**FIAP Cloud Games (FCG) MVP** Introdução

* **Objetivo:**

Na Fase 1, o foco foi desenvolver uma API RESTful para o gerenciamento de usuários e sua biblioteca de jogos digitais.

Na Fase 2, o projeto evoluiu com foco em:

- Deploy em nuvem (Azure)

- Containerização com Docker

- Observabilidade e métricas com Application Insights + Grafana

- Integração Contínua e Entrega Contínua (CI/CD) via GitHub Actions

**Tecnologias utilizadas**

- .NET 8 + ASP.NET Core Web API  
- Entity Framework Core 8  
- JWT (Json Web Token)  
- Docker + Docker Compose  
- Azure Container Registry + Azure Container Apps  
- Azure Database for MySQL  
- Application Insights  
- Grafana (via Azure Monitor datasource)  
- GitHub Actions CI/CD  
- Serilog  
- xUnit + Moq  
- Clean Architecture + DDD

# Arquitetura

* **Padrão usado**: Clean Architecture + DDD (Domain-Driven Design)
* **Camadas**

**Domain:** Entidades, Enum, Value Objects

**Application:** Casos de uso (Handlers, Requests, Responses)

**Infra:** Repositórios e Contexto EF Core **API:** Controllers, Middleware, Configuração

**Fluxo:** Código -> Github actions -> Docker build and push -> ACR -> Azure db mysql -> Application insights -> Grafana

# Funcionalidades Implementadas

**Usuários:**

Criar, atualizar, deletar, buscar por ID e email

Login com geração de token JWT Promoção de usuário para Admin

**Jogos:**

CRUD de jogos (somente admins)

**UserGames:**

Comprar jogo, listar jogos do usuário, remover jogo

Autenticação/Autorização

JWT Policies por Role (Admin / User)

**Middleware de exceção global**

**Swagger UI com autenticação via Bearer Token**

Clicar em autenticação e apenas colar o token depois de logar.

**Log com Serilog**

Implementado ainda em poucas funcionalidades para teste.

# Endpoints

Requer

|  |  |  |  |
| --- | --- | --- | --- |
| Método | Rota | Descrição Perfil  Autenticação? | |
| POST | api/users | Cria um  novo Não  usuário | Público |
| POST | api/users/login | Autentica e  retorna Não  token | Público |
| POST | api/games | Cria novo  Sim jogo | Admin |
| POST | /api/Admin/{userId}/games/{gameId} | Adiciona  um Jogo a Sim  uma conta | Admin |
| POST | /api/UserGames/games/{gameId} | Adiciona  um jogo a  Sim conta autenticada | Ambos |
| GET | /api/Admin/{userId}/games | Lista jogos  de um Sim  usuário | Admin |
| GET | /api/Games | Lista todos  Sim os jogos | Ambos |
| GET | /api/Games/{id} | Retorna informações Sim do jogo | Ambos |
| GET | /api/User/Games/me/games | Mostra a  biblioteca  Sim do user autenticado | Ambos |
| GET | /api/users | Mostra  todos Sim  usuários | Admin |
| GET | /api/users/me | Mostra as  informações  Sim do usuário autenticado | Ambos |
| GET | /api/users/{id} | Mostra  informações  Sim do usuário buscado | Admin |

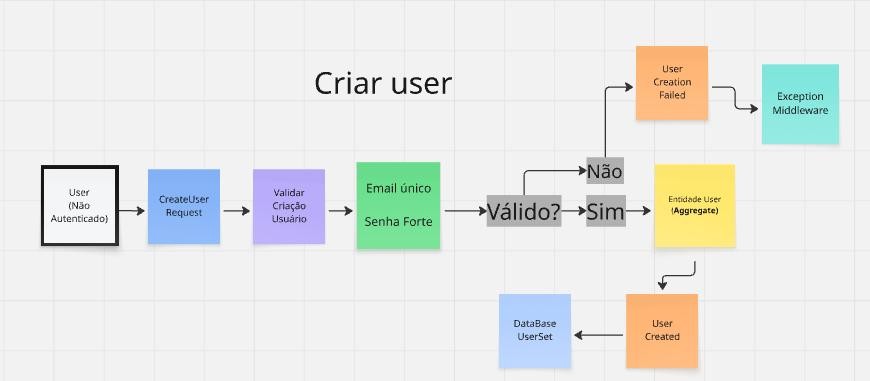
|  |  |  |  |
| --- | --- | --- | --- |
| Método | Rota | Descrição Perfil  Autenticação? | |
| por id |  |
| GET | /api/Users/email/{email} | Mostra  informações do usuário Sim buscado por email | Admin |
| PUT | /api/Admin/promote/{id} | Promover user para Sim admin | Admin |
| PUT | /api/Admin/users/{id} | Alterar  nome e password Sim  de usuário por id | Admin |
| PUT | /api/Games/{id} | Alterar  dados de  Sim um jogo por  id | Admin |
| PUT | /api/Users/me | Atualizar  nome e password Sim  de usuário autenticado | Ambos |
| Deletar um | |

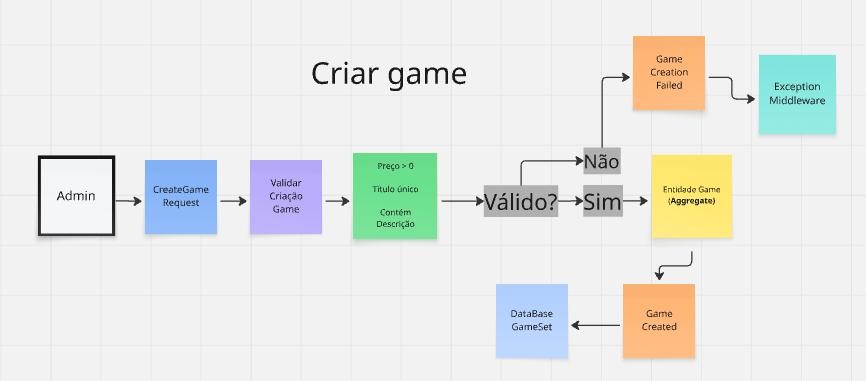
Requer

|  |  |  |
| --- | --- | --- |
| DELETE jogo de um Sim | | Admin |
|  | usuario |
| DELETE /api/Admin/{id} | Deletar um  Sim usuário | Admin |
| DELETE /api/Games/{id} | Deleta um  Sim jogo | Admin |
| DELETE /api/UserGames/games/{gameId} | Deletar jogo da  biblioteca Sim do usuário autenticado | Ambos |

/api/Admin/users/{userId}/games/{gameId}

# Event Storm





# Testes Automatizados

**Frameworks: xUnit + Moq**

**Testes implementados**:

ValueObjects (Email, Password, Title, Price, Description)

Handlers dos use cases (CreateUser, LoginUser, etc.)

Para rodar os testes abra o terminal na FCG.Tests e de dotnet test

# Execução do Projeto

**Pré-requisitos:**

.NET 8 SDK

MySQL

**Passos:**

**Restaurar pacotes**: dotnet restore **Configurar connection string:**

"Default":

"server=SEU\_SERVER;port=SUA\_PORTA;database=fcgdb;user=SEU\_USE

R;password=SEU\_PASS;"

**Criar Migration**: dotnet ef migrations “text”

**Subir para o Db**: dotnet ef database update

**Rodar o projeto:** dotnet run

**Acessar:** <https://localhost:porta/swagger>

# Pré‑requisito: Docker Desktop

docker compose up –build

CI / CD:

- .github/workflows/ci.yml

restore ➜ build ➜ test ➜ publish‑artifact

- .github/workflows/deploy.yml

docker build ➜ push ➜ az login ➜ az containerapp update

Código Fonte:

<https://github.com/devjoaomelo/fiap-cloud-games>

Pontos a serem melhorados no curto prazo:

* Documentação descritiva no swagger
* Interface front-end para melhor visualização
* Adição de saldo para usuário para conseguir adquirir game
* Refatorar código principalmente seção de log/tratamento de erros

João Vitor Gonçalves de Melo