PROJECTO DE API PARA UMA CARTEIRA DIGITAL CLIENTE-AGENTE COM FUNCIONAMENTO SIMILAR AO M-PESA

Autor: Narciso Cadeado

No presente artigo, serão apresentados os detalhes de implementação da API de uma carteira digital com funcionamento similar ao m-pesa, em que os clientes podem depositar, levantar, transferir e enviar dinheiro.

O presente documento é um artigo de nível técnico destinado a programadores e engenheiros de software que queriam implementar a API em uma determinada linguagem de programação.

1. CAPITULO I – FUNCIONAMENTO DO NEGÓCIO

Antes de iniciar com os detalhes têcnicos da implementação da API, é necessário compreender como o negócio funciona na sua essência. É necessário compreender como os dados fluem desde o momento que o cliente se cadastra, como ele deposita dinheiro, como ele levanta o dinheiro em um agente e como ele efectua operações.

1.1. Cadastro de Utilizadores

O cadastro é o primeiro passo a ser feito.

Primeiro, o utilizador precisa acessar a carteira digital utlizando algum meio (um front-end), seja por uma aplicação da carteira digital baixada na playstore /appstore ou utilizando algum navegador web. Ao tentar cadastrar-se, o utilizador terá a opção de se cadastrar como cliente ou como agente.

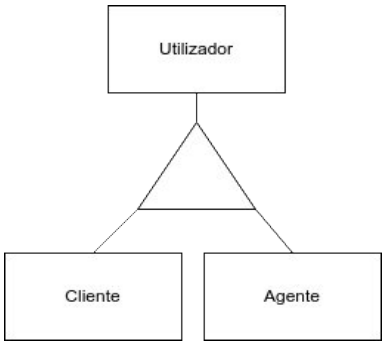


Fig 1: Tipos de utilizadores.

A figura 1 mostra que, no negócio existem 2 tipos de utilizadores, Cliente e Agente, onde :

- **Os clientes,** são os utilizadores finais que podem realizar as operações basicas como levantamento e transferencias ;
- **Os Agentes,** são utilizadores que permitem que os clientes levantem e depositem dinheiro na carteira digital.

1.1.1. Cadastro de Clientes

Para que se possam cadastrar, os clientes precisam fornecer os seguintes dados:

- Nome e sobrenome – Serão utilizados para identificação do cliente;
- Nome do utilizador (username) - Utilizado para identificar o utilizador a nivel do sistema;
- Email - Será utilizado para recuperação de senha, 2FA e comunicação com o cliente.
- Numero de Celular – Para validação de operações com OTP.

Ao criar a conta cliente, automaticamente uma conta será aberta. Um cliente pode ter uma ou mais contas.

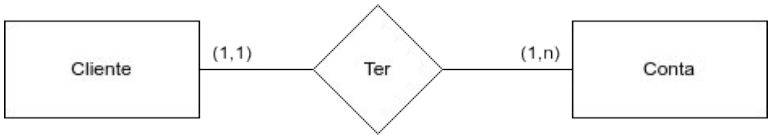


Fig 2: Relação entre cliente e conta.

1.1.2. Cadastro de Agentes

Para que um utilizador consiga cadastrar-se como agente, para além dos dados basicos como username, nome e sobrenome, é necessário que tenha um *token/chave.* A chave é um identificador único que só pode ser utilizado uma única vez para abrir uma conta agente.

Durante a abertura de uma conta agente, o saldo inicial pode ser inicializado na conta do mesmo. Se o agente não tiver o saldo inicial, não poderá efectuar depositos na conta de um determinado cliente.

1.2 Operações

1.2.1 Deposito

O deposito só poderá ser realizado em um agente com saldo inicial > 0. O cliente fornecerá o montante e o número de conta ao agente e o agente irá efectuar o deposito.

Ao afectuar o deposito, o saldo do agente será transferido para a conta do cliente.

1.2.2. Levantamento

O levantamente é a operação contraria ao deposito. O cliente que quiser efectuar o levantamento, terá que ter o saldo disponivel na conta e deverá ter o número do agente onde pretende levantar.

Ao efectuar o levantamento, o saldo do cliente será transferido para a conta do agente. Neste momento, uma taxa será cobrada. Um determinado valor desta taxa será lucro da carteira digital e a outra parte do valor irá para o agente.

Para que os agentes levantem dinheiro das suas contas é necessário que efectuem uma transferencia do valor para uma conta bancária (mais detalhes sobre isso estarão descritos no capitulo sobre a integração com outras plataformas).

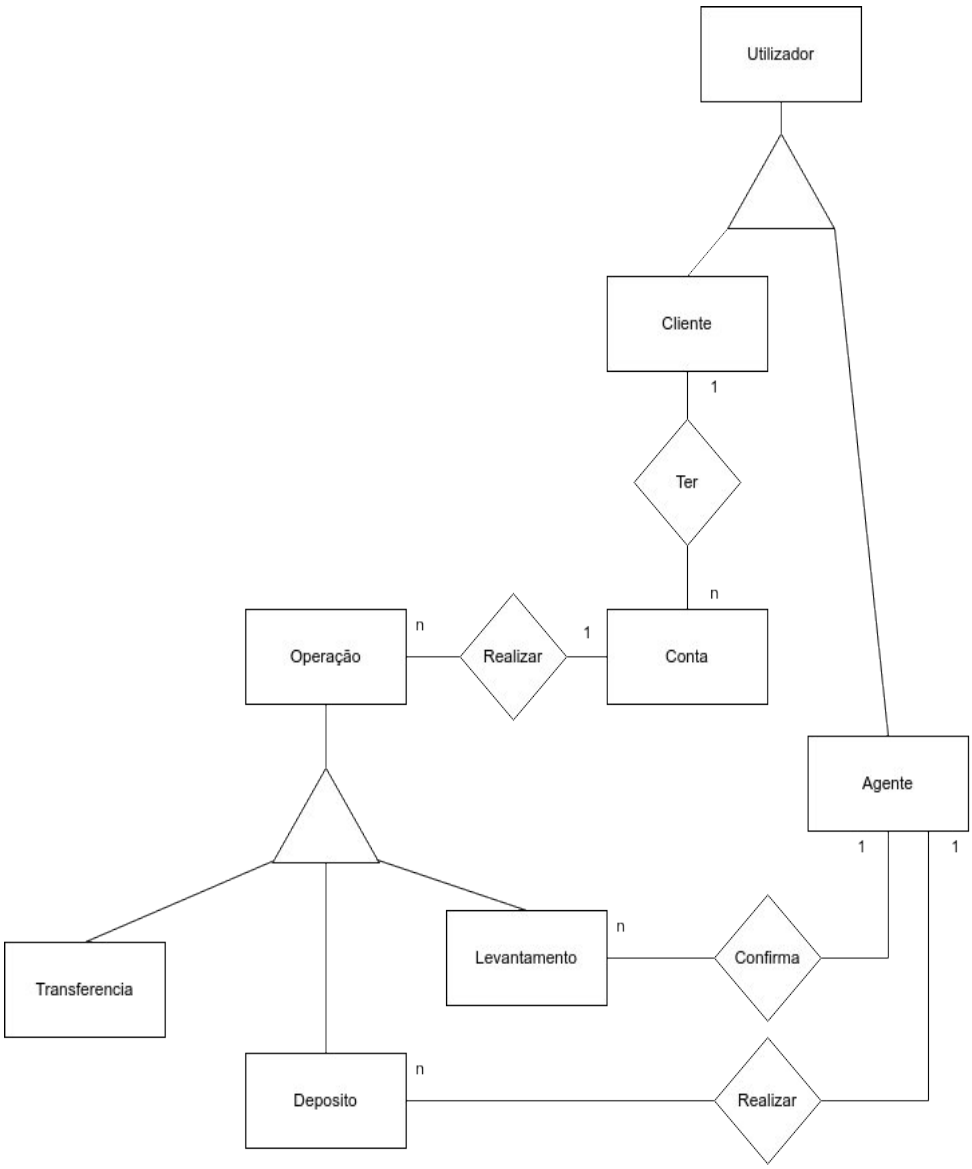
1.2.3. Transferencias

Um cliente poderá transferir saldo de uma conta para outra.

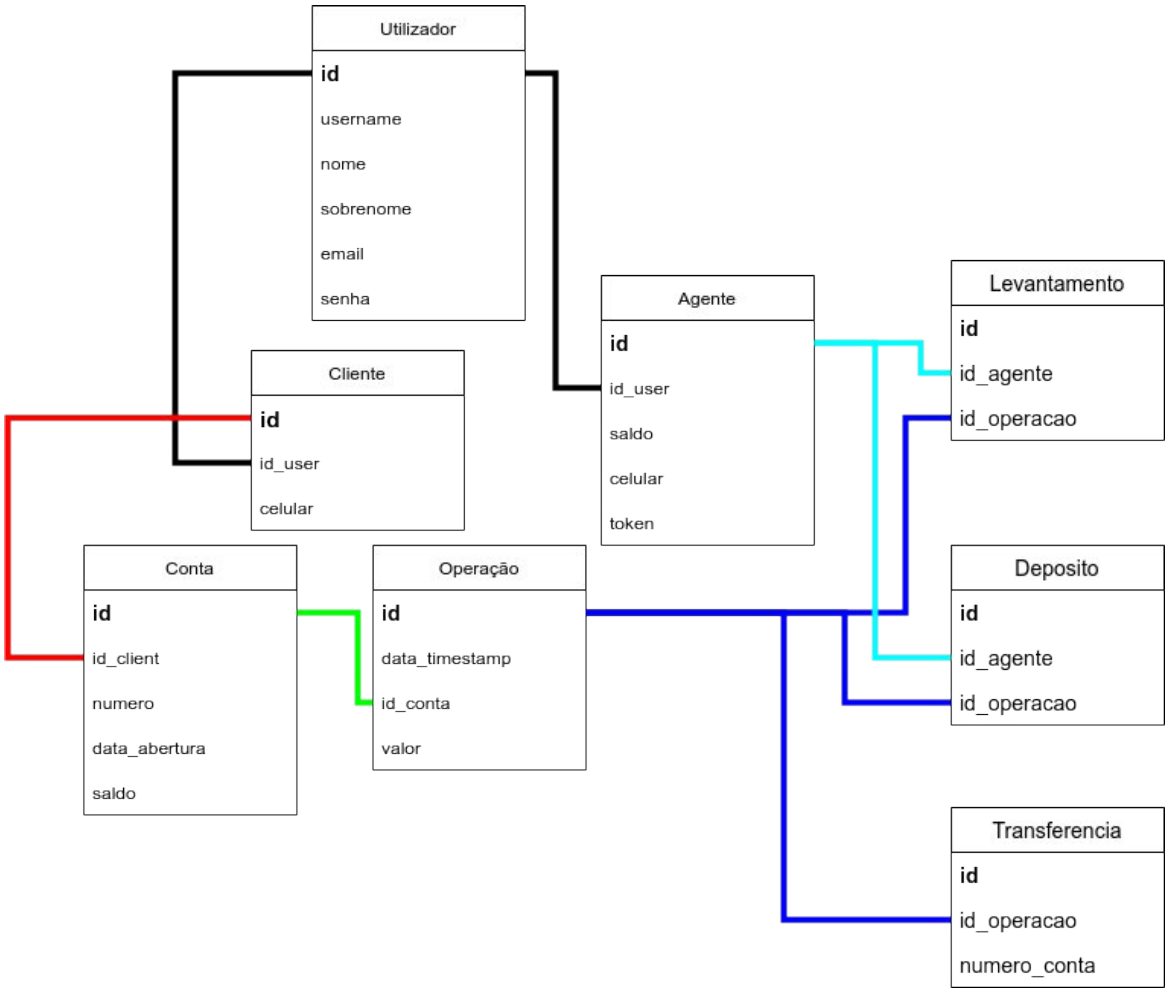
Ao efectuar a transferencia, será necessário ter o número de conta do cliente na qual pretendemos enviar o saldo. O saldo será debitado na conta actual, a taxa de transferencia será aplicada e, por fim, o saldo será acrescentado no conta de destino .

2. CAPITULO 2 – MODELO RELACIONAL

Tendo em conta as informações descritas no capitulo I, foi concebido o seguinte modelo relacional:

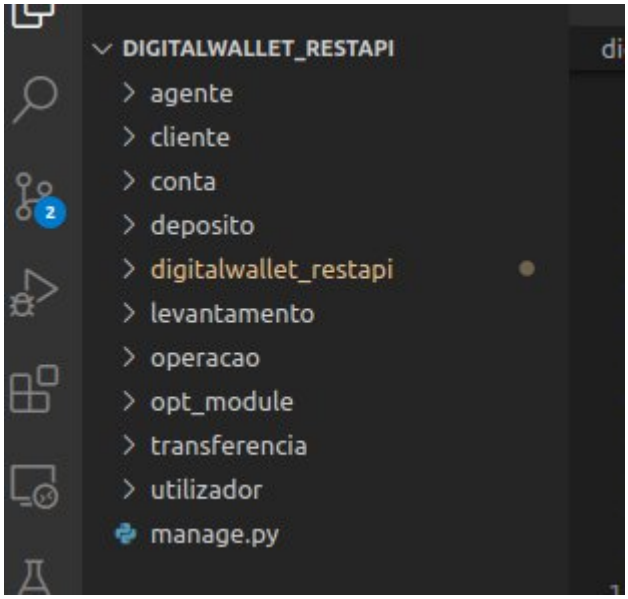


3. CAPITULO 3 – BASE DE DADOS RELACIONAL E NORMALIZAÇÃO



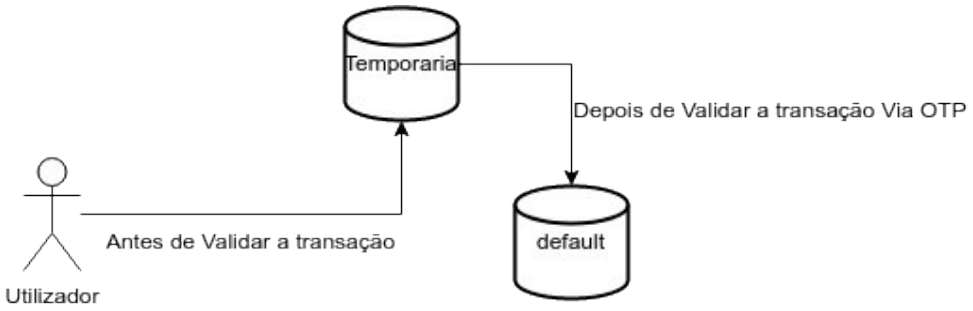
4. CAPITULO 4 – PROTOTIPO DA API UTILIZANDO DJANGO

Para tornar a API mais organizada, cada entidade do projecto será tratada em um aplicativo especifico. Com isso, foram criados 8 aplicativos django dentro do mesmo projecto.



4.1 Sistema de Validação de Transações Via OPT

Para a implementação do sistema de autenticação de transações, serão usados duas base de dados : “default” e “Temporaria”. Cada entidade na base de dados default, terá uma “irma” na base de dados temporaria que vai armazenar os dados temporarios.



Base de Dados Provisoria

A base de dados temporaria, irá armazenar os dados dos utilizadores, operações e outros dados, cuja persistência precisa de validação. Por exemplo, quando a pessoa se cadastrar como utilizador, terá que inserir o número de celular, para que este número, seja usado para validar todas as transações da pessoa inclusive o proprio cadastro. Assim sendo, quando a pessoa efectuar o cadastro, uma mensagem OTP será enviada ao celular para que possa confirmar o cadastro com o código OTP. Enquanto a pessoa não valida o cadastro, os dados estarão no armazenamento temporario, que será apagado após um certo tempo.

Base de Dados default

Na base default, do contrário da temporaria, armazena os dados já persistidos e validados via OTP. Por exemplo se a pessoa afectuou o cadastro e validou o número de celular via OPT, os dados do cadastro serão enviados para a base de dados default.

5. ***CAPITULO 5 – AUMENTAR A PERFORMANCE DA API UILIZANDO: REDUNDÂNCIA, BALANCEMANTO DE CARCA, CACHEAMENTO E REPLICAÇÃO DE BADE DE DADOS***
6. ***CAPITULO 6 – INTEGRAÇÃO DA API COM OUTRAS PLATAFORMAS m-pesa, e-mola, m-kesh, rede Visa...***
7. ***CAPITULO 7- DEPLOY E SEGURANÇA*.**