

Documentação Arquitetónica da Online Trading Platform

No seguinte documento, “Online Trading Platform” estará abreviado como OTP.

22 de Outubro, 2017

Ana Rita Marques, A74218

Arquiteturas de Software
Engenharia de Sistemas de Software

Mestrado Integrado em Engenharia Informática
Universidade do Minho

Introdução e Objetivos	4
1. Requirements Overview	4
1. 1. Requisitos Funcionais	4
1. 2. Requisitos não funcionais	4
2. Stakeholders	5
2. Restrições	5
2. 1. Restrições Técnicas	5
2. 2. Restrições Organizacionais	6
2. 3. Convenções	6
3. Context	7
3. 1. Business Context	7
3. 2. Contexto de Implementação	7
4. Estratégia de Solução	8
5. Building Block View	9
5. 1. Whitebox OTP	9
5. 1. 1. Blackbox Sessao	9
5. 1. 2. Blackbox Trader	10
5. 1. 1. Blackbox Contrato	11
5. 1. 1. Blackbox Ativo	12
5. 1. 1. Blackbox Connection	12
6. Runtime View	13
6. 1. Diagrama de Classes	13
6. 2. Diagrama de Use Cases	13
6. 2. 1. Diagramas de Sequência de Use Case	14
6. 2. 1. 1. Ver contrato	15
6. 2. 1. 2. Criar conta	15
6. 3. Diagramas de Sequência de Sistema	17
6. 3. 1. Comprar contrato	17
6. 3. 2. Ver lista de ativos	17
7. Visão de Implementação	18
7. 1. Base de Dados	19
8. Conceitos	19
8. 1. Modelo de Domínio	19
8. 2. User Interface	20

8. 3. Internacionalização	20
9. Decisões do design	20
10. Cenários de qualidade	20
10. 1. Quality Tree	20
11. Riscos Técnicos	20
12. Mockups	21
12. 1. Página Inicial	21
12. 2. Iniciar sessão	21
12. 3. Menu	22
12. 4. Ativos	22
12. 5. Abrir Contrato	23
13. Glossário	23
14. Anexos	24
Schema da Base de Dados	24
2. Dados já na DB	27

1. Introdução e Objetivos

OTP é um projeto que permite **investidores** e **traders** abrir, fechar e gerir posições no mercado financeiro, podendo envolver a compra e venda de ativos financeiros. Os ativos financeiros poderão ser:

- Ações;
- *Commodities* (ouro, prata, petróleo);

Esta plataforma será oferecida gratuitamente aos seus utilizadores.

1. 1. Requirements Overview

O principal objetivo da *OTP* é permitir a gestão de posições no mercado financeiro.

Além disso, há um conjunto de requisitos que a plataforma deverá satisfazer.

1. 1. 1. Requisitos Funcionais

Os utilizadores da plataforma deverão poder:

- Visualizar a lista de ativos a serem negociados via CFDs, assim como os seus valores;
- Abrir conta de *trader*/investidor com um plafond inicial para negociação/investimento;
- Abrir posições (CFDs) sobre os ativos disponíveis.
- Monitorizar em 'tempo real' o seu portfólio de CFDs e para cada ativo visualizar o valor atual do(s) ativo(s) adquirido(s).

A plataforma deverá manter um ilimitado número de ativos, assim como permitir um ilimitado número de *traders*/investidores

1. 1. 2. Requisitos não funcionais

Nr.	Qualidade	Motivação
1	Precisão	O sistema guarda e mostra informações e cálculos precisos.
2	Facilidade de	Os utilizadores devem conseguir utilizar a

	aprendizagem	aplicação com algumas ajudas por parte da equipa de desenvolvimento
3	Modificabilidade	Quando necessárias alterações no código, é fácil perceber em que zona do código devem ser feitas.

1. 2. Stakeholders

A seguinte tabela contém os principais intervenientes desta aplicação.

Papel	Objetivos
<i>Trader/Investidor</i>	Procuram uma plataforma para a gestão de ativos financeiros.
<i>Developer/Software Architect</i>	Desenvolve a plataforma de <i>trading</i> .

2. Restrições

2. 1. Restrições Técnicas

	Restrição	Background e/ou motivação
TC1	Implementação em Python	Desenvolvimento sob versão 3.4.0 do Python.
TC2	Software de terceiros disponível gratuitamente sob uma licença de código aberto compatível	O software externo adicionado à solução deve ser gratuito e disponível gratuitamente. O desenvolvedor ou arquiteto interessado deve poder verificar as fontes, compilar e executar o aplicativo sem problemas para compilar ou instalar dependências.

TC3	Desenvolvimento independentemente do OS	A aplicação deve ser compilável nos sistemas operativos Mac OS X, Linux e Windows.
-----	---	--

2. 2. Restrições Organizacionais

	Restrição	Background e/ou motivação
OC1	Equipa	Ana Rita Marques
OC2	Horário	Desenvolvimento começou no início de Outubro e terminou a 20 de Outubro.
OC3	Modelo de processo	arc42 é usado para documentar a arquitetura.
OC4	Ferramentas de desenvolvimento	Criação do código fonte Python no PyCharm.
OC6	Publicado sob licença Open Source	O código e a documentação devem ser publicados como Open Source.

2. 3. Convenções

	Convenções	Background e/ou motivação
C1	Documentação de arquitetura	Estrutura baseada na versão inglesa arc42-Template, version 6.5.
C2	Convenções de programação	O projeto usa <u>convenções de programação em Python</u> .
C3	Linguagem	Português.

3. Context

3. 1. Business Context

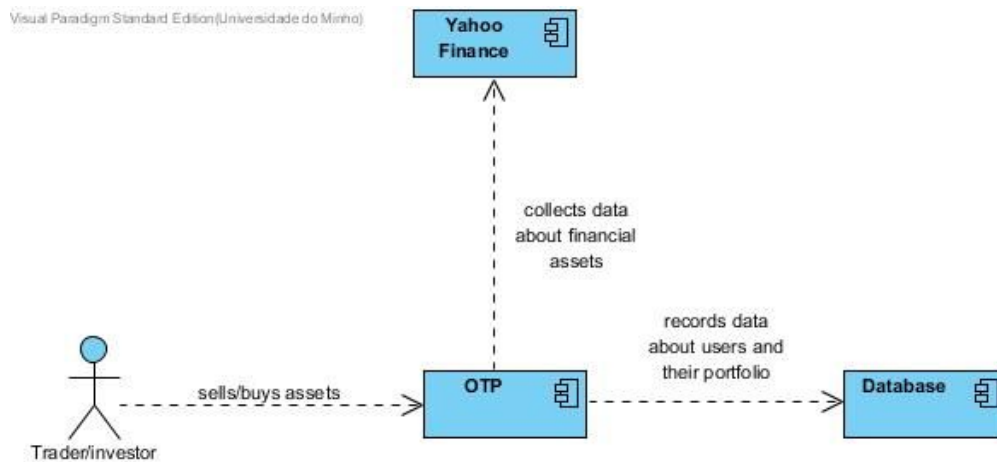


Fig. 1. Business Context Diagram

Component	Description
Trader/investidor	O trader/investidor usa a OTP para vender, comprar e gerir ativos financeiros, mantidos no seu portfólio.
Yahoo Finance	Yahoo Finance é a data source da plataforma. Fornece os valores dos ativos.
Database	As informações sobre os ativos, utilizadores e seus portefólios estarão armazenados numa base de dados.

3. 2. Contexto de Implementação

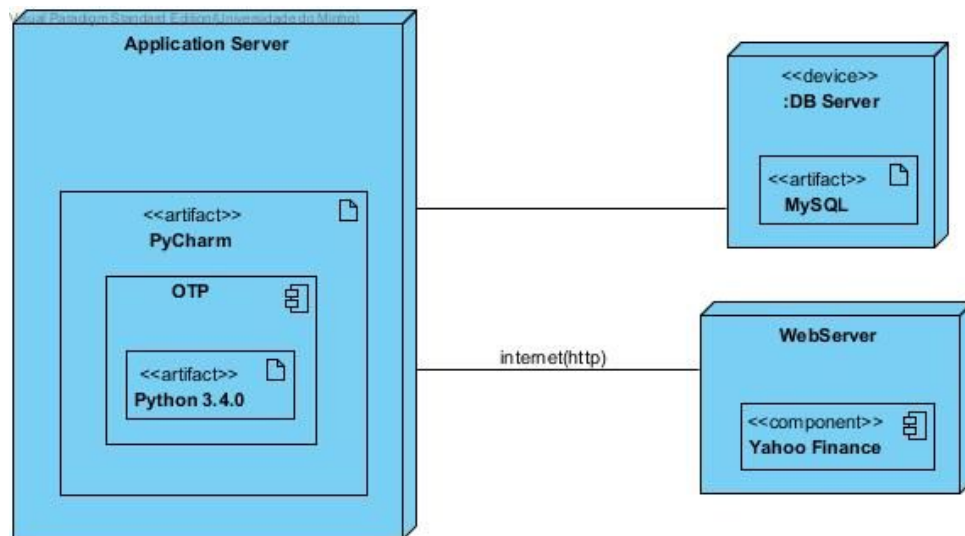


Fig. 2. Deployment Context Diagram

- **Application Server:** Local onde o programa irá ser utilizado. O utilizador deverá ter a ferramenta PyCharm, onde poderá executar a aplicação. Deverá ter também a versão Python 3.4.0 instalada, e fazer import do MySQL Connector;
- **Web Server:** A aplicação utiliza dados dos ativos, fornecidos pelo Yahoo Finance;
- **DB Server:** Os dados dos ativos, contratos e utilizadores estarão armazenados numa base de dados MySQL.

4. Estratégia de Solução

Numa plataforma online de trading é fundamental ter os valores dos ativos financeiros corretos. Para assegurar isso, a aplicação lê, constantemente, ficheiros .csv com valores (e outras informações relevantes) dos ativos e atualizados na base de dados. Quando estas alterações são feitas, os contratos em aberto de todos os utilizadores são verificados e atualizados. Deste modo, quando um utilizador faz login, os seus contratos estão atualizados.

5. Building Block View

5. 1. Whitebox OTP

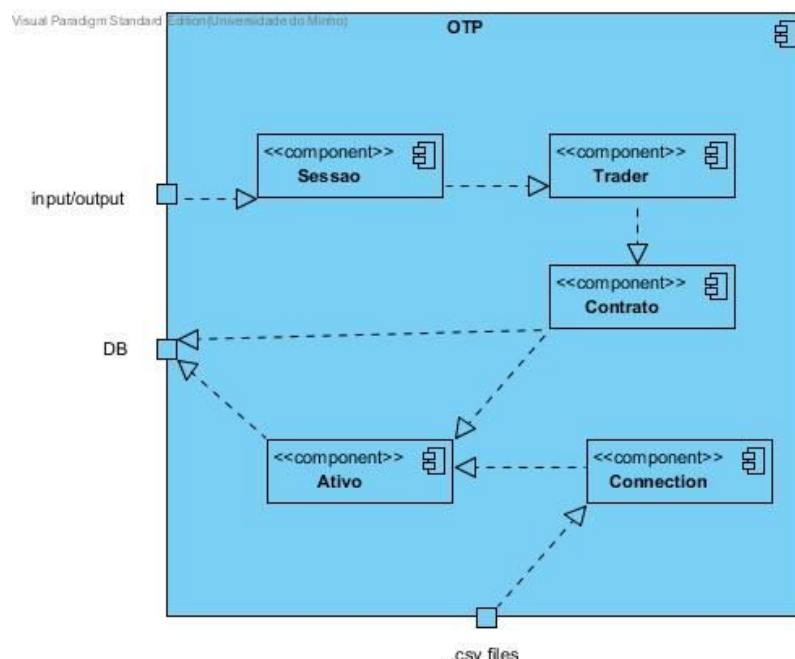


Fig. 3. Whitebox OTP

Classe	Descrição
Sessao	Realiza comunicação entre cliente e a plataforma.
Trader	Responsável pelos métodos relativos a funcionalidades.
Contrato	Responsável pelos métodos relativos aos contratos.
Ativo	Responsável pelos métodos relativos a ativos financeiros.
Connection	Atualiza os valores dos ativos e dos contratos.

5. 1. 1. Blackbox Sessao

Esta classe é a responsável por iniciar a sessão do utilizador e fazer a comunicação entre utilizador e a plataforma.

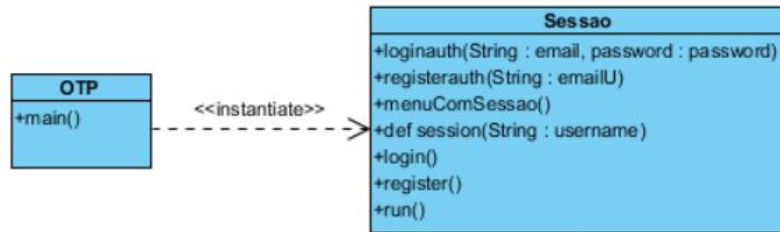


Fig. 4. Blackox Sessao

Método	Descrição
run	Fornece o menu inicial ao utilizador, com as opções de registar ou iniciar sessão.
login	Responsável pelo login do utilizador.
register	Responsável pelo registo do utilizador.
loginauth	Verifica se o email e a password do utilizador existem e correspondem existem na base de dados.
registerauth	Verifica se o email é válido, isto é, ainda não está na base de dados.
session	Guarda os dados da pessoa que iniciou sessão.
menuComSessao	Mostra as funcionalidades que o utilizador logado pode executar.

5. 1. 2. Blackbox Trader

Esta classe é responsável pelas funções que o trader pode realizar na aplicação. É instanciada pela classe Sessao.

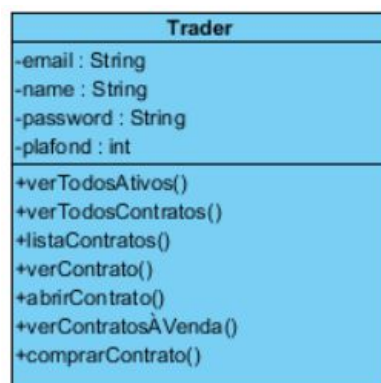


Fig. 5. Blackbox Trader

5. 1. 1. Blackbox Contrato

Esta classe é responsável por todos os métodos relacionadas com contratos. é instanciada pela classe Trader.

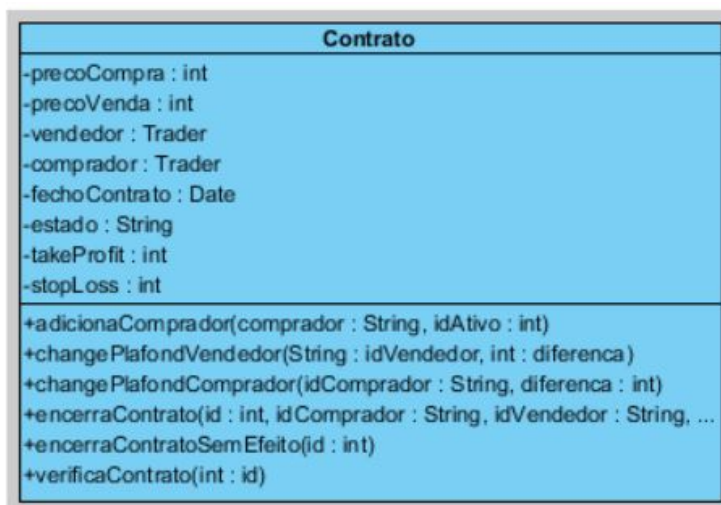


Fig. 6. Blackbox Contrato

Método	Descrição
adicionaComprador	Quando um comprador aceita um contrato, este método atualiza os dados do contrato, adicionando quem o aceitou e quantos contratos fez.
changePlafondVendedor	Quando um contrato é encerrado, este método atualiza o plafond do vendedor com o que ele ganhou/perdeu.
changePlafondComprador	Quando um contrato é encerrado, este método atualiza o plafond do comprador com o que ele ganhou/perdeu.
encerrarContrato	Fecha um contrato e atualiza o estado do mesmo.
encerrarContratoSemEfeito	Encerra um contrato que ninguém aceitou. Contrato sem efeito.
verificaContrato	Verifica se o contrato está em condições de ser encerrado. Este método é várias vezes chamado pela classe Connection.

5. 1. 1. Blackbox Ativo

Esta será a classe responsável pelos ativos. O seu único método é o `atualizaAtivo`, chamado pela classe `Connection`. Este método é responsável por alterar o valor de uma ativo em específico, na base de dados.



Fig. 7. Blackbox Ativo

5. 1. 1. Blackbox Connection

Esta classe estabelece conexão com a base de dados, e, lendo ficheiros `.csv` com os dados dos ativos, atualiza os valores na base de dados. Para além disso, lê os contratos abertos e verifica se os deve fechar. Em caso afirmativo instancia a classe `Contrato`, que irá fazer a gestão necessária.

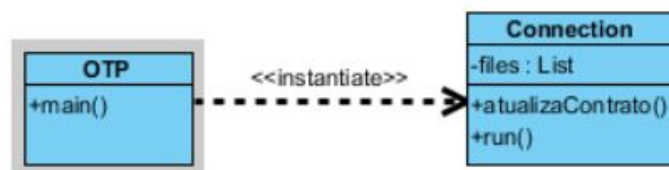


Fig. 8. Blackbox Connection

6. Runtime View

6. 1. Diagrama de Classes

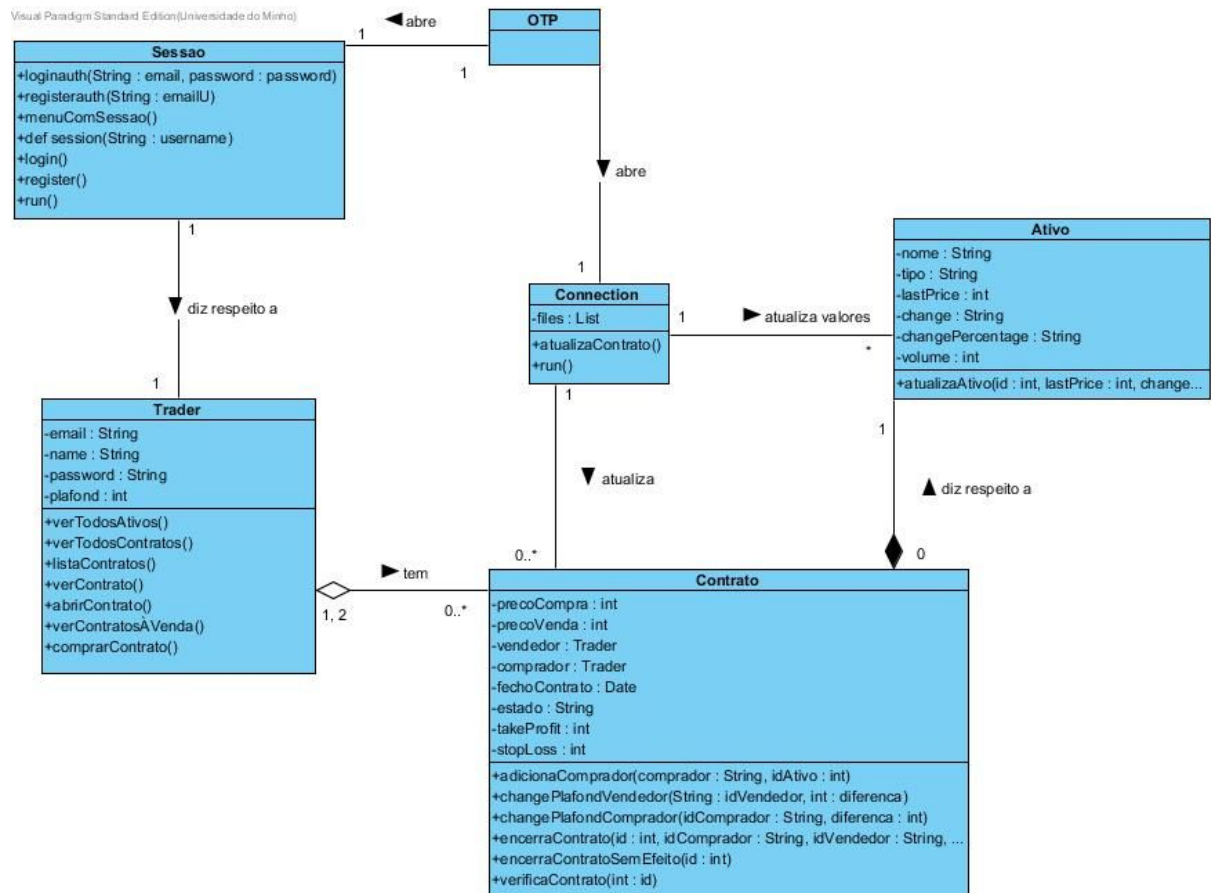


Fig. 9. Diagrama de Classes

6. 2. Diagrama de Use Cases

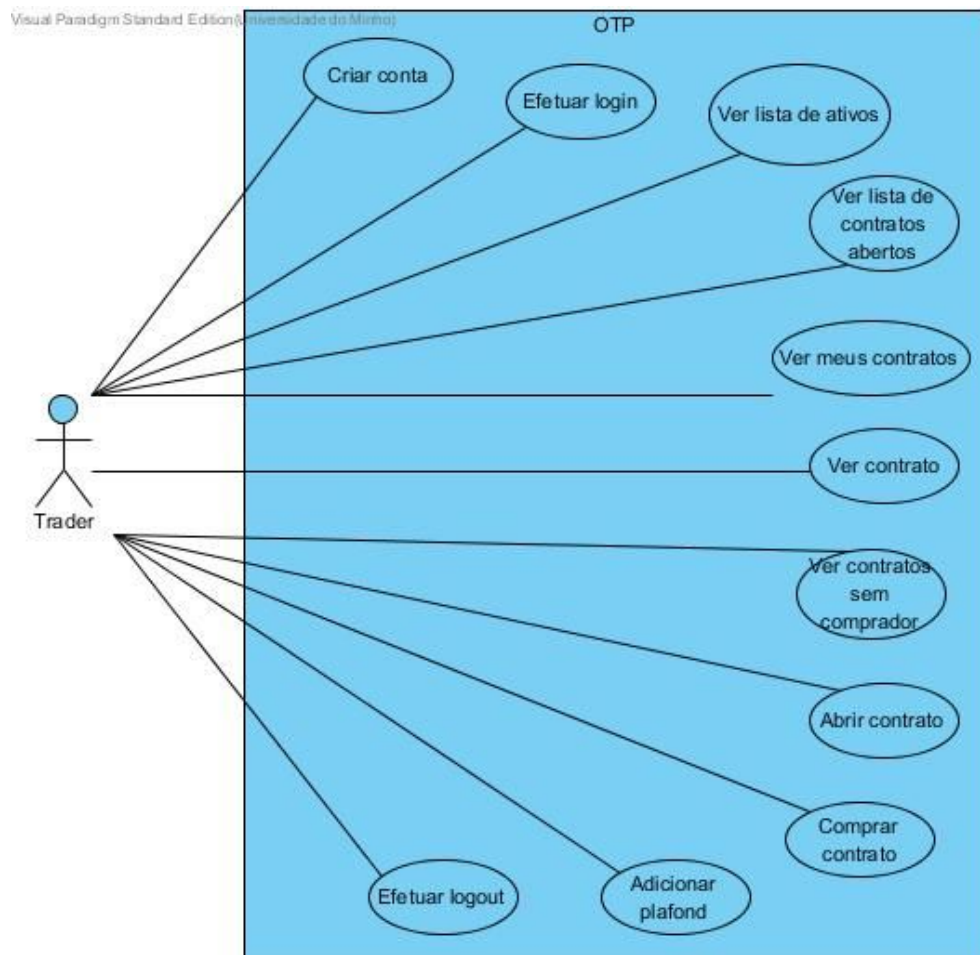


Fig. 10. Diagrama de Use Cases

6. 2. 1. Diagramas de Sequência de Use Case

6. 2. 1. 1. Ver contrato

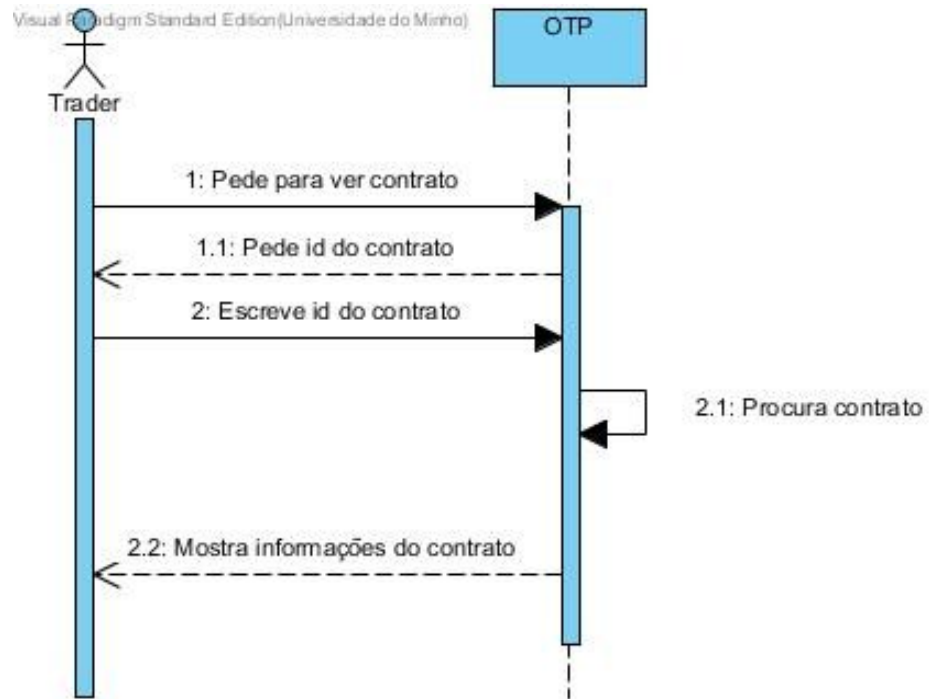


Fig. 11. Diagrama de Sequência de Ver Contrato

6. 2. 1. 2. Criar conta

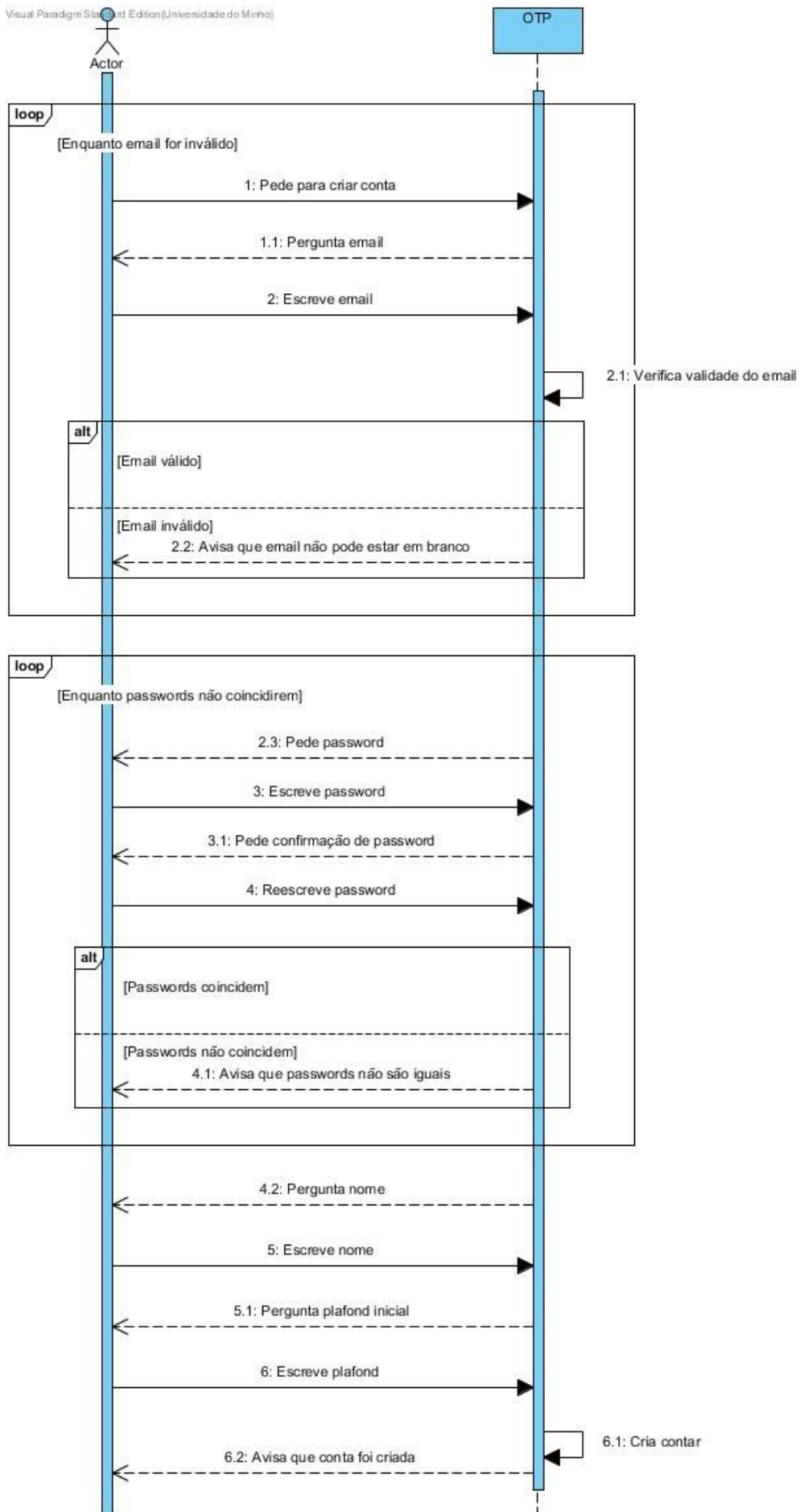


Fig. 12. Diagrama de Sequência de Criar Conta

6. 3. Diagramas de Sequência de Sistema

6. 3. 1. Comprar contrato

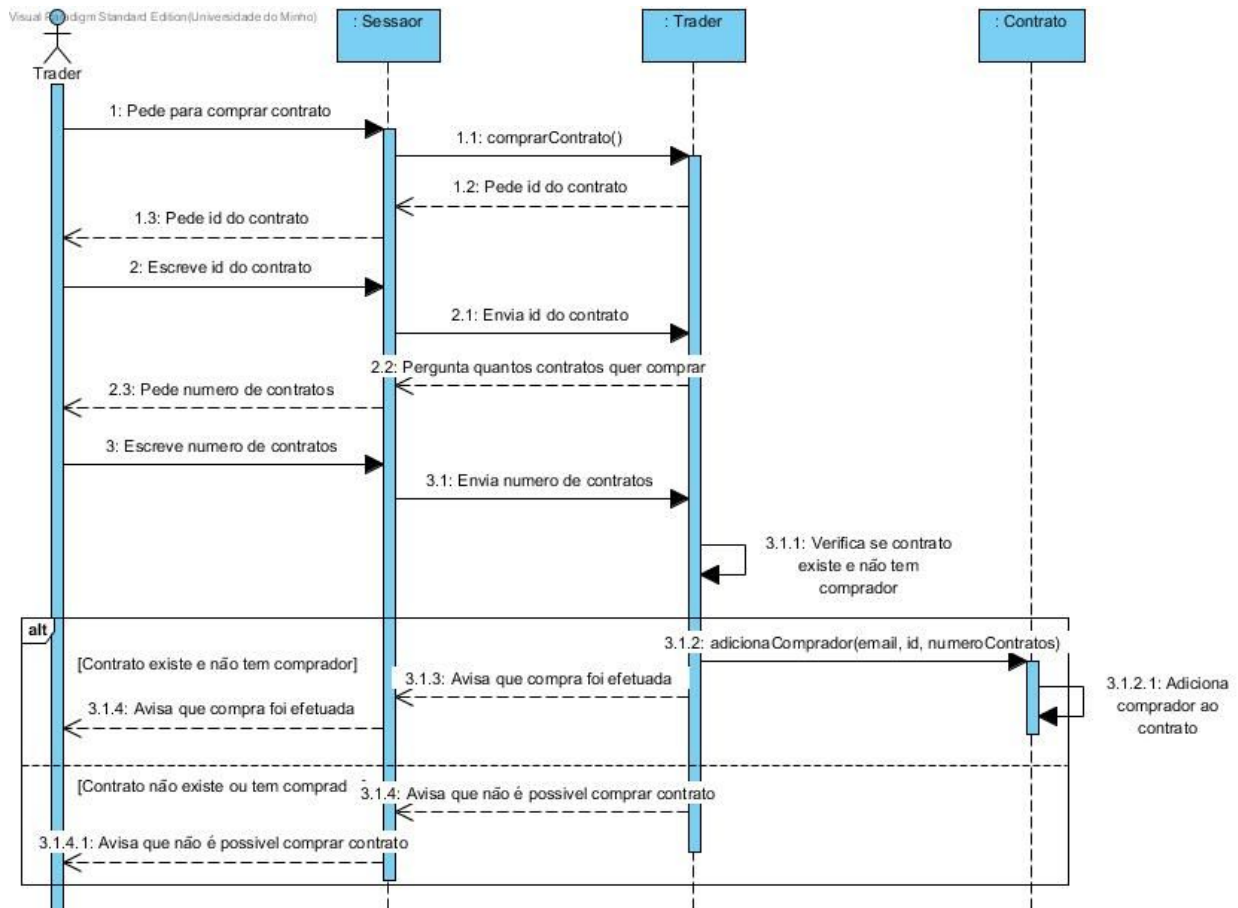


Fig. 13. Diagrama de Sequência de Comprar Contrato

6. 3. 2. Ver lista de ativos



Fig. 14. Diagrama de Sequência de Ver Lista de Ativos

7. Visão de Implementação

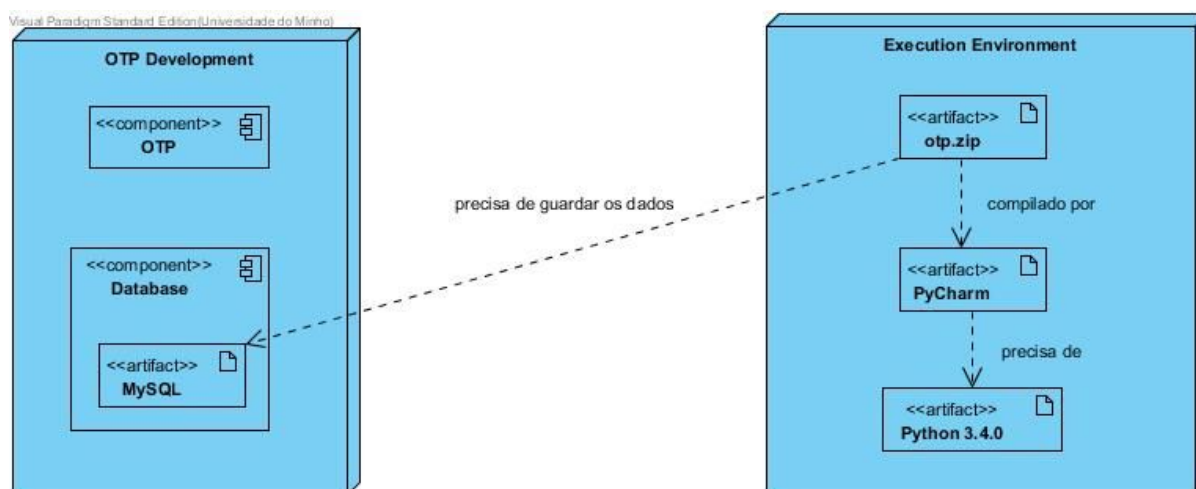


Fig. 15. Deployment Diagram

Node/Artifact	Descrição
OTP Development	Onde a OTP foi desenvolvida. Computador com Python 3.4.0 e onde a IDE PyCharm foi usada para compilar e correr o programa.
Execution Environment	Onde o utilizador irá correr a plataforma.
otp.zip	Onde estão todos os ficheiros necessários para a compilação do programa.
PyCharm	IDE sugerida para compilar e correr a plataforma.

7. 1. Base de Dados

É necessário que o utilizador tenha uma base de dados MySQL, já com um schema criado. O schema utilizado encontra-se em [anexo](#).

A base de dados deverá também ter já alguns dados, que seguem também em [anexo](#).

NOTA: Poderá ser necessário alterar os dados de conexão ao MySQL nos ficheiros .py.

8. Conceitos

8. 1. Modelo de Domínio

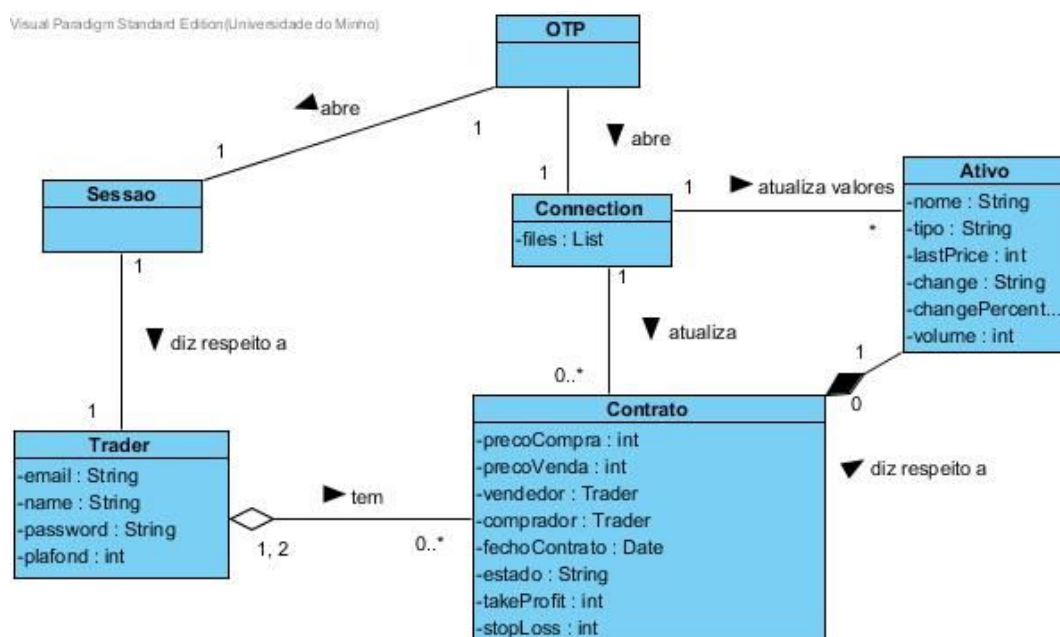


Fig. 16. Modelo de Domínio

No modelo de domínio encontramos classes de implementação, que é o caso da OTP, Sessao e Connection.

As restantes são conceitos de domínio.

A OTP é a classe que abre o sistema, isto é, instancia uma sessao e uma connection.

Na Connection é onde os dados dos ativos são lidos e atualizados na base de dados. É também a responsável por atualizar os contratos.

A Sessao é responsável por gerir, como o nome indica, a sessão do utilizador, desde o login ao menu das funcionalidades.

8. 2. User Interface

A plataforma não tem User Interface, pelo que terá de ser corrida numa shell.

8. 3. Internacionalização

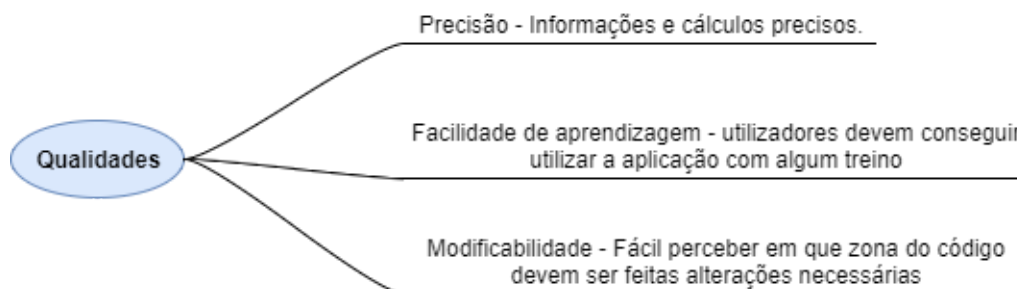
A língua suportada é português. Inglês poderá ser implementado.

9. Decisões do design

A principal decisão neste projeto foi sobre a extração dos dados do Yahoo Finance. Após várias pesquisas a única possibilidade descoberta foi a extração de dados a partir de ficheiros .csv. Para cada ativo seria necessário um url diferente para o download dos dados, e por isso foram selecionados alguns ativos. Os ativos selecionados foram: Ouro, Prata, Crude, Google, Apple, Nvidia, Alibaba e IBM.

10. Cenários de qualidade

10. 1. Quality Tree



11. Riscos Técnicos

A plataforma não foi testada por outros utilizadores e por isso desconhece-se potenciais riscos. Há a possibilidade de, por perdas de atualização de dados, alguns contratos sejam fechados com valores ligeiramente diferentes, prejudicando

investimentos de grande valor. No entanto, estas correções seriam feitas num projeto com maior duração.

12. Mockups

12. 1. Página Inicial

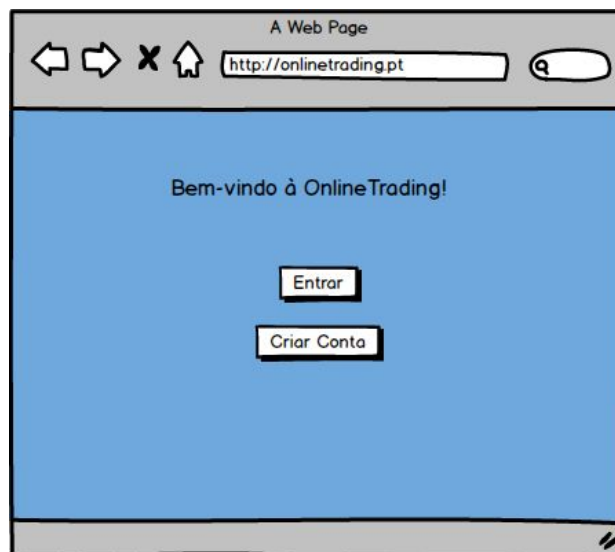


Fig. 17. Mockup de Página Inicial

12. 2. Iniciar sessão

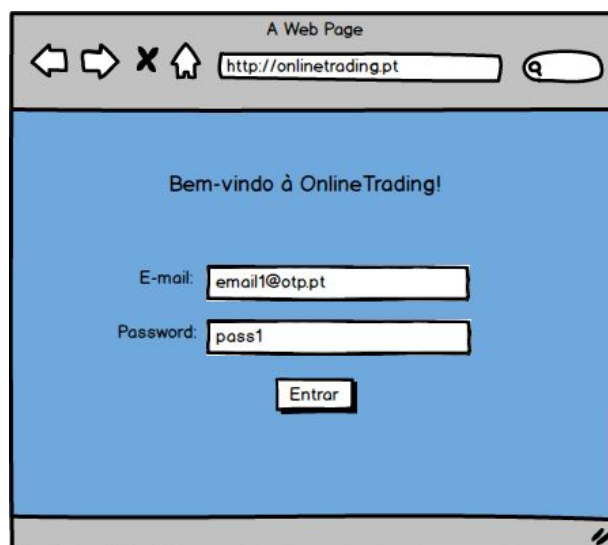


Fig. 18. Mockup de Iniciar Sessão

12. 3. Menu

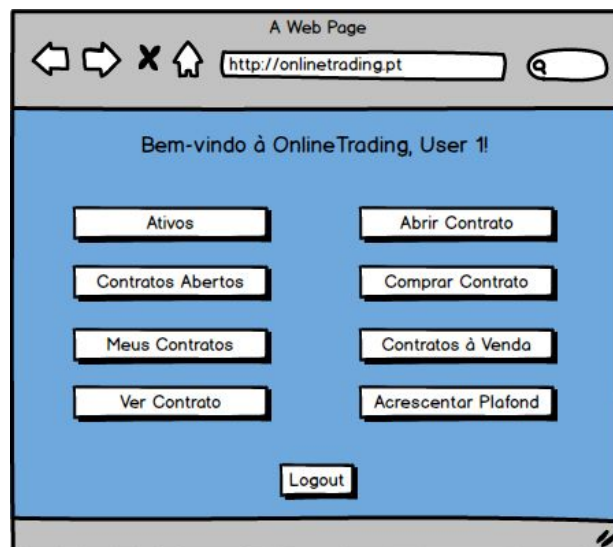


Fig. 19. Mockup de Menu

12. 4. Ativos

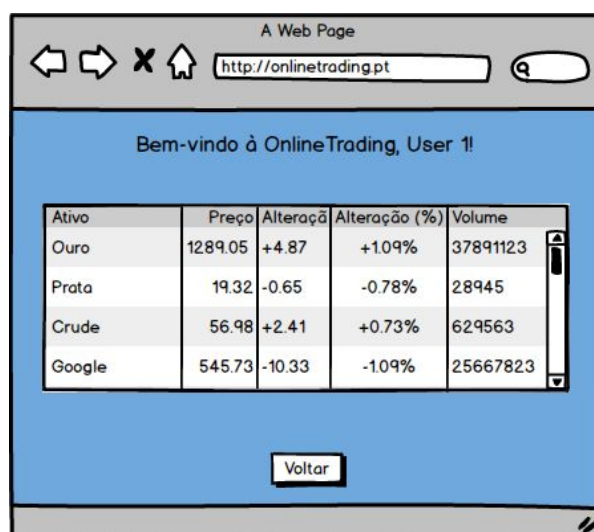


Fig. 20. Mockup de Ver Lista de Ativos

12. 5. Abrir Contrato



Fig. 21. Mockup de Abrir Contrato

13. Glossário

Termo	Descrição
MySQL	Sistema open source de gestão de bases de dados relacionais.
Yahoo Finance	Propriedade de media que faz parte da rede Yahoo!. Ele fornece notícias, dados e comentários financeiros, incluindo cotações de ações, comunicados de imprensa, relatórios financeiros e conteúdo original. Ele também oferece algumas ferramentas online para gerenciamento de finanças pessoais.
Python	Linguagem de programação de alto nível de propósito geral..
PyCharm	IDE usada especificamente para Python.

14. Anexos

1. Schema da Base de Dados

-- MySQL Workbench Forward Engineering

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';
```

```
-- -----
-- Schema mydb
-- -----
```

```
-- Schema otp_db
-- -----
```

```
-- Schema otp_db
-- -----
```

```
CREATE SCHEMA IF NOT EXISTS `otp_db` DEFAULT CHARACTER SET utf8 ;
USE `otp_db` ;
```

```
-- -----
-- Table `otp_db`.`tipo`
-- -----
```

```
CREATE TABLE IF NOT EXISTS `otp_db`.`tipo` (
  `idTipo` INT(11) NOT NULL,
  `nome` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idTipo`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;
```

```
-- -----
-- Table `otp_db`.`ativo`
-- -----
```

```
CREATE TABLE IF NOT EXISTS `otp_db`.`ativo` (
  `idAtivo` INT(11) NOT NULL,
  `idTipo` INT(11) NOT NULL,
```



```

`ultimoPreco` FLOAT NOT NULL,
`change` VARCHAR(45) NULL DEFAULT NULL,
`changePercentagem` VARCHAR(45) NULL DEFAULT NULL,
`volume` INT(11) NULL DEFAULT NULL,
PRIMARY KEY (`idAtivo`),
INDEX `idTipo_idx` (`idTipo` ASC),
CONSTRAINT `idTipo`
  FOREIGN KEY (`idTipo`)
  REFERENCES `otp_db`.`tipo` (`idTipo`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION)
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;

```

```

-----
-- Table `otp_db`.`trader`
-----

```

```

CREATE TABLE IF NOT EXISTS `otp_db`.`trader` (
  `email` VARCHAR(45) NOT NULL,
  `password` VARCHAR(45) NOT NULL,
  `nome` VARCHAR(45) NOT NULL,
  `plafond` FLOAT NOT NULL,
  PRIMARY KEY (`email`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8;

```

```

-----
-- Table `otp_db`.`contrato`
-----

```

```

CREATE TABLE IF NOT EXISTS `otp_db`.`contrato` (
  `idContrato` INT(11) NOT NULL AUTO_INCREMENT,
  `idAtivo` INT(11) NOT NULL,
  `idVendedor` VARCHAR(45) NOT NULL,
  `idComprador` VARCHAR(45) NULL DEFAULT NULL,
  `precoVenda` INT(11) NOT NULL,
  `precoCompra` INT(11) NULL DEFAULT NULL,
  `dataFechoContrato` DATE NOT NULL,
  `takeProfit` INT(11) NOT NULL,
  `stopLoss` INT(11) NOT NULL,
  `estado` VARCHAR(45) NOT NULL,

```

```

`numeroContratos` INT(11) NULL DEFAULT NULL,
PRIMARY KEY (`idContrato`),
INDEX `idAtivo_idx` (`idAtivo` ASC),
INDEX `idVendedor_idx` (`idVendedor` ASC),
INDEX `idComprador_idx` (`idComprador` ASC),
CONSTRAINT `idAtivo`
  FOREIGN KEY (`idAtivo`)
  REFERENCES `otp_db`.`ativo` (`idAtivo`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION,
CONSTRAINT `idComprador`
  FOREIGN KEY (`idComprador`)
  REFERENCES `otp_db`.`trader` (`email`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION,
CONSTRAINT `idVendedor`
  FOREIGN KEY (`idVendedor`)
  REFERENCES `otp_db`.`trader` (`email`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION)
ENGINE = InnoDB
AUTO_INCREMENT = 5
DEFAULT CHARACTER SET = utf8;

SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;

```

2. Dados já na DB

```

INSERT INTO trader VALUES ('email1@otp.pt', 'pass1', 'nome1', '2000'),
('email2@otp.pt', 'pass2', 'nome2', '1000'),
('email3@otp.pt', 'pass3', 'nome3', '500'), ('email4@otp.pt', 'pass4', 'nome4', '10000'),
('email5@otp.pt', 'pass5', 'nome5', '4500');

```

```

INSERT INTO tipo VALUES (1, 'Gold'), (2, 'Silver'), (3, 'Crude Oil'), (4, 'Google'), (5,
'Apple'),

```

(6, 'Nvidia'), (7, 'Alibaba'), (8, 'IBM');

```
INSERT INTO ativo VALUES(1, 1, 1297.00, '-7.60', '-0.58', 262723), (2, 2, 17.25,
'-0.16', '-0.92', 75763),
(3, 3, 51.85, '+0.40', '+0.78', 492353), (4, 4, 991.92, '+2.24', '+0.23', 836220),
(5, 5, 159.88, '+2.89', '+1.84', 23898054), (6, 6, 197.93, '+3.34', '+1.72', 14279079),
(7, 7, 179.56, '+1.11', '+0.62', 12728649), (8, 8, 146.76, '-0.34', '-0.23', 2581419);
```

```
INSERT INTO traderativo VALUES (1, 'email1@otp.pt', 1), (2, 'email1@otp.pt', 1),
(3, 'email1@otp.pt', 2), (4, 'email2@otp.pt', 3), (5, 'email3@otp.pt', 4),
(6, 'email4@otp.pt', 4), (7, 'email5@otp.pt', 5), (8, 'email1@otp.pt', 4),
(9, 'email2@otp.pt', 5), (10, 'email5@otp.pt', 8), (11, 'email3@otp.pt', 7);
```

```
INSERT INTO contrato VALUES (1, 1, 'email1@otp.pt', 'email2@otp.pt', 1290,
NULL, "2017-11-01", 1310, 1280, 'aberto', 5);
INSERT INTO contrato (idAtivo, idVendedor, precoVenda, dataFechoContrato,
takeProfit, stopLoss, estado)
VALUES("3", "email2@otp.pt", 56.6, "2017-10-29", 55, 45, "aberto");
```

```
INSERT INTO contrato (idAtivo, idVendedor, precoVenda, dataFechoContrato,
takeProfit, stopLoss, estado)
VALUES(5, "email2@otp.pt", 159.76, "2017-10-29", 165, 155, "aberto");
```