**Documentação do Projeto Mobile**

**1. Informações Iniciais**

Nome do Projeto: Gamerail-app

Dupla:Marcos Antônio e Marcus Pascoal

Turma: 3º Ano

Professor: Eduardo Almeida e Carlos Renan

**2. Visão Geral do Projeto**

Aplicativo mobile desenvolvido no FlutterFlow para o armazenamento de jogos e produtos, como acessórios e itens colecionáveis. O sistema permite cadastro de usuários, listagem e detalhamento de jogos e produtos e seus preços, com backend em Laravel responsável pelo armazenamento e validação dos dados.

**3. Funcionalidades Implementadas**

Funcionalidades atuais:

- Tela de login

- Tela de Cadastro

- Botão de logout

- Tela de listagem de Games

- Tela de listagem de Products

- Tela de Criar Games

- Tela de Editar Games

- Botão de deletar Games

- Tela de Criar Products

- Tela de Editar Products

- Botão de deletar Products

- Integração com API  
- Banco Supabase

Funcionalidades em andamento:

- Autenticação  
- Cadastro

-Logout

**4. Como Executar o Projeto**

**O que preciso fazer para executar seu projeto ?**

4.1. FlutterFlow (Frontend)

1. Acesse o projeto no FlutterFlow: https://app.flutterflow.io/project/projeto-marcos-e-marcus-kwyeu

2. Telas disponíveis:

Login,

Registrar,

Dashboard,

Listagem de Jogos(games),

Criar Jogos(games),

Editar Jogos(games),

Listagem de Produtos de Jogos (Products),

Criar Produtos de Jogos (Products),

Editar Produtos de Jogos (Products).

4.2. API Backend

1. Clone o repositório: git clone https://github.com/marcosanton1o/gamerail-flutterflow.git
2. Vá em env.example e troque o nome para .env (ou copia e cola o arquivo env.example, assim trocando o nome para .env)
3. Abra o terminal do projeto e use Composer install
4. Ainda no terminal do projeto use php artisan generate:key para gerar uma chave no arquivo .env
5. No terminal use php artisan serve para a api do projeto ficar disponível na máquina local
6. Agora irá fazer o login pelo terminal com o ngrok (caso tenha o ngrok logado na máquina, utilize em um terminal qualquer “ngrok config unset authtoken” para sair da conta na qual está logada).  
   1. Para logar vá para um terminal qualquer e em seguida utilize o auth token do ngrok “taltal” para logar na máquina a conta existente para o funcionamento do link que é usado no ngrok
   2. Para conseguir compartilhar seu projeto com o flutterflow a api deve ser disponível globalmente, para isso, em um terminal qualquer (não sendo o do php artisan serve) utilize o link do ngrok “ngrok http --url=longhorn-bright-hog.ngrok-free.app 8000” (sem as aspas) no terminal
7. Agora apenas vá ao flutterflow e está pronto para começar a aplicação

**5. Testes**

Foram realizados testes de integração para as entidades principais: Games e Products, incluindo criação, atualização, listagem e remoção.

Categorias de testes realizadas:

* Validação de requisição: Foram criados testes específicos para verificar se o backend rejeita dados incompletos ou inválidos, usando os arquivos GameStoreRequestTest, GameUpdateRequestTest, ProductStoreRequestTest e ProductUpdateRequestTest.
* CRUD funcional: Testes para garantir que é possível criar, atualizar, visualizar e excluir registros corretamente, com arquivos como GameTest e ProductTest.

Exemplos de verificação feita nos testes:

* Verificação de status HTTP (como 200, 201, 204, 422)
* Verificação de erros de validação retornados pelo JSON
* Confirmação de que os dados foram armazenados corretamente no banco de dados
* Testes com dados parciais válidos e dados totalmente inválidos

Comando de execução dos testes:

php artisan test

Organização dos arquivos de teste:

* tests/Feature/GameTest.php
* tests/Feature/ProductTest.php
* tests/Feature/GameStoreRequestTest.php
* tests/Feature/GameUpdateRequestTest.php
* tests/Feature/ProductStoreRequestTest.php
* tests/Feature/ProductUpdateRequestTest.php

**6. Estrutura do Banco de Dados (Supabase)**

Tabelas Criadas:

Tabela: users  
- id  
- name  
- email  
- password

Tabela: games  
- id  
- title  
- price

- image

- developer

- publisher

- category  
- total\_sales

Tabela: products  
- id  
- name  
- description

- image

- price

- category  
- total\_sales

Link do projeto Supabase: <https://supabase.com/dashboard/project/mhvgvlxruwputxwhknfc>

**7. Integração App + API**

A integração entre o aplicativo (desenvolvido com FlutterFlow) e a API (desenvolvida com Laravel) foi realizada utilizando o serviço ngrok, que permitiu a exposição temporária da API local por meio de uma URL pública. Essa abordagem facilitou a comunicação entre o frontend e o backend durante o desenvolvimento.

Endpoints Utilizados

A API foi estruturada com os seguintes endpoints RESTful:

Games

* GET /games – Lista todos os jogos
* POST /games – Cria um novo jogo
* GET /games/{id} – Retorna os dados de um jogo específico
* PUT /games/{id} – Atualiza os dados de um jogo
* DELETE /games/{id} – Remove um jogo

Products

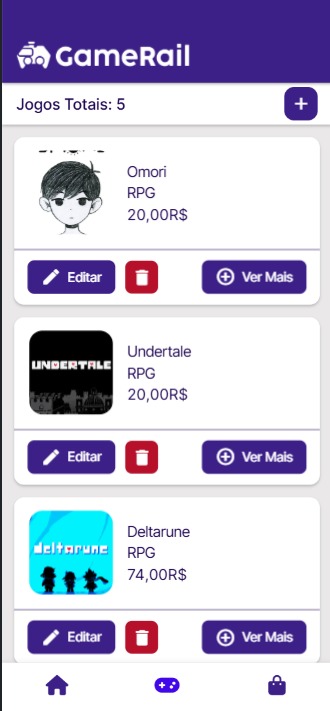
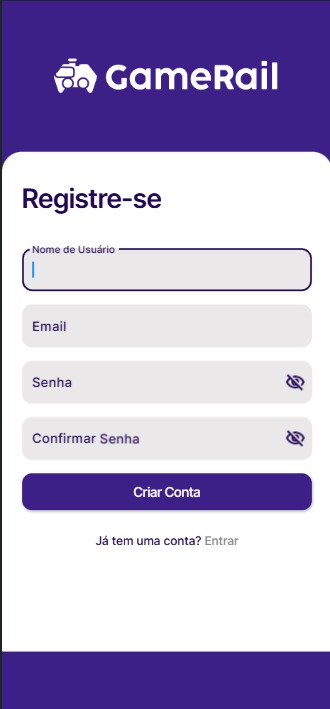
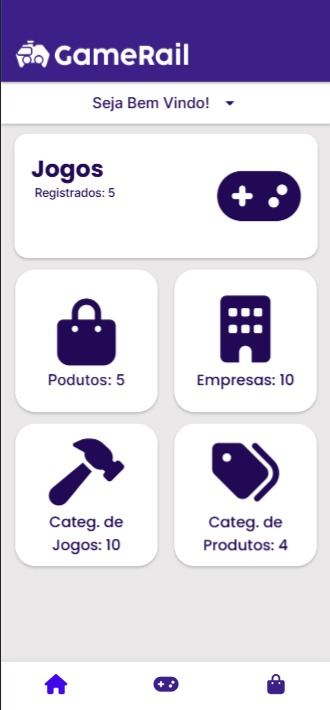
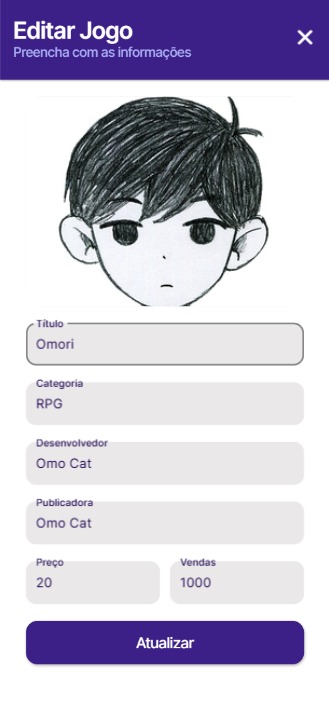
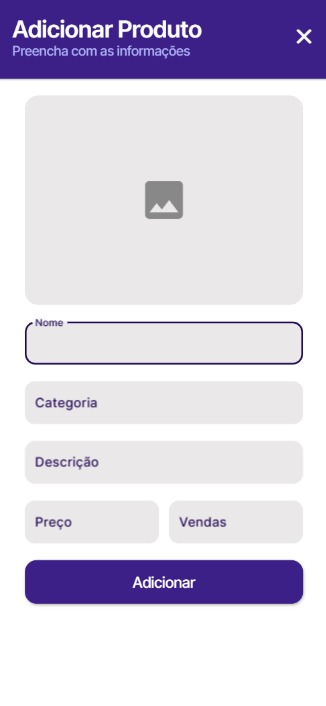
* GET /products – Lista todos os produtos
* POST /products – Cria um novo produto
* GET /products/{id} – Retorna os dados de um produto específico
* PUT /products/{id} – Atualiza os dados de um produto
* DELETE /products/{id} – Remove um produto

Segurança

Durante a fase de desenvolvimento, o acesso à API foi feito por meio de uma URL pública gerada pelo ngrok. Para fins de segurança:

* As requisições foram feitas via HTTPS, garantindo a criptografia dos dados em trânsito.
* Foi implementado o controle de acesso na API utilizando middleware de autenticação, protegendo as rotas sensíveis.
* Os dados recebidos da aplicação foram validados usando o recurso FormRequest do laravel, evitando falhas de segurança.

**8. Prints do App**



**9. Exportação de Rotas (Postman)**

As rotas podem ser testadas após a ativação do ngrok, os arquivos json do postman se encontram na pasta postman do projeto disponível no github

REST API basics- CRUD, test & variable.postman\_collection (1) (é o teste de rotas dos jogos);

REST API basics- CRUD, test & variable.postman\_collection (é o teste de rotas dos produtos dos jogos).