deti universidade de aveiro departamento de eletrónica, telecomunicações e informática

# TQS: Product specification report

***Diogo Cunha [95278], Pedro Tavares [93103]***
***José Carlos [87456], , Juan Lessa [91359], Gonçalo Pereira [93310]***,
v2021-05-28

**1 Introduction** **1**
1.1 Overview of the project 1
1.2 Limitations 1

**2 Product concept** **2**
2.1 Vision statement 2
2.2 Personas 2
2.3 Main scenarios 2
2.4 Project epics and priorities 2

**3 Domain model** **2**
**4 Architecture notebook** **3**
4.1 Key requirements and constraints 3
4.2 Architectural view 3
4.3 Deployment architecture 4

**5 API for developers** **4**
**6 References and resources** **4**

## 1 Introduction

### 1.1 Overview of the project

The project's context is inserted in TQS's (Testes e Qualidade de Software) which encourages the use of TDD (Test Driven Development) meaning that the project's development process should be done in accordance with the TDD methodology.

With that being said, our application will be called Biodrop and will be an online platform to make orders for groceries by associating stores and their products to drivers that will receive delivery requests from customers that cannot or do not want to go to the store and do the groceries.

By using our application a store representative should be able to add the available items to the online store, so that the clients know what items are available. When the client requests drivers are

notified that there is a new order request, giving the driver the choice to accept or deny the order request.

Will be possible for people to register as clients who want to make orders or as a driver who wants to make deliveries.

One similar product is UberEats, which served us an inspiration, an application capable of offering orders and stores of multiple qualities, and capable of handling and incredible delivery capacity.

### 1.2 Limitations

Right now there are some unimplemented features that we think would improve our application if they were available.

The drivers don't have the option to accept or decline an order, which was something we wanted to implement. The riders are assigned the orders and they can't decide whether to deliver them or not.

The clients don't get notified that their orders have arrived.

## 2 Product concept

### 2.1 Vision statement

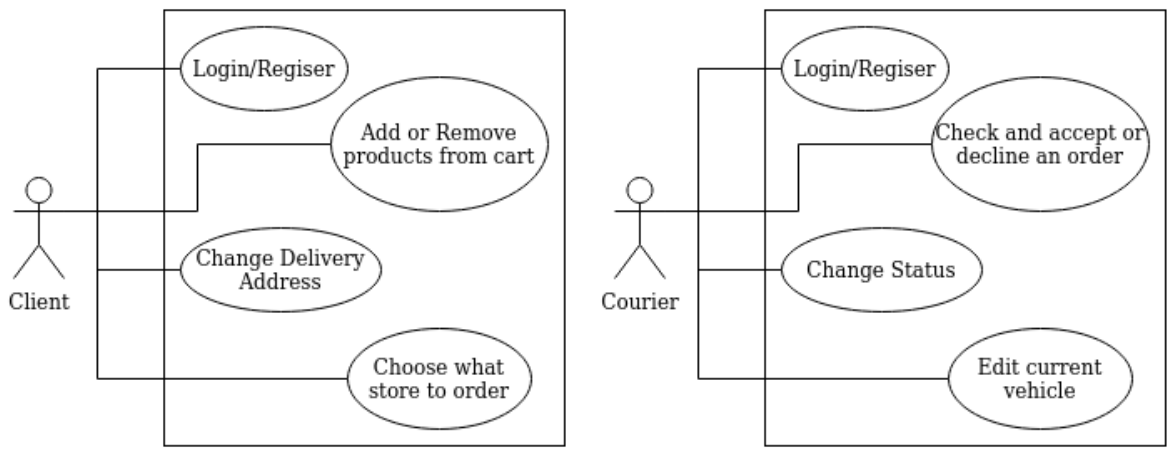
Our product will be divided into two blocks. One block is the drivers' block (Deliveries Engine) and the client block (Business Initiative). Drivers who want to make deliveries should register in the application as a driver and then will be able to accept or deny jobs when they are requested.

The client should also register itself, this time on the client-side of the application enabling the possibility of making orders to local grocery shops from the brand.

When it comes to stores, store employees should have access to a specific administrative part of the platform where it is possible to add store locations, edit the available items in-store, and check its activity through the online store.

With this project, we aim to provide a better shopping experience for everyday grocery shopping by having the groceries delivered to the customer's home by a courier. Meaning that by charging the client an additional fee for the delivery calculated through the distance the courier must cover to conclude the delivery.

This fee is the courier's reward for the job, which encourages couriers to accept more requests to work more, hence making more money.



## 2.2 Personas

### Worker

### Persona 1

Rui was born in Portugal. He is a 20 years old student. He is currently studying Informatic Engineering at Aveiro University. In his spare time Rui enjoys spending time with his family and friends. Since he is currently studying, he doesn't have much free time to work. So he  is looking for an easy way to make money and have a job with an open workload that can be administered together with his university.

**Motivation:** Rui is looking for a job that is quick and easy, without a schedule obligation.

### Client

### Persona 2

Joana was born in Portugal. She is 28 years old. Joana currently works in a bank as an accountant and would like to be promoted to manager. Joana enjoys spending time with her family and taking care of them. As she doesn't have that much free time to shop and doesn't like either, she would like to find some easy way to buy at grocery stores in the comfort of her home.

**Motivation:** Joana is looking for an easy way to shop groceries.

### Owner

### Persona 3

João was born in Portugal. João is 40 years old and owns a grocery store. João would like to expand his business and not have to invest more, in recent time he hasn't had a good audience either. He is now looking for a cheap and fast method to attract more customers.

**Motivation:** João is looking for a way to invest in his grocery store and gain more customers.

## 2.3 Main scenarios

**Rui**, as a **courier**, will enjoy working in his free time while studying. He will log into the app and change his status to available. Then he will receive orders and will be able to accept them. Once he

accepts the order will receive directions to complete the job. After the job is completed the status of the driver will be available again

**Joana**, as a **client**, creates/logs into her account to be able to get her groceries straight at home. To do so, she fills in the client registration form and is then able to place her orders and receive them home. She searches for the items she wants to be delivered and adds them to the shopping cart. Finally, confirm the purchase and wait for the products to arrive home.

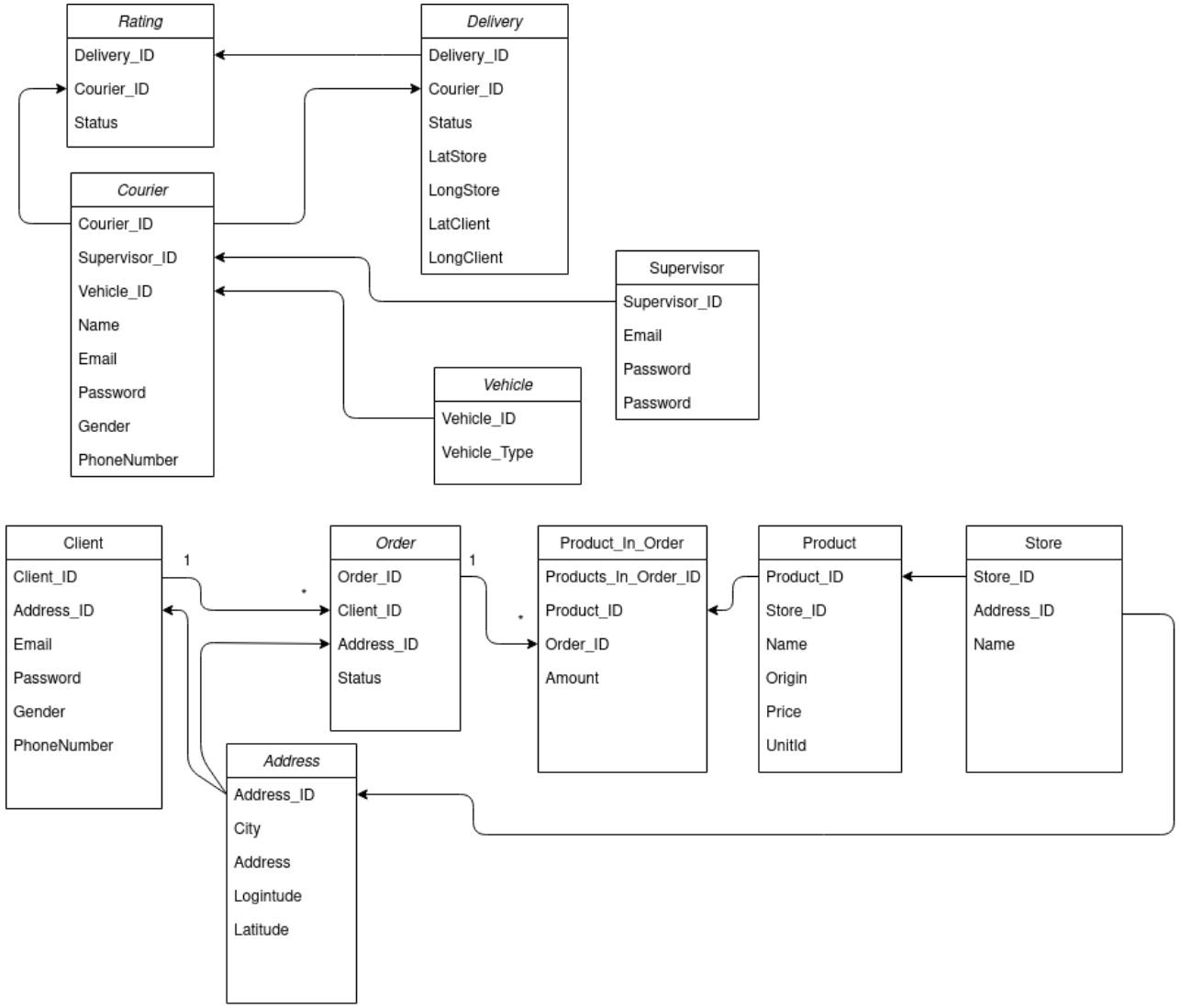
### 2.4 Project epics and priorities

Epics considered:

- User interface prototype
- Platform login and registry ( riders and clients )
- Creation of orders
- Distribution of work to riders
- Accepting orders
- History and statistics page
- Creation of shop store
- Search Stores
- Distribution of stores and products inside all of them

## 3 Domain model

Concerning the system model, there are two major classes: the Client and the Courier. The user can be a client, to buy groceries, or can be the courier to deliver groceries.. Clients can order different types of groceries and set their quantities. Couriers can deliver differente deliveries and set their attributes.




# 4 Architecture notebook

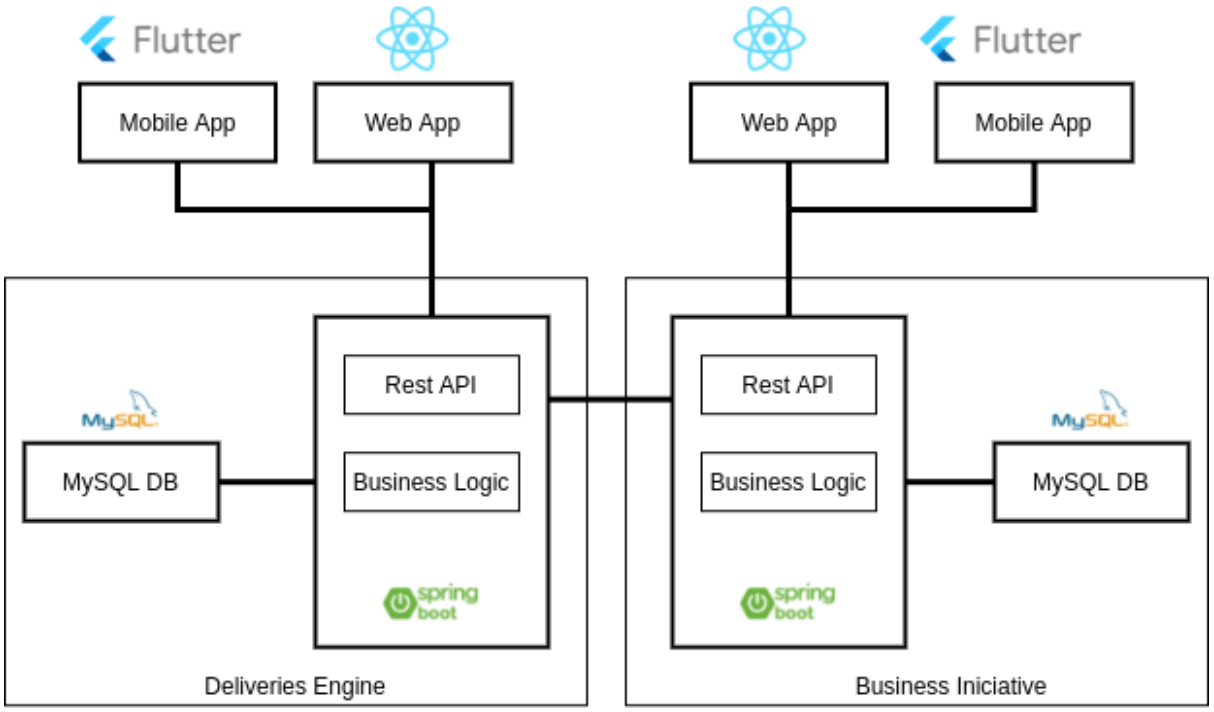
## 4.1 Key requirements and constraints

- Availability: The system must be highly available with almost 100% percentile guarantee. Loss of availability has a direct impact in our business and may result in financial losses.
- Consistency: Data needs to be across all regions, the system needs to be available in several places.
- Data Freshness: Most of our use cases require products to be delivered quickly and all the time. This requirement is critical, most of all since it ensures the ability to respond to certain events such as demand-supply skews, business metric alerts and security incidents.
- Scalability: in essence, scalability reflects in our organization's ambition to grow and the need for solutions to support the growth. In our scenario, it is to increasingly support the high number of deliveries.
- Cost: Our company needs to be a low margin business. It is necessary to ensure low data processing costs and serve to maintain a highly operation efficiency.

- Flexibility: Only possible with programmatic as well as declarative (SQL like) interface, all of that to ensure the diverse user groups our company supports good products.

The system can be accessed through an web interface, interacting directly to the service layer, making use of the API provided that will interact with the service layer. Concerning the API use and the way our system interacts with it, documentation for the endpoints are provided and available(see point 5). Docker containers were used to make continuous deployment.

### 4.2 Architectural view




This is our project architecture having 2 separate parts, the business and the delivered engine.
Flutter wasn't implemented but we have 2 websites that were implemented with react.
The two webapps interact with both api's making requests for both of them.

### 4.3 Deployment architecture

Our application consists of two Springboot applications and 2 web front-ends (reactjs). For deployment we use NGINX to serve the 2 fron-ends. In order to create a continuous delivery flow we created a github actions workflow to, whenever the main branch is updated, build the frontend and update the files served by NGINX. We also keep 2 mysql docker containers which are used by the backends.

## 5 API for developers

# 6 References and resources

https://material-ui.com/pt/

https://docs.mapbox.com/

https://react-bootstrap.github.io/