INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ
CAMPUS TERESINA CENTRAL
TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

VICTOR HUGO VIEIRA DE SOUSA

TRADING FIGHT CLUB - UM SIMULADOR DE CORRETORA FINANCEIRA

TERESINA, PIAUÍ

2022

VICTOR HUGO VIEIRA DE SOUSA

TRADING FIGHT CLUB - UM SIMULADOR DE CORRETORA FINANCEIRA

Projeto apresentado à Banca Examinadora como requisito para aprovação na disciplina de Trabalho de Conclusão de Curso II do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia do Piauí.

Orientador: Prof. Dr. Fábio de Jesus Lima Gomes

TERESINA, PIAUÍ

2022

Dados Internacionais de Catalogação na Publicação (CIP) de acordo com ISBD

Sousa, Victor Hugo Vieira de

S725t Trading fight club : um simulador de corretora financeira / Victor Hugo Vieira de Sousa. - 2022.
46 f.: il. color.

Trabalho de Conclusão de Curso (Tecnologia em Análise e Desenvolvimento de Sistemas) - Instituto Federal de Educação, Ciência e Tecnologia do Piauí, Campus Teresina Central, 2022.
Orientador : Prof Dr. Fábio de Jesus Lima Gomes.

1. Aplicativo. 2. Simulador financeiro. 3. B3. 4. Bolsa de Valores. I.Título.

CDD - 004.21

Elaborado por Maria Rosismar Farias CRB 3/631

VICTOR HUGO VIEIRA DE SOUSA

TRADING FIGHT CLUB - UM SIMULADOR DE CORRETORA FINANCEIRA

Projeto apresentado à Banca Examinadora como requisito para aprovação na disciplina de Trabalho de Conclusão de Curso II do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia do Piauí.

Aprovado pela banca examinadora em 16 de Dezembro de 2022.

BANCA EXAMINADORA:

Prof. Dr. Fábio de Jesus Lima Gomes
Instituto Federal de Educação, Ciência e Tecnologia do Piauí - IFPI

Prof. Dr. Adalton de Sena Almeida
Instituto Federal de Educação, Ciência e Tecnologia do Piauí - IFPI

Prof. M^{e}. Fernando Castelo Branco Gonçalves Santana
Instituto Federal de Educação, Ciência e Tecnologia do Piauí - IFPI

TERESINA, PIAUÍ

2022

Dedico este trabalho ao meu filho Henry.

AGRADECIMENTOS

Agradeço primeiramente aos meus pais pela criação humilde e ensino de valores inestimáveis, como a importância do estudo e seu papel como transformador social. A minha esposa Hildelanne por estar ao meu lado em todos os momentos dos últimos *N* anos e ser esta pessoa maravilhosa.

Agradeço ao meu orientador pela paciência, aos amigos, professores, sócios, colegas de trabalho, desconhecidos e anônimos que contribuíram para o meu crescimento pessoal, acadêmico e profissional, partilhando de conhecimento, ideias e paixões que me deram motivos para todos os dias levantar da cama e seguir em frente.

“Más notícias são as melhores amigas de um investidor”
(Warren Buffett)

RESUMO

A Bolsa de valores brasileira, conhecida como B3 registra nos últimos 4 anos uma grande onda de novos ingressantes. Que de acordo com dados divulgados em dezembro de 2020, mais de 2 milhões de pessoas iniciaram sua jornada de investimento na bolsa de valores, alcançando a marca de 3.2 milhões de investidores naquele ano.

Aliado ao relatório da Associação Brasileira das Entidades dos Mercados Financeiros e de Capitais (ANBIMA) em 2022, quase metade das pessoas que investem, utilizam aplicativo do banco ou da corretora para efetuar operações de compra ou venda.

Diante do cenário de crescimento e popularidade do mercado de ações no Brasil, este trabalho apresenta todo o processo de criação de um aplicativo móvel de código aberto que simula uma corretora financeira, com foco na interação simples e direta com o mercado de ações, sendo uma ferramenta para estudo e prática na compra e venda de ativos financeiros.

Palavras-chaves: Aplicativo. Simulador financeiro. B3. Bolsa de valores.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
MVVM	Model View View Model
GCP	Google Cloud Platform
HTTP	Hypertext Transfer Protocol API
SAAS	Software As A Service
IaaS	Infrastructure As A Service
PaaS	Platform As A Service
Pub/Sub	Publish–Subscribe
CF	Firebase Cloud Function
LLVM	Low Level Virtual Machine
CI	Continuous Integration
CD	Continuous Delivery
IU	Interface do Usuário
PR	Pull Request
B3	Brasil Bolsa Balcão
ANBIMA	Associação Brasileira das Entidades dos Mercados Financeiro e de Capitais
COPOM	Comitê de Política Monetária
SELIC	Sistema Especial de Liquidação de Custódia
CDI	Certificado de Depósito Interbancário
LCI	Letras de Crédito Imobiliário
LCA	Letras de Crédito do Agronegócio
LC	Letras de Câmbio
CDB	Certificado de Depósito Bancário
CRI	Certificado de Recebíveis Imobiliários
CRA	Certificado de Recebíveis do Agronegócio

SUMÁRIO

1 INTRODUÇÃO

Em 2015 a taxa, considerada taxa básica de juros, Selic (Sistema Especial de Liquidação de Custódia) havia chegado a 14,25% e permaneceu nesse nível por 15 meses, até que em outubro de 2016 a taxa começou a baixar gradualmente e em março de 2021, o Comitê de Política Monetária (Copom) promoveu novo corte, chegando a baixa histórica de 1,90% [1].

Com esta redução expressiva da taxa Selic, muitos investimentos de renda fixa tem sua lucratividade afetada por possuírem rentabilidades diretamente associada à taxa básica de juros, tais como: caderneta de poupança, títulos públicos, CDI (Certificado de Depósito Interbancário), LCI (Letras de Crédito Imobiliário), LCA (Letras de Crédito do Agronegócio), LC (Letras de Câmbio), CDB (Certificado de Depósito Bancário), CRI (Certificado de Recebíveis Imobiliários), CRA (Certificado de Recebíveis do Agronegócio) e Debêntures.

Sobre a poupança, vale ressaltar que por lei toda vez que a Selic for igual ou inferior a 8,5%, a remuneração da aplicação passa a ser de 70% da Selic acrescida da Taxa Referencial (TR)[2], que em momentos de crescimento econômico como em 2019 chegou em zero. Com isso, o rendimento ficou abaixo de 3%, inferior à inflação de 3,4% projetada para aquele mesmo ano, segundo relatório focus [3]. Desta forma, a modalidade de investimento mais utilizada no Brasil, teve retirada histórica em outubro de 2022 ultrapassando R$22 bilhões como divulgado pelo Banco Central [4].

Além disso, a redução da taxa, cria um ambiente favorável ao crescimento privado e por sua vez a renda variável. Tal cenário é evidenciado com a grande onda de novos ingressantes na bolsa de valores brasileira, conhecida como B3 (Brasil Bolsa Balcão). Que de acordo com dados divulgados em dezembro de 2020 [5], mais de 2 milhões de pessoas iniciaram sua jornada de investimentos na bolsa de valores entre abril de 2019 e abril de 2020, alcançando a marca de 3,2 milhões de investidores ainda naquele ano.

Segundo relatório divulgado pela Associação Brasileira das Entidades dos Mercados Financeiro e de Capitais (ANBIMA)[6] em 2022, 48% dos brasileiros vão pessoalmente até o banco para aplicar efetivamente seu dinheiro. Mas há quem prefira usar a tecnologia a seu favor: quase metade das pessoas investe pelo aplicativo do banco (45%) ou da corretora (9%) ou até pelo site do banco (14%) ou da corretora (9%).

Diante do cenário de crescimento e popularidade do mercado de ações no Brasil e no mundo, além da possibilidade de utilizar tecnologias móveis como aplicativos de negociação para interação simples e direta com o mercado de ações, uma ferramenta para estudo e prática na compra e venda de ativos financeiros se faz oportuna.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

O objetivo geral deste trabalho é prover uma aplicação Android de código aberto que viabilize a compra e venda de ativos financeiros em um ambiente emulado, que trabalhe com informações reais, gratuito e sem riscos de prejuízo.

1.1.2 Objetivos Específicos

- Planejar e desenvolver uma aplicação Android com as funcionalidades necessárias para operações de compra e venda de ativos financeiros;
- Disponibilizar uma versão do aplicativo para download juntamente com seu código fonte.

2 TECNOLOGIAS ENVOLVIDAS

Para o desenvolvimento do TradingFight.club, foram seguidos os guias oficiais de arquitetura o mais próximo possível. Com o foco em se manter de fácil entendimento e a possibilidade de muitos desenvolvedores trabalharem no mesmo projeto, nenhuma tecnologia experimental foi utilizada. Facilitando assim, testes unitários e instrumentados, seja em ambiente de desenvolvimento ou em uma *pipeline* de integração contínua.

2.1 GOOGLE CLOUD PLATFORM

O Google Cloud Platform (GCP)[1] é um conjunto de recursos de computação do Google, disponibilizados por meio de serviços ao público em geral como uma oferta de nuvem pública.

Os recursos do GCP consistem em infraestrutura de hardware físico — computadores, unidades de disco rígido, unidades de estado sólido e redes — contidas nos data centers globalmente distribuídos do Google, onde qualquer um dos componentes é projetado de forma personalizada usando padrões semelhantes aos disponíveis no *Open Compute Project*.

Esse hardware é disponibilizado aos clientes na forma de recursos virtualizados, como máquinas virtuais (VMs), como alternativa aos clientes que constroem e mantêm sua própria infraestrutura física.

Como uma oferta de nuvem pública, os produtos de software e hardware são fornecidos como serviços integrados que fornecem acesso aos recursos subjacentes. O GCP oferece mais de 50 serviços, incluindo ofertas de infraestrutura como serviço (IaaS), plataforma como serviço (PaaS) e software como serviço (SaaS) nas categorias de computação, armazenamento e bancos de dados, rede, big data, aprendizado de máquina, identidade e Segurança e Ferramentas de Gerenciamento e Desenvolvedor.

O GCP está hospedado na mesma infraestrutura subjacente que o Google usa internamente para produtos de usuário final, incluindo a sua ferramenta principal de pesquisa, Google Maps e o YouTube.

2.1.1 Google Scheduler

O Google Scheduler[2] é um agendador de tarefas recorrentes altamente customizável que permite acionar chamadas HTTP, disparar mensagens em tópicos de Pub/Sub ou serviços App Engine.

[1] https://cloud.google.com
[2] https://cloud.google.com/scheduler/docs

2.1.2 Google Pub/Sub

O Google Cloud Pub/Sub[3] é um serviço de mensagens para troca de dados de eventos entre aplicativos e serviços. Ao desacoplar remetentes e destinatários, ele permite uma comunicação segura e altamente disponível entre aplicativos escritos de forma independente. O Google Cloud Pub/Sub oferece mensagens de baixa latência/durabilidade e é comumente usado por desenvolvedores na implementação de fluxos de trabalho assíncronos, distribuição de notificações de eventos e transmissão de dados de vários processos ou dispositivos.

2.1.3 BigQuery

O BigQuery[4] é um tipo de sistema de gerenciamento de dados projetado para centralizar e consolidar grandes quantidades de dados de várias fontes. Seus recursos analíticos permitem que atividades de BI (*business intelligence*), consultas e análises avançadas em grandes quantidades de dados históricos sejam realizadas.

A arquitetura *serveless* do BigQuery permite executar rapidamente consultas SQL e analisar milhões de linhas de dados em segundos. O BigQuery também tem excelentes integrações com outros produtos do GCP, como Data Flow e Looker Studio, o que o torna uma ótima opção para tarefas de análise de dados.

2.1.4 Looker Studio

O Looker Studio[5], antigamente chamado de Data Studio, é um serviço gratuito de criação e visualização de relatórios com diversos tipos de gráficos, controles dinâmicos customizáveis. Por possuir integração com muitos serviços terceiros e aceitar diversas fontes e formatos de dados disponibilizada gratuitamente, é uma excelente ferramenta de *business intelligence*.

2.2 FIREBASE

O Firebase[6] é um conjunto de ferramentas da Google que ajuda você a criar, melhorar e expandir seu aplicativo. As ferramentas que ele oferece cobrem uma grande parte dos serviços que os desenvolvedores normalmente teriam que construir do zero, ao invés de focar na experiência do aplicativo em si. Isso inclui ferramentas como análise, autenticação, bancos de dados, configuração remota, armazenamento de arquivos, notificações, dentre outros. Os serviços são hospedados na nuvem e são dimensionados com pouco ou nenhum esforço por parte do desenvolvedor.

[3] https://cloud.google.com/pubsub/docs/overview
[4] https://cloud.google.com/bigquery
[5] https://cloud.google.com/looker-studio
[6] https://firebase.google.com/

2.2.1 Google Cloud Functions

O Google Cloud Functions[7] é um ambiente de execução *serveless* para criar e conectar serviços em nuvem. Com o Cloud Functions, você escreve funções simples e de propósito único que são anexadas a eventos emitidos de sua infraestrutura e serviços de nuvem, como: eventos de autenticação, banco de dados, pub/sub. Sua *cloud function* é acionada quando um evento que está sendo observado é acionado em um ambiente totalmente gerenciado. Não há necessidade de provisionar nenhuma infraestrutura ou se preocupar em gerenciar qualquer servidor.

2.2.2 Firebase Authentication

O Firebase Authentication[8] simplifica o processo de autênticação, provendo SDKs simples e bibliotecas de interface prontas para autenticar usuários. Ele suporta autenticação usando senhas, números de telefone, provedores de identificação populares como Google, Facebook, Twitter, Github e outros.

2.2.3 Cloud Firestore

Cloud Firestore[9] é um banco de dados NoSQL em tempo real altamente escalável e hospedado na nuvem. Para estruturar seus dados, você define coleções (semelhantes a tabelas em SQL) que contêm documentos (semelhantes a linhas). Cada documento contém campos que contêm os dados. Você pode fazer referência a um documento individual usando seu caminho exclusivo ou pode consultar uma coleção de documentos cujos campos contenham os dados que você está procurando.

2.2.4 Firebase Remote Config

O Firebase Remote Config[10] é um serviço de nuvem que permite alterar o comportamento ou a aparência do seu aplicativo sem exigir que os usuários façam o download de uma atualização do aplicativo. Ao usar o Remote Config, você cria valores padrão no aplicativo que controlam o comportamento ou a aparência do seu aplicativo. Em seguida, você pode usar o painel administrativo do Firebase ou as APIs disponíveis para substituir os valores padrão no aplicativo para todos os usuários do aplicativo ou para segmentos da sua base de usuários. Seu aplicativo controla quando as atualizações são aplicadas e pode verificar atualizações com frequência e aplicá-las com baixo impacto no desempenho.

[7] https://cloud.google.com/functions
[8] https://firebase.google.com/docs/auth
[9] https://firebase.google.com/docs/firestore
[10] https://firebase.google.com/docs/remote-config/

2.3 LINGUAGENS

Neste projeto, o aplicativo android foi inteiramente desenvolvido na linguagem de programação Kotlin, inclusive suas telas através da biblioteca gráfica Jetpack Compose[11].

Entretando, para funções mais complexas, como: compra e venda de ativos, criação de nova conta e *crawler* de preço dos ativos. Foi adotado a linguagem Python do lado do servidor através das Cloud Functions.

2.3.1 Kotlin

Kotlin[12] é uma linguagem de programação multiplataforma, orientada a objetos e funcional, concisa e estaticamente tipada (variáveis com tipos específicos), desenvolvida pela JetBrains em 2011, que compila para a Máquina virtual Java e que também pode ser traduzida para a linguagem JavaScript e compilada para código nativo (via LLVM). Foi anunciada em 2017 pela Google como a linguagem oficial do sistema Android.

A versão da JVM (Java Virtual Machine) de sua biblioteca padrão depende da Java Class Library, mas a inferência de tipos permite que sua sintaxe seja mais concisa. Apesar de possuir uma sintaxe mais concisa e um pouco diferente da linguagem Java, Kotlin é projetada para ter uma interoperabilidade total com código Java, agilizando assim a sua adoção.

2.3.2 Python

Python[13] é uma linguagem de programação de alto nível, interpretada de *script*, imperativa, orientada a objetos, funcional, de tipagem dinâmica e forte. Foi lançada por Guido Van Rossum em 1991. Atualmente, possui um modelo de desenvolvimento comunitário, aberto e gerenciado pela organização sem fins lucrativos Python Software Foundation. Apesar de várias partes da linguagem possuírem padrões e especificações formais, a linguagem, como um todo, não é formalmente especificada. O padrão na pratica é a implementação CPython[14].

2.4 ANDROID

Android [15] é um sistema operacional baseado no núcleo Linux, projetado principalmente para dispositivos eletrônicos móveis (como smartphones e tablets) com tela sensível ao toque ou interface de usuário baseada na manipulação direta; desenvolvido por um consórcio de desenvolvedores conhecido como Open Handset Alliance, sendo o principal colaborador o Google. Possui interface específica para aparelho televisivo (Android TV), carros (Android Auto) e relógios inteligentes (Wear OS).

[11] https://developer.android.com/jetpack/compose
[12] https://kotlinlang.org/
[13] https://www.python.org/
[14] https://github.com/python/cpython
[15] https://www.android.com/

2.4.1 Android Jetpack

O Android Jetpack[16] é um pacote de bibliotecas que ajuda desenvolvedores a seguir as práticas recomendadas, reduzir códigos *boilerplate* e programar códigos que funcionam de maneira consistente em diferentes dispositivos e versões do Android. Assim, os desenvolvedores podem se concentrar no código que realmente importa para eles.

2.4.2 Jetpack Compose

O Jetpack Compose[17] é um kit de ferramentas moderno para criar interface de usuário (IU) nativa do Android. O Jetpack Compose simplifica e acelera o desenvolvimento de IU no Android com menos código, ferramentas poderosas e APIs Kotlin intuitivas.

Jetpack Compose é uma programação totalmente declarativa, o que significa que você pode descrever sua interface de usuário invocando um conjunto de funções chamadas *composable*. Nos últimos dez anos, temos usado uma forma tradicional de design de interface do usuário imperativo.

2.5 GITHUB ACTIONS

O GitHub Actions[18] é uma plataforma de integração contínua e entrega contínua (CI/CD) que permite automatizar seu pipeline de compilação, teste e implantação. É possível implantar fluxos de trabalho no mesmo local em que armazena código e colabora em solicitações de *pull requests*, agendadas, entre outros eventos.

No GitHub Actions, um fluxo de trabalho é um processo automatizado que você configura no repositório do GitHub. Você pode criar, testar, empacotar, lançar ou implantar qualquer projeto no GitHub com um fluxo de trabalho.

Cada fluxo de trabalho é composto por ações individuais que são executadas após ocorrer um evento específico (pull requests, merges, etc). As ações individuais são *scripts* empacotados que automatizam tarefas de desenvolvimento de software.

2.6 SONAR CLOUD

O SonarCloud[19] é um serviço de análise de código baseado em nuvem projetado para detectar problemas de qualidade de código em mais de 20 linguagens de programação diferentes, garantindo continuamente a capacidade de manutenção, confiabilidade e segurança do seu código. Além de possuir integração com Github Actions e ser totalmente gratuito para projetos open-source.

[16] https://developer.android.com/jetpack
[17] https://developer.android.com/jetpack/compose
[18] https://github.com/features/actions
[19] https://www.sonarsource.com/products/sonarcloud/

3 MODELAGEM DO PROJETO

3.1 LEVANTAMENTO DE REQUISITOS

3.1.1 Requisitos Funcionais

Existe apenas um tipo de usuário no sistema e é preciso que esteja autenticado para poder utilizar as funcionalidades do sistema: Visualizar saldo, histórico de operações, classificação dos usuários da plataforma, efetuar operações de compra e venda de ativos financeiros, bem como outros casos de usos:

Código	Requisito	Descrição
RF1	Autenticar	Registro e validação de acesso ao sistema
RF2	Explorar	Procura e visualização de informações detalhadas de um ativo financeiro
RF3	Negociar ativos financeiros	Compra e venda de ativos financeiros
RF4	Ver carteira	Visualiza do portfólio e ordens de negociação em andamento
RF5	Ver ranqueamento	Lista ordenada de usuários com base em seu percentual de acertos
RF6	Visualizar progresso	Visualiza o progresso do usuário

Tabela 1 – Requisitos funcionais

3.1.2 Requisitos Não Funcionais

Sobre os requisitos relacionados ao ambiente onde a aplicação está inserida, como: Um servidor mais robusto, medidas de segurança ou um usuário especializado para realização de determinadas ações. Não devem ser ignorados por não fazerem parte diretamente da aplicação, mas devem ser considerados por compor o seu ambiente e, por vezes, determinante para a sua utilização. Como os relacionados na tabela a seguir:

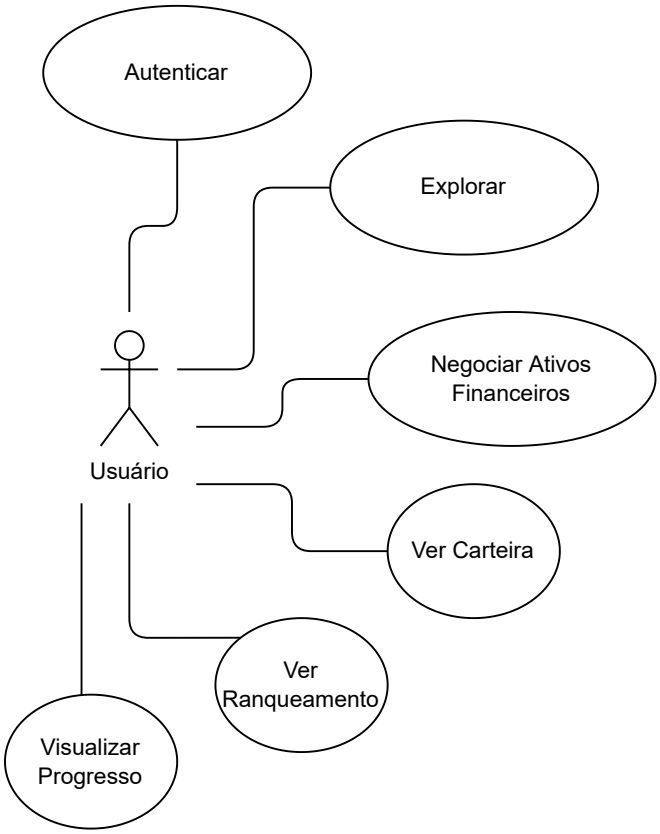
Código	Requisito	Descrição
RNF1	O sistema deve ser simples e intuitivo, provendo rapidez e facilidade nas interações	Usabilidade
RNF2	A Aplicação deve garantir a identificação do usuário	Segurança
RNF3	A Aplicação deve realizar as operações de compra e venda em até 15 minutos	Desempenho
RNF4	A Aplicação deve ser compatível com Android 7.0 ou superior	Compatibilidade

Tabela 2 – Requisitos não-funcionais

3.2 DIAGRAMAS DE CASOS DE USO

O diagrama de caso de uso desenvolvido mostrado na figura 1 descreve as possíveis interações do usuário através da ferramenta.

Figura 1 – Diagrama UML de Casos de Uso



Fonte: Elaborado pelo Autor

A partir deste diagrama foi criado os seguintes casos de uso:

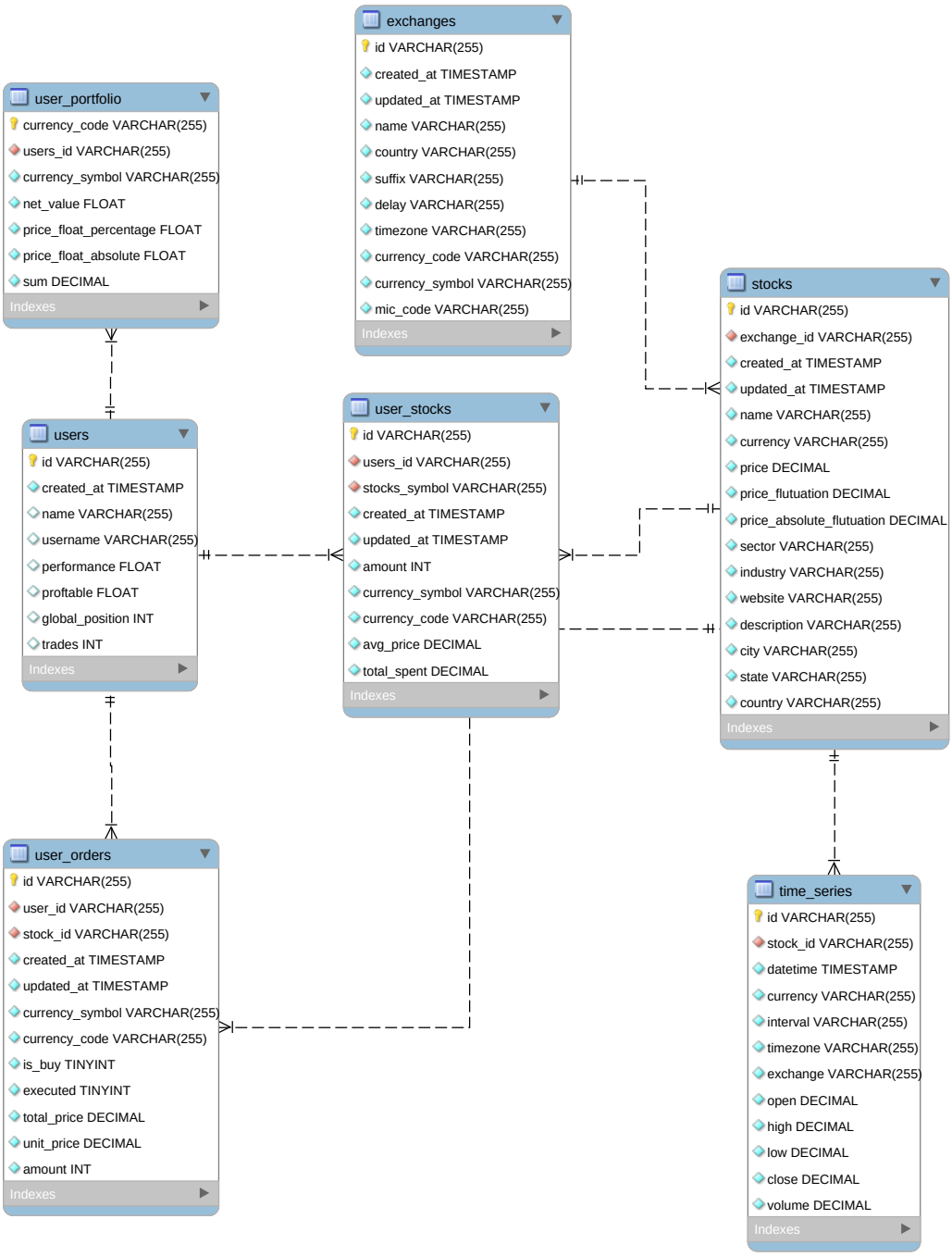
- **Caso de Uso 01 - Autenticar**: O usuário utilizando sua conta Google já existente;
- **Caso de Uso 02 - Explorar**: O usuário autenticado pode buscar e visualizar detalhes de um ativo financeiro;
- **Caso de Uso 03 - Negociar ativos financeiros**: O usuário autenticado pode efetuar comprar e venda de ativos financeiros;
- **Caso de Uso 04 - Ver Carteira**: O usuário acompanha os ativos comprados, seu preço e valorização atual, bem como ordens pendentes de execução e valor total do portfólio;
- **Caso de Uso 05 - Ver Ranqueamento**: É disponibilizado uma listagem de usuários ordenado pela taxa de performance financeira;
- **Caso de Uso 06 - Visualizar Progresso**: O usuário pode visualizar indicadores como: dinheiro disponível, valor total do patrimônio, valorização percentual, total de negociações, taxa de acertos e posição na classificação global. Além de acompanhar o histórico de suas negociações ao longo do tempo;

Os documentos com as especificações completas estão disponíveis no Anexo A deste trabalho.

3.3 DIAGRAMA DE ENTIDADES-RELACIONAMENTOS

Na figura 2 podemos ver a estrutura lógica usada como base para o banco de dados.

Figura 2 – Diagrama de Entidade Relacionamento



Fonte: Elaborado pelo Autor

3.4 ARQUITETURA DO SISTEMA

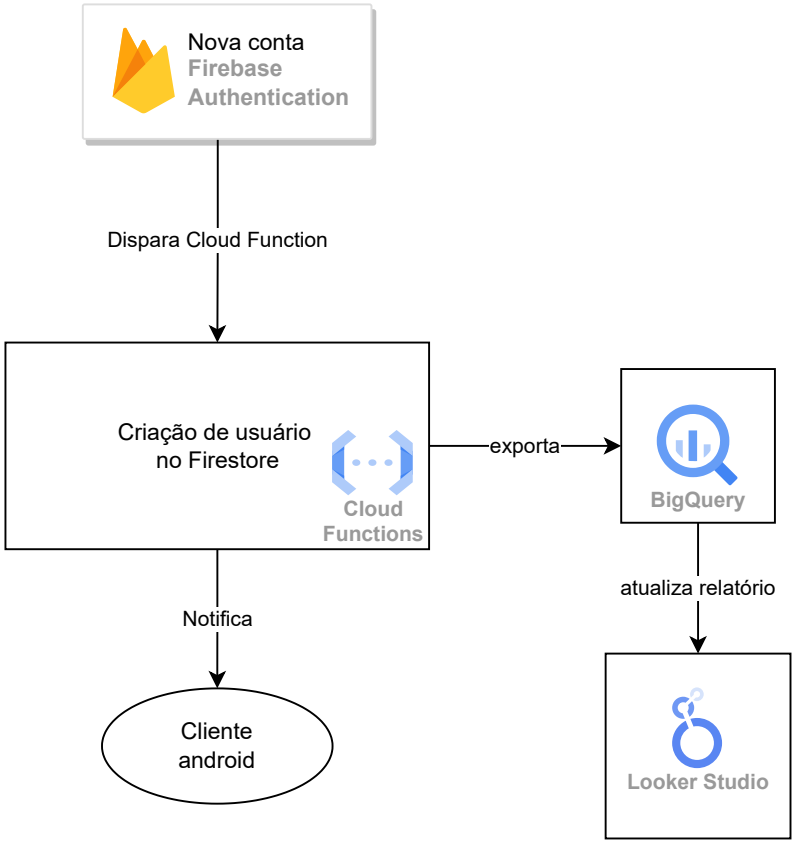
3.4.1 Processo de criação de conta

Apesar do Firebase prover muitas SDKs que dispensam a implementação de regras de negócio do lado do servidor, algumas vezes se faz necessário, seja por consistência de dados, complexidade de fluxo ou integrações externas, tais como envio de emails, notificações, entre outros.

Nestes cenários, utilizamos a Google Cloud Functions para executar nosso código de back-end sempre que um gatilho configurável for disparado, seja um item em um tópico de pub/sub, operação em banco de dados, conta de acesso, entre outros.

No fluxograma da figura 3, podemos ver como se dá a criação de um novo usuário.

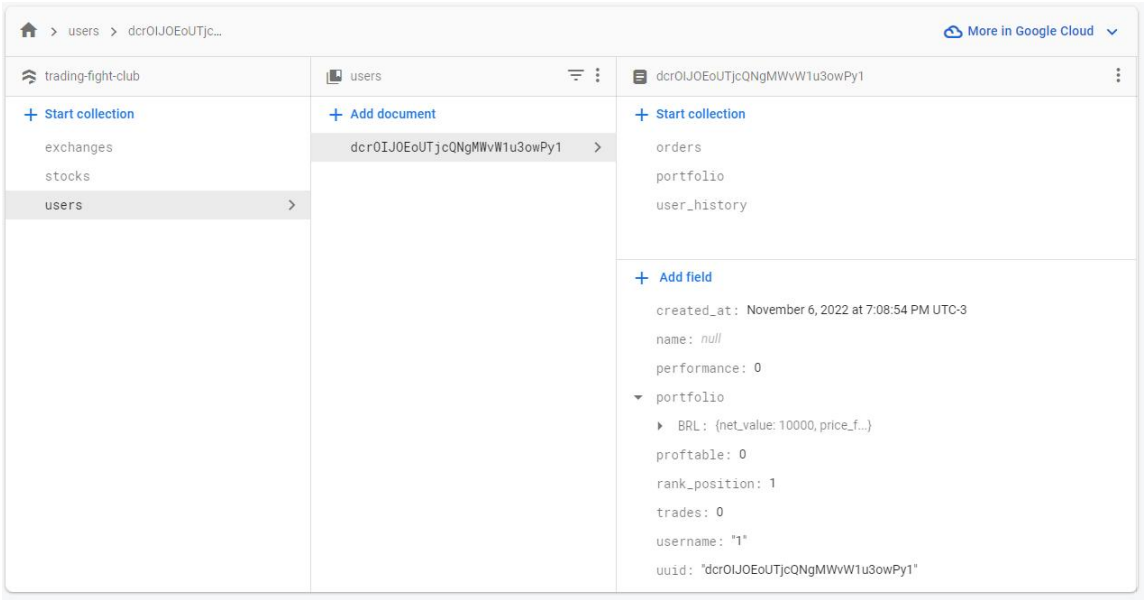
Figura 3 – Fluxograma de uma nova conta



Fonte: Elaborado pelo Autor

Sempre que um novo usuário é registrado atraves do SDK do Firebase Authentication pela tela de login do aplicativo, é disparado um gatilho que aciona uma função da Google Cloud Function para criação de um registro no banco de dados FireStore, como na Figura 6.

Figura 4 – Usuário criado via automação



Fonte: Elaborado pelo Autor

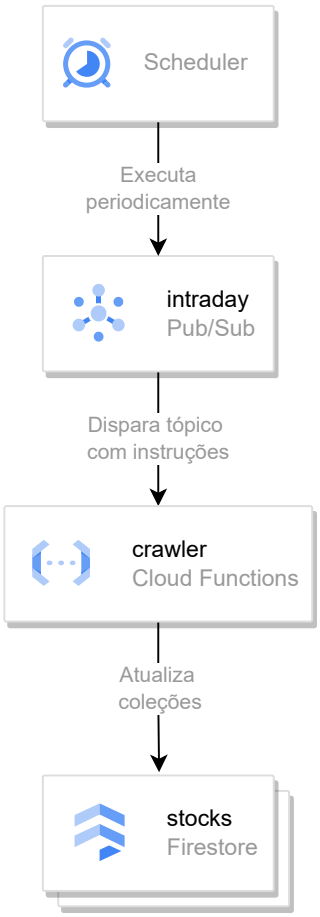
Assim que gerado este novo documento é disparado um evento que exporta os dados deste usuário ao BigQuery que por sua vez é consultado pelos relatórios da Looker Studio. Bem como a SDK do Firebase Firestore notifica o aplicativo Android com os dados atualizados do usuário.

3.4.2 Processo de atualização dos preços de ações

Outro processo que não é possível ser mantido do lado do cliente, é a obtenção dos preços dos ativos financeiros frequentemente, seja por cota de uso ou consistência das informações.

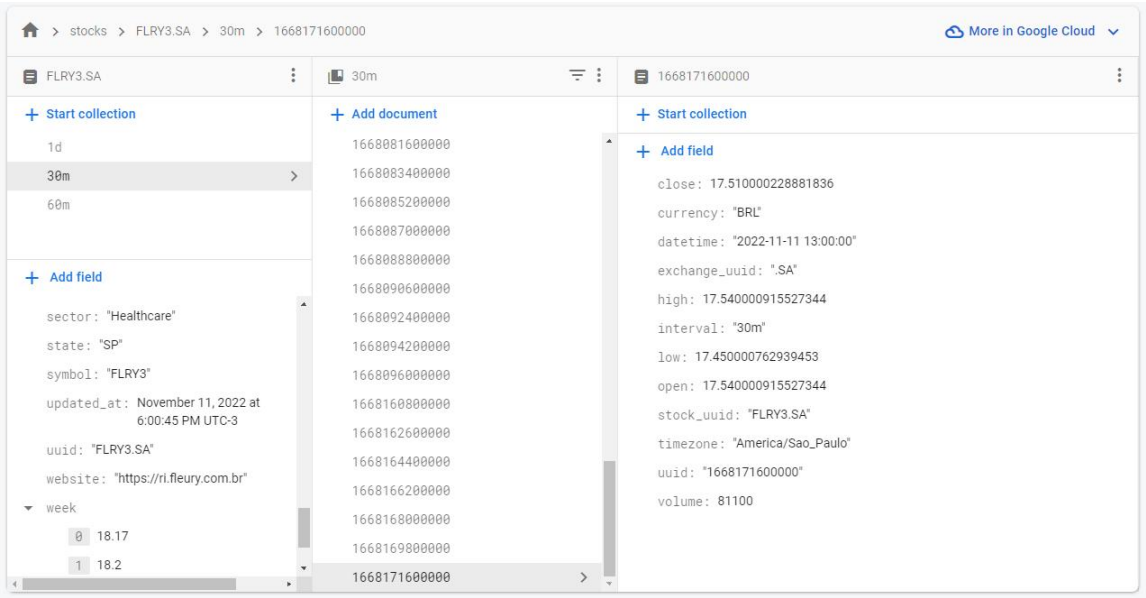
No fluxograma da figura 5 mostra como se dá a cadeira de chamadas que se inicia com um agendador (Scheduler) que dispara de tempos em tempos uma mensagem com os parâmetros `intervalo`, `período` e `quantidade de ações` dentro do serviço de mensageria Pub/Sub da GCP. Esta nova mensagem dispara uma CF que é responsável por realizar a captura dos preços destes ativos e então atualizar a base de dados do FireStore que por sua vez notifica o aplicativo.

Figura 5 – Fluxograma de atualização de preços de ativos



Fonte: Elaborado pelo Autor

Figura 6 – Preço de ações atualizados via automação



Fonte: Elaborado pelo Autor

3.4.3 Arquitetura do aplicativo

A arquitetura de app recomendada pela Google se subdivide camada de interface do usuário, responsável por mostrar os dados do aplicativo na tela. Camada de dados, responsável por conter lógica de negócios do app e expor os dados do aplicativo. Sendo possível adicionar uma camada extra conhecida como camada de domínio para simplificar e reutilizar as interações entre a IU e as camadas de dados.

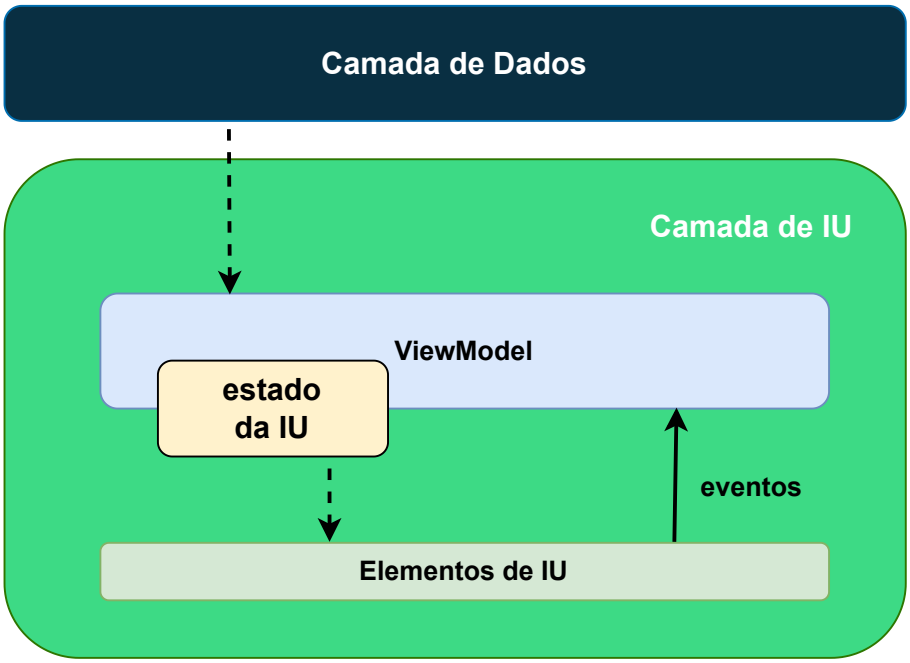
3.4.3.1 Camada de Apresentação

Conhecida como camada de interface do usuário ou camada de apresentação, seu objetivo é lidar com as interações do usuário e exibir os dados em tela. Esta camada é composta por dois items: Elementos de IU e *state holder*. Sendo comunicados via fluxo de dados unidirecional (UDF, na sigla em inglês), um padrão de arquitetura que ajuda a aplicar a separação saudável de responsabilidades.

State holders são classes responsáveis pela produção do estado da IU e que contêm a lógica necessária para exibição. A implementação típica é uma instância de uma classe `ViewModel`. Este tipo é a implementação recomendada para o gerenciamento do estado da IU no nível da tela com acesso à camada de dados. Além disso, ela resiste às mudanças de configuração automaticamente.

As classes `ViewModel` definem a lógica a ser aplicada aos eventos no app e produzem o estado atualizado como resultado. A interação entre IU e seu `ViewModel` pode ser entendida como uma entrada do evento (*event input*) e a consequente saída de estado (*state output*), a relação pode ser representada conforme mostrado no diagrama a seguir:

Figura 7 – Diagrama de como o UDF funciona na arquitetura do app



Fonte: Elaborado pelo Autor

UDF é um padrão em que o fluxo do estado desce e dos eventos sobe. As implicações desse padrão para a arquitetura do app são as seguintes:

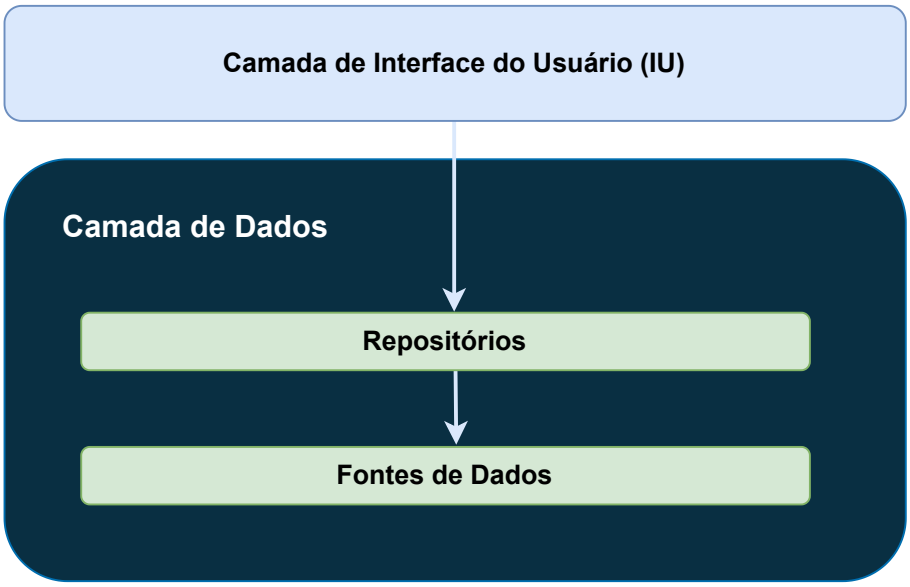
1. A ViewModel retém e expõe o estado a ser consumido pela IU. O estado da IU são os dados do app transformados pela ViewModel.
2. A IU notifica a ViewModel sobre eventos do usuário.
3. A ViewModel processa as ações do usuário e atualiza o estado.
4. O estado atualizado é retornado à IU para renderização.
5. O processo acima é repetido para qualquer evento que cause uma mutação de estado.

3.4.3.2 Camada de dados

A camada de dados de um app contém a lógica de negócios. A lógica de negócios é o que agrega valor ao app. Ela é composta por regras que determinam como o app cria, armazena e muda dados.

A camada de dados é composta por repositórios que podem conter de zero a muitas fontes de dados.

Figura 8 – Comunicação na camada de dados



Fonte: Elaborado pelo Autor

As classes de repositório são responsáveis pelas tarefas abaixo:

- Expor dados ao restante do app.
- Centralizar mudanças nos dados.
- Resolver conflitos entre várias fontes de dados.
- Abstrair fontes de dados do restante do app.
- Conter uma lógica de negócios.

A classe de fonte de dados é a ponte entre o aplicativo e o sistema para operações de dados e deve ser responsável por trabalhar com apenas uma origem, no nosso caso, o Firebase Firestore.

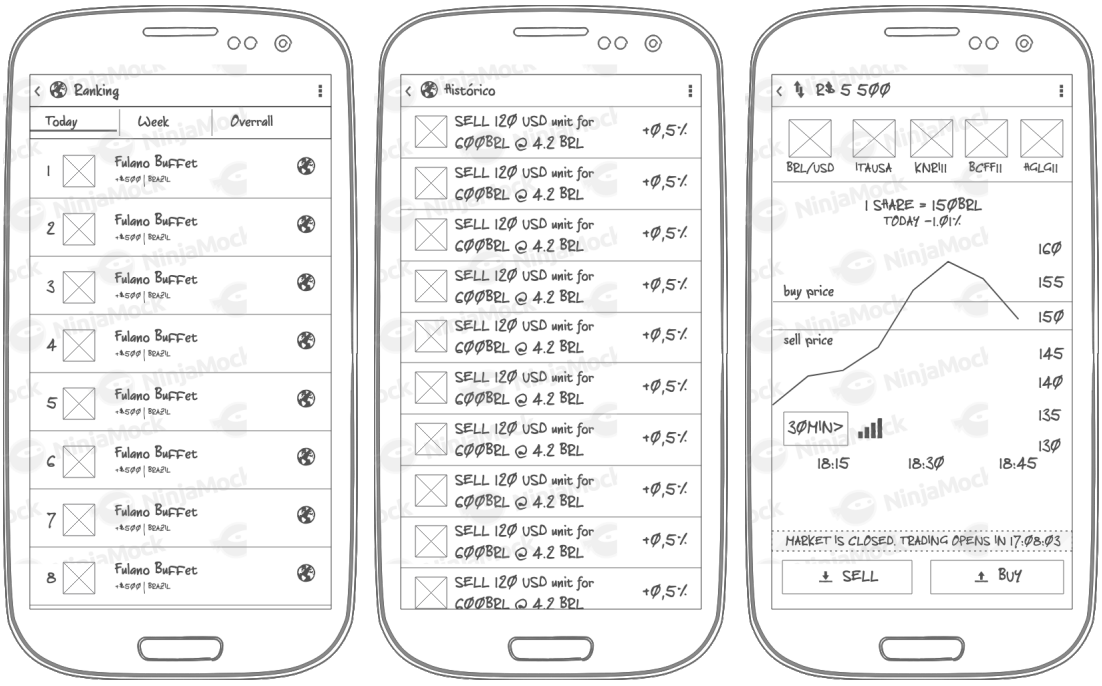
3.5 INTERFACE

Com o foco no desenvolvimento rápido e econômico, foi realizada a prototipação de baixa fidelidade para validar fluxos e telas na fase inicial do projeto, utilizando a ferramenta online e gratuita NinjaMock[1] que fornece componentes simples para o esboço das característica básicas do sistema, como na figura 9.

[1] https://ninjamock.com/

Figura 9 – Protótipo de baixa fidelidade

(a) Tela de autenticação (b) Tela principal (c) Tela de perfil



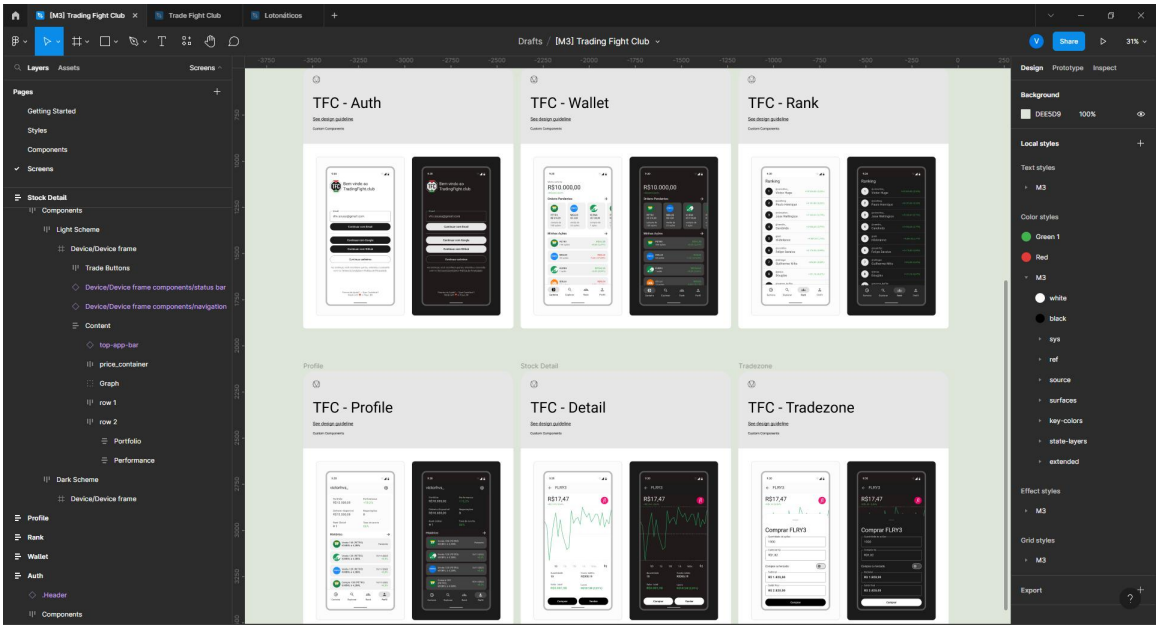
(d) Tela de ranqueamento (e) Tela de histórico (f) Tela de negociações

Fonte: Elaborado pelo autor no NinjaMock

Após a modelagem do banco de dados e documentação dos casos de uso, se fez possível a elaboração de um protótipo de alta fidelidade, mirando na produção fiel dos componentes, trazendo a representação da versão final do produto, apresentando estilo, fluxo e demais aspectos

da interface disponibilizada ao usuário. Nesta etapa, a ferramenta online Figma[2] foi utilizada seguindo o modelo guia gratuito disponível na documentação oficial do Material Design 3[3]. Como na figura 10.

Figura 10 – Protótipo de alta fidelidade



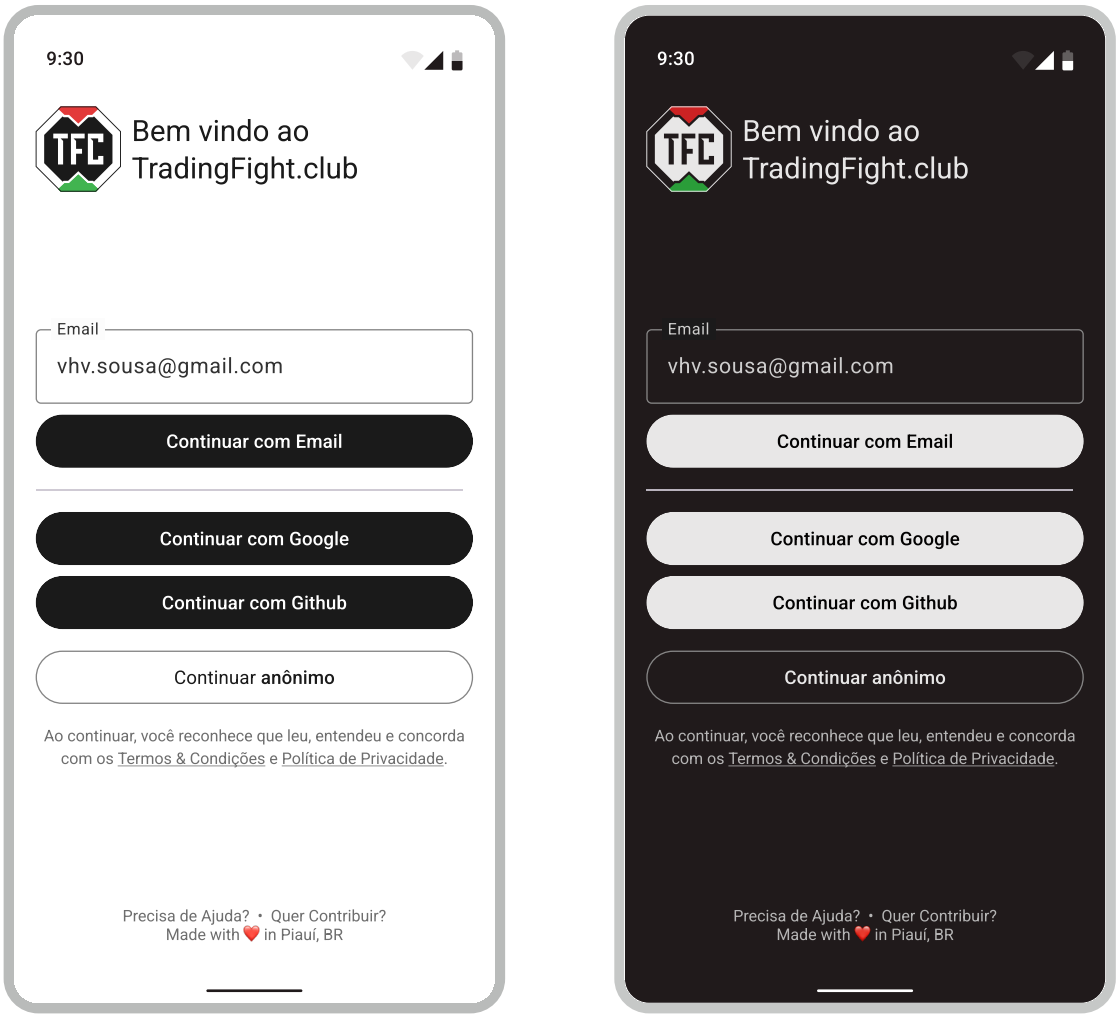
Fonte: Elaborado pelo Autor no Figma

3.5.1 Tela de Autenticação

Tela responsável pelo cadastro e autenticação do usuário, é exibida sempre que uma verificação de identificação do usuário é realizada e o mesmo não esteja autenticado. Sendo encaminhado para este fluxo na qual, independente da escola, um registro de usuário é criado, caso não o tenha.

[2] https://www.figma.com/
[3] https://www.figma.com/community/file/1035203688168086460

Figura 11 – Tela de Autenticação



(a) Tema Branco (b) Tema Noturno

Fonte: Elaborado pelo Autor

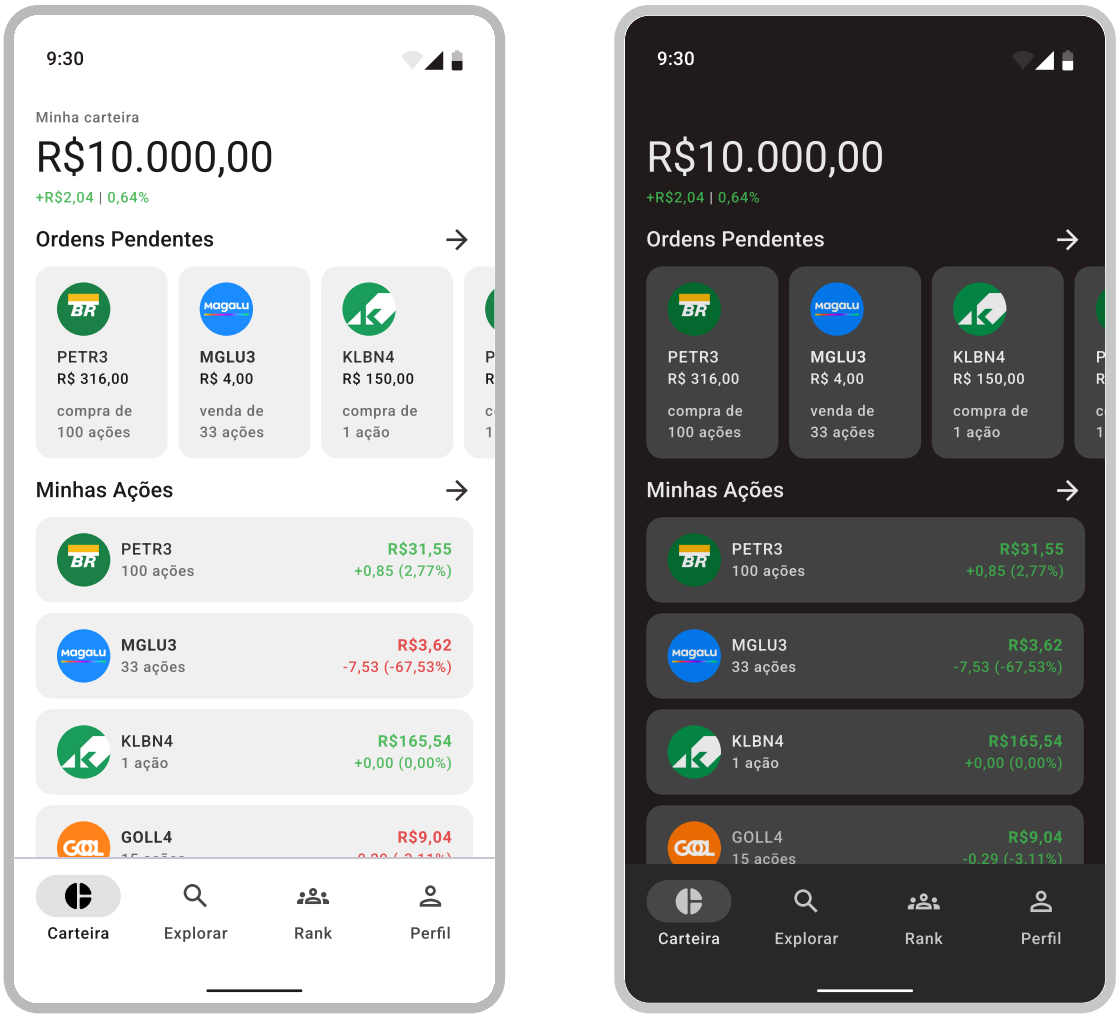
3.5.2 Tela Principal

A primeira tela que um usuário autenticado tem acesso é a com sua carteira ou tela principal. Nela são exibidos os seguintes elementos:

- Somatório das ações e dinheiro líquido;
- Valor percentual da variação do portfólio nas últimas 24 horas;
- Valor absoluto da variação do portfólio nas últimas 24 horas;
- Listagem de 5 ordens com status pendentes ordenadas por data;
- Listagem de 5 ações ordenadas por quantidade;

Nessa mesma tela, também podemos ver um menu em *tabs* na parte inferior do aplicativo que auxilia na navegação entre as telas: principal, explorar, *rank* (ranqueamento) e perfil.

Figura 12 – Tela Principal



(a) Tema Branco (b) Tema Noturno

Fonte: Elaborado pelo Autor

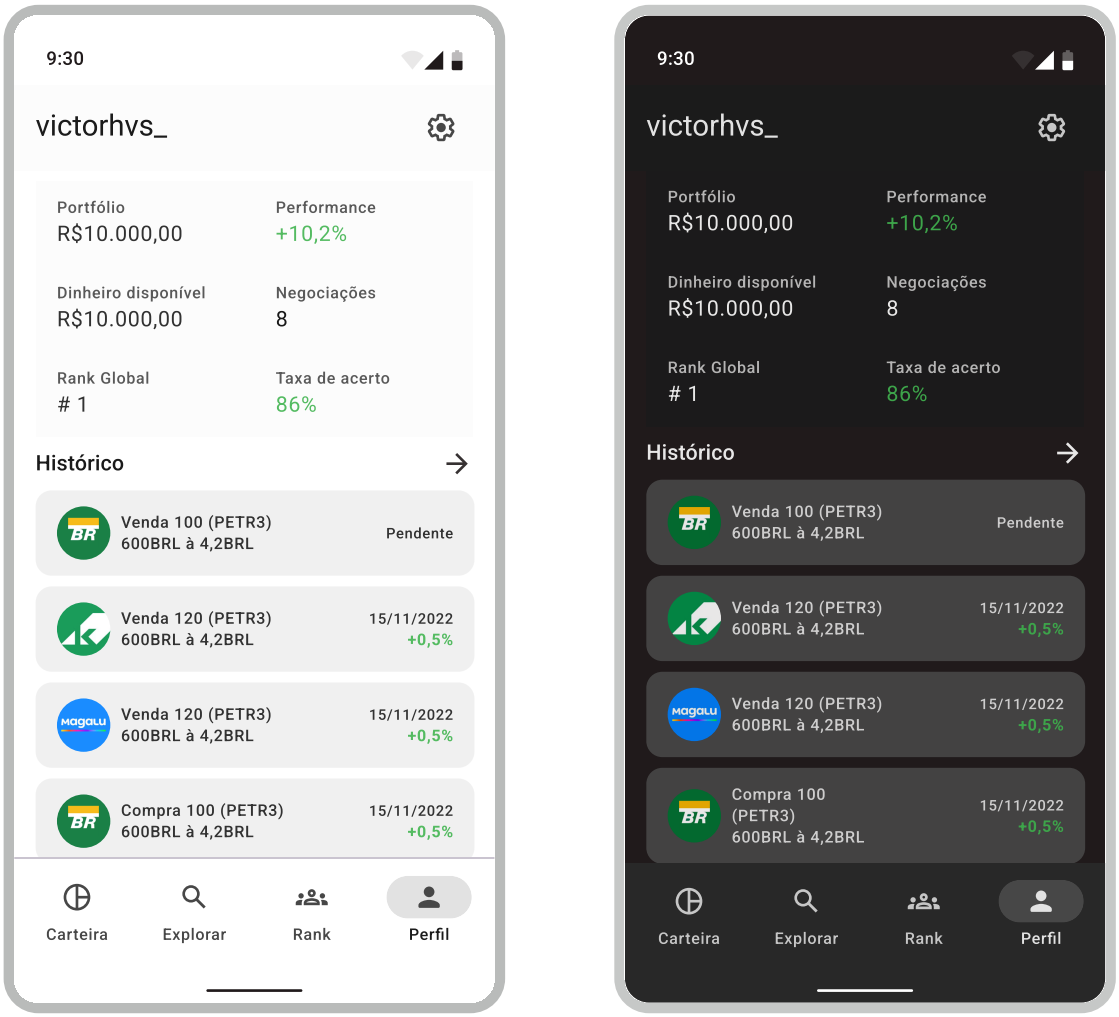
3.5.3 Tela de Perfil

Na tela de perfil o usuário visualiza seu histórico de operações, bem como indicadores, sendo eles:

- Portfólio: Somatório do valor líquido com o valor dos ativos comprados;
- Dinheiro disponível: Valor total disponível para operação de compra;
- Performance: Porcentagem comparativa do saldo atual com o saldo inicial;
- Negociações: Total de negociações com status concluído;
- *Rank Global*: Posição referente a performance realizada e comparada com todos os usuários da plataforma;
- Taxa de acerto: Porcentagem de ações com saldo positivo;

Abaixo dos indicadores, o usuário tem acesso a todas as negociações já realizadas ordenadas de forma decrescente por data.

Figura 13 – Perfil do Usuário



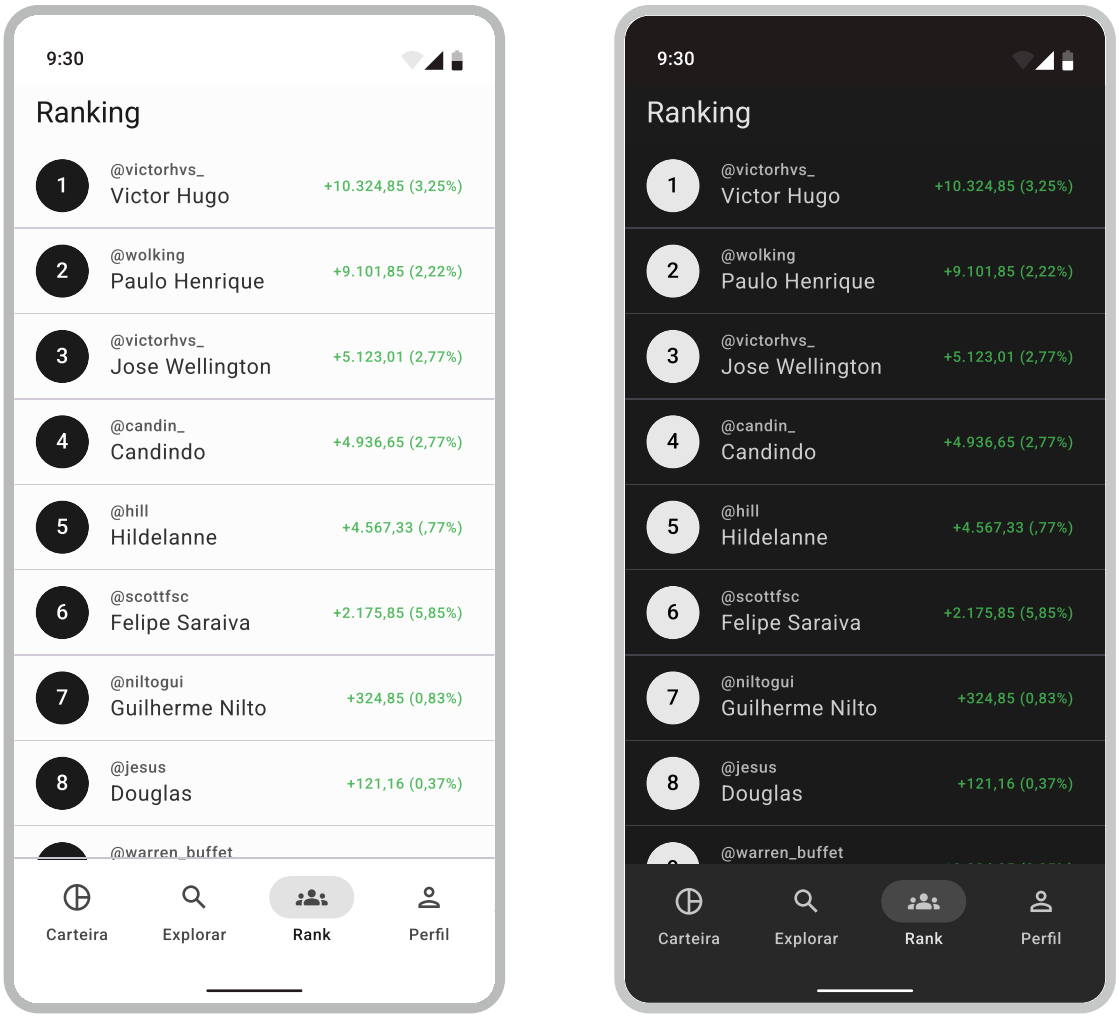
(a) Tema Branco (b) Tema Noturno

Fonte: Elaborado pelo Autor

3.5.4 Tela de Ranqueamento

A tela de ranqueamento ou *Ranking* pode ser acessada através da navegação principal do aplicativo e lista os usuários de forma ordenada pelo valor total do portfólio, nela contém o nome, usuário, posição, valor absoluto do portfólio e a variação percentual.

Figura 14 – Tela de Ranqueamento



(a) Tema Branco (b) Tema Noturno

Fonte: Elaborado pelo Autor

3.5.5 Tela de Detalhes do Ativo Financeiro

Na tela de detalhes concentra o histórico de preço que pode ser acompanhado em tempo real por um gráfico de linha ou velas em intervalos de 1 dia, 1 semana, 1 mês ou valor máximo registrado na base de dados.

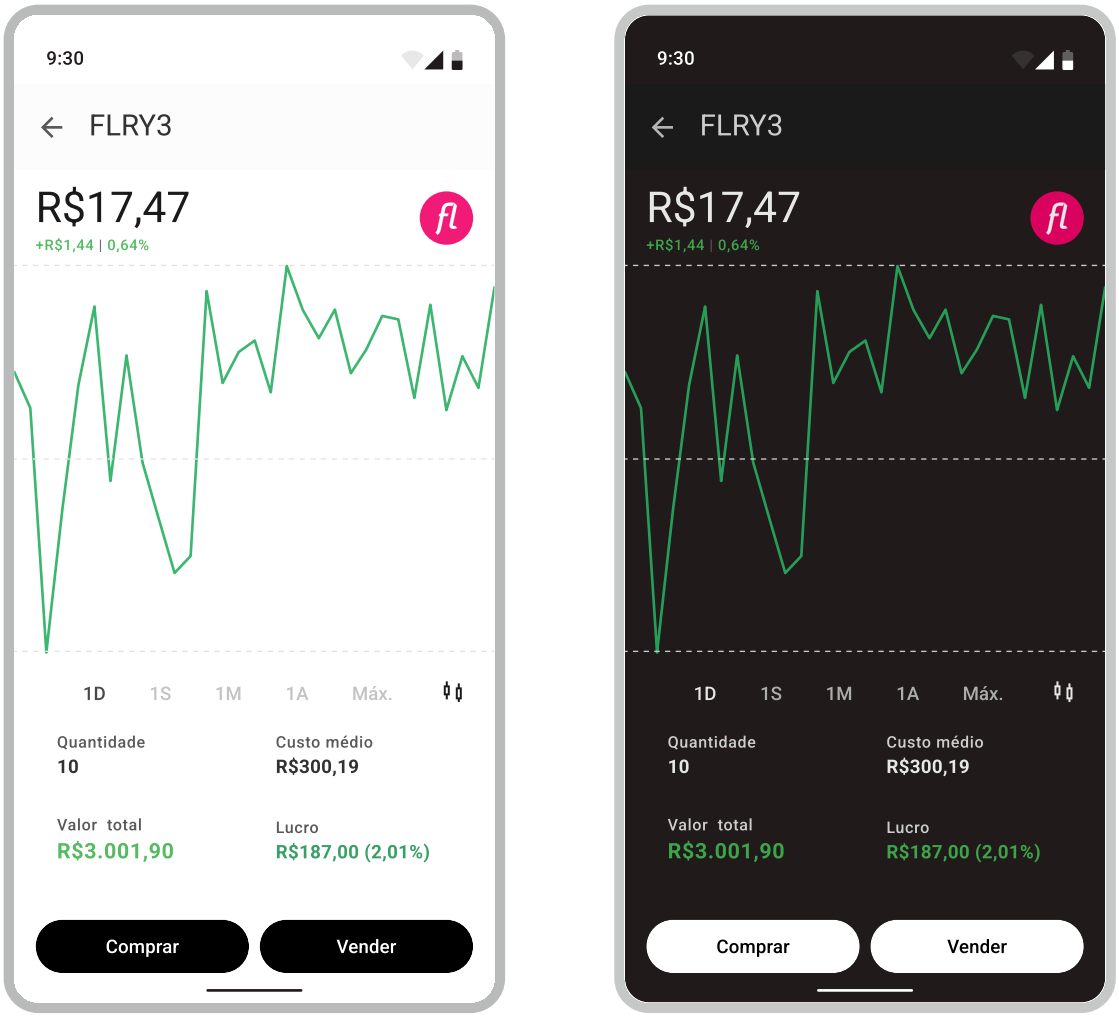
Em destaque na parte superior possui informações como Código da ação na bolsa de valores, cotação atual e valorização absoluta e percentual em comparação com o dia anterior.

No meio da tela possui um gráfico com o histórico de preço do ativo que é atualizado sempre que o banco de dados registra uma atualização, bem como a opção de alterar o intervalo das informações no gráfico e dois modos de visualização: em linha ou em barras.

Na parte inferior possui informações referente a posse do usuário. Quantidade total de ações, custo médio, somatório e lucro. Abaixo possui os botões de compra e venda. Como

podemos ver na figura 15.

Figura 15 – Tela de Detalhes do ativo



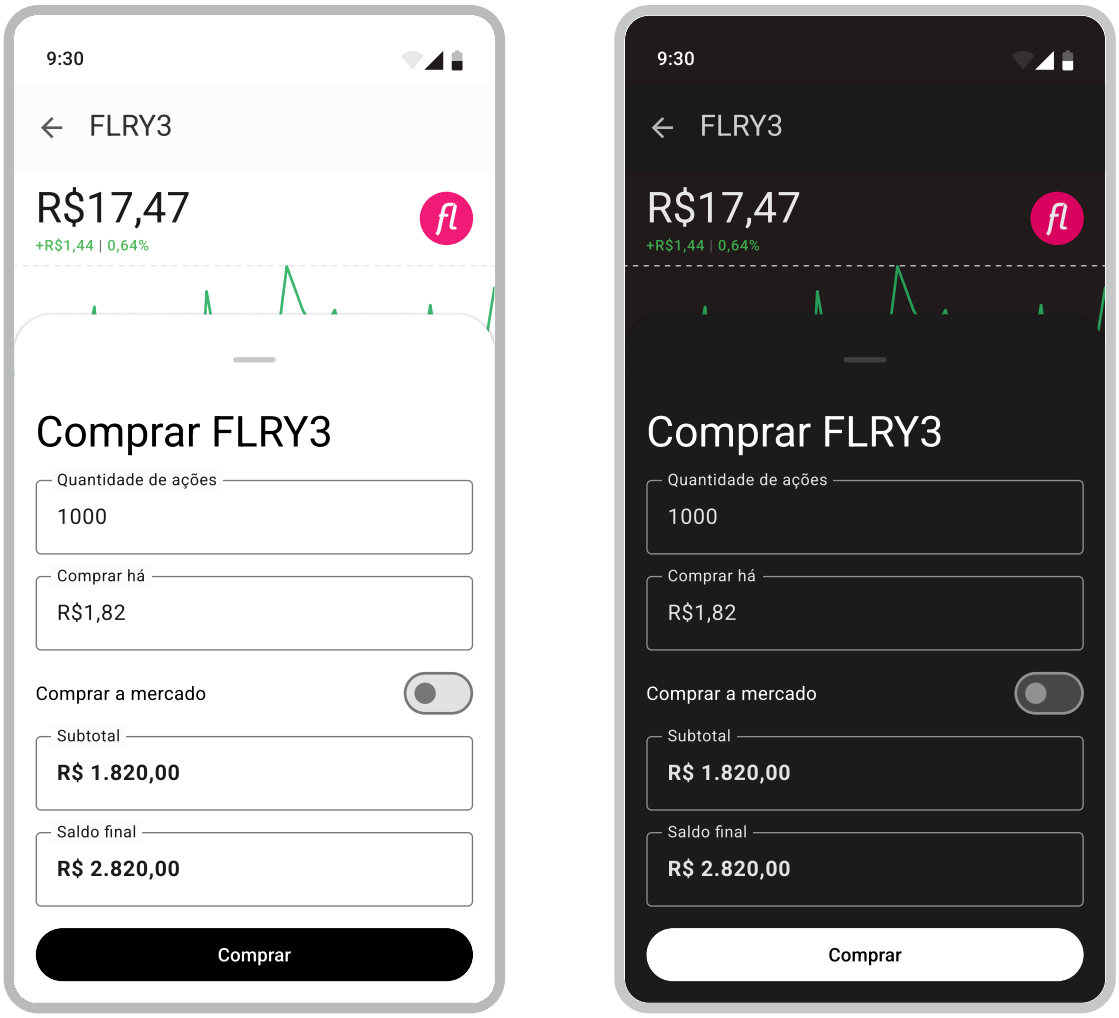
(a) Tema Branco (b) Tema Noturno

Fonte: Elaborado pelo Autor

3.5.6 Tela de Negociações

Ao clicar no botão de comprar ou vender, na tela de detalhes do ativo, é exibido um formulário com os campos: Quantidade, preço, comprar/vender a mercado, subtotal e saldo final. Tendo validações de quantidade para habilitar o botão de executar a ordem, seja de compra ou venda. Como na figura 16.

Figura 16 – Tela de Negociações



(a) Tema Branco (b) Tema Noturno

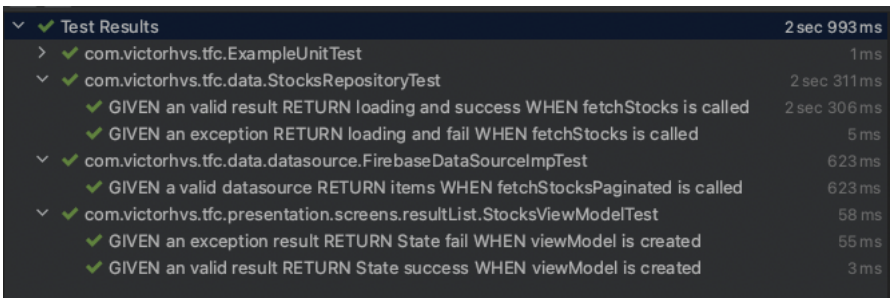
Fonte: Elaborado pelo Autor

4 SOFTWARE

4.1 TESTES

No desenvolvimento deste projeto foram realizados testes unitários nas classes que proveem as informações para a camada de interface, sendo as classes de maior destaque: `DataSources`, `Repositories` e `ViewModels`, buscando sempre testar os mais diversos cenários, como podemos ver na figura 17.

Figura 17 – Testes unitários

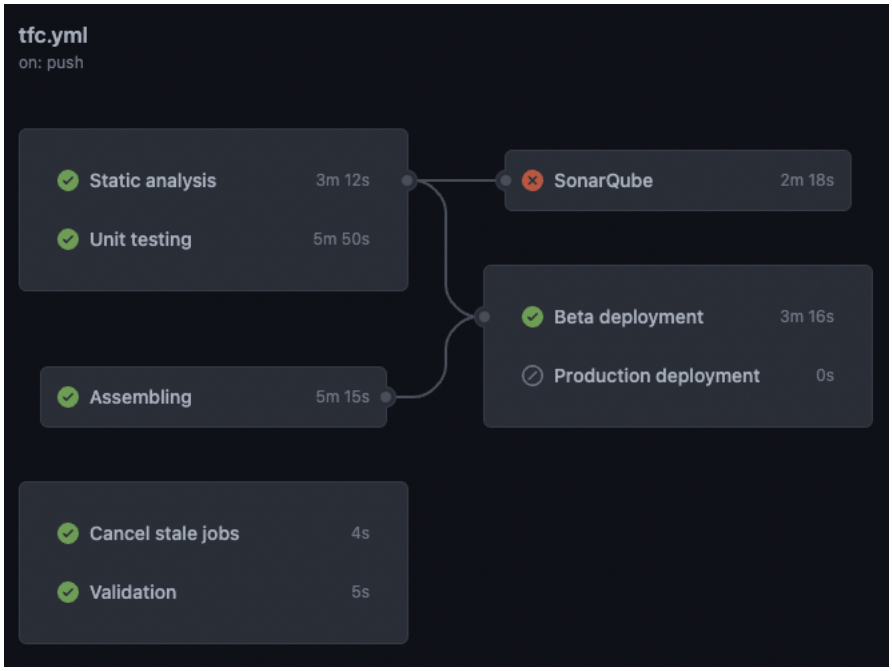


Fonte: Elaborado pelo Autor

Todo o processo de integração contínua do código se dá de forma automática na plataforma do *Github Actions* sempre que é aberto um pedido de mesclagem de uma ramificação temporária com a principal (`main`), através de um *pull request*. Este processo de integração contínua, que pode ser vista na figura 18, é divida em 4 grandes grupos de tarefas:

1. ***Assembling***: Geração de uma versão de depuração e relatórios e métricas, para garantir que o aplicativo é construído com sucesso;
2. ***Static analysis***: *Scripts* de análise estática do código que levanta débitos técnicos e gera relatórios;
3. ***Unit testing***: Executa testes unitários, testes instrumentados, produz relatórios de execução e cobertura;
4. ***SonarQube***: Agrega todos os relatórios produzidos nas etapas anteriores e envia ao serviço SonarQube.

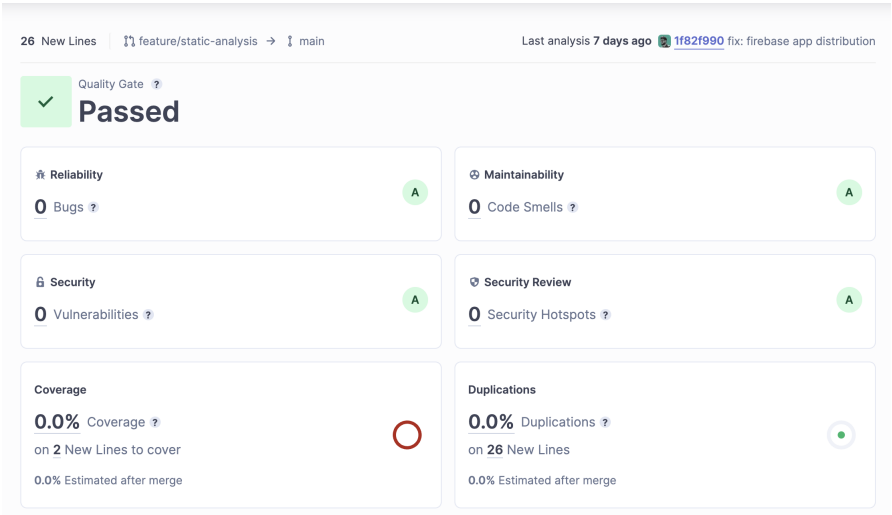
Figura 18 – Pipeline



Fonte: Elaborado pelo Autor

Juntamente com o relatório de cobertura produzido pelos testes unitários, é gerado um relatório da análise estática do código utilizando a ferramenta Detekt e enviado ao serviço de acompanhamento de métricas de qualidade Sonar, que testa a evolução do código em aspectos como segurança, duplicação de código, cobertura de testes e possíveis código que venha a ter falhas de execução, como podemos ver na figura 19.

Figura 19 – Resultado Sonar



Fonte: Elaborado pelo Autor

4.2 IMPLANTAÇÃO

O processo de implantação do software se dá de forma automática e contínua sempre que um novo código é aprovado pelo processo de integração contínua e enviado a *branch main*, executando assim a tarefa *Beta Deployment* que é responsável por gerar uma versão depurável de teste e envia-la ao *Firebase App Distribution*, que por sua vez notifica todos os usuários cadastrados no grupo de teste por e-mail.

Assim como sempre que uma nova *tag* é criada ou recebe atualizações, a *pipeline* é executada juntamente com a tarefa *Production deployment*, responsável por gerar uma versão oficial de produção e enviar para loja de aplicativos da Google.

O código-fonte do aplicativo android está disponível no endereço <https://github.com/VictorHVS/trading-fight-club-mobile>, o código-fonte das *cloud functions* estão disponíveis em <https://github.com/VictorHVS/tfc-functions> e o aplicativo android pode ser baixado loja Google Play no link <https://play.google.com/store/apps/details?id=com.victorhvs.tfc>.

5 CONSIDERAÇÕES FINAIS

Este trabalho apresentou a ferramenta *TradingFight.club*, uma plataforma que simula a experiência de uma corretora de ativos financeiros, com preços e histórico real de mais de 1.000 ações da bolsa de valores brasileira, totalmente gratuita e de código aberto. Além disso, o sistema desenvolvido utilizou de boas práticas recomendadas para o desenvolvimento nativo de aplicativos Android, com uso de técnicas e ferramentas de ponta que possibilitam o escalonamento para atender a demanda de milhares de usuários a um custo baixo.

Considera-se que a solução alcança seu objetivo de prover uma experiência similar a uma corretora de ativos, de fácil uso, com performance e escalável por ser construída com uma arquitetura moderna que permite a consulta e visualização em tempo real da variação dos preços das ações, bem como compra e venda e visualizações de progresso financeiro.

Por fim, o trabalho também contribui com a propagação do conhecimento em modelagem, utilização de ferramentas avançadas e boas práticas para implementação de um aplicativo Android nativo se utilizando de diversas ferramentas para autenticação, persistência, qualidade, geração de relatórios, integração e entrega contínua.

Em suas próximas versões, é pretendido diminuir o tempo de atualização dos preços, incluir uma variedade maior de indicadores a respeito dos ativos, bem como incluir novos relatórios a respeito do portfólio do usuário e o aprimoramento das funcionalidades existentes.

O código-fonte do aplicativo android está disponível no endereço <https://github.com/VictorHVS/trading-fight-club-mobile>, o código-fonte das *cloud functions* estão disponíveis em <https://github.com/VictorHVS/tfc-functions> e o aplicativo android pode ser baixado loja Google Play no link <https://play.google.com/store/apps/details?id=com.victorhvs.tfc>.

REFERÊNCIAS

1 BRASIL, B. C. do. *Dados diários da Selic*. 2022. Banco Central. Disponível em: <https://www.bcb.gov.br/acessoinformacao/legado?url=https://www.bcb.gov.br/htms/selic/selicdiarios.asp>. Acesso em: 05 dez. 2022. Citado na página 12.

2 BRASIL, B. C. do. *Remuneração dos Depósitos de Poupança*. 2022. Banco Central. Disponível em: <https://www.bcb.gov.br/estatisticas/remuneradepositospoupanca>. Acesso em: 05 dez. 2022. Citado na página 12.

3 BRASIL, B. C. do. *Relatório Focus*. 2022. Banco Central. Disponível em: <https://www.bcb.gov.br/publicacoes/focus/14022020>. Acesso em: 05 dez. 2022. Citado na página 12.

4 BRASIL, B. C. do. *Relatório de poupança*. 2022. Banco Central. Disponível em: <https://www.bcb.gov.br/estatisticas/relatoriopoupanca>. Acesso em: 05 dez. 2022. Citado na página 12.

5 BALCãO, B. B. *Total de investidor pessoa física cresce 43% no primeiro semestre, mostra estudo da B3*. 2022. Brasil Bolsa Balcão. Disponível em: <https://www.b3.com.br/pt_br/noticias/investidores.htm>. Acesso em: 05 dez. 2022. Citado na página 12.

6 ANBIMA. *Raio X do investidor brasileiro 5ª edição*. 2022. ANBIMA. Disponível em: <https://www.anbima.com.br/pt_br/especial/raio-x-do-investidor-2022.htm>. Acesso em: 05 dez. 2022. Citado na página 12.

ANEXO A – ESPECIFICAÇÃO DE CASOS DE USO

ESPECIFICAÇÃO DO CASO DE USO 01 — AUTENTICAR

Objetivo

Caso de uso responsável pelo cadastro e início de sessão do usuário não identificado. Através de um provedor de conta disponível para ter um acesso identificado no aplicativo.

Identificação dos Atores

- **Usuário**: Qualquer pessoa não autenticada que utiliza o aplicativo;

Identificação dos Fluxos

- **Autenticar com rede social**: Este fluxo descreve como se dá a autenticação do usuário usando sua conta já existente de uma rede social.

Detalhamento dos Fluxos

- **Autenticar com rede social**: Este fluxo especifica a ação de iniciar uma sessão no aplicativo utilizando credenciais de sua conta já existente em algum provedor disponível para ter um acesso identificado no aplicativo.
  - **Atores**: Usuário;
  - **Pré-Condições**: Não estar autenticado;
  - **Pós-Condições**: Sessão ativa, conta criada (caso não exista) e redirecionado ao caso de uso 2 (Tela principal);
  - **Requisitos Funcionais**: O sistema deve prover uma interface que faz chamada ao serviço do provedor selecionado responsável por autenticação.

  Fluxo Básico

  1. O usuário é encaminhado para tela de autenticação, onde contêm uma listagem de opções para acesso ao aplicativo;
  2. O Usuário decide acessar o sistema utilizando sua conta já existente em uma rede social;
  3. O aplicativo executa e exibe em primeiro plano o serviço de autenticação do provedor selecionado, que por sua vez possui seu próprio fluxo de autenticação;

4. O serviço de autenticação do provedor retorna o status da solicitação juntamente com alguns dados básicos do usuário;
5. O aplicativo valida os dados;
6. Caso não tenha cadastro, o aplicativo registra todas as informações no banco de dados do Firestore;
7. O Usuário é direcionado a tela principal do aplicativo;
8. O caso de uso se encerra.

Fluxo Alternativo A
Caso exista algum erro de autenticação ou comunicação:

1. Uma mensagem de erro é exibida ao Usuário informando da falha;
2. O fluxo retorna ao passo 1.

ESPECIFICAÇÃO DO CASO DE USO 02 — EXPLORAR

Objetivo

Lista e acessar detalhes das ações disponíveis na plataforma.

Identificação dos Atores

- **Usuário**: Qualquer pessoa cadastrada e autenticada;

Detalhamento do Fluxo

- **Explorar**: Este fluxo especifica a ação listar e visualizar os detalhes como preço atual e histórico de preços de um determinado ativo.
  - **Atores**: Usuário;
  - **Pré-Condições**: O usuário deve estar autenticado na aplicação;

  Fluxo Básico

  1. O usuário autenticado navega para tela de listagem através do menu inferior principal;
  2. O aplicativo exibe uma listagem por ordem alfabética de todos os ativos disponíveis na plataforma, contento: logomarca, nome, código na B3, preço atual e valorização compara com as últimas 24h;
  3. Ao clicar em um item, é encaminhado para tela de detalhes do ativo selecionado;
  4. Nesta tela, informações adicionais são exibidas ao usuário, como: gráfico de linha ou velas com o histórico do preço do ativo, quantidade de ações que já foi comprado pelo mesmo, o custo médio, total e lucro.
  5. O caso de uso se encerra.

ESPECIFICAÇÃO DO CASO DE USO 03 — NEGOCIAR ATIVOS FINANCEIROS

Objetivo

Efetuar comprar e venda de ativos financeiros;

Identificação dos Atores

- **Usuário**: Qualquer pessoa cadastrada e autenticada;

Detalhamento do Fluxo

- **Negociar ativos financeiros**: Este fluxo especifica a ação de comprar ou vender um ativo financeiro.
  - **Atores**: Usuário;
  - **Pré-Condições**: O usuário deve estar autenticado na aplicação e possuir saldo disponível ou ações no portfólio para efetuar negociações;
  - **Pós-Condições**: Uma ordem de negociação é colocada na fila de execução;

  Fluxo Básico

  1. O usuário autenticado navega para tela de detalhe de uma ação;
  2. Na tela exibe a quantidade de ações que o usuário possui daquele ativo selecionado;
  3. No canto inferior são exibidos os botões de compra e venda;
  4. Ao clicar em um deles é exibido a tela com um formulário para negociação com os seguintes campos: Quantidade, preço, comprar/vender a mercado, subtotal, e saldo final;
  5. Dependendo da opção selecionada (compra ou venda) são feitas validações de quantidade para habilitar o botão de submeter o pedido;
  6. Ao submeter a ordem, o formulário se fecha;
  7. O caso de uso se encerra.

ESPECIFICAÇÃO DO CASO DE USO 04 — VER CARTEIRA

Objetivo

Visualizar os ativos comprados, seu preço e valorização atual, bem como ordens pendentes de execução e valor total do portfólio;

Identificação dos Atores

- **Usuário**: Qualquer pessoa cadastrada e autenticada;

Detalhamento do Fluxo

- **Ver Carteira**: Este fluxo especifica a ação de acompanhar os itens que compõe o portfólio do usuário: ações, ordens pendentes e valorização total.
  - **Atores**: Usuário;

- **Pré-Condições**: O usuário deve estar autenticado na aplicação;

Fluxo Básico

1. O usuário autenticado navega para tela principal ou carteira;
2. Na tela exibe uma listagem de todas as ações que o usuário possui, bem como ordens de compra ou venda pendentes e o somatório da carteira com a variação percentual e absoluta em relação ao dia anterior;
3. O caso de uso se encerra.

ESPECIFICAÇÃO DO CASO DE USO 05 — VER RANQUEAMENTO

Objetivo

Visualizar a listagem de usuários ordenado pela taxa de performance financeira;

Identificação dos Atores

- **Usuário**: Qualquer pessoa cadastrada e autenticada;

Detalhamento do Fluxo

- **Ver Ranqueamento**: Este fluxo especifica a ação de acompanhar o ranqueamento dos usuários.
  - **Atores**: Usuário;
  - **Pré-Condições**: O usuário deve estar autenticado na aplicação;

  Fluxo Básico

  1. O usuário autenticado navega para tela de ranqueamento através do menu inferior principal;
  2. Na tela exibe uma listagem ordenada de usuários com base no desempenho do portfólio, contendo posição, nome, *username*, somatório absoluto da carteira e variação percentual;
  3. O caso de uso se encerra.

ESPECIFICAÇÃO DO CASO DE USO 06 — VISUALIZAR PROGRESSO

Objetivo

Visualizar indicadores como: dinheiro disponível, valor total do patrimônio, valorização percentual, total de negociações, taxa de acertos e posição na classificação global. Além de acompanhar o histórico de suas negociações ao longo do tempo;

Identificação dos Atores

- **Usuário**: Qualquer pessoa cadastrada e autenticada;

Detalhamento do Fluxo

- **Visualizar Progresso**: Este fluxo especifica a ação de acompanhar de perto maiores detalhes de desempenho da conta do usuário.
  - **Atores**: Usuário;
  - **Pré-Condições**: O usuário deve estar autenticado na aplicação;

  Fluxo Básico

  1. O usuário autenticado navega para tela de perfil através do menu inferior principal;
  2. Na tela além de acompanhar o histórico de suas negociações ao longo do tempo, exibe indicadores de desempenho tais como como: dinheiro disponível, valor total do patrimônio, valorização percentual, total de negociações, taxa de acertos e posição na classificação global;
  3. O caso de uso se encerra.