deti universidade de aveiro
departamento de eletrónica, telecomunicações e informática

TQS: Relatório de Especificação do Produto

Manuel Diaz [103645], Pedro Coelho [104247], Diogo Silva [98644], Luca Pereira [97689]
v2023-06-01

1 Introdução

1.1 Overview do projeto

No contexto da disciplina de TQS, “Teste e Qualidade de Software”, foi pedido aos alunos para fazerem um trabalho de grupo de modo a especificar e implementar um projeto de tamanho médio, composto pelo:

- Desenvolvimento de um produto de software viável, que inclua a sua especificação funcional, arquitetura e implementação do sistema;
- Especificação e aplicação de uma estratégia de Garantia de Qualidade de Software (SQA), aplicada em todo o processo de engenharia de software.

Neste projeto em questão deverá ser implementado uma rede de ***pick up points*** e, para tal, desenvolvemos a nossa aplicação, mailMover, que pretende facilitar a vida aos clientes dos seus parceiros.

1.2 Limitações

Como é possível observar na dashboard do SonarCloud, o nosso backend apresenta algumas limitações, nomeadamente no que diz respeito à duplicação de código e à segurança. Estes aspectos carecem de maior atenção e revisão, o que infelizmente não foi possível por questões temporais.

Relativamente ao frontend da nossa loja, é necessário concluir o código para a implementação do carrinho de compras, falta desenvolver o código para a barra de pesquisa e ainda realizar melhoramentos estéticos. Para dar apoio ao frontend ainda é necessário criar uma base de dados com tabelas para os produtos, users e ainda encomendas (neste momento temos só um ficheiro em JavasScript com os produtos).

Também é importante salientar que, devido a um erro que não conseguimos corrigir até ao momento, não efetuamos o deploy da nossa solução na VM, portanto também não conseguimos implementar o CD. Porém queremos tentar corrigir estas questões o mais rapidamente possível.

2 Concepção do produto

2.1 Visão

A plataforma **mailMover** foi desenvolvida como uma solução para diversos negócios de venda de produtos, com vista a melhorar a experiência dos seus clientes. O sistema tem como objetivo simplificar o processo de encomendas e de seleção de pontos de recolha para as compras.

Deste modo, pretendemos oferecer aos clientes uma forma mais flexível e acessível de receberem os seus pedidos. Em vez de depender apenas de serviços de entrega tradicionais, os clientes podem escolher pontos de entrega que lhes sejam convenientes, como lojas locais ou locais específicos de recolha. Assim, este modelo reduz, significativamente, a possibilidade de haver uma falha na entrega, visto, por exemplo, não existir o risco de a pessoa que vai efetuar a

entrega não conseguir encontrar a morada ou não haver ninguém para a receber, e é também mais eficiente na perspetiva de logística para quem faz as entregas.

Este modelo de negócio tem vindo a ganhar popularidade, por exemplo, com o serviço *Amazon Locker.* Este serviço permite aos clientes da loja online da *Amazon* que tenham os seus produtos entregues em pontos de entrega geridos pela *Amazon.* Este modelo tem semelhanças claras com a nossa proposta, porém tem a grande diferença de ser um serviço apenas para clientes da loja da *Amazon* e não um serviço para outras lojas online terem acesso a uma rede de *pick up points*. Outra plataforma já existente que é mais similar à nossa proposta é a *Parcel Pending,* este serviço permite aos clientes das suas lojas parceiras terem as suas encomendas entregues num *pick up point* que lhes seja conveniente.

2.2 Personas e cenários

Como primeira persona do nosso projeto, apresentamos o Jorge, que tem 22 anos, é estudante na universidade de Aveiro e, uma vez que está constantemente a mudar de localização e gosta de encomendar sapatilhas, pretende arranjar uma forma para que lhe seja fácil recolher as suas encomendas.

Jorge, tem 22 anos e está no segundo ano de mestrado de desenvolvimento de jogos digitais na Universidade de Aveiro. Nasceu no Porto e é filho único. Uma vez que estuda em Aveiro, divide o seu tempo entre a casa dos pais no Porto e os dormitórios da universidade. Passa a maior parte do seu tempo ao ar livre a correr e a andar de skate.
Além disso, o Jorge é um entusiasta de sapatilhas, tendo uma coleção diversa no seu armário. Como estudante universitário, Jorge está sempre à procura da melhor oferta por um bom preço, comparando os preços disponíveis em diversas páginas web.

Objetivos: O Jorge deseja conseguir encontrar as sapatilhas de coleção de que gosta. Para além disso, como está frequentemente a mudar de localização, pretende que as suas encomendas sejam direcionadas para um ponto de recolha onde ele as possa levantar quando lhe for mais conveniente.

Tendo em consideração a persona definida e os objetivos do nosso projeto definimos os seguintes cenários:

Cenário 1: Jorge procura por sapatilhas e pretende ver os seus detalhes

Jorge está a comparar preços de diversas lojas online para descobrir qual é a melhor oferta para umas sapatilhas específicas. Sendo assim, o Jorge navega pela página principal, onde estão colocadas em display várias sapatilhas, à procura das sapatilhas que deseja. Depois de não encontrar as sapatilhas desejadas, este vai usar a barra de procura onde escreve o nome das sapatilhas que quer e ao clicar nelas pode ver os diversos detalhes, incluindo o preço que queria saber.

Cenário 2: Jorge faz o login e escolhe um ponto de recolha

Jorge já sabe que sapatilhas quer comprar, e para o fazer, vai fazer o login no website de modo a poder efetuar a compra. Ele vai introduzir as suas credenciais, nomeadamente o nome de utilizador e a palavra-passe.

Após o login, Jorge é redirecionado para a página principal do site onde pode ver algumas sapatilhas que estão disponíveis. Ao clicar nas que pretende, avança para a página dos detalhes das mesmas, onde vai clicar em “Encomendar”. De seguida, introduz os seus dados e é-lhe apresentado uma lista de pontos de recolha, incluindo lojas físicas ou pontos específicos. Ele seleciona o ponto de recolha que lhe é mais conveniente, que neste caso é à entrada dos dormitórios da universidade de Aveiro.

Como segunda persona do nosso projeto, temos a Sofia, de 28 anos, uma administradora da mailMover, que se dedica ao máximo em oferecer uma solução de entrega de encomendas eficiente e conveniente para clientes e empresas de venda de produtos.

Sofia, tem 28 anos e é uma jovem empreendedora apaixonada por tecnologia e inovação, que tirou formação universitária na área de gestão na Universidade de Aveiro. Ela nasceu em Lisboa e é por lá que vive após ter terminado os estudos. Adora a energia da cidade e todas as oportunidades que ela oferece. É uma pessoa sociável e gosta de se envolver em comunidades tecnológicas, participando em eventos relacionados com a sua área.
A Sofia é uma pessoa determinada e persistente, que está sempre em busca de melhorias para a plataforma da mailMover.

Objetivos: A Sofia pretende fornecer uma plataforma eficiente e confiável para todos os negócios parceiros. Deste modo, ela empenha-se em garantir que os clientes tenham uma experiência satisfatória ao escolherem um ponto de recolha para as suas encomendas, sendo o seu principal objetivo oferecer uma solução inovadora que simplifique a vida das pessoas e impulsione o crescimento dos negócios.

Com base nesta persona desenvolvemos os seguintes cenários:

Cenário 1: Sofia faz login e acompanha as encomendas

Sofia, como administradora da plataforma, realiza o login no sistema usando as suas credenciais. Uma vez autenticada, ela tem acesso ao painel de administração. Nesse painel, Sofia pode visualizar a lista de todas as encomendas feitas pelos clientes, podendo filtrar as encomendas por ACP específico, permitindo-lhe obter uma visão sobre as encomendas associadas a um determinado ponto de recolha.

Ao analisar a lista de encomendas, Sofia identifica uma encomenda com um problema. Utilizando a funcionalidade de filtrar por ID, ela pesquisa a encomenda pelo seu ID único. Uma vez encontrada, ela visualiza os detalhes dessa encomenda em específico e, se necessário, toma medidas para corrigir ou resolver o problema.

Cenário 2: Sofia adiciona novos pontos ACP e altera o estado duma encomenda

Sofia identifica a necessidade de adicionar um novo ponto de recolha à plataforma. Utilizando a funcionalidade de adicionar ACP’s, ela insere as informações necessárias, como nome e endereço.

Após adicionar o novo ACP, Sofia recebe uma solicitação para alterar o estado de uma encomenda, que acabou de sair de uma loja parceira onde foi efetuada a compra. De seguida, pesquisando pelo ID da mesma, ela altera o seu estado de *Store* para Courier, identificando desta forma que a encomenda está a caminho.

2.3 *Epics* do projeto e prioridades

De modo a cumprir com os objetivos, adotamos uma metodologia ágil para o desenvolvimento e implementação do nosso sistema. Com base nesta abordagem, planejamos implementar a solução em iterações, priorizando as funcionalidades mais importantes e de alto valor para os utilizadores.

Iteração 1

- Definir conceito do produto;
- Desenhar a arquitetura;
- Recursos da equipa, como o repositório, canal de comunicação, etc.

Iteração 2

- Definir a arquitetura do sistema;
- Definir as práticas de SQE;
- Começar a desenvolver o *Product Specification Report;*
- Sistema de backlog pronto;
- Protótipo da UI.

Iteração 3

- Implementar *user stories* envolvendo o acesso a dados;
    - Como utilizador, devo conseguir fazer login na página da loja;
    - Como utilizador, devo poder ver a lista de produtos na página da loja;
    - Como admin, devo poder autenticar-me com username e password;
    - Como admin, devo poder ver a lista de encomendas no ponto ACP na página desse mesmo ACP
    - Como admin, devo poder ver a lista de todas as encomendas que estão a ser processadas na página de admin
- Começar a desenvolver o *QA Manual*;
- Preparar a *pipeline CI*

Iteração 4

- Implementar *user stories* envolvendo o acesso a dados;
    - Como utilizador, devo conseguir realizar uma encomenda;

    - Como admin, devo poder autenticar-me com username e password na página de gestão de ACP;
    - Como admin, devo poder ver a lista de todos os ACP's na página de admin;
    - Como admin, devo poder adicionar um ACP na página de admin.
- Preparar a *pipeline CD;*
- API pronta.

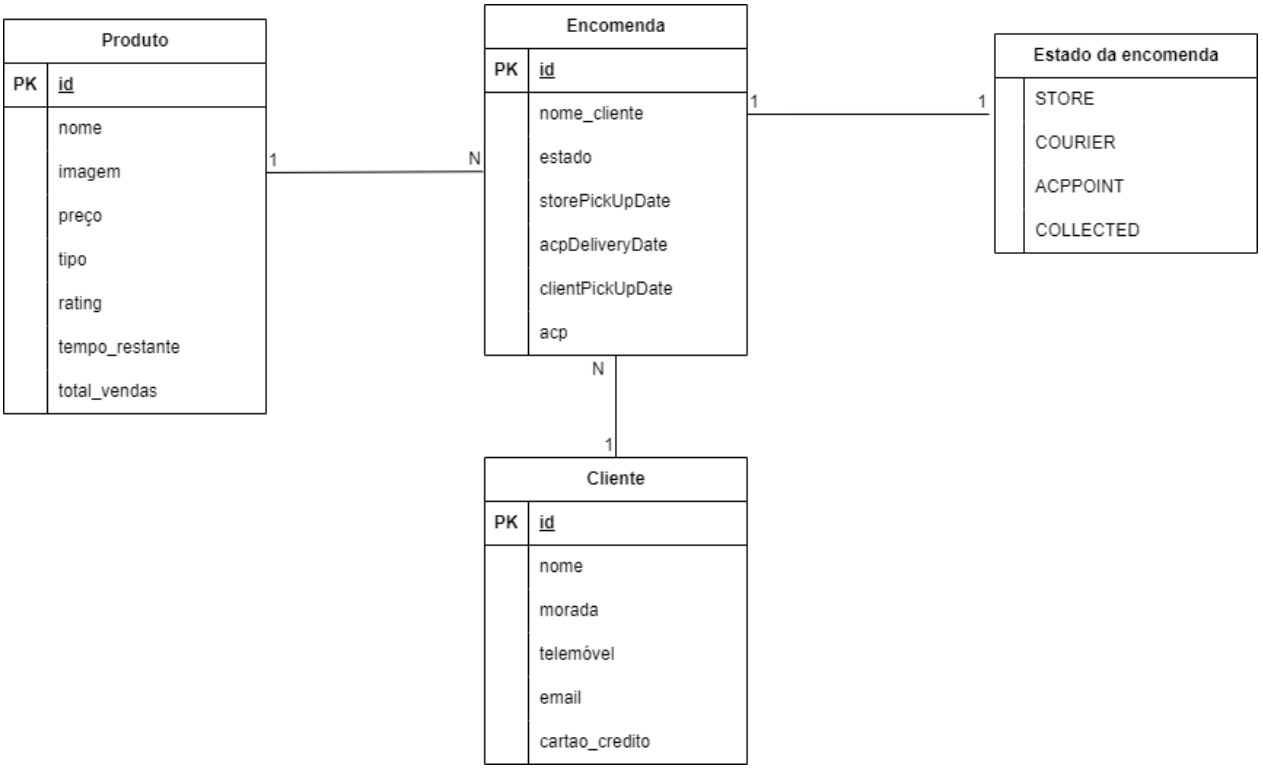
Iteração 5

- Versão final do *Product Specification Report;*
- Versão final do *QA Manual;*
- Estabilizar o *Minimal Viable Product* (MVP)

Iteração 6

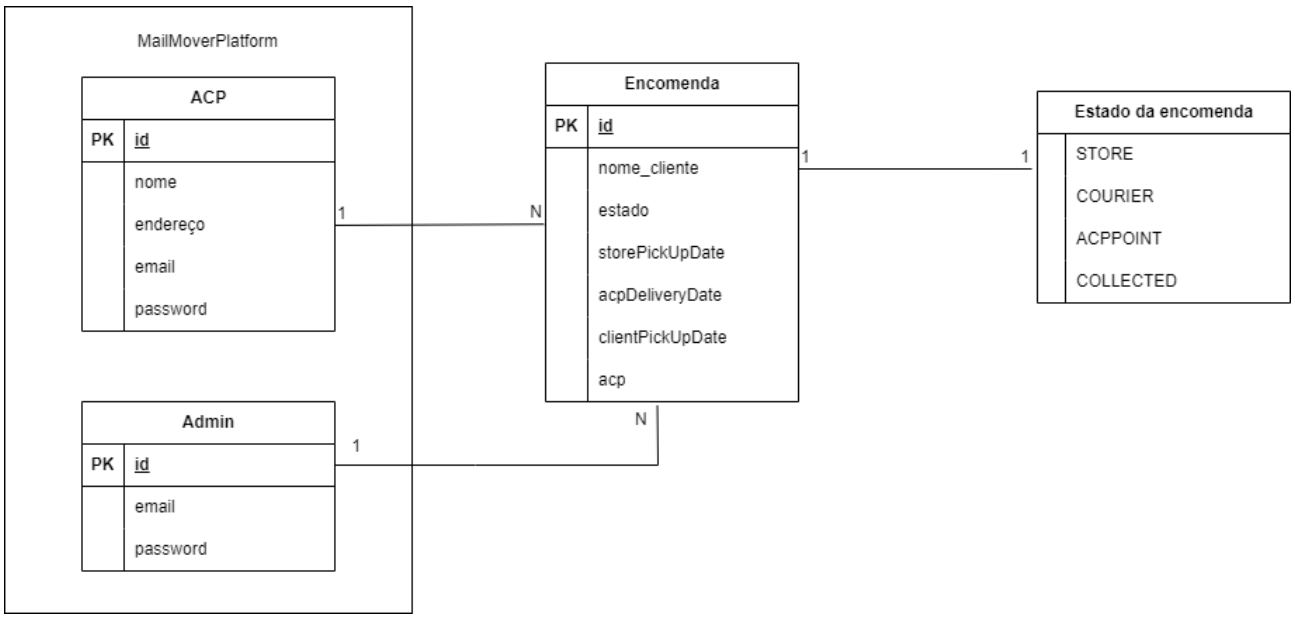
- Backend do *Minimal Viable Product* (MVP) deployed;
- Apresentação final / defesa

3 Domain model



Relativamente ao primeiro modelo de domínio, temos 4 módulos principais: o **Produto**, a **Encomenda**, o **Cliente** e o Estado da **Encomenda**. O Produto representa os produtos na nossa loja (sapatilhas) e são definidos com um id, nome, imagem, preço, tipo, rating, tempo restante de compra e vendas totais do produto. Além dos produtos, a nossa loja terá o Cliente que representa os clientes da loja caracterizados por um id, nome, morada, número de telemóvel, email e número do cartão de crédito. Estes poderão fazer várias encomendas que irão ter um produto cada. Já a

Encomenda é representada por um id, nome do cliente associado, estado de encomenda que poderá ser “STORE”, “COURIER”, “ACPPOINT” e “COLLECTED”, irá também ter uma data de saída da loja, uma data de chegada ao ponto de recolha e uma data de recolha do cliente e ainda uma ACP associada.



De acordo com o segundo modelo de domínio, temos 4 módulos principais: o **ACP** e o **Admin**, que pertencem à mailMoverPlatform, a **Encomenda** e o Estado da **Encomenda**. O ACP representa os ACP’s que vão estar disponíveis para o nosso serviço e são definidos com um id, nome, endereço, email e password. Cada ACP pode ter várias encomendas associadas, funcionando como um destino para as mesmas. Por sua vez, a Encomenda é representada por um id, nome do cliente, um estado de encomenda, que poderá ser “STORE”, “COURIER”, “ACPPOINT” e “COLLECTED”, e irá também ter uma data de saída da loja, uma data de chegada ao ponto de recolha e uma data de recolha do cliente e o seu único ACP associado. Por último, temos o Admin que é responsável por tratar de operações relacionadas com várias encomendas, sendo caracterizado pelo email e password.

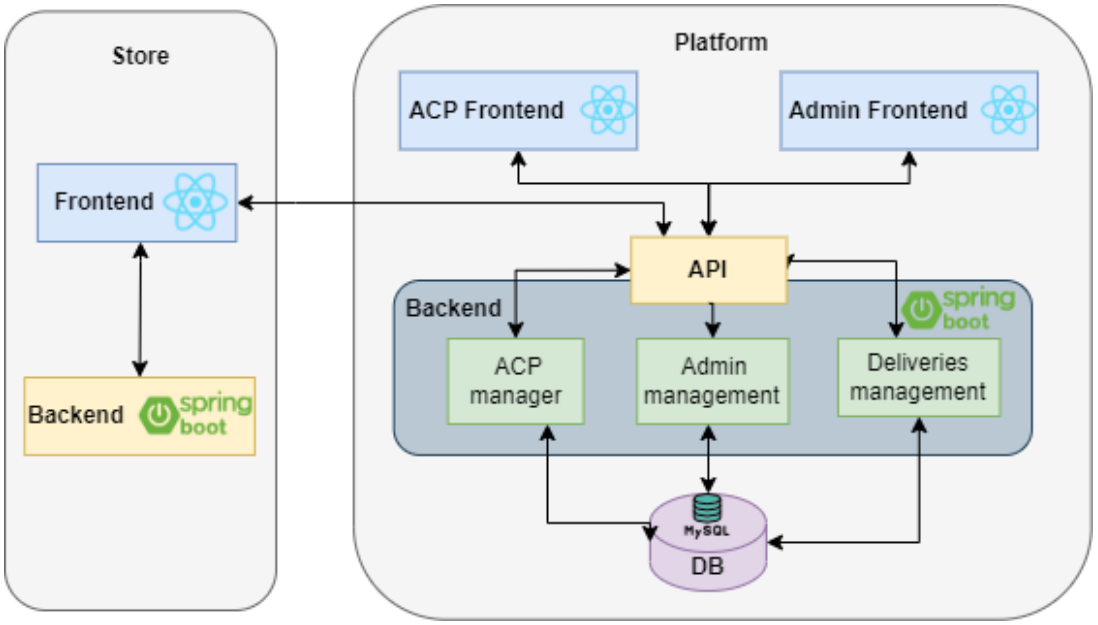
4 Architecture notebook

4.1 Requisitos chave e limitações

- O sistema da *store* e da *platform* (mailMover) devem ser acessíveis através de PC’s;
- O sistema deve ter um bom desempenho, para que nenhum pedido seja perdido;
- Não deve haver substituição de informações para que todos os clientes recebam o que compraram;

- O sistema deve garantir proteção completa dos dados das encomendas, por exemplo, do endereço do ACP associado, para que não haja enganos aquando da recolha por parte do cliente que fez a encomenda;
- O sistema deve ser de fácil utilização, de modo a garantir o conforto dos utilizadores;
- O sistema deve ser rápido e garantir boa performance capaz de suportar muitos utilizadores ao mesmo tempo.

4.2 Visão da Arquitetura



O nosso diagrama de arquitetura retrata os dois principais componentes do nosso projeto: Store e Platform.

Começando pelo componente da Store, desenvolvemos em ReactJs o frontend da mesma devido a fácil utilização. Além disso, realizamos os testes para o mesmo utilizando o Spring Boot (mais precisamente o Selenium IDE). Este frontend irá comunicar, através do uso de endpoints, com uma API que foi desenvolvida também em Spring Boot (localizada já na componente da Platform).

Por fim, a componente da Platform é dividida em duas partes: frontend e backend. Na parte do frontend foram desenvolvidos dois blocos, um frontend para as ACP e outro para os administradores. Estes frontends irão comunicar com o backend através da API referida em cima, tendo esta a função de disponibilizar endpoints para todos os frontends. Além disso, este backend armazena todo o tipo de informação que necessita numa base de dados, desenvolvida em MySQL.

5 API para desenvolvedores

PublicController:

Endpoint	Método	Função Associada
v1/mailMover/new/{clientName}/{acp_id}	POST	createOrder
v1/mailMover/byAcp/{acp_id}	GET	getByAcpId
v1/mailMover/byId/{id}	GET	getById
v1/mailMover/storeToCourier/{id}/{ts}	GET	changeState_STORE_to_COURIER_PUBLIC
v1/mailMover/all	GET	getAllAcps

OrdersController:

Endpoint	Método	Função Associada
v1/orders/new/{clientName}/{acp_id}	POST	createOrder
v1/orders/all	GET	getAllOrders
v1/orders/byAcp/{acp_id}	GET	getByAcpId
v1/orders/byId/{id}	GET	getById
v1/orders/deleteAll	GET	deleteAllOrders
v1/orders/storeToCourier/{id}/{ts}	GET	changeState_STORE_to_COURIER
v1/orders/courierToAcp/{id}/{ts}	GET	changeState_COURIER_to_ACPPOINT
v1/orders/acpToCollected/{id}/{ts}	GET	changeState_ACPPOINT_to_COLLECTED

AdminController:

Endpoint	Método	Função Associada
v1/admin/new	POST	createAdmin
v1/admin/login	POST	login

AcpController:

Endpoint	Método	Função Associada
v1/acp/new	POST	createACP

v1/acp/all	GET	getAllAcps
v1/acp/{id}	GET	getAcpById
v1/acp/login	POST	login

6 References and resources

https://react.dev/learn/thinking-in-react

https://react-icons.github.io/react-icons/icons?name=fa

https://www.youtube.com/watch?v=SqcY0GlETPk&ab_channel=ProgrammingwithMosh