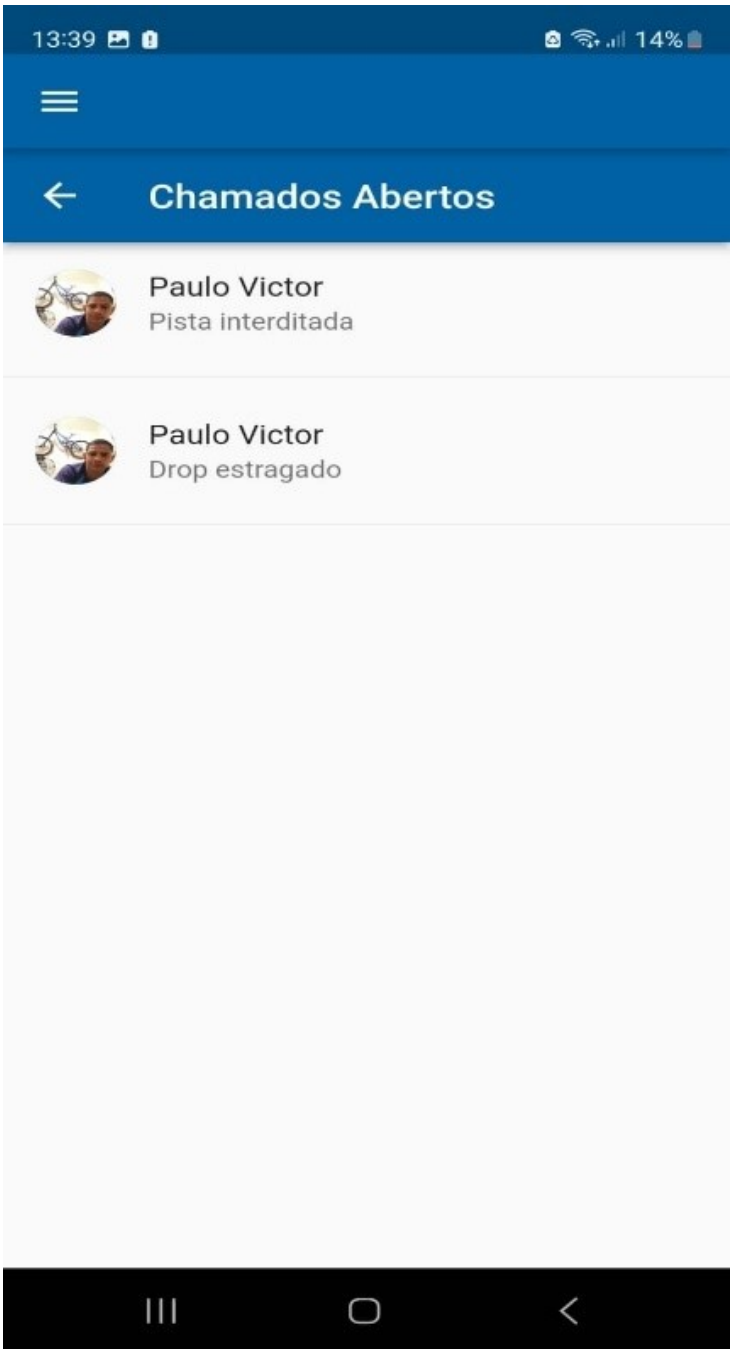
Figura 5.12: Cadastro grupo.



Fonte: Autor.

A Figura 5.13 representa a tela de Chamados do administrador do local. Nela, é possível abrir o chamado enviado e verificar se é realmente relevante. Caso não seja, ele pode ser excluído antes de ser encaminhado para uma equipe de manutenção de pista.

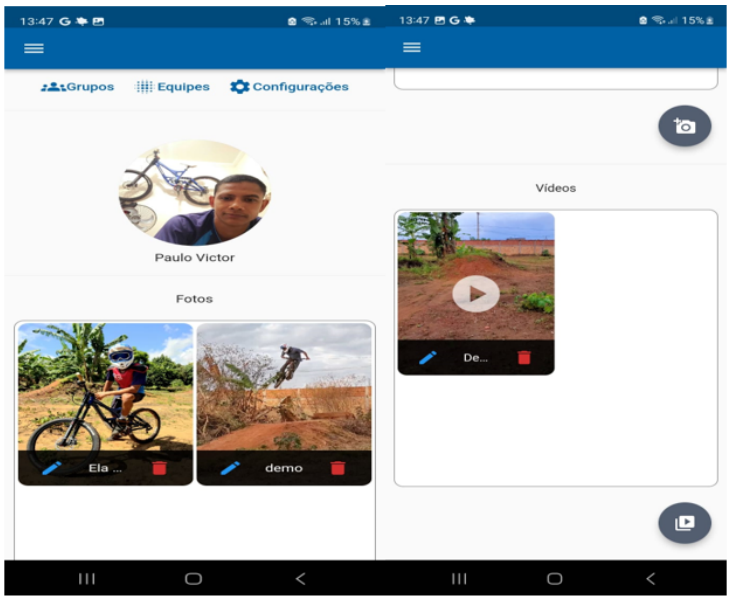
Figura 5.13: Chamados abertos.



Fonte: Autor.

A Figura 5.14 mostra o perfil do usuário, incluindo os grupos, equipes e configurações da conta. Na tela de perfil, o usuário tem a opção de adicionar fotos e vídeos. Essas opções são permitidas apenas para o proprietário da conta.

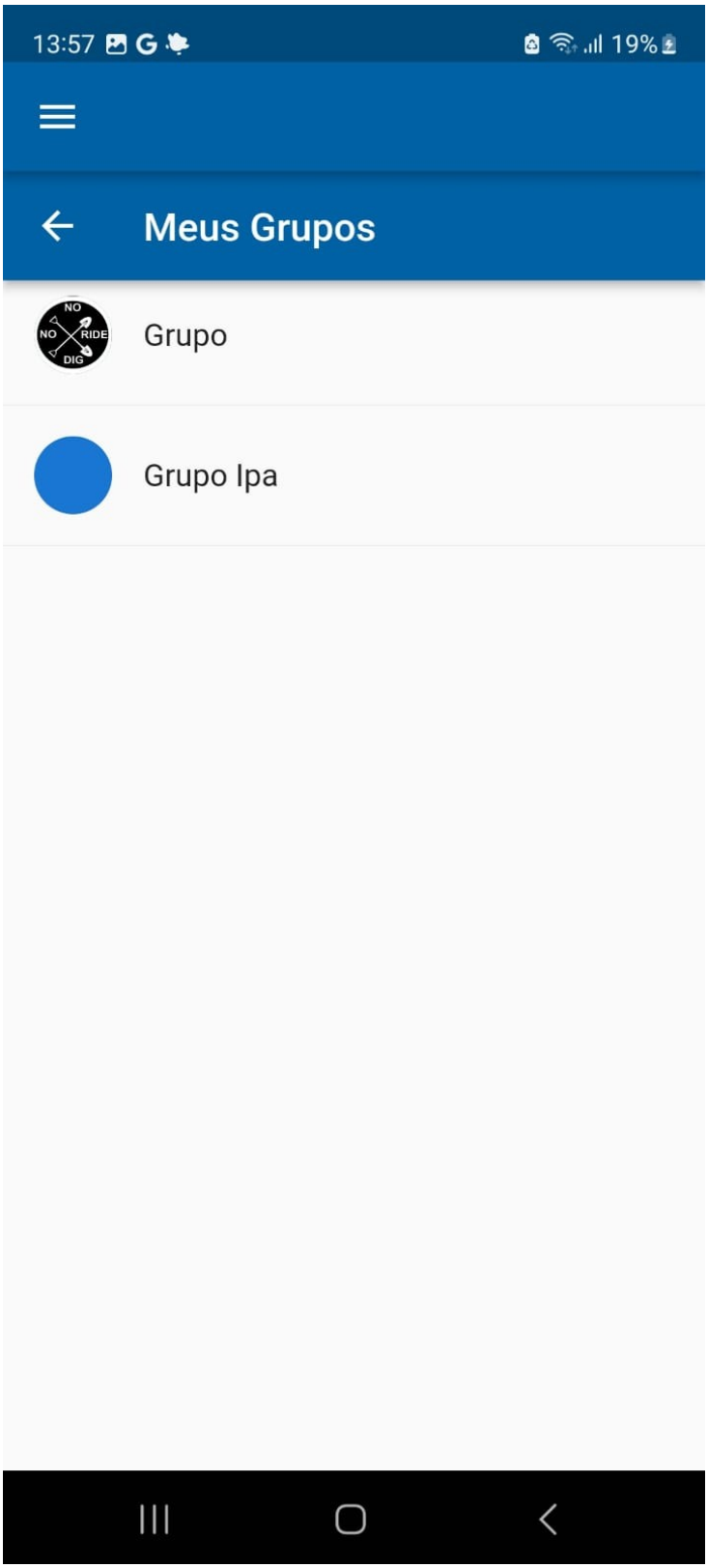
Figura 5.14: Perfil.



Fonte: Autor.

A Figura 5.15 mostra a tela dos grupos que o usuário participa. Para se tonar membro de um grupo, é necessário enviar uma solicitação à equipe de manutenção do local de esporte. Ao clicar em um grupo, o usuário pode interagir com os demais participantes.

Figura 5.15: Meus grupos.

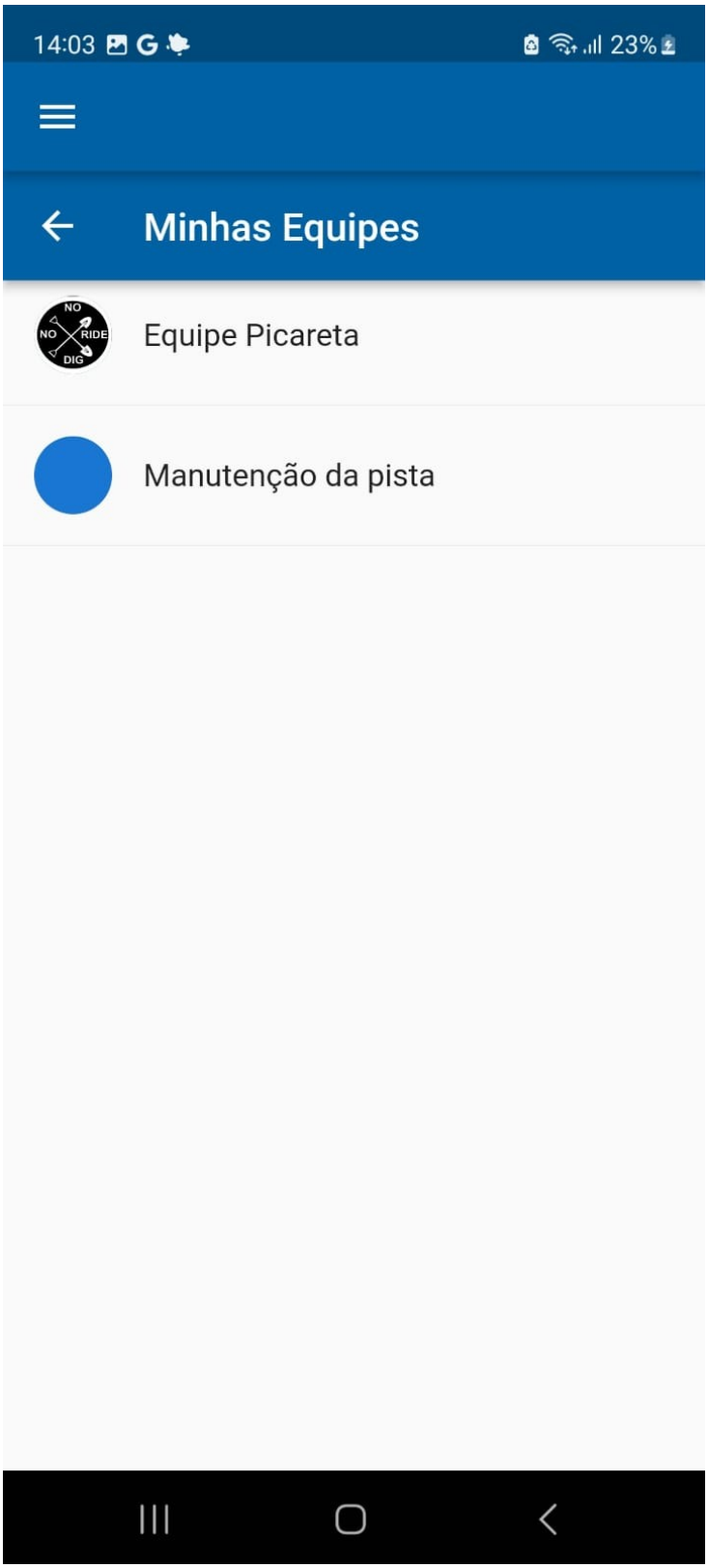


Fonte: Autor.

A Figura 5.16 ilustra a tela de equipes de manutenção. Na tela de "Minhas Equipes",

são exibidas as equipes às quais o usuário pertence. Ao clicar em uma equipe de manutenção, é possível visualizar os membros da equipe, os serviços que devem ser realizados e os serviços que já foram concluídos pela equipe.

Figura 5.16: Minhas equipes.

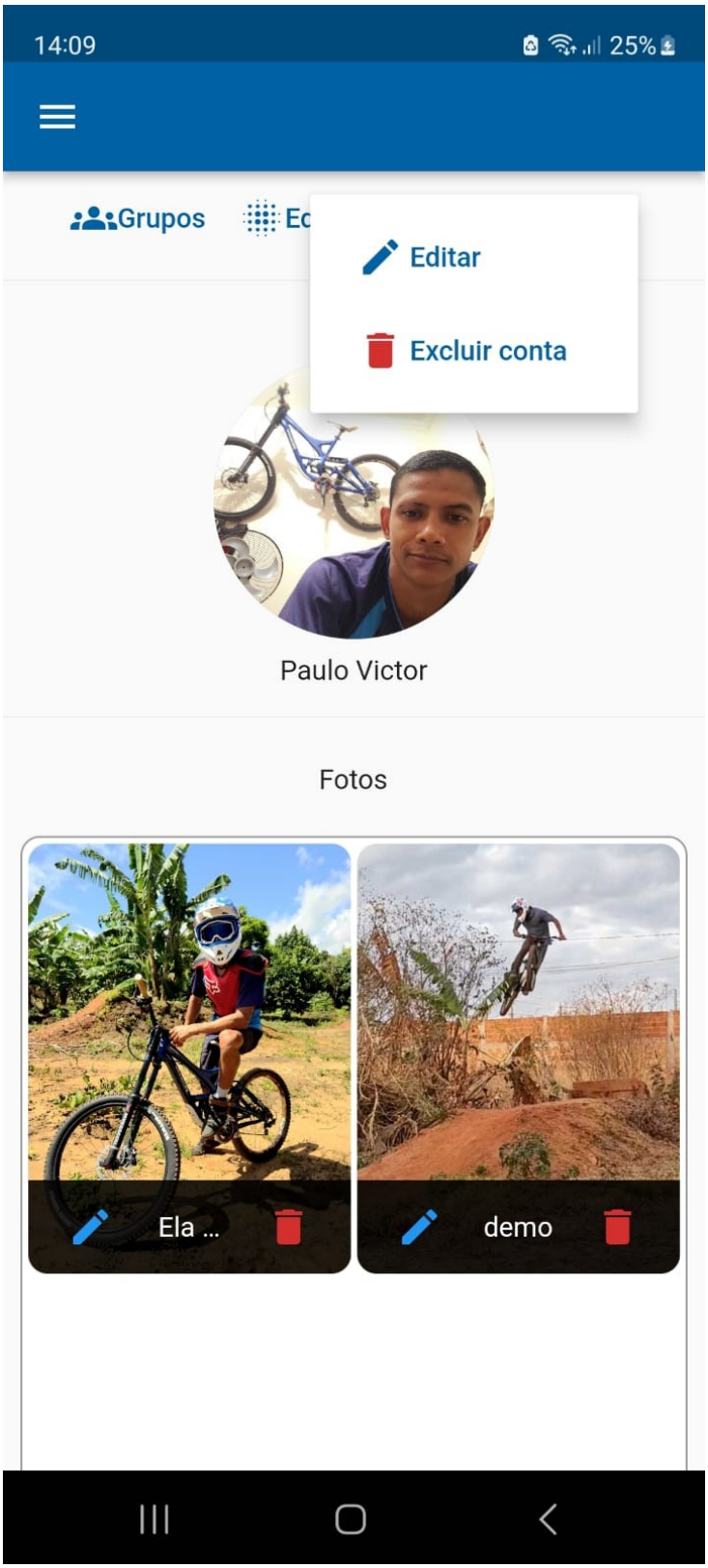


Fonte: Autor.

A Figura 5.17 mostra a tela de configurações, onde oferece duas opções. Primeiro, o

usuário pode editar o perfil, adicionar uma foto de perfil e alterar os dados da conta. Por último, há a opção de excluir a conta.

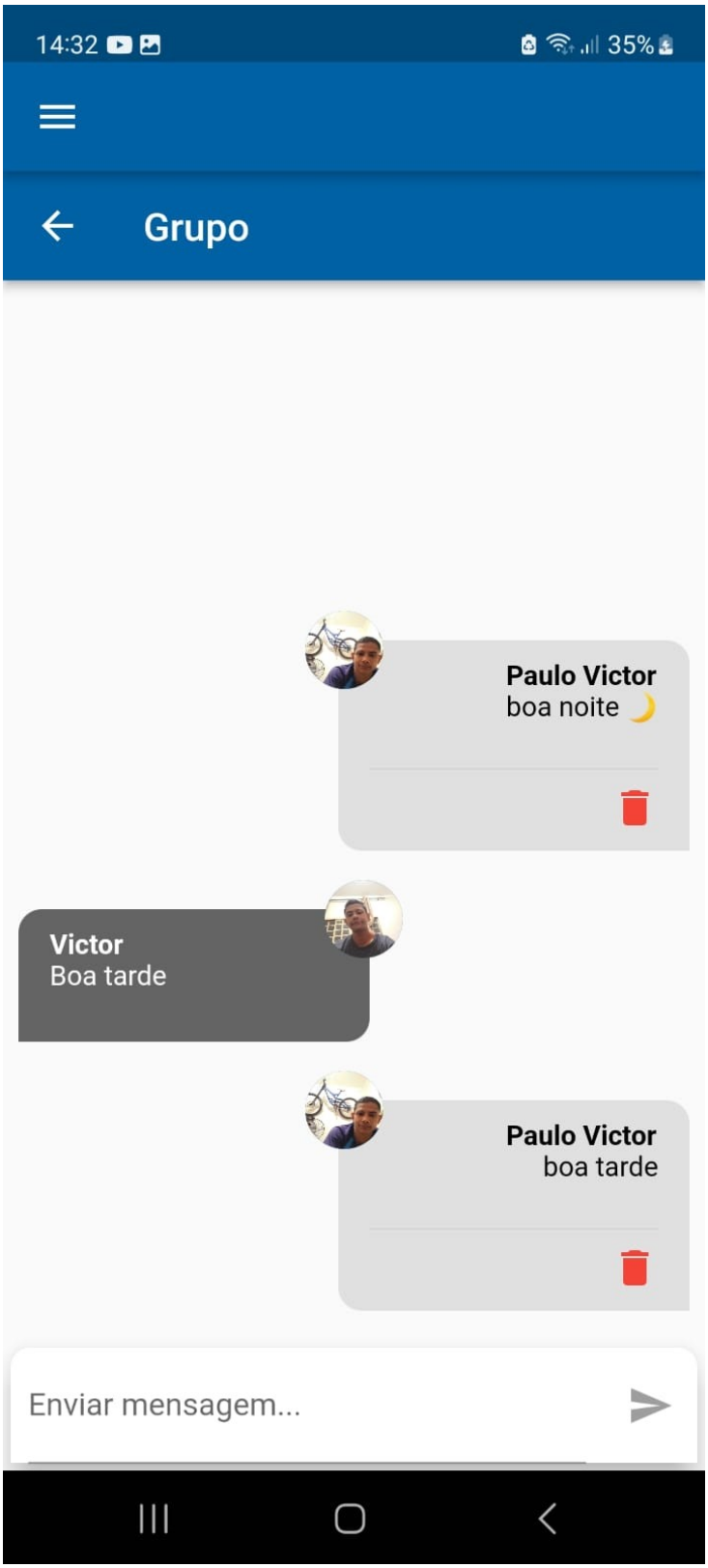
Figura 5.17: Configurações do usuário.



Fonte: Autor.

A Figura 5.18 mostra a tela de mensagens, onde o usuário pode se comunicar com os demais integrantes do grupo da equipe que ele faz parte.

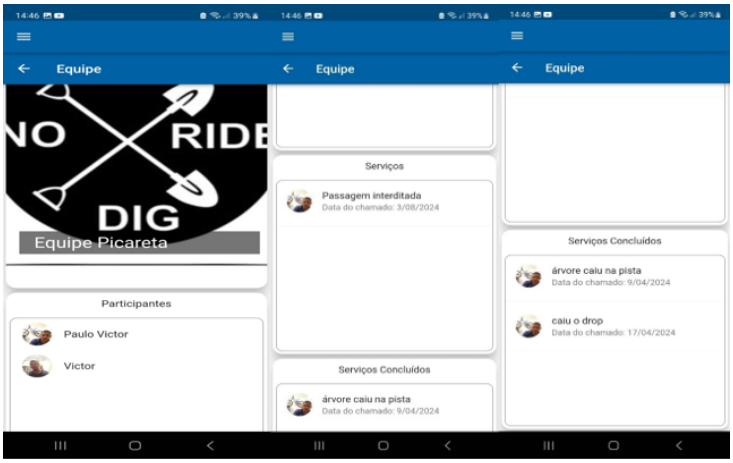
Figura 5.18: Mensagens do grupo.



Fonte: Autor.

A Figura 5.19 mostra a tela da equipe de manutenção, que exibe os participantes da equipe, os serviços a serem realizados em uma pista e os serviços concluídos. Ao clicar em um serviço, podemos ver detalhadamente o que deve ser feito, e, ao clicar em um serviço concluído, podemos ver com detalhes o que foi realizado.

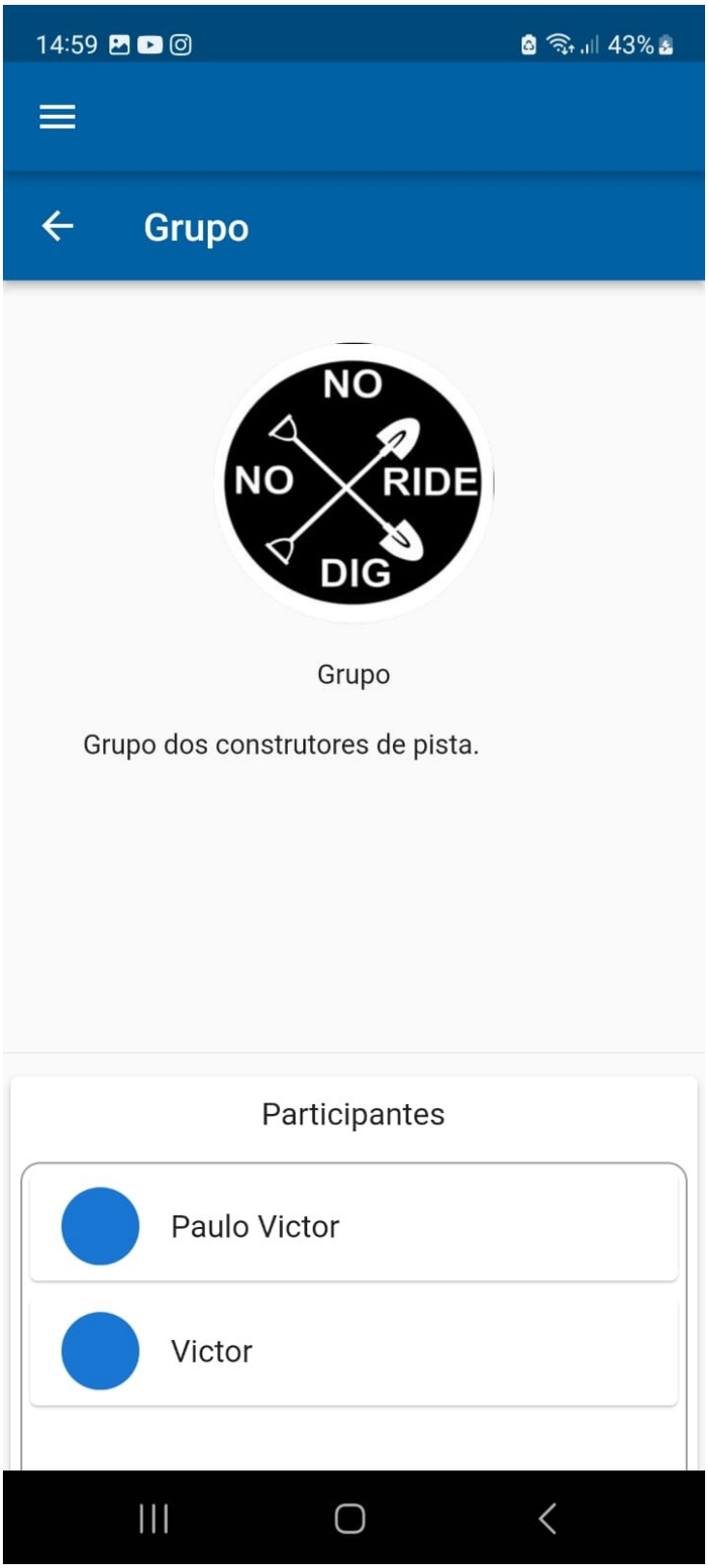
Figura 5.19: Equipe de manutenção.



Fonte: Autor.

A Figura 5.20 mostra a tela de detalhes do grupo, onde o aplicativo exibe o nome, a descrição e os participantes do grupo.

Figura 5.20: Detalhes do grupo.

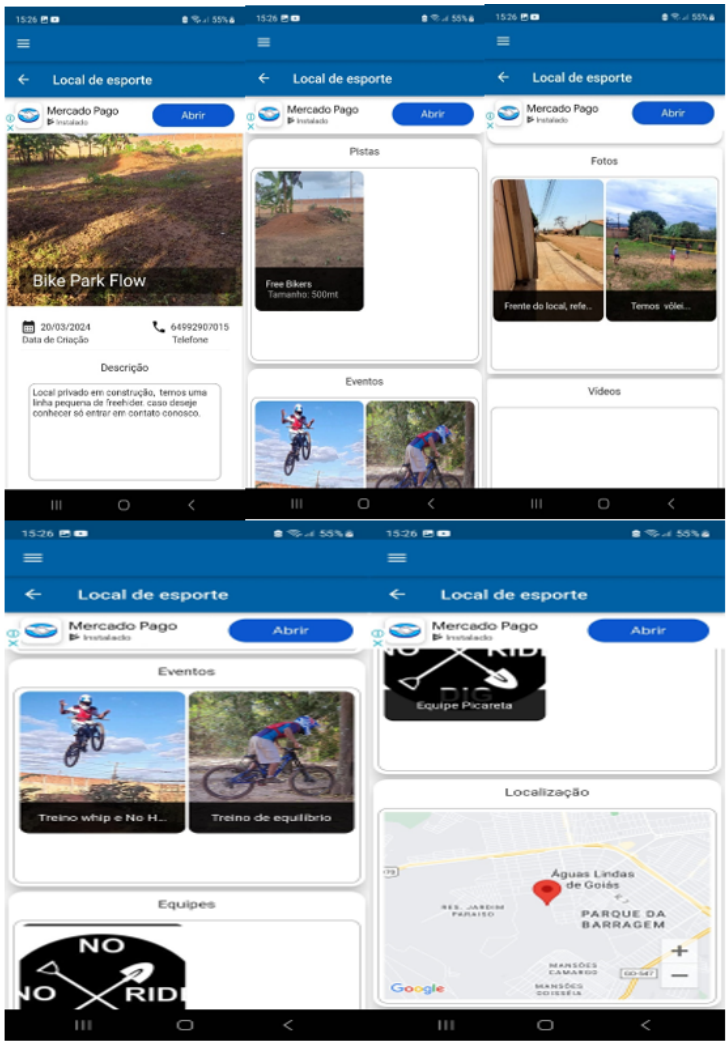


Fonte: Autor.

A Figura 5.21 mostra a tela de detalhes do local de esporte. O aplicativo exibe as

informações do local, o telefone do administrador, a descrição, fotos, vídeos, e a localização no mapa. Além disso, são apresentados os eventos, as pistas e as equipes de manutenção do local. Clicando em uma equipe do local, há a opção de enviar uma solicitação para participar da equipe de manutenção.

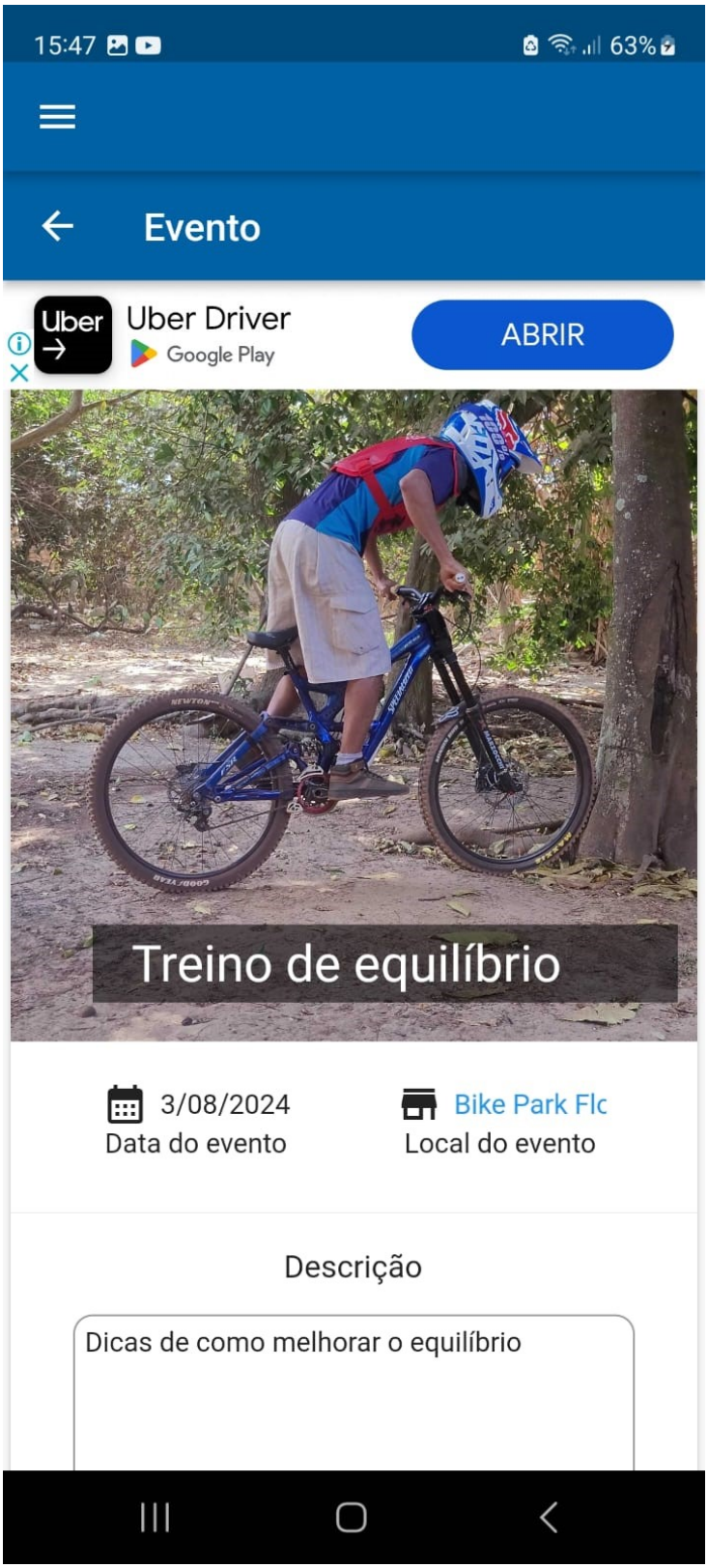
Figura 5.21: Detalhe do local de esporte.



Fonte: Autor.

A Figura 5.22 mostra a tela detalhada do evento, incluindo informações como o local onde o evento ocorrerá, a data e uma descrição.

Figura 5.22: Detalhe do evento.

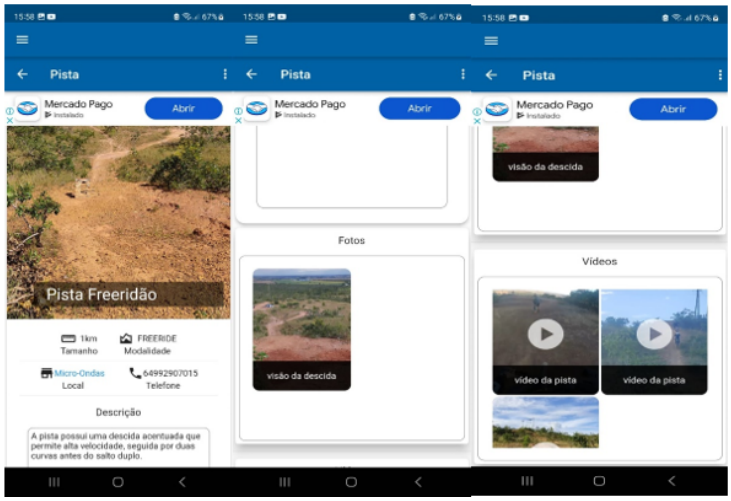


Fonte: Autor.

A Figura 5.23 exibe a tela com os detalhes da pista, mostrando informações como o

local, tamanho, telefone, descrição, fotos e vídeos. No canto da tela, na barra superior, há três pontinhos que permitem aos usuários realizar uma reclamação para o administrador do local.

Figura 5.23: Detalhe da pista.



Fonte: Autor.

5.2 Ambiente de desenvolvimento

O hardware utilizado no ambiente de desenvolvimento foi um notebook Dell modelo 7567, equipado com um processador Intel i7-7700HQ de 2.80 Ghz, 16GB de RAM e um SSD de 500GB. O sistema operacional em uso foi o Windows 10 Pro. Para codificação, utilizei o Visual Studio Code como editor. Na modelagem do banco de dados, foi empregado o MySQL Workbench 8.0. Foi usado a versão do framework Flutter SDK 3.13.8 e o Dart SDK 3.1.4. Para controle de versão, foi utilizado o Git (GIT, 2024), e para hospedagem do código, o GitHub (GITHUB, 2024).

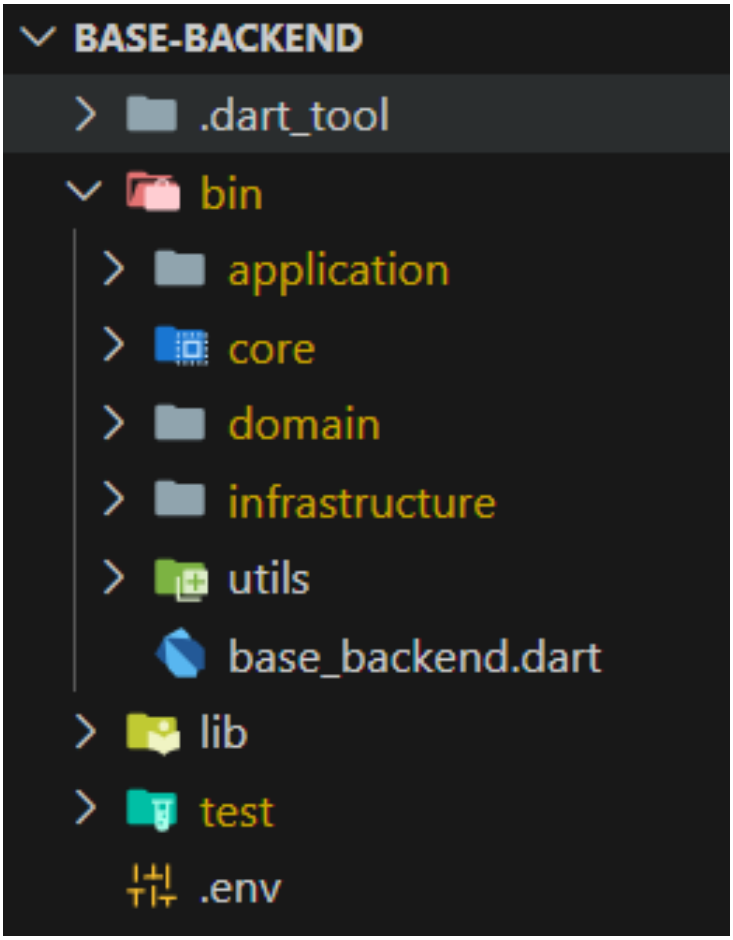
5.3 Codificação

O desenvolvimento da aplicação começou pelo *backend*, que foi construído com base na arquitetura hexagonal. Em seguida, foram criados os diretórios da aplicação de acordo com as camadas da arquitetura hexagonal. Após o desenvolvimento do *backend*, teve início a codificação do *frontend*, que foi construído com base na arquitetura limpa, e assim como no *backend* foi criado os diretórios da arquitetura limpa com forme as camadas da arquitetura.

5.4 Estrutura das pastas

A Figura 5.24 mostra a estrutura das pastas da arquitetura hexagonal, onde cada pasta representa uma camada da arquitetura. Analisando as pastas, temos a pasta *"domain"* , que representa a camada mais interna responsável pelas regras de negócio da aplicação. Essa camada não depende de nada além dela mesma e não conhece nenhuma outra camada, ou seja, no domínio temos apenas código puro. Na pasta *"application"*, encontramos a camada primaria responsável pelas APIs que realizam a comunicação com o mundo externo. Esta camada se conecta com o domínio através de portas. Finalmente, na camada secundaria, temos a pasta *"infrastructure"*, que é responsável por realizar a comunicação com mundo externo, atendendo às necessidades do sistema, como a comunicação com o banco de dados.

Figura 5.24: Estrutura de pasta.

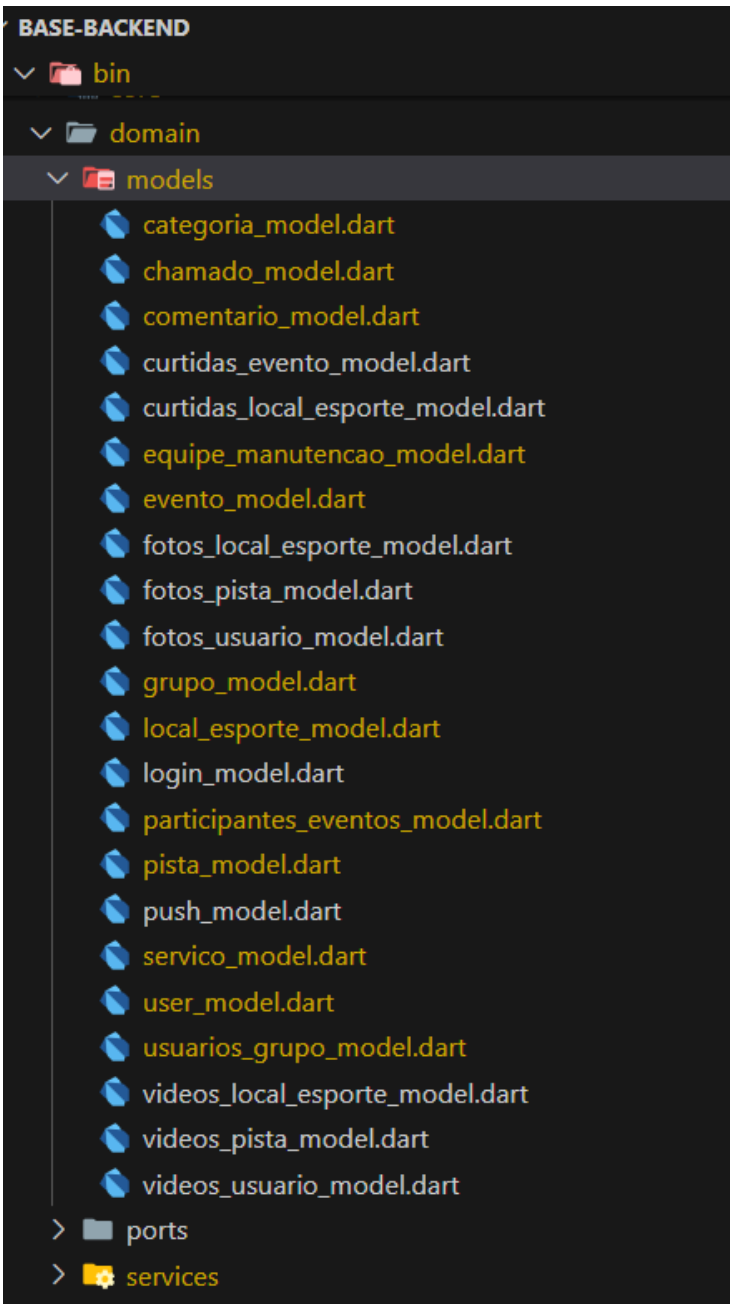


Fonte: Autor.

A Figura 5.25 mostra detalhadamente a pasta *"models"* da camada de *"domain"*, que representa as entidades da nossa aplicação. Na pasta *"models"*, estão a parte central das regras

de negócio e os modelos, que foram codificados na linguagem Dart.

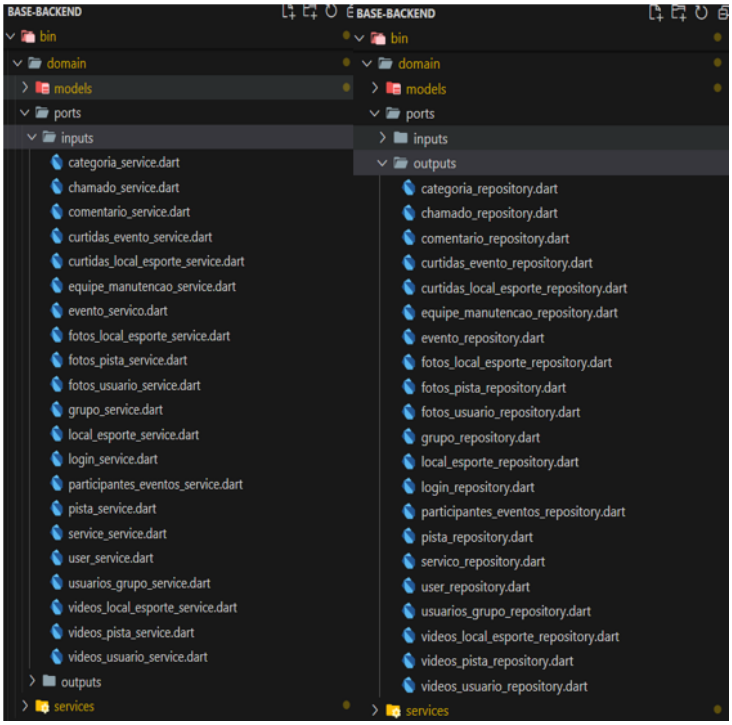
Figura 5.25: Modelos.



Fonte: Autor.

A Figura 5.26 mostra as portas de entrada e saída da camada de domínio. Na arquitetura hexagonal, as portas são interfaces que definem contratos e possibilitam vários benefícios. Entre eles, destacam-se: independência de tecnologia e frameworks, modularização, testabilidade, escalabilidade e separação de preocupações.

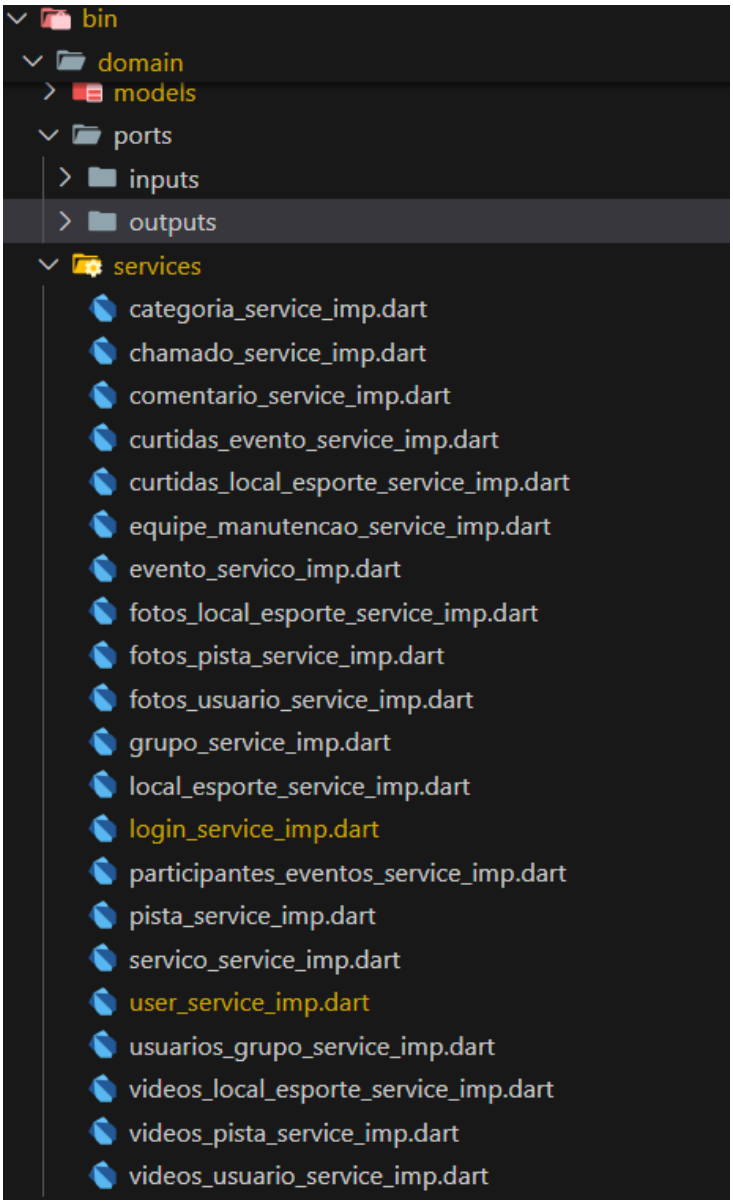
Figura 5.26: Portas.



Fonte: Autor.

A Figura 5.27 mostra a pasta *"services"*, que contém todos os serviços da camada de domínio. A camada de serviço é responsável por realizar manipulações no sistema. Para isso, ela implementa um contrato da porta de entrada e injeta uma porta de saída por meio da inversão de dependência (DIP) do SOLID, ou seja, é fornecida uma porta de saída que define um contrato para realizar operações na camada secundária.

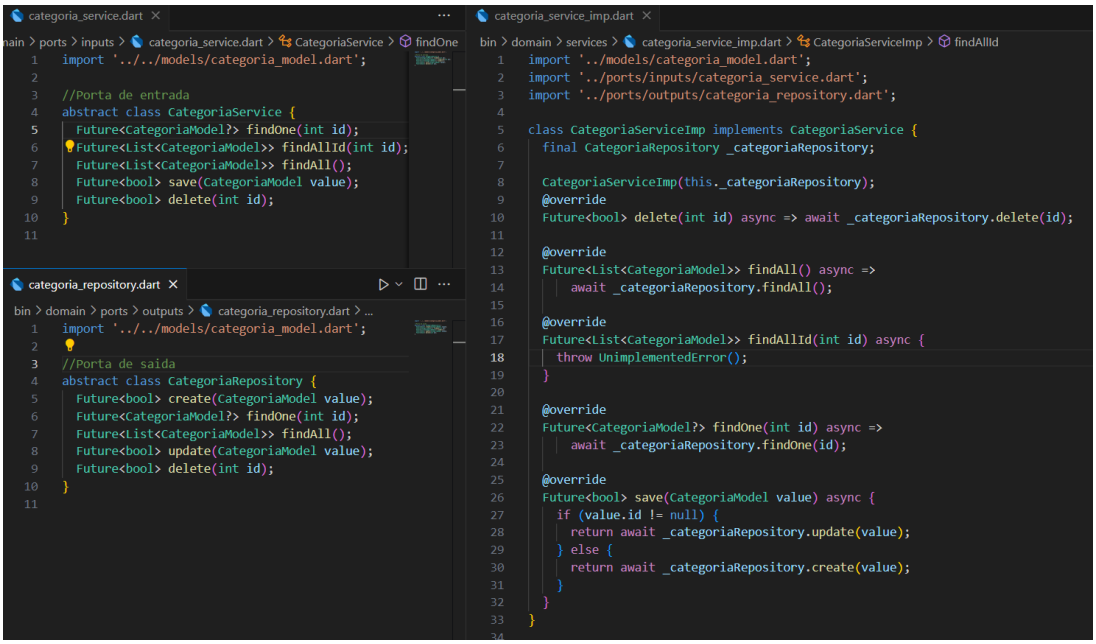
Figura 5.27: Serviços.



Fonte: Autor.

A Figura 5.28 mostra o código da implementação da porta de entrada da camada de domínio e como a porta de saída é passada via construtor, seguindo os princípios do SOLID.

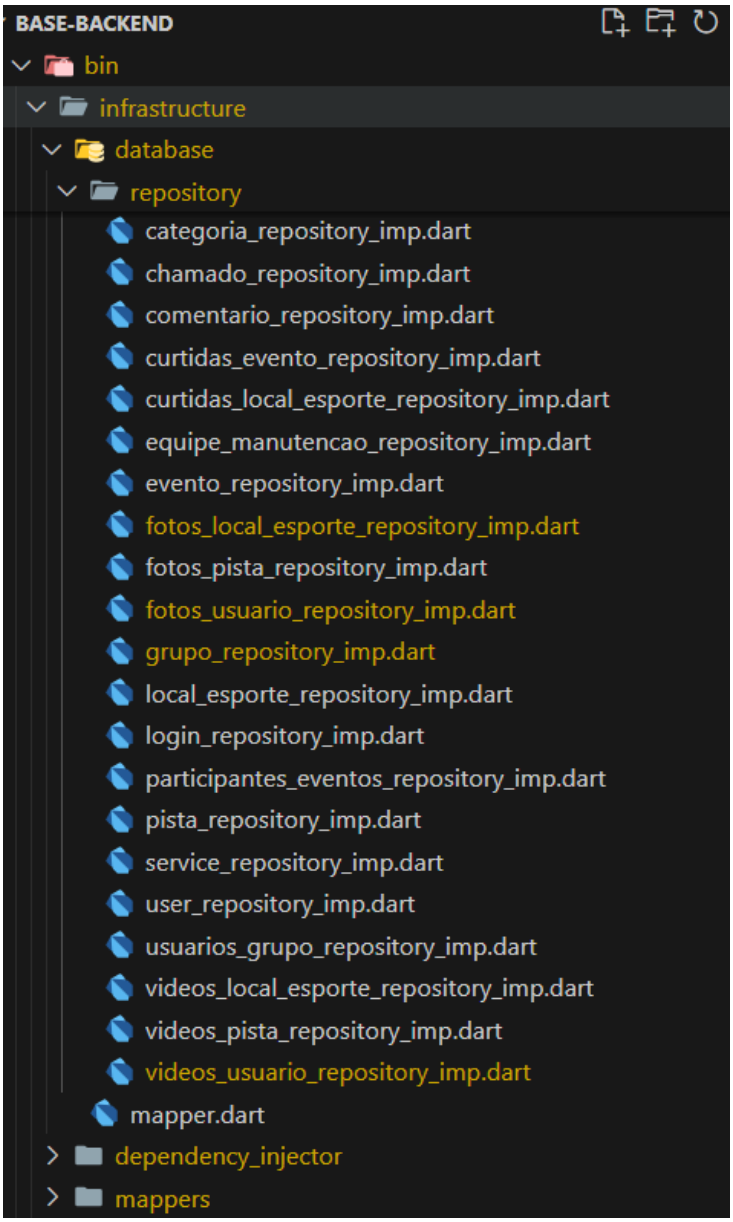
Figura 5.28: Implementação das portas de entrada e saída.



Fonte: Autor.

A Figura 5.29 mostra as classes da camada de infraestrutura que implementam as portas de saída da camada de domínio. Essas classes são responsáveis por manipular as informações no banco de dados.

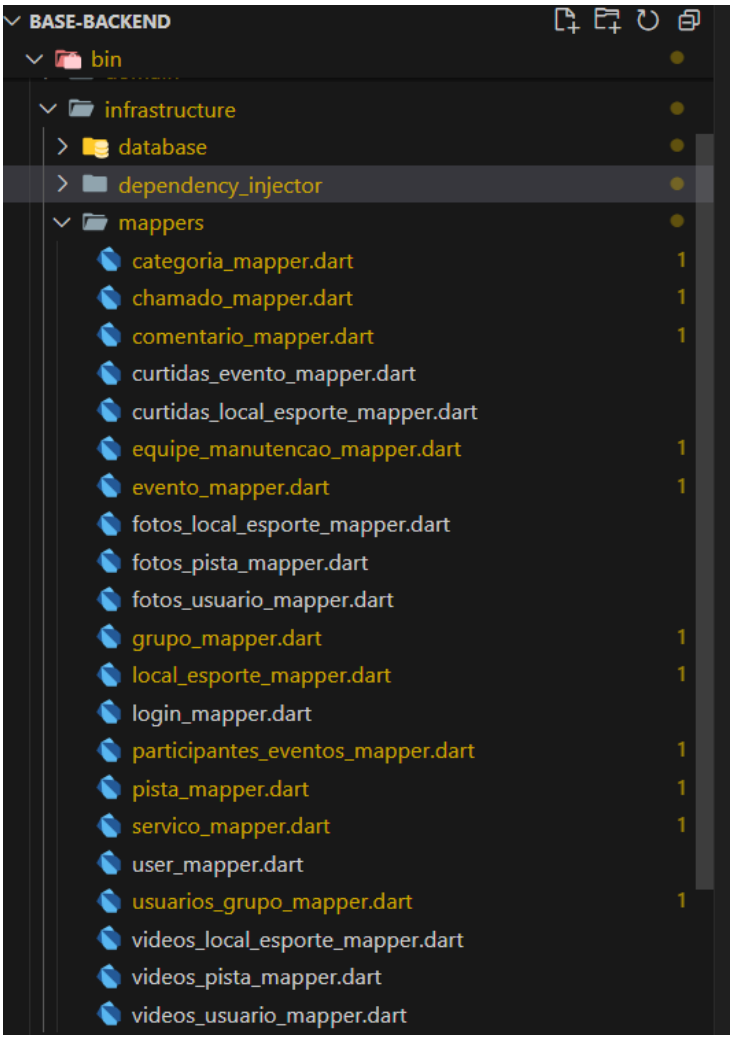
Figura 5.29: Repositório da camada de infraestrutura.



Fonte: Autor.

A Figura 5.30 mostra as classes responsáveis por mapear os dados vindos do banco de dados para objetos do sistema. Esses mapas são utilizados pelas classes da pasta repositório, na camada de infraestrutura, que são responsáveis pela manipulação dos dados.

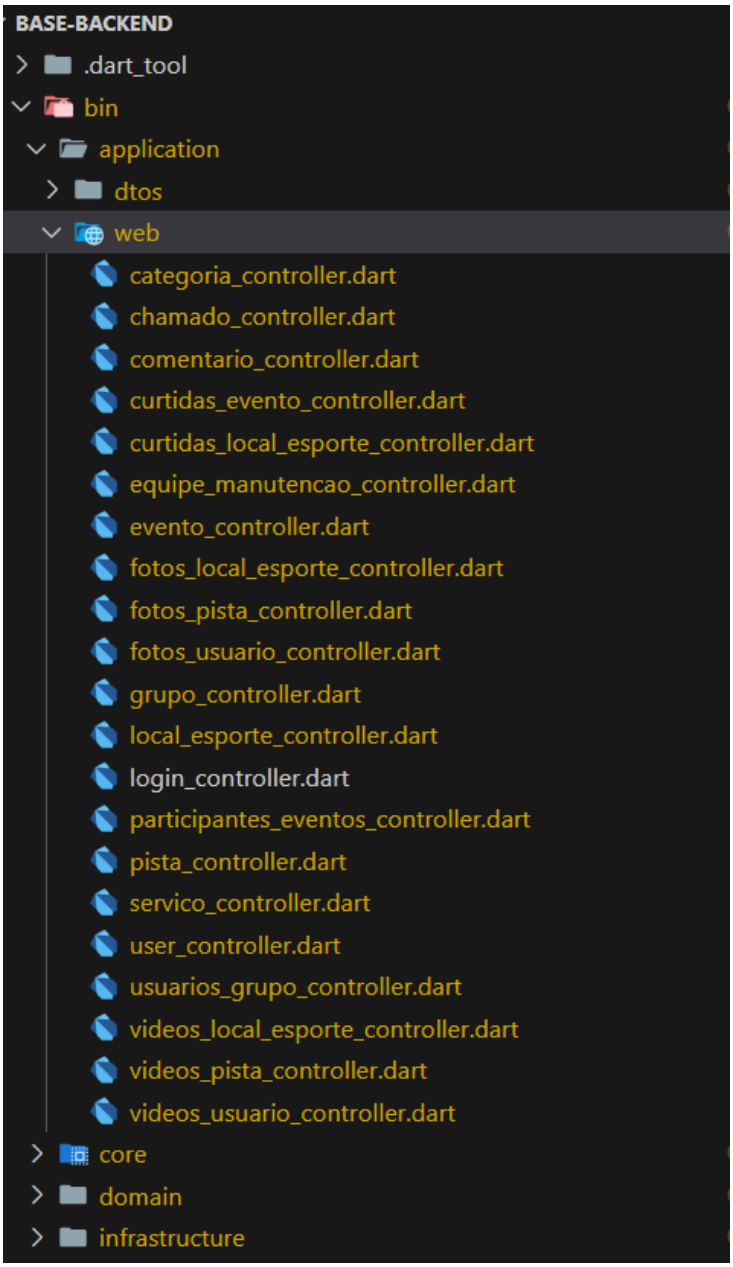
Figura 5.30: Mapeadores.



Fonte: Autor.

A Figura 5.31 mostra a pasta responsável por representar a camada de aplicação. Nessa pasta, há duas subpastas. A pasta "dtos", que contém objetos de transferência, é responsável por transformar os dados em JSON para um objeto do sistema e também por converter um objeto do sistema em um formato que será convertido para JSON.

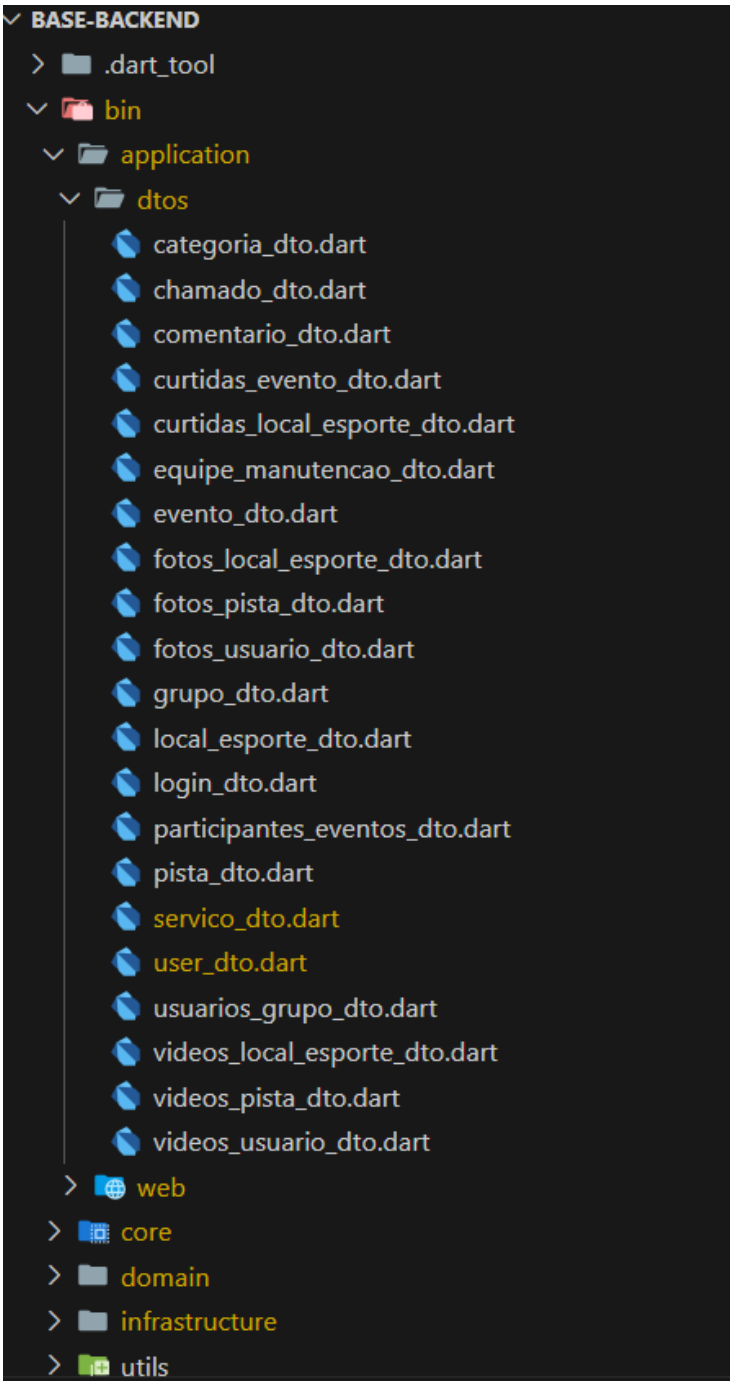
Figura 5.31: Camadas de aplicação.



Fonte: Autor.

A Figura 5.32 mostra todas as classes responsáveis por converter os objetos do sistema em um formato aceitável para transferência e também por converter os dados recebidos de volta em objetos do sistema.

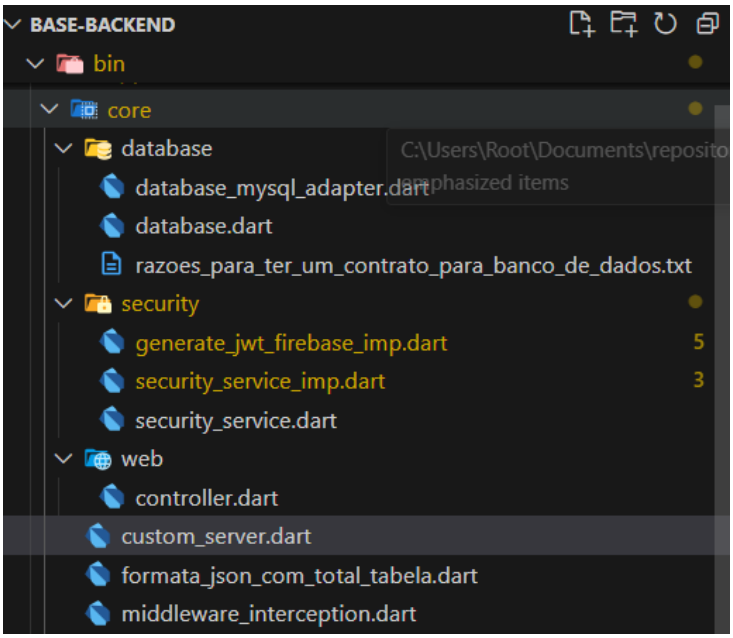
Figura 5.32: Objetos de transferência.



Fonte: Autor.

A Figura 5.33 mostra a pasta "core", que é responsável pelos requisitos globais do sistema, como conexão com banco de dados, segurança, conexão com o servidor, assinaturas de tokens e até mesmo a formatação dos dados transmitidos pelas APIs do sistema.

Figura 5.33: Core.

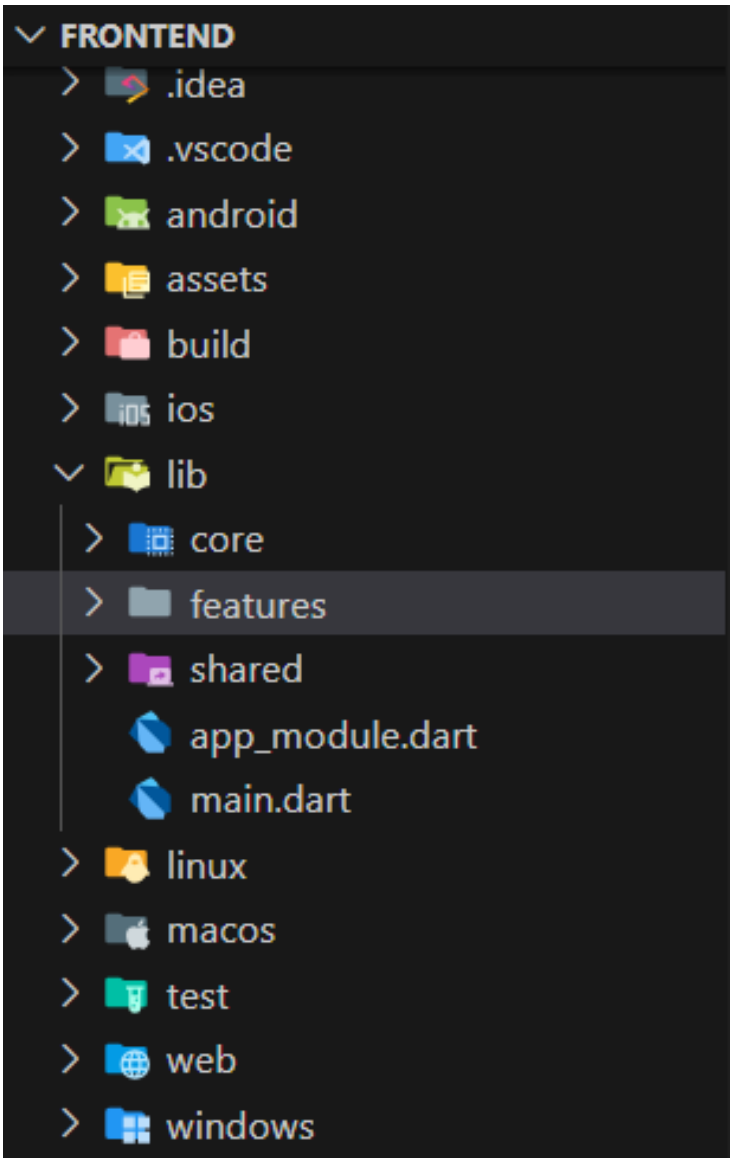


Fonte: Autor.

5.5 A construção do frontend utilizando a arquitetura limpa

A Figura 5.34 mostra a pasta "lib", onde é construído o código no Framework Flutter. Nela, há outras três pastas: a primeira é a pasta "core", responsável por todas as funcionalidades globais da aplicação; a segunda é a pasta "features", que lida com cada contexto do sistema; e a última é a pasta "shared", que contém classes compartilhadas entre diferentes contextos do sistema. Além dessas pastas, há duas classes: a primeira, "main.dart", é responsável por iniciar a aplicação, funcionando como a classe principal; a segunda classe inicia os módulos do sistema.

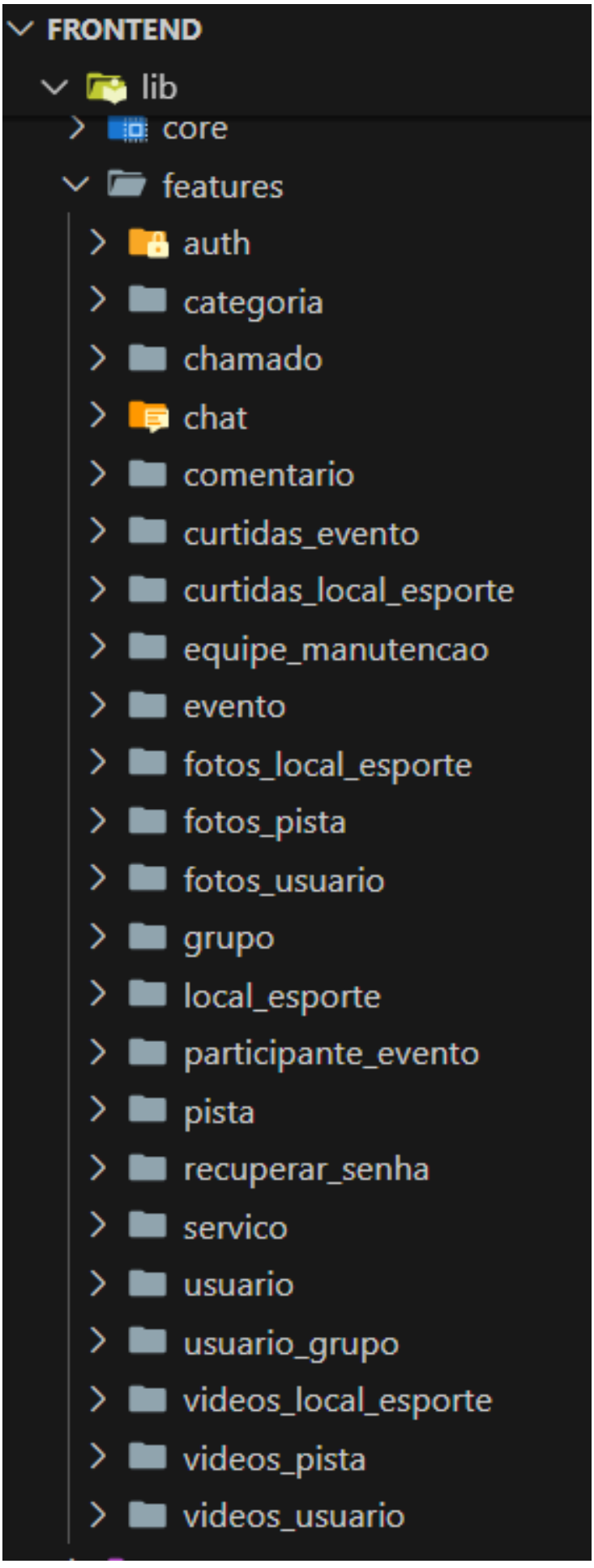
Figura 5.34: Pastas iniciais do sistema.



Fonte: Autor.

A Figura 5.35 mostra os contextos da aplicação. Essa separação por contexto é um conceito de design de software que visa a divisão da aplicação em módulos. De acordo com o Domain-Driven Design (DDD), essa divisão em contextos delimitados ajuda a reduzir a complexidade do sistema, facilita a manutenção, permite que as equipes de desenvolvimento trabalhem de forma independente e contribui para a escalabilidade do sistema.

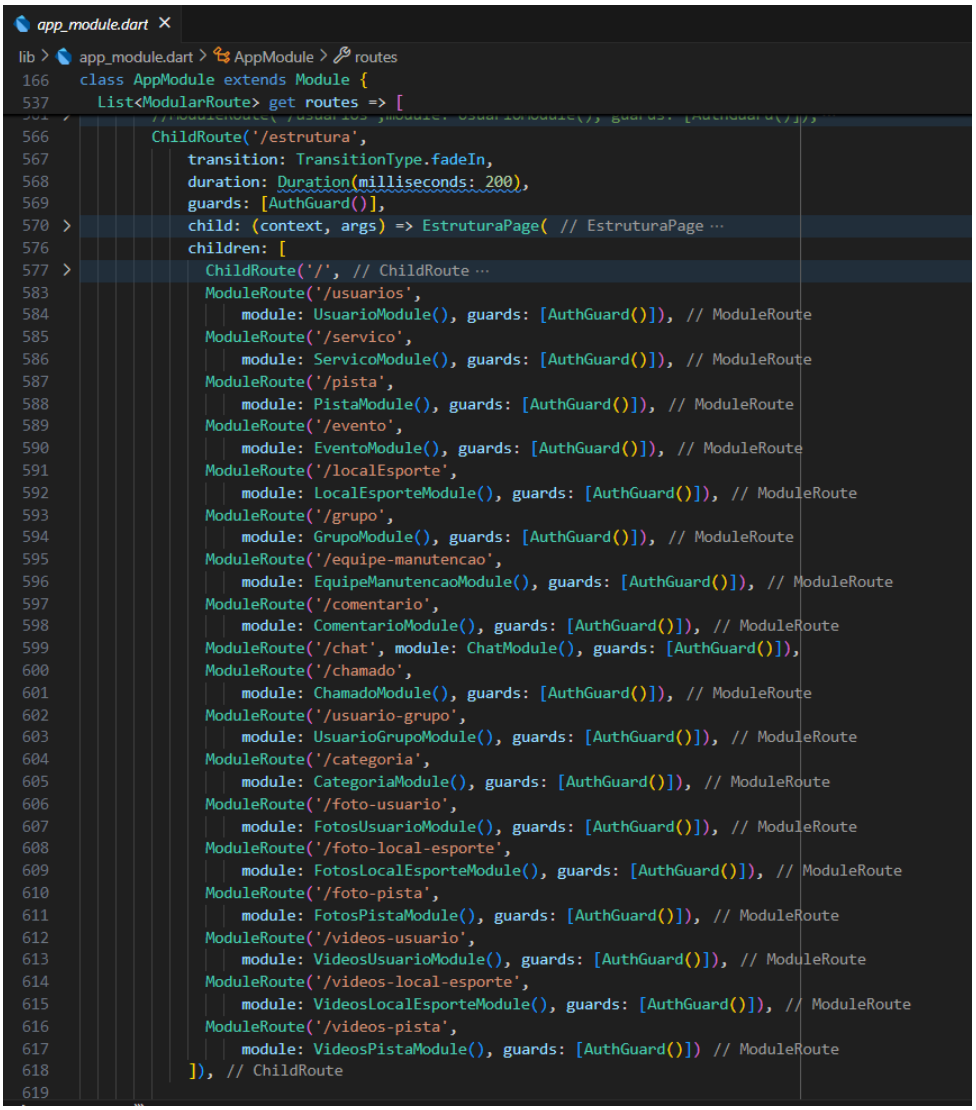
Figura 5.35: Contextos da aplicação.



Fonte: Autor.

A Figura 5.36 mostra o código da classe AppModule que é responsável por gerenciar as rotas de todos os outros módulos da aplicação.

Figura 5.36: AppModule.



Fonte: Autor.

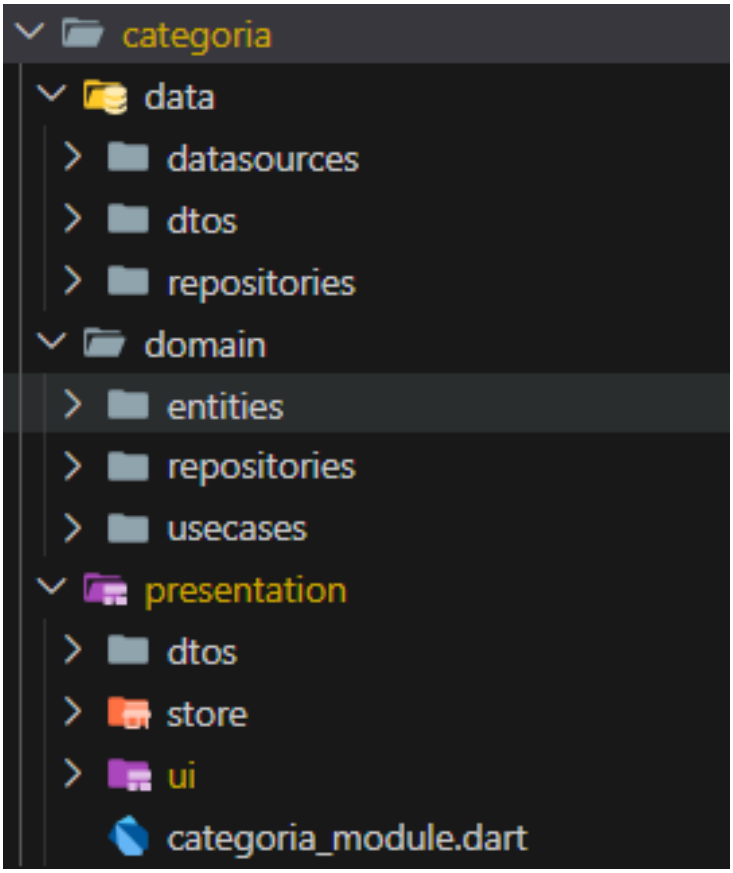
A Figura 5.37 mostra as pastas que representa as camadas da arquitetura limpa. Analisando a imagem, na pasta "categoria", há outras três pastas principais: "data", "domain"e "presentation". A pasta "domain"é responsável pelas regras de negócio relacionadas ao contexto de categoria. Na pasta "domain", não há dependências externa; ou seja, não se faz uso de tecnologias como o Flutter, apenas código puro em Dart.

A pasta "data"é responsável por manipular os dados provenientes do armazenamento local, do Firebase e do servidor remoto da aplicação PlaceRiders. Esta camada também é responsável por implementar as interfaces de repositório da camada de domínio. Os repositórios

na pasta "data"são encarregados de realizar verificações, como verificar se há acesso à internet, antes de encaminhar os dados para a camada responsável por manipulações na base de dados. A pasta "dtos"contém as classes responsáveis por mapear os dados para objetos do sistema e prepara os objetos do sistema para o formato esperado pela base de dados remota. Por último, a camada "presentation", é responsável pela parte visual e pela parte reativa da aplicação, que é construída utilizando o MobX, uma biblioteca de gerenciamento de estado.

Na pasta "UI"estão contidas as telas de categorias. Na raiz da pasta "categoria", está a classe "categoriamodule.dart", responsável por gerenciar o módulo de categoria. Este arquivo define tanto as rotas quanto os binds, que são utilizados para registrar e fornecer as dependências necessárias.

Figura 5.37: Estruturas de pastas da arquitetura limpa.

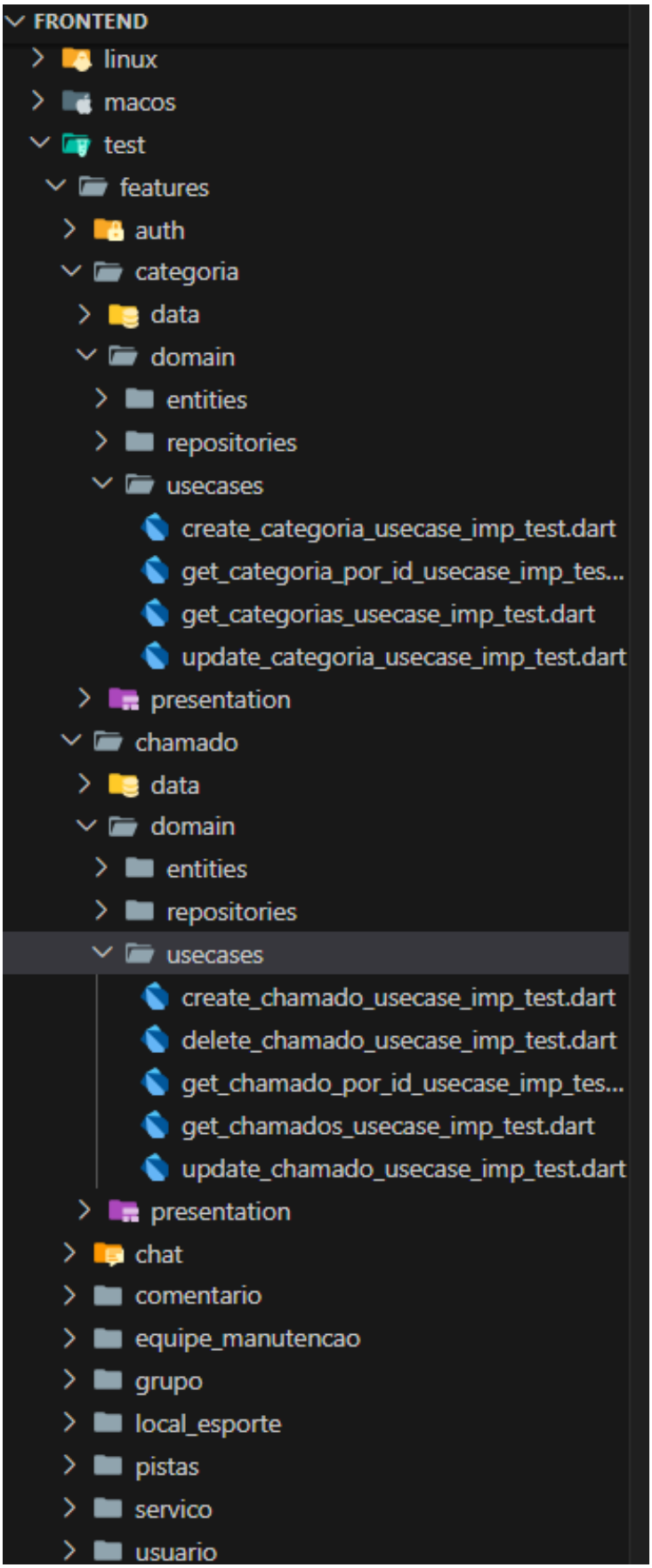


Fonte: Autor.

5.6 Testes

A Figura 5.38 mostra as estruturas das pastas "test", responsáveis por conter os contextos do sistema que serão testados. Os testes realizados na aplicação na camada de caso de usos.

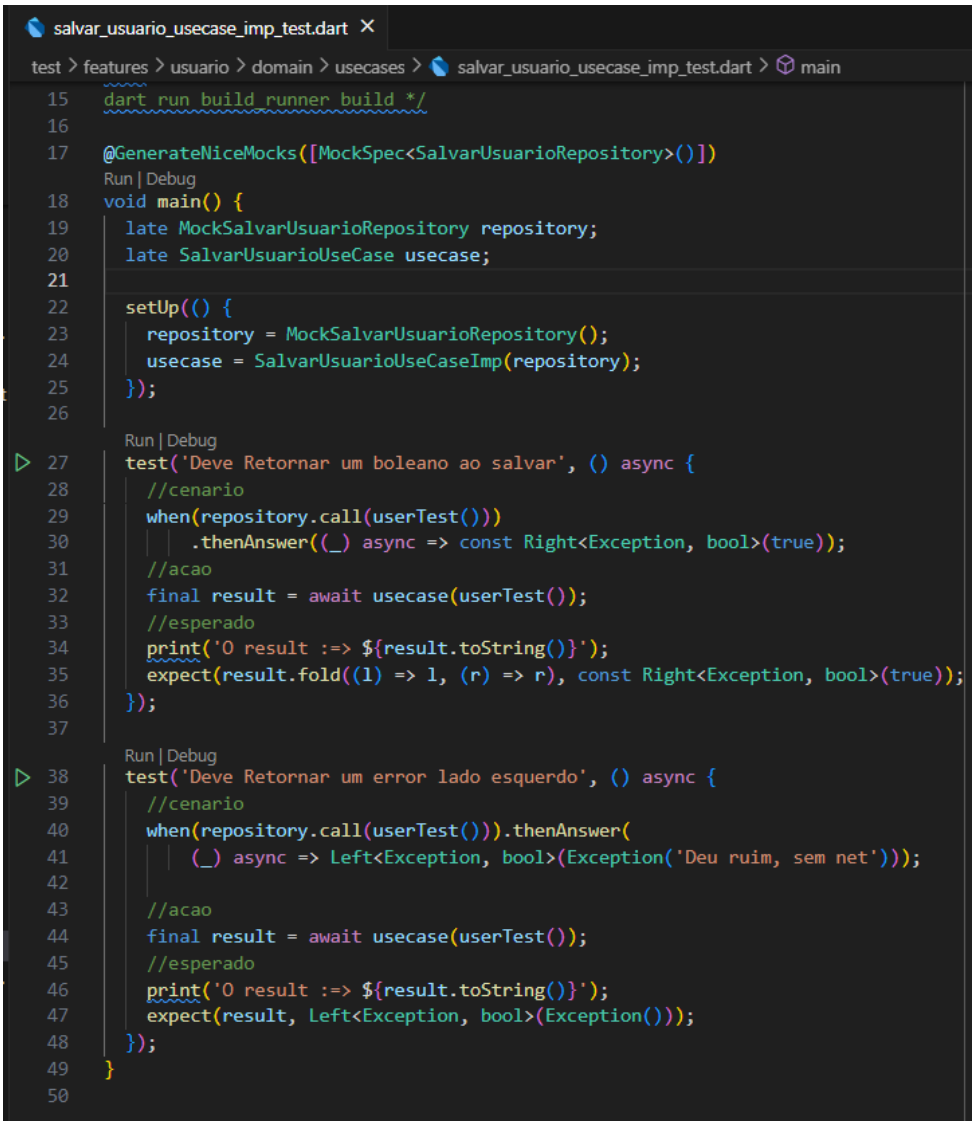
Figura 5.38: Teste.



Fonte: Autor.

A Figura 5.39 mostra o código usado para testar o caso de uso de salvar um usuário. Para simular os valores esperados do servidor, foi utilizada uma biblioteca de testes que gera esses valores para a realização dos testes. O teste consiste de três etapas: a primeira etapa é o cenário, onde são fornecidos os dados necessários; a segunda etapa é a ação, que consiste na operação de salvar o usuário; e, por fim, a terceira etapa é a validação do resultado, verificando se ele está conforme os esperados.

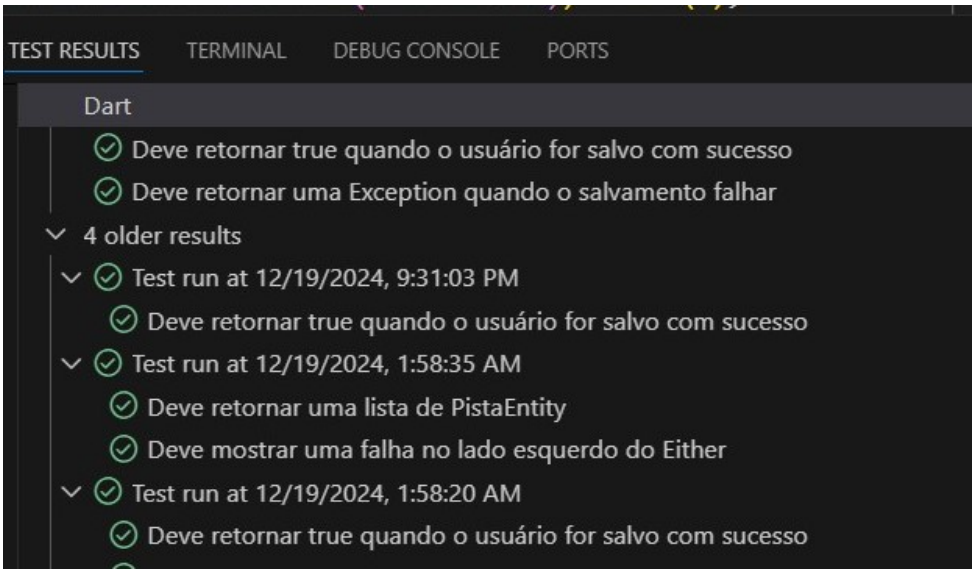
Figura 5.39: Código de teste salvar usuário.



Fonte: Autor.

A Figura 5.40 apresenta os resultados obtidos nos testes, os quais foram aprovados, conforme podem ser observados.

Figura 5.40: Resultados dos testes.



Fonte: Autor.

A Figura 5.41 apresenta o resultado obtido no teste de estabilidade realizado pelo Google Play Console. Esse teste garante que o aplicativo funcione de maneira confiável em diferentes dispositivos.

Figura 5.41: Resultado do teste de estabilidade.

Fonte: Google Play Console.

A Figura 5.42 apresenta o resultado do teste de desempenho realizado pelo Google Play Console. Esse teste tem como objetivo garantir a eficiência do aplicativo em diversos dispositivos.

Figura 5.42: Resultado do teste de desempenho.

Desempenho

Renderização lenta
0
dispositivos com problemas

Tempo de inicialização a frio
0
dispositivos com problemas

Aparelho	CPU média	Média de rede enviada	Média de rede recebida	Memória média	Tempo de inicialização a frio
Dispositivos de teste sem problemas					
motorola Motorola G20	1,78%	0 B	0 B	117 MB	1,74 mil ms
google Pixel 5	1,65%	0 B	0 B	187 MB	1,03 mil ms
google Pixel 6	1,22%	0 B	0 B	201 MB	401 ms
samsung Galaxy S20	1,83%	0 B	0 B	178 MB	814 ms

Fonte: Google Play Console.

CAPÍTULO 6

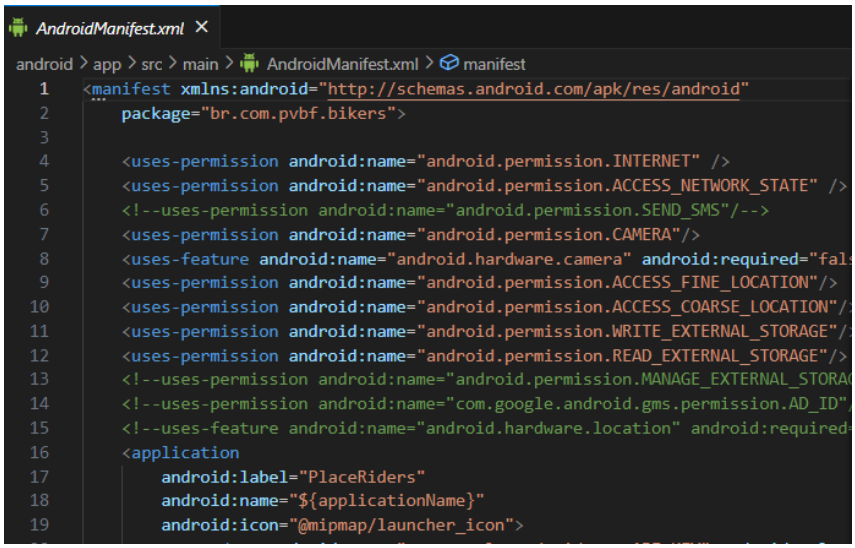
IMPLEMENTAÇÃO

6.1 Deploy

Depois que o aplicativo foi aprovado em todos os testes e confirmado que tudo está funcionando com segurança, inicie-se o processo de implementação. De acordo com a documentação (FLUTTER,), temos as etapas para a implementação do aplicativo nas lojas.

Definir o nome e identificador da aplicação: A Figura 6.1 mostra o arquivo "AndroidManifest.xml", onde são definidos o nome e o identificador da aplicação. Na linha 17, a variável "label"recebe o nome da aplicação, que no caso desta aplicação tem o nome de "PlaceRiders". O objetivo principal do nome é transmitir a ideia de que o aplicativo é um local voltado para ciclistas. O identificador da aplicação deve ser único. Na linha 2, definimos o identificador da aplicação na variável "package", que neste caso é "br.com.pvbf.bikers".

Figura 6.1: Nome e identificador da aplicação.



Fonte: Autor.

- Adicionando ícones e Imagens usadas na aplicação: Para criação dos ícones da aplicação foi usado uma biblioteca que facilita a criação dos ícones. É necessário fornecer a imagem do ícone que deseja criar com o pacote (flutterlaunchericons) para isso basta executando o comando "flutter pub run flutterlaunchericons". Na Figura 6.2 temos o ícone da aplicação criado.

Figura 6.2: Ícone do aplicativo.

Fonte: Autor.

- Reduza o código: Conforme recomendado na documentação do (FLUTTER,), é importante utilizar a ferramenta R8 da Google para realizar a redução do código. Isso resulta em um aplicativo menor, que ocupa menos espaço de armazenamento e diminui o tempo de download.

- Assinando o aplicativo: A assinatura é a identidade do aplicativo, por isso é essencial assiná-lo para garantir sua autenticidade. Além disso, ela é crucial para o Google Play Console, assegurando que apenas quem possui a chave de assinatura possa realizar atualizações no aplicativo.

- Criação de uma política de uso do aplicativo: Os termos de uso de uma aplicação são os meios que garantem proteção legal e uma interação adequada entre o usuário e o aplicativo. Ao criar uma política de uso, é essencial definir claramente os requisitos e as permissões que o usuário deve fornecer e com as quais deve concordar para que a aplicação funcione corretamente, sem comprometer a privacidade do usuário.

- Configuração da Conta de desenvolvedor no Google Play: Para publicar o aplicativo no Google Play, é necessário seguir algumas etapas. Primeiro, realizar o cadastro no

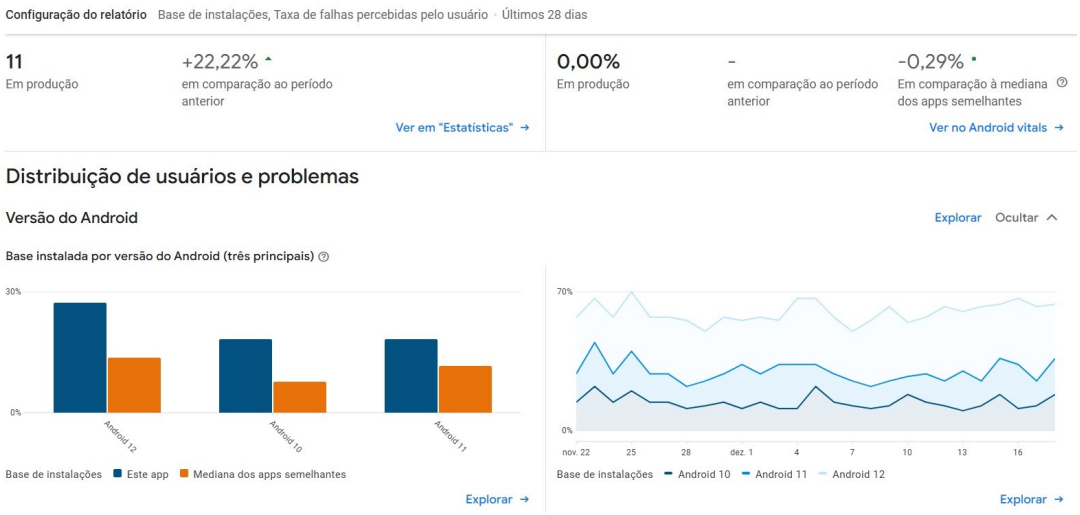
Play Console como desenvolvedor e aceitar os termos. Em seguida, pagar a taxa única de inscrição e fornecer as informações requisitadas para concluir o cadastro. Após a conclusão dessas etapas e a aprovação da conta de desenvolvedor pelo Google Play, o proprietário da conta pode acessar o Google Play Console para publicar seus aplicativos.

- Publicação no Google Play: Para publicar o aplicativo é necessário acessar a conta de desenvolvedor no Google Play Console e seguir alguns passos:
  - No Google Play Console, acesse a opção "Criar App". Preencha os dados requisitados: nome do app, idioma padrão, categoria (app ou jogo), tipo (gratuito ou pago), confirme se o app atende às políticas do programa para desenvolvedores e aceite a legislação de exportação dos EUA.
  - Após criar o aplicativo, você terá a opção de publicar uma versão de teste ou uma versão de produção. Clicando na opção "Produção", o Google Play Console exibirá a opção "Criar nova versão". Ao selecionar essa opção, o sistema exibirá um formulário com alguns requisitos: *upload* do aplicativo, nome da versão e uma nota informativa sobre a versão que está sendo publicada. Após preencher todos os campos requisitados, basta clicar na opção "Próximo". O aplicativo será então enviado para revisão, e após a aprovação pela equipe do Google Play, você poderá clicar na opção "Confirmar"para que o aplicativo seja publicado na loja.

6.2 Publicação no Google Play

Com o aplicativo já desenvolvido e os testes aprovados, temos a primeira versão pronta para publicação. Inicialmente, o aplicativo será lançado em apenas uma loja, a Play Store. Seguindo as instruções da documentação de implementação do Flutter (FLUTTER,), o aplicativo foi implementado no Google Play, a loja oficial de aplicativos para Android (Google Play Store, 2024).

Figura 6.3: Visão geral do alcance e dos dispositivos.



Fonte: Google Play Console.

A Figura 6.3 apresenta a visão geral do aplicativo em produção. Conforme mostrado na figura, existem 11 instalações da versão em produção do aplicativo. Logo abaixo, são exibidos os gráficos de distribuição de usuários e problemas. Entre os gráficos, destaca-se o que ilustra o alcance dos dispositivos: a coluna azul representa este aplicativo, enquanto a coluna laranja mostra as médias dos aplicativos semelhantes.

CONSIDERAÇÕES FINAIS

O trabalho proposto consistiu na criação de um aplicativo para a divulgação e manutenção de locais destinados à prática de esportes com a bicicleta. Durante o processo de desenvolvimento, foram trabalhados diversos conceitos e padrões aplicados ao desenvolvimento do aplicativo. Além dos requisitos iniciais que o aplicativo se propôs a atender, outros foram adicionados, como a divulgação de eventos realizados nos locais de esporte, sendo os campeonatos de ciclismo um exemplo desses eventos. Também foi incluída a funcionalidade de permitir que os usuários realizem comentários sobre os locais de esporte e os eventos. Outra funcionalidade adicionada foi a possibilidade de os usuários incluírem fotos e vídeos dos locais de esporte, pistas, eventos e em seus próprios perfis.

6.3 Sugestões para trabalhos futuros

- Compreender melhor o público-alvo e suas necessidades, analisando se a aplicação atende a essas necessidades em relação aos locais para a prática de esportes com bicicleta.
- Analisar o impacto que o aplicativo pode causar no cicloturismo.
- Publicar o aplicativo em outras lojas e lançar sua versão web.

REFERÊNCIAS BIBLIOGRÁFICAS

ABRACICLO. *Produção de bicicletas deve crescer 7,3% em 2020*. 2020. Acesso em: 25 nov. 2024. Disponível em: <https://abraciclo.com.br/press-releases-2020/2020/producao-de-bicicletas-deve-crescer-73-em-2020/>.

BENINGTON, H. D. Production of large computer programs. 1956. Discussed the challenges associated with producing large and complex computer programs.

COCKBURN, A. *Hexagonal Architecture – The Pattern: Ports and Adapters*. 2005. [on-line]. Disponível em: <https://alistair.cockburn.us/hexagonal-architecture>.

DART. *Dart language tour*. 2024. Acesso em: 5 set. 2024. Disponível em: <https://dart.dev/guides/language>.

DELONG, P. C. C. *TRAIL BUILDING - MUITO MAIS QUE CONSTRUIR TRILHAS*. 2012. Disponível em: <https://www.pedal.com.br/trail-building-muito-mais-que-construir-trilhas_texto14492.html>.

DIGITAL, D. *Aplicativos Mobile: Multiplataforma x Nativo, qual utilizar?* 2024. Acessado em: 27 nov. 2024. Disponível em: <https://www.dtidigital.com.br/blog/aplicativos-mobile-multiplataforma-vs-nativo-qual-utilizar>.

ELLIOTT, E. *Programming JavaScript applications: Robust web architecture with node, HTML5, and modern JS libraries*. [S.l.]: "O'Reilly Media, Inc.", 2014.

EVANS, E. *Domain-driven design: tackling complexity in the heart of software*. [S.l.]: Addison-Wesley Professional, 2004.

FEIXA, C. La aventura imaginaria. una visión antropológica de las actividades físicas de a ventura en la naturaleza. *Apunts. Educación Física y Deportes*, v. 3, n. 41, p. 36–43, 1995.

FLUTTER. *Android Deployment*. Disponível em: https://docs.flutter.dev/deployment/android. Disponível em: <https://docs.flutter.dev/deployment/android>.

FLUTTER. *Flutter Showcase*. 2024. Acessado em: 25 nov. 2024. Disponível em: <https://flutter.dev/showcase>.

FOWLER, M. *Patterns of enterprise application architecture*. [S.l.]: Addison-Wesley, 2012.

Google Play Store. *PlaceRiders - Aplicativos*. 2024. [Acessado em: 5 set. 2024]. Disponível em: <https://play.google.com/store/search?q=placeRiders&c=apps&hl=pt_BR>.

HILLIARD, R. Ieee-std-1471-2000 recommended practice for architectural description of software-intensive systems. *IEEE, http://standards. ieee. org*, v. 12, n. 16-20, p. 2000, 2000.

MARTIN, R. C. *The Clean Architecture*. 2012. [S. L.]: [S. I.]. Disponível em: <https://blog cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html>.

SHAW, M.; GARLAN, D. *Software architecture: perspectives on an emerging discipline*. [S.l.]: Prentice-Hall, Inc., 1996.

Strava Inc. *Strava: Run, Ride, Hike*. 2024. *Aplicativo móvel*. Disponível em: <https://www.strava com>. Acesso em: 25 nov. 2024.

TERUYA, R. O lazer nas atividades em integração com a natureza, 2000. 44 f. *Trabalho de Conclusão de Curso (Graduação em Educação Física)–Instituto de Biociências, Universidade Estadual Paulista, Rio Claro*, 2000.

TRAILFORKS. Trailforks, 2024. Disponível para Android e iOS. Disponível em: <https: //www.trailforks.com/>.