deti universidade de aveiro
departamento de eletrónica, telecomunicações e informática

TQS: Product Specification Report

Alexandre Serras [97505], Gonçalo Leal [98008], João Reis [98474], Pedro Duarte [97673]
v2022-05-20

Index

1 Introduction

1.1 Overview of the project

This project aims to create an online shop and delivery system for Coviran. To add some abstraction we will create two Spring Boot APIs: a generic one and another one Coviran specific. In order to achieve a quality result, we will write tests for our code (implementing Test Driven Deployment) and implement a CI/CD flow.

With our solution, we want to offer a system capable of allowing users to order their groceries online and have them delivered at the location of their choice. Our system should be a good task assignment platform, assigning couriers to an order efficiently.

1.2 Limitations

<explain the known limitations/unimplemented (but planned) features>

2 Product concept

2.1 Vision statement

With the exponential advance of technology, one of the problems of super/hypermarkets today is the need to expand their business into the digital market. One of these companies is *Coviran*, a Spanish supermarket chain, also spread throughout Portugal, that doesn't have this digital side in their business, yet.

Therefore, the goal of this product is to innovate the digital market of *Coviran* supermarkets, providing them with a service to deliver their products to their customers' homes.

The super/hypermarkets companies already have applications with similar goals, for example, the portuguese hypermarkets chain, *Pingo Doce*, provides a mobile application with the goal of listing all products and promotions/discounts available in their markets. The existing application that most closely resembles the purpose of this product is *Continente Online*, another big Portuguese hypermarkets chain, where it is possible, not only to list all the products available and on sale, but also the delivery service is available.

2.2 Personas

2.2.1 Lucília Santos

Lucília is a 74 year old retired woman with reduced mobility from Aveiro, in Portugal. She is a widow, has 1 adult son and uses a cane to help her walk.

She finds it hard enough to go to the supermarket due to her reduced mobility. So, her son usually does her shopping every two weeks at Coviran, the nearest supermarket. She always has the same list of products.

MOTIVATION: Lucilia doesn't want to keep asking her son to go shopping for her. She wants to be independent in that aspect and not worry her son anymore.

2.2.2 André Vaz

André is a 39 year old manager of the Coviran's supermarket in Aveiro, in Portugal. He is married and has one son, one daughter and one dog.

He's a very active and focused person not only in his job, but also in his personal life. He never stops! If he isn't working, he is doing sports or spending quality time with his family.

MOTIVATION: As manager, André wants to control all your delivery's employees, check if everything is working fine and analyze some statistics.

2.2.3 Duarte Henriques

Duarte is a 19 year old student from Faro. Last year, he finished High School, and now he is studying Public Administration at Universidade de Coimbra. He is a person very independent and focused on his future. He has a small van and a rented room in Coimbra.

As an independent student, he wants to start saving money to help his parents pay for university, but he knows that it is not easy to combine studies and work. So, he is looking for a simple job that does not require a lot of resources, only his van and his computer/phone.

MOTIVATION: Duarte wants a temporary job that doesn't need much resources.

2.3 Main scenarios

- Lucília needs some food for her lunch tomorrow, but her son is working. As the list of products is always the same, she wants to save it to order every week.
    - opens the app, logs in, creates a new product list, searches for the products he needs, chooses the quantity of each one, adds them to the list, chooses the list that she wants to order, selects the time we want to receive the order, and finally orders her products.

- Duarte is working for Coviran right now. He is a courier.
    - opens the app, logs in as a courier, receive a delivery order, accept or deny it:
        - If he accepts it, he will receive the list of products and the client's address. In the end, he marks that order as done, and he is available again for another delivery.
        - If he denies it, he waits for another delivery order.

- At the end of each working day, André will check the delivery statistics of his store.
    - opens the app, logs in as a manager, selects his supermarket and checks the delivery statistics on that day.

- Lucília is finishing her lunch and realizes that the dishwashing liquid is empty.
    - chooses the category of the wanted product (in this case, personal hygiene), chooses the product, adds it to the cart, and orders the product.

2.4 Project epics and priorities

[Apresentar um plano indicativo para a implementação incremental da solução ao longo de várias iterações/releases, explicando as funcionalidades a atingir por *epics*]

3 Domain model

Como se pode observar pela imagem X, o sistema de entregas vai ter a sua base de dados o mais genérica possível e onde apenas lhe interessa ter aquilo que é extremamente essencial para um serviço de entregas , ou seja, a encomenda, quem fez a encomenda, o estado da encomenda e o estafeta responsável por esta encomenda, sendo por isso que apenas existem 4 tabelas, sendo estas : Delivery , DeliveryStatus , Provider e Courier.

No que toca a informações específicas da encomenda nada disso vai ser armazenado aqui, pois isso vai variar de loja para loja sendo que a ideia principal é que seja possível que várias lojas usem este sistema daí pretender-se ter o sistema de entregas o mais genérico possível.

Decidiu-se que vai ser utilizado MySQL para a base de dados.

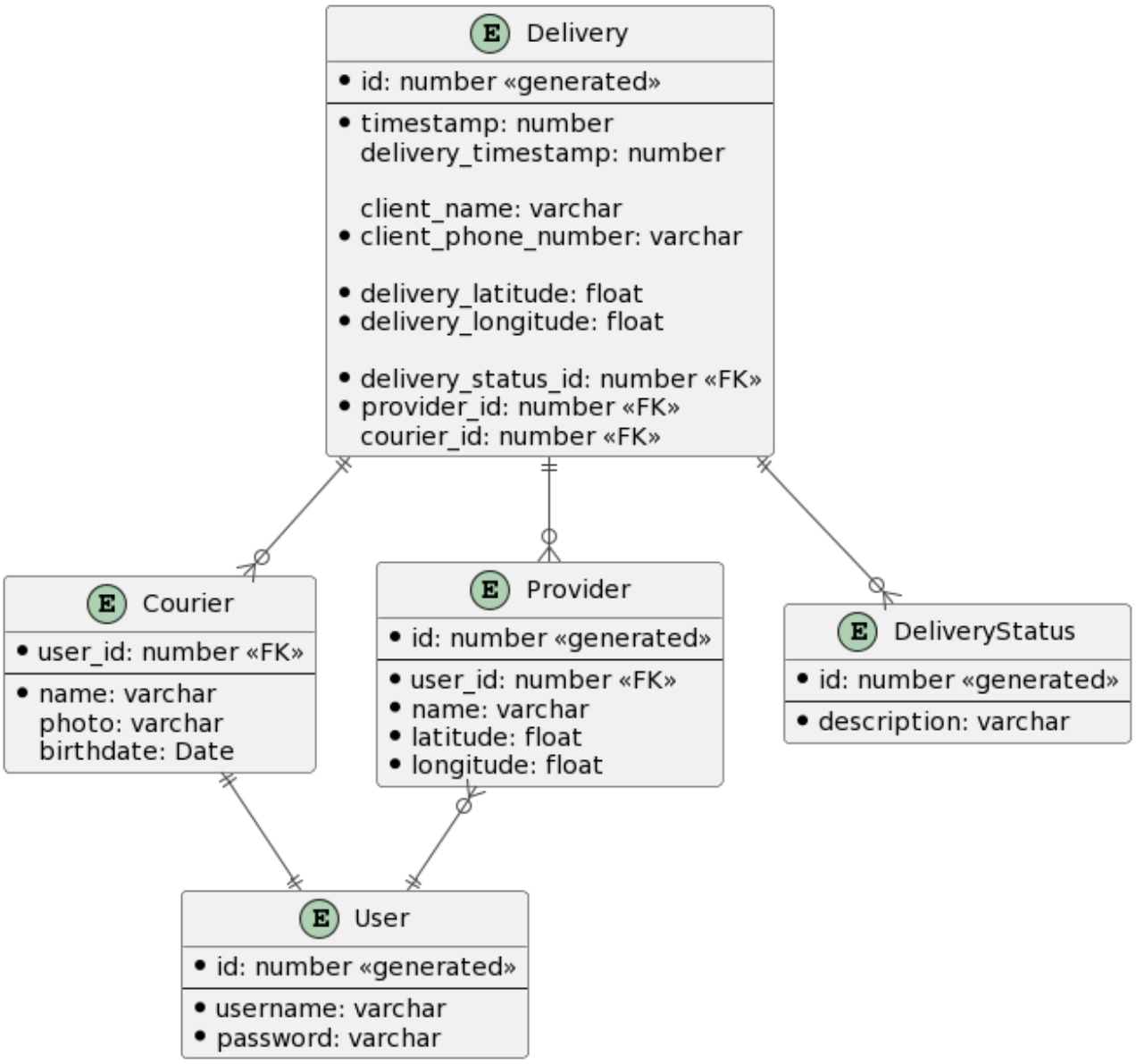


Figura X , ER da DB do sistema de entregas

Olhando agora para a imagem Y, consegue-se observar o modelo de dados e como tudo interage entre cada tabela escolhida para a nossa loja, ou seja, a app específica.

Foram criadas cerca de 12 tabelas, para fornecer tudo aquilo que a nossa loja precisa para funcionar corretamente.

Também deste lado vai ser utilizado MySQL para a base de dados.

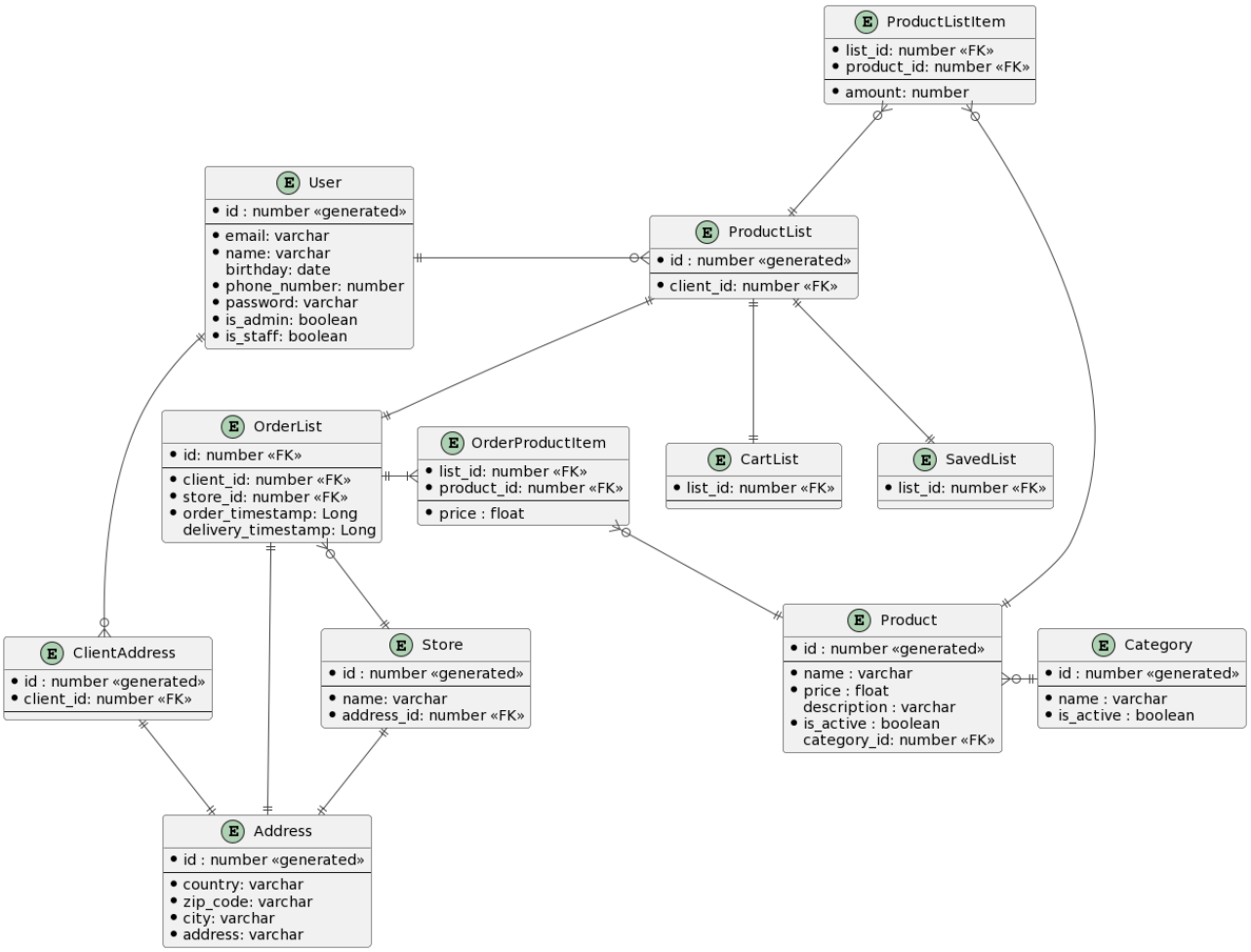


Figura y , ER da DB da loja

4 Architecture notebook

4.1 Key requirements and constraints

Requisitos e restrições do Sistema de Entregas:

- O sistema de entregas pretende-se que seja possível integrar com qualquer loja não apenas com a aquela que vai ser desenvolvido por nós
- Para se ser estafeta é necessário o registo na aplicação
- Como se estão a lidar com dados pessoais vai ser obrigatório autenticação tanto dos estafetas como dos administradores

- Plataforma para os estafetas aceitarem os serviços, e onde os vai informar do que vão ter de transportar e para onde
- Plataforma para os administradores vejam as estatísticas do sistema
- Como a API vai ser pública, informações confidenciais do sistema não podem ser passadas nos pedidos à respectiva API

Requisitos e restrições da Loja:

- Obrigatório utilizar a API do Sistema de Entregas para pedir entregas
- Autenticação para fazer compras na Loja

4.2 Architectural view

A solução de software pensada para o problema colocado divide-se em 2 grupos como se pode observar pela Figura Z, Generic App e a Specific App.

Para as bases de dados vai ser sempre utilizado o MySQL, backend era o único local onde havia um requisito obrigatório da tecnologia que era o spring boot, nas Web Applications vai ser utilizado o Angular e na mobile app o React Native foi a tecnologia escolhida.

Todas as tecnologias foram escolhidas livremente pelo grupo, a escolha deve-se essencialmente em todas elas, a serem aquelas que os membros do grupo tem mais conhecimento e onde apenas no React Native como apenas 1 elemento tem experiência com aplicações mobile sugeriu esta e o grupo aceitou.

Ao nível dos workflows que o sistema vai precisar vão ser mais fáceis de entender através dos diagramas de sequência contudo a principal dificuldade está na ligação da loja específica para com a genérica visto que se pretende que qualquer loja consiga utilizar a app genérica.

O backend vai interagir com as DB's através do data access que esta primeira camada têm e o frontend vai sempre interagir com o backend através da layer da API, por último a forma como as 2 aplicações vão comunicar é sempre através do backend de uma e de outra, e assim desta forma parece-se que conseguimos da melhor forma fazer a comunicação entre os diferentes componentes do sistema.

O fluxo habitual de como o sistema funciona pode ser visualizado pelo diagrama de sequência, Diagrama QQ, que se encontra abaixo.

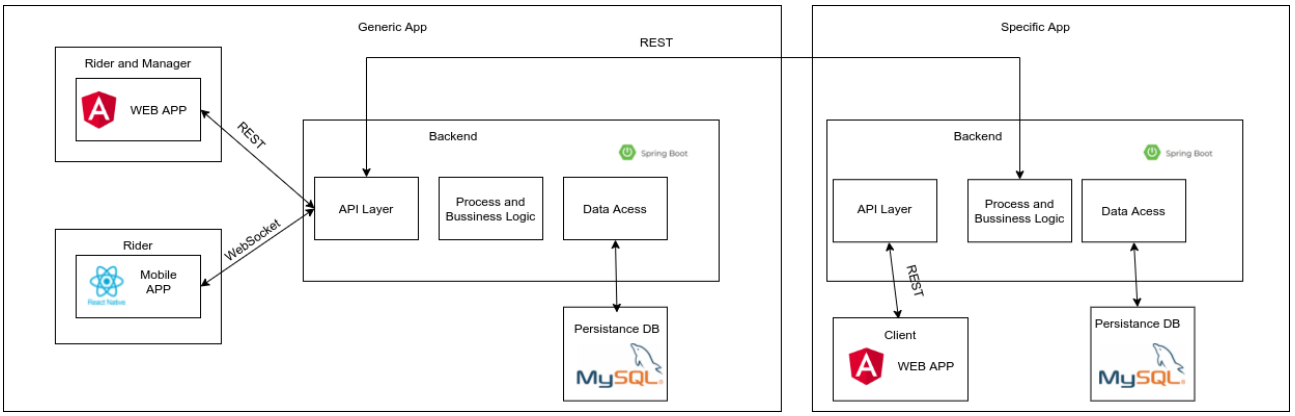


Figura Z, Diagrama da arquitetura do sistema

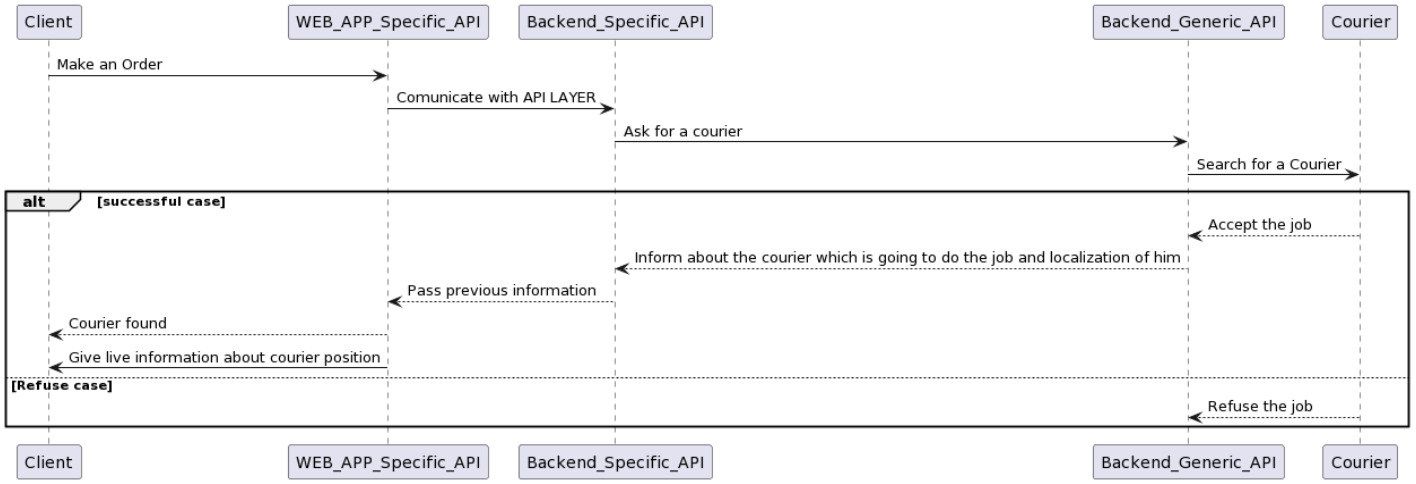


Diagrama QQ, Diagrama sequência de como é feita a comunicação entre as diferentes componentes do sistema

4.3 Deployment architecture

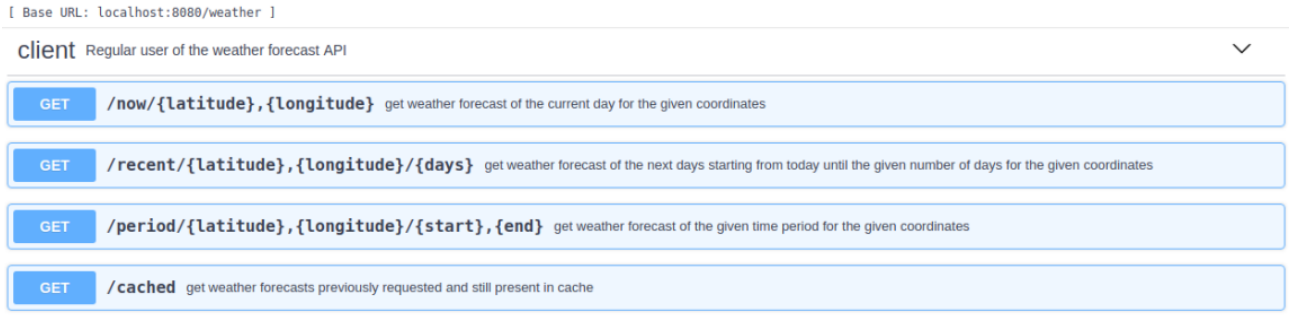
Our code is managed through github, where the code is divided in different repositories which belong to the same organization. Using github we are able to implement our CI/CD flow, namely through github actions and github agents which allow running tests and validations and deploy our solution.

We have a development deployment to have the opportunity to validate if the result is the expected. This deployment is made in the same way as the production one.

The code will be then deployed through several docker containers.

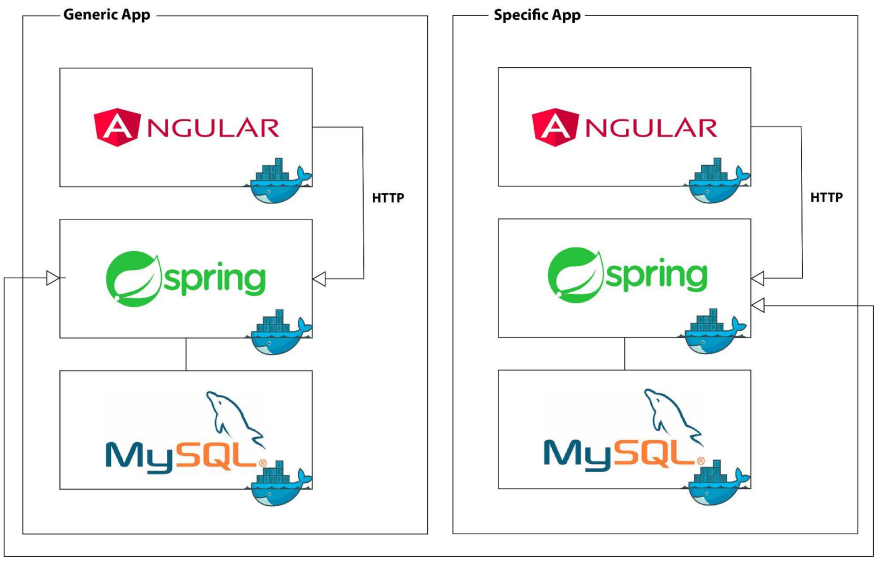
5 API for developers

[Explicar a organização da API. Os detalhes detalhes/documentação dos métodos devem ficar numa solução *hosted* de documentação de APIs, como o Swagger, Postman documentation, ou incluída no próprio desenvolvimento (e.g.: maven site)
<what services/resources can a developer obtain from your REST-API?>
<document the support endpoints>



6 References and resources

<document the key components (e.g.: libraries, web services) or key references (e.g.: blog post) used that were really helpful and certainly would help other students pursuing a similar



work>