**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS**NÚCLEO DE EDUCAÇÃO A DISTÂNCIA

Pós-graduação *Lato Sensu* em Desenvolvimento Web Full Stack

**TAYLOR RODRIGUES LOPES**

**APLICATIVO JUNTAR DINHEIRO**

Belo Horizonte

2024

**SUMÁRIO**

[1. Apresentação 3](#__RefHeading___Toc3240_2624087216)

[1.1. Contexto 3](#__RefHeading___Toc3242_2624087216)

[1.2. Público alvo 4](#__RefHeading___Toc3244_2624087216)

[1.3. Requisitos 4](#__RefHeading___Toc3246_2624087216)

[2. Modelagem 6](#__RefHeading___Toc3248_2624087216)

[2.1. Diagrama de casos de uso 6](#__RefHeading___Toc3250_2624087216)

[2.2. Atores 7](#__RefHeading___Toc3252_2624087216)

[2.3. Detalhamento dos casos de uso 7](#__RefHeading___Toc3254_2624087216)

[2.4. Projeto de Interface 8](#__RefHeading___Toc3256_2624087216)

[2.5. Diagrama de classes 10](#__RefHeading___Toc3258_2624087216)

[3. Projeto 10](#__RefHeading___Toc3260_2624087216)

[3.1. Arquitetura de *software* 10](#__RefHeading___Toc3262_2624087216)

[3.2. Arquitetura de Dados 14](#__RefHeading___Toc3264_2624087216)

[4. Testes 15](#__RefHeading___Toc3266_2624087216)

[5. URLs 16](#__RefHeading___Toc3268_2624087216)

[5.1. Aplicação web 16](#__RefHeading___Toc3270_2624087216)

[5.2. Repositório código-fonte 16](#__RefHeading___Toc3272_2624087216)

[5.3. Vídeo de apresentação do trabalho 16](#__RefHeading___Toc3274_2624087216)

[REFERÊNCIAS 17](#__RefHeading___Toc3276_2624087216)

[ANEXO A 18](#__RefHeading___Toc2247_1309324954)

# 1. Apresentação

## 1.1. Contexto

Segundo a B3 [1], bolsa de valores brasileira, o número de investidores vem crescendo anualmente, chegando em 2023 a 8,1 milhões, um aumento de 8%. Contudo, considerando o Censo 2022 do IBGE [2] que estimou uma população de 203 milhões no Brasil, pode-se dizer que menos de 4% da população de fato investe. Nota-se, portanto, um nicho de mercado com potencial a ser explorado.

Para controlar os investimentos, muitas corretoras oferecem aplicativos para compra e venda de ativos em tempo real, disponibilizando importantes indicadores para subsidiar as transações, além de prover formas, às vezes complexas, de gerenciar os ativos. Em geral, esses aplicativos não tem foco em uma estratégia ou plano de investimento que possa ser facilmente monitorado, o que é desejável por uma parcela de usuários que basicamente só pretende “*juntar dinheiro*” de modo simples e eficaz.

Dado o contexto e o “*gap*” de mercado, o presente trabalho tem elaborado o aplicativo web denominado “Juntar Dinheiro”. Esse aplicativo baseia-se no método de alocação de ativos de *Benjamin Graham*, descrito no livro "*O Investidor Inteligente"* [3]. A proposta é criar uma carteira de investimento para médio e longo prazo, composta, por exemplo, de 50% de renda fixa (menor risco) e 50% de renda variável (maior risco), ajustável pelo investidor mediante sua aversão à perda. Essa estratégia também prevê aportes regulares, sempre buscando recompor os percentuais inicialmente definidos, evitando que o investidor reaja emocionalmente à flutuação do mercado financeiro, cabendo-lhe apenas seguir sua estratégia (% definidos) de composição da carteira.

O aplicativo “Juntar Dinheiro” trata-se, portanto, de uma solução para controlar investimentos guiados por uma estratégia de alocação de ativos. Tal modelo de negócio pode ser melhor compreendido pelo *Lean Canvas* descrito no Anexo A, bem como por aspectos técnicos, diagramas e protótipos a seguir apresentados.

## 1.2. Público alvo

Por ser um aplicativo relacionado a investimentos, seu público-alvo abrange indivíduos interessados em compor uma carteira de ativos diversificada seguindo uma estratégia de médio e longo prazo (*buy and hold*), não sendo indicado para investidores com perfil especulativo (*day trading*).

Espera-se que o usuário do aplicativo tenha alguma familiaridade com operações financeiras via Internet, seja por bancos digitais, corretora de bolsa de valores, tesouro direto ou outras modalidades. Embora não seja escopo do aplicativo realizar essas operações, é importante o prévio conhecimento desse tipo negócio e tecnologia para melhor construir uma boa estratégia e registrar as transações realizadas (compra e venda de ativos) para que o aplicativo faça o monitoramento.

## 1.3. Requisitos

A seguir, são descritos os requisitos funcionais e não funcionais da aplicação:

**1.3.1 Requisitos Funcionais**

RF-01 – O aplicativo deve permitir o usuário realizar login.

RF-02 – O aplicativo deve permitir o usuário realizar logoff.

RF-03 – O aplicativo deve permitir o usuário se cadastrar no sistema.

RF-04 – O aplicativo deve permitir o usuário recuperar a senha.

RF-05 – O aplicativo deve permitir o usuário manter a carteira de ativos.

RF-06 – O aplicativo deve permitir o usuário manter os ativos.

RF-07 – O aplicativo deve permitir o usuário manter as classes de ativos.

RF-08 – O aplicativo deve permitir o usuário manter aportes (transações).

RF-09 – O aplicativo deve permitir o usuário exibir o painel (*dashboard*).

**1.3.2 Requisitos Não Funcionais**

RNF-01 – O aplicativo será desenvolvido utilizando a linguagem de programação Python ,versão 10+.

RNF-02 – O aplicativo será desenvolvido utilizando o framework FastAPI na plataforma Visual Studio Code.

RNF-03 – O aplicativo utilizará banco de dados SQL Server, versão 2022, como SGBD principal.

RNF-04 – O aplicativo fará uso do framework Bootstrap, vesrão 5.3, para o desenvolvimento da interface do usuário.

RNF-05 – O Git será adotado como sistema de controle de versão, hospedado no repositório do GitHub.

RNF-06 – A documentação do projeto deve ser mantida no formato *markdown* no diretório principal.

RNF-07 – O aplicativo deve ser implantado utilizando tecnologia de container com Docker.

RNF-08 – O código do sistema deve ter cobertura de teste de no mínimo 80%.

RNF-09 – O tempo de resposta para qualquer interação do usuário não deve exceder 5 segundos.

RNF-10 – Prever a interoperabilidade para principais funcionalidades por meio de APIs com autenticação/autorização com Oauth2 e JWT.

RNF-11– O aplicativo web deve ser compatível com as versões recentes dos navegadores Chrome, Firefox e Safari.

RNF-12 – O aplicativo web deve ser responsivo e compatível com smartphones e tablets.

# 2. Modelagem

## 2.1. Diagrama de casos de uso

O diagrama de casos de uso oferece uma visão global dos casos de uso e dos atores que participam da aplicação:

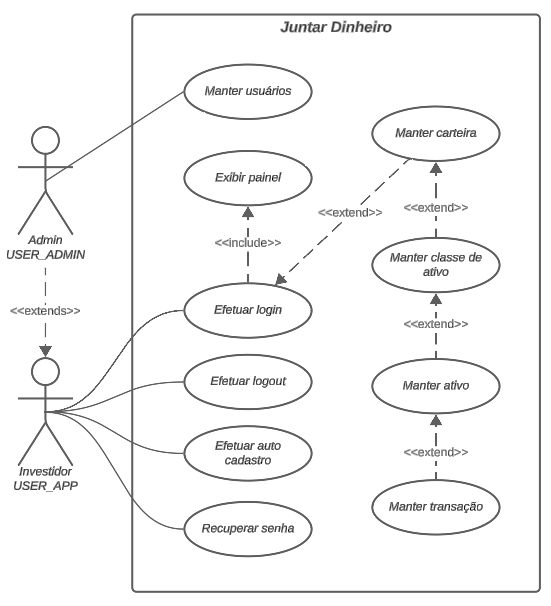
****

Fig. 1 – Diagrama de Caso de Uso  
Produzido com Lucid, em [https://lucid.app](https://lucid.app/)

## 2.2. Atores

O aplicativo Juntar Dinheiro” possui dois tipos de usuários: (i) o usuário investidor (USER\_APP), que faz uso convencional do aplicativo como investidor; e (ii) o usuário administrador (USER\_ADMIN), que possui ações restritas de gerenciamento de usuário e outros processos administrativos que futuramente venham a ser implementados, sempre preservando a privacidade dos dados financeiros dos usuários investidores.

## 2.3. Detalhamento dos casos de uso

Nesta seção, dois dos principais casos de uso são detalhados.

|  |  |
| --- | --- |
| **Caso de Uso (UC)** | RF-01 |
| **Funcionalidade** | O aplicativo deve permitir o usuário realizar login. |
| **Ator(es)** | Usuário investidor (USER\_APP) e admin (USER\_ADMIN). |
| **Breve descrição** | Esta funcionalidade permite que o usuário acesse os recursos do aplicativo por meio de login. |
| **Fluxo básico** | - O aplicativo exibe um formulário de login com os campos “usuário” e “senha”.  - O usuário fornece as credenciais de acesso e submete o formulário.  - O aplicativo autentica as informações no banco de dados.  - O aplicativo redireciona para o painel do usuário. |
| **Fluxos alternativos** | - Se as credenciais inseridas pelo usuário forem inválidas, o aplicativo exibe uma mensagem de erro.  - Deve-se fornecer um link “*Esqueceu a senha?*” para recuperação de senha.  - Deve-se fornecer um link “*Não tem cadastro?*” para cadastro de novo usuário. |
| **Pré-condições** | - A página de login deve estar acessível.  - Os campos do formulário devem estar vazios. |
| **Pós-condições** | - Os campos do formulário de login permanecem preenchidos com as credenciais inválidas, se autenticação falhar. |
| **Relacionamentos** | Este UC tem vínculos com RF-05 (carteiras) e funcionalidades vinculadas na Fig. 1 que dependem de login. |

|  |  |
| --- | --- |
| **Caso de Uso (UC)** | RF-05 |
| **Funcionalidade** | O aplicativo deve permitir o usuário manter a carteira de ativos. |
| **Ator(es)** | Usuário investidor (USER\_APP). |
| **Breve descrição** | Esta funcionalidade permite que o usuário crie, liste, atualize e exclua uma carteira de investimento. |
| **Fluxo básico** | - O aplicativo exibe painel com o nome da carteira e respectivas classes de ativos e ativos propriamente.  - Se não tiver nenhuma carteira cadastrada, o aplicativo redireciona para página de criar nova carteira.  - O painel apresenta botões para criar “Nova carteira”, “Editar” e “Excluir” a carteira.  - Ao criar “Nova carteira” ou “Editar” a carteira:   * o usuário define o campo “Nome da carteira”; * o usuário define o seu perfil de investidor ( “conservador”, “moderado” ou “arrojado”); * o usuário pode manter (CRUD) a classe de ativos; * o usuário define o % de cada classe de ativo; * o usuário pode manter (CRUD) os ativos; * o usuário define o % de cada ativo; * o usuário submete (salva) o formulário e o aplicativo armazena na base de dados. * o aplicativo redireciona para o painel. |
| **Fluxos alternativos** | - Não há. |
| **Pré-condições** | - Ao criar nova carteira, os campos do formulário devem estar vazios.  - Ao editar carteira existente, os campos já devem vir preenchidos com valores do banco de dados. |
| **Pós-condições** | - Não há. |
| **Relacionamentos** | Este UC tem vínculos com RF-06 (ativos), RF-07 (classes de ativos) e e RF-08 (transações). |

## 2.4. Projeto de Interface

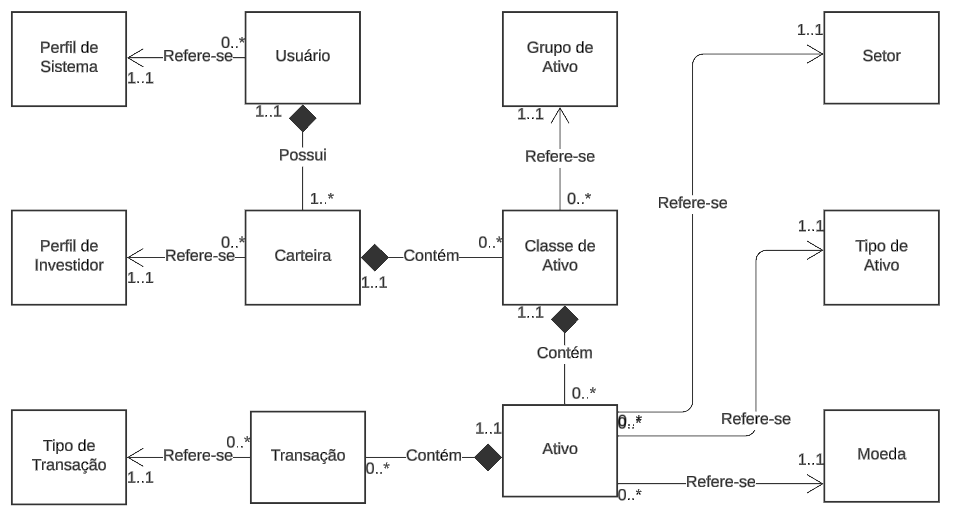
Esta seção apresenta esboços de baixa fidelidade de algumas das principais interfaces do aplicativo utilizando *wireframes*. Cabe destacar, que embora a aplicação seja majoritariamente voltada para web, foi utilizada o conceito de “*mobile first*”, vislumbrando futuramente a possibilidade de uso em dispositivos móveis.Os desenhos fora produzidos no site Balsamiq, em https://balsamiq.com/.

|  |  |
| --- | --- |
| Fig. 2 – Tela inicial | Fig. 3 – Tela de login |

|  |  |
| --- | --- |
| Fig. 4 – Tela Painel | Fig. 5 – Tela Nova carteira |

## 2.5. Diagrama de classes

O presente diagrama de classes (Figura 6) ilustra as classes e as relações entre elas, sem enfatizar os atributos e métodos. No aplicativo “Juntar Dinheiro”, um **usuário** pode possuir uma ou mais **carteiras** de investimentos. Cada carteira contém **classes** **de** **ativos**, que por sua vez contém **ativos**, que por sua vez contém **transações** (aportes). Cabe notar a associação do tipo composição representada pelo losango “fechado” ao lado do *objeto-todo*, caracterizando que as partes não existem sem o todo.

Fig. 6 – Diagrama de Classes  
Produzido com Lucid, em [https://lucid.app](https://lucid.app/)

# 3. Projeto

## 3.1. Arquitetura de *software*

O padrão arquitetural do aplicativo “Juntar Dinheiro” está baseada no MVC (*Model-View-Controlle*), visando melhor separar as camadas de **apresentação** (*interface do usuário),* ***lógica*** *(roteamento e chamadas às funcionalidades) e* ***modelos*** *(regra de negócio abrangendo entidades e dados).*

Os diagramas das Figuras 7 e 8 retratam em alto nível a visão arquitetural modelada em notação C4 Model (*context* e *container*). O diagrama de contexto (Nível 1) mostra a relação uma visão geral, possibilitando visualizar as fronteiras e os relacionamentos com cada item externo, no caso, uma API da *Mailtrap* para envio de e-mail e outra API da B3 para obter informações de ativos financeiros.

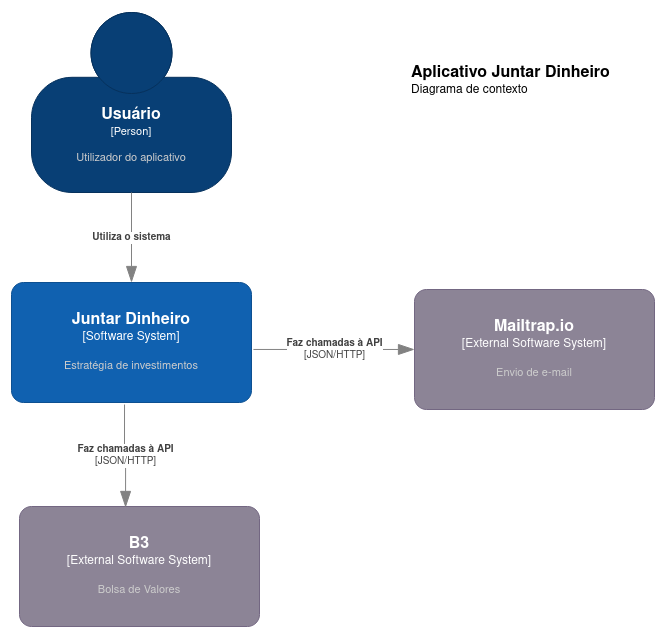


Fig. 7 – Diagrama de contexto  
Produzido com <https://app.diagrams.net/?libs=c4>

Para implantar o aplicativo foi contratado um Servidor Virtual Privado (VPS) Linux Debian da Hostinger (<https://www.hostinger.com.br/>) a fim de utilizar recursos personalizados como a linguagem Python e virtualização com Docker. Também foi registrado o domínio **juntardinheiro.com**.

Já a camada container (Nível 2) mostra as tecnologias usadas e como ocorre a comunicação. Nota-se que os dados são persistidos com Microsoft SQL Server, bem como existe uma API própria consumida pelo aplicativo Juntar Dinheiro.

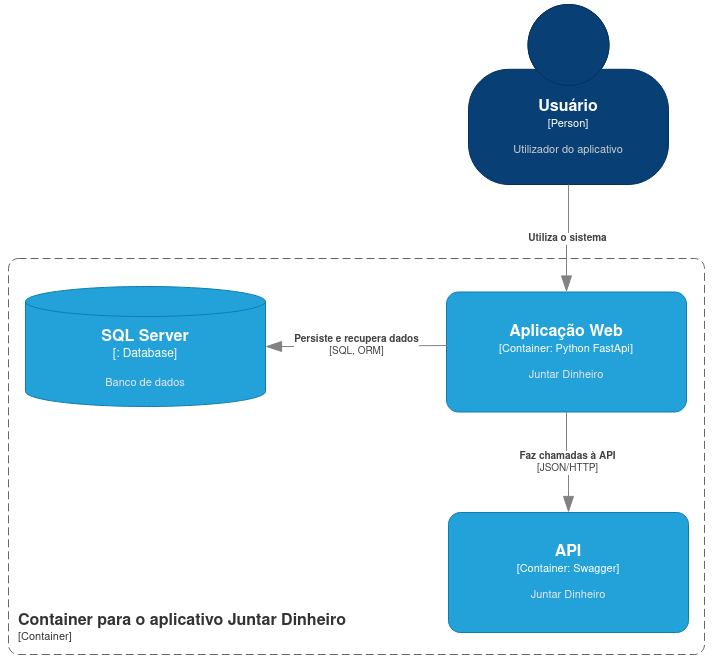


Fig. 8 – Diagrama de contexto  
Produzido com <https://app.diagrams.net/?libs=c4>

Essa API permite maior interoperabilidade e seus diversos *endpoints* estão devidamente documentados utilizando as especificações OpenAPI por intermédio do [Swagger](https://swagger.io/), disponível em <https://juntardinheiro.com:8000/docs>.

A autenticação e autorização para acesso aos recursos do aplicativo são realizados com [OAuth 2.0](https://oauth.net/2/) e token [JWT](https://jwt.io/), gerenciados por bibliotecas providas pelo Framework Python [FastAPI](https://fastapi.tiangolo.com/). A comunicação com os módulos é feita essencialmente via protocolo HTTP e JSON, além de todo tráfego ocorrer de forma segura utilizando certificados autoassinados (HTTPS).

Em termos de sistema de arquivos, o aplicativo Juntar Dinheiro está estruturado do seguinte modo:

── app

│ ├── api Módulo de APIs versionadas

│ │ └── v1 Versão 1 da API

│ ├── config.py Configuração geral do aplicativo

│ ├── database.py Configuração do banco de dados

│ ├── libs Bibliotecas especializadas

│ │ ├── security.py OAuth e JWT

│ │ └── utils.py Funcionalidades globais

│ ├── main.py Programa principal

│ ├── models.py Modelagem ORM

│ ├── schemas.py Modelagem de tipo e validação de dados

│ └── web Módulo de páginas e recursos estáticos

│ ├── carteira.json Dados-exemplo da carteira de investimento

│ ├── pages.py Rotas para as páginas

│ ├── static CSS, Java Script, Fonts, etc

│ └── templates Templates HTML (Jinja2)

├── docker-compose.yaml Configuração de criação do container Docker

├── Dockerfile Configuração de criação da imagem Docker

├── docs Documentação

├── pytest.ini Configuração de testes (pytest)

├── requirements.txt Lista de pacotes Python do aplicativo

├── ssl Repositório de certificados

└── tests Módulo de testes

## 3.2. Arquitetura d**e Dados**

A estrutura de dados do aplicativo Juntar Dinheiro estão graficamente representadas pelo Diagrama Entidade-Relacionamento – DER da Figura 9:

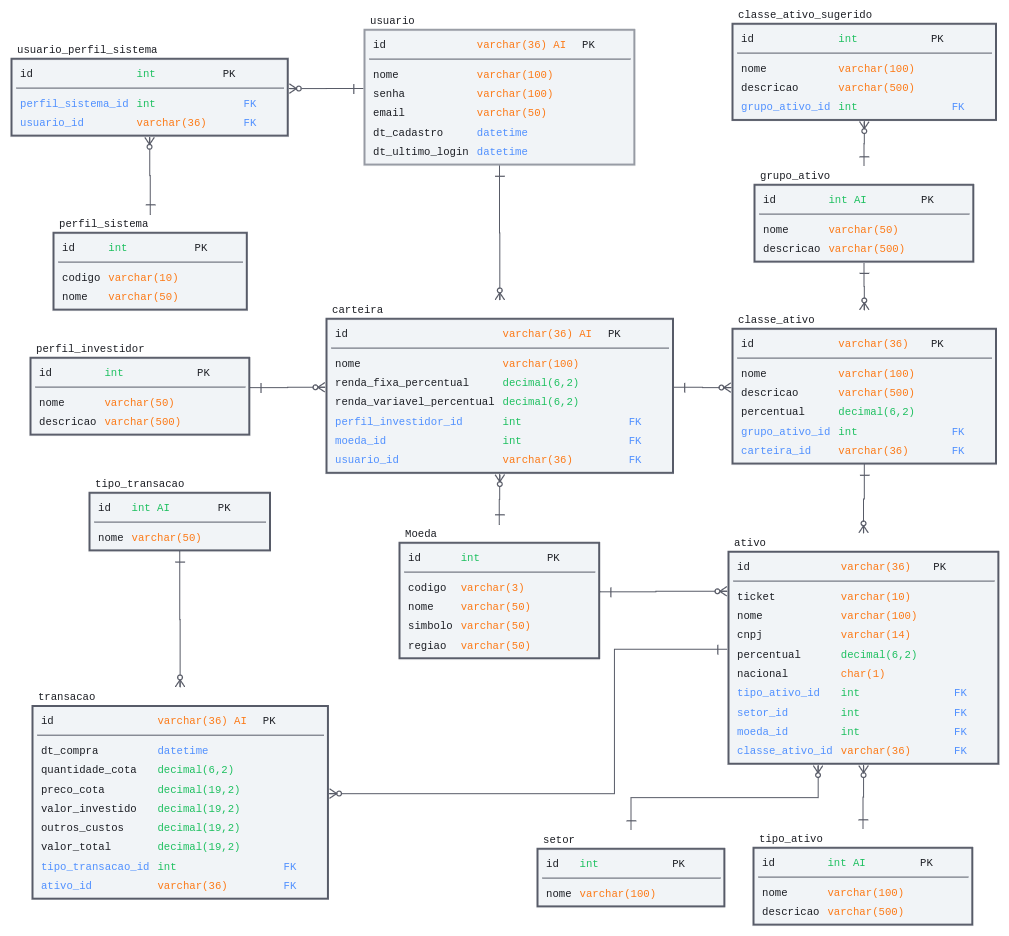
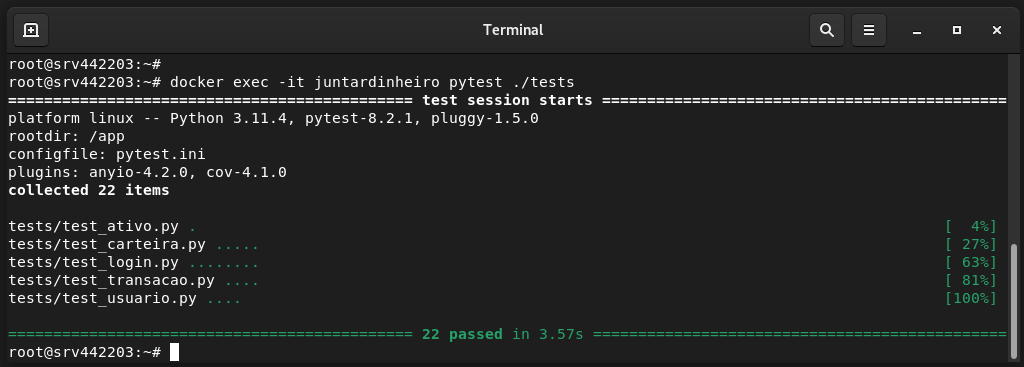


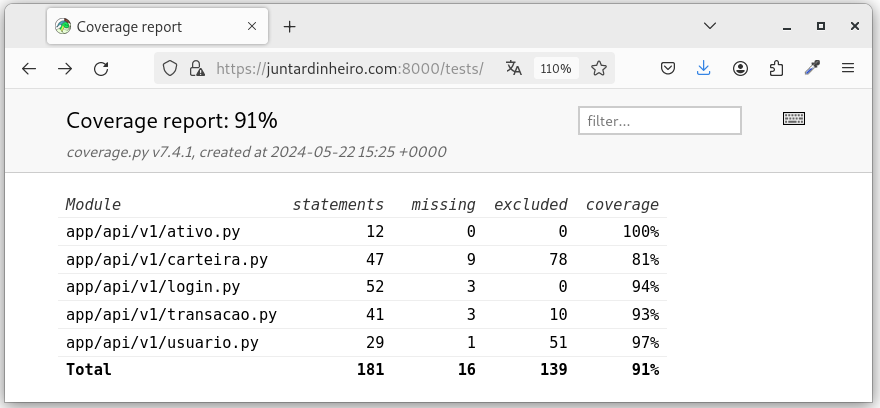
Fig. 8 – Diagrama Entidade-Relacionamento – DER  
Produzido com SqlDBM, em [https://sqldbm.com](https://sqldbm.com/)

# 4. Testes

Conforme mostra Figuras 9 e 10, foram realizados testes unitário e integração (api/endpoints) com cobertura média superior a 90%, disponível em <https://juntardinheiro.com:8000/tests/>, gerados com *pytest* e *pytest-cov*, por meio do seguinte comando:

|  |
| --- |
| docker exec -it juntardinheiro pytest ./tests --cov=app/api/v1 --cov-report html:tests/htmlcov |

Fig. 9 – Testes de integração

Fig. 10 – Cobertura de testes

# 5. URLs

## 5.1. Aplicação web

a. Endereço do aplicativo Juntar Dinheiro:

[https://juntardinheiro.com:8000](https://juntardinheiro.com:8000/)

b. Dados para login:

usuário: taylorlopes@gmail.com

senha: 123456

c. Endereço da API (Swagger)

<https://juntardinheiro.com:8000/docs>

Observação:

Requer autenticação (gerar token) para acessar a maioria dos *endpoints*.

## 5.2. Repositório código-fonte

O código do aplicativo Juntar Dinheiro está disponível no seguinte repositório do GitHub: <https://github.com/taylorlopes/juntardinheiro>

## 5.3. Vídeo de apresentação do trabalho

Inclua o link para o vídeo com a apresentação da sua aplicação. Seu vídeo deve ter duração máxima de 5 minutos e deve apresentar de forma sucinta o seu projeto. Concentre-se principalmente na apresentação da sua aplicação, mostrando o funcionamento de cada um dos requisitos solicitados na proposta escolhida por você.

O software OBS Studio é uma boa sugestão de software para gravação de screencast.

# REFERÊNCIAS

[1] B3, **Número de investidores na B3 cresce 34% em renda fixa e 23% em renda variável em 12 meses**, 05/06/2023. Disponível em https://www.gov.br/pt-br/noticias/financas-impostos-e-gestao-publica/2023/06/censo-2022-indica-que-o-brasil-totaliza-203-milhoes-de-habitantes. Acesso em: 11 de fev. de 2024.

[2] GOB.BR, **Censo 2022 indica que o Brasil totaliza 203 milhões de habitantes, 28/06/2023**. Disponível em https://www.gov.br/pt-br/noticias/financas-impostos-e-gestao-publica/2023/06/censo-2022-indica-que-o-brasil-totaliza-203-milhoes-de-habitantes. Acesso em: 11 de fev 2024.

[3] Benjamin Graham, **O investidor inteligente**. Disponível em https://www.amazon.com.br/investidor-inteligente-Lourdes-Sette/dp/8595080801. Acesso em: 11 de fev 2024.

# ANEXO A

Lean Canvas - Aplicativo Juntar Dinheiro

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PROBLEMA**  Estratégia ou plano de investimento que permita monitoramento | **SOLUÇÃO**  Aplicativo “Juntar Dinheiro” baseada no método de alocação de ativos de *Benjamin Graham* | **PROPOSTA DE VALOR**  Definição de percentuais de investimento (metas) em vários níveis para acompanhamento | | **VANTAGEM VENCEDORA** Evita que o investidor reaja emocionalmente à flutuação do mercado financeiro, bastando seguir sua estratégia (% definido) de composição da carteira, podendo operar com várias moedas simultaneamente | **CLIENTES**  Pessoa física que invistam a longo e médio prazo no mercado financeiro |
| **MÉTRICAS**  Número de usuários registrados | **CANAIS**  Redes sociais, blog, artigos, e-mail marketing, etc |
| **CUSTOS**  - Desenvolvimento do aplicativo Juntar Dinheiro  - Contratação de VPS ou Serverless como AWS e Azure - Registro de domínio - Aquisição de Certificado SSL - Contratação de APIs financeiras (*market data*) - Contratação de APIs para envio de e-mail - Investimento em divulgação e propaganda | | | **RECEITAS**  - Ganhos com plano pago de acesso “*premium*” em etapa futura - Ganhos com publicação de propagandas em contas gratuitas - Ganhos com patrocínios de agentes do mercado financeiro | | |