Product specification (Draft)

Version	Date	Author	Description
0.1	23 May 2022	Rodrigo Lima	Draft Document - placeholders and Qourier specifications
0.2	28 May 2022	Afonso Boto	Added Story board for LaundryAtHome
0.3	29 May 2022	Afonso Boto	Added Story Board for LaundryAtHome view status of order
1	23 June 2022	Tomás Candeias	Success criteria, testing, updates and maintenance

Contents

1. Introduction

This document has the goal to capture the main project requirements and considerations. It tries to go into detail about every aspect of the project.

Every change to the document must be consented by the consensus of the working project group.

2. Project Objectives

The project is composed of two sub projects. The first sub project is called Quorier and its being built with the goal of bridging the independent every day courier and big companies who need something to be delivered on a small scale.

The project will have four key deliverables:

1. Prototype project, focused on UX and story driven (26/5)
2. Core stories implementation and CI integration to backend (02/6)
3. API final implementation, CD pipeline integration and QA Manual final version (09/6)
4. Stable MVP, deployment of project on the cloud, final product specifications report (17/6)

3. Success Criteria

Quorier

- Register and Login made in an intuitive way for the customer and rider, and good management for account status (Pending/Active/Refused/Suspended)
- Dynamic bidding system: if more than one rider has an interest in order, within 10 minutes, the one that is more close to the location is the one that gains the bid. WebApp riders have priority compared to mobile app riders.
- The rider that gains the bid has to be notified when the delivery is assigned.
- If no one has an interest in the delivery, the delivery is deleted.
- When a rider is delivering a package the tracking has to be possible with a pub/sub system
- The admin page can manage accounts and see the system status

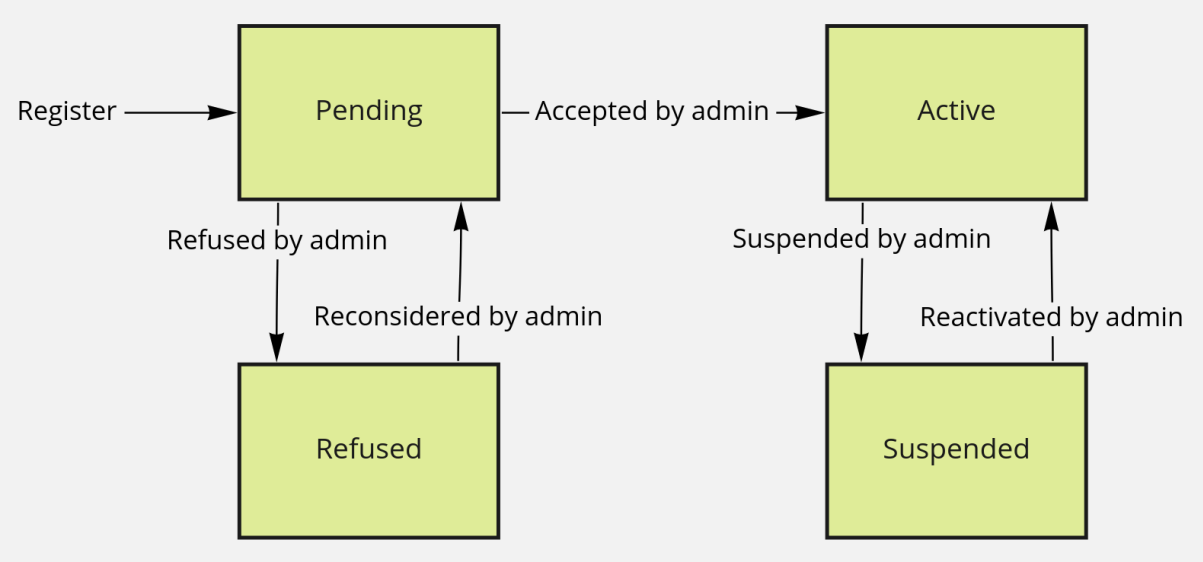
LaundryAtHome

- Register and Login using JWT cookies
- User can choose between four different types of service
- In a specific service he can make an order by putting the address, and by adding items
- To add items he has to be able to specify the type of item, the color and finally the number of items associated to that specification
- When the user clicks add he can see those items being added to the order
- When he finally makes the order he sees a pop-up and then he is redirected to the Orders page
- In the orders page, he can see all his order and the status, he can see the details by clicking “see details” on the specific order he wants to know
- On the details page the user can see the Date he made the order, the state of its’ order, the delivery location address and finally the orders’ total cost
- Finally the user can add a complain to the order he is seeing, by clicking add complaint, here he chooses the title and writes a description and finalizes it by clicking ok.

4. Functional Specification

The sub project Quorier will have the following functionalities:

- Register delivery jobs and manage status
- Check delivery progress
- Register account as rider or customer
- Log in into the account and be redirected to the right page, this will be done by checking the account role and status on the database
- Manage Riders and Customers account status:



- Check account details
- Check platform performance and API metrics
- Generate scores for riders with efficiency and customer rating into account
- Biding system will decide on who will deliver the package by first seing who is closer or if the position isn't given or in case of a stale mate it will be decided by the highest ranking, in case of the same ranking it will be random.

5. Technical Specification

The sub project Quorier will have the following technologies:

- Spring for website copntroller;
- Thymeleaf for website templating;
- Grafana for graph information;
- Rabbitmq for pub sub status information;
- MySQL for the database;
- Swagger for API documentation;
- Prometheus for metrics broker;
- Android application built in kotlin for the riders application;

6. Wireframes/storyboards

The sub project Quorier will have the following personas:

- Alberto (administrator)
- Diego (rider)
- Cristina (concrete delivery service)

Diego user stories:

- Bid a delivery job
  - Visual feedback with "activated" theming
  - Registered the Rider's bid in the database
- Accept a delivery job
  - Bid operations restricted
  - Visual feedback with "accepted" message
  - Registered as the Rider for the job in the database
  - Delivery job not up for bidding
- Confirm a delivery job
  - Registered the delivery job as done
- Check profile and stats
  - Is in profile page, with the details inputted at registration
  - Contains a section with statistics about, at least, the number of deliveries done and average time spent

Cristina user stories:

- Register delivery
  - Delivery up for bidding
- Check progress of deliveries
  - Contains a table with all deliveries requested, in progress and completed
  - Deliveries in progress contain a progress bar

- Check profile and stats
  - Is in profile page, with the details inputted at registration
  - Contains a section with statistics about, at least, the number of deliveries requested and time taken until delivery completion
  - Contains the API key for the operations, which is hidden by default

Alberto user stories:

- Check Riders' progress
  - Has a table with all riders that accepted a delivery and their progress, with a progress bar
- Check each Rider's performance
  - The Rider has a section with statistics, such as the number of deliveries done and average time spent
- Check each Rider's profile
  - The Rider has a profile with all details inputted in their registration
- Check each Customer's stats
  - The Customer has a section with statistics, such as the number of deliveries requested and time taken until delivery completion
- Check each Customer's profile
  - The Customer has a profile with all details inputted in their registration
- Accept/Refuse Rider/Customer applications
  - There is an Applications page with two tables, one for Riders and another for Customers, where each row is a concrete application
  - Clicking on an application summons a modal with details of the application, with two buttons from which the application can be accepted/refused
  - Contains per-table filtering options for, at least, Accepted or Refused applications. Only accepted applications are shown by default
- Suspend/Reactivate accounts of Riders/Customers
  - Within the Rider/Customer profile, there should be a button to suspend/activate their account

    - Suspending an account disables all operations from the account
    - Activating an account enables all operations from the account
- Check platform performance and number of API requests
    - Check platform performance metrics in graphs
    - Check the rate of API requests in graphs

The subproject LaundryAtHome has the following personas:

- Jonas(Customer)
- Sofia (Customer

Jonas’ user story:

Background:

- He doesn’t have a laundry room at home.
- He has colored shirts to laundry
- He has white jackets to laundry

- Request Laundry Service
    - He selects the service “wash and laundry”.
    - he chooses the type of item , and finally
    - he chooses the color whether it is white or coloured
    - he chooses the number of items that correspond to the description.
    - He does the same for the white jackets
    - He clicks make order

- Finalising order
    - To finalize the order he chooses whether he is going to pick it it up himself or get it delivered at home.

    - In this case he chooses to get it delivered at home
    - He chooses the address of where he wants it delivered
    - He finalizes the order by adding the payment method

Sofia’s user story

Background:

- She is a recurrent customer
- She made a couple of orders a few days ago
- She wants to see the status of a order made to her home

- See previous order
    - She goes to her account and clicks on her orders
    - On the list provided she clicks on the one she is curious about
- See status of the order
    - On the specific order tab she can see the details of that order
    - That details contais the date of which she placed the order
    - It contais the latest status update (date and status)
    - It also contais the estimated time until it’s delivered

7. Testing

There were many tests made on the Delivery Platform side and the PoC side as well.

Both web apps have different kinds of features and each one scenarios for the functional testing, made with selenium and cucumber.

All the repositories used for manipulation of the data were tested

Unit tests were made for the controllers and the services

Integration Tests were made for the controllers on both apps.

MessageCenter tests were made as well, but in this one, a RabbitMQ instance has to be running.

8. Updates and Maintenance

For incompatibility reasons the Swagger had to be removed, and therefore our both API, delivery and laundry platform, don't have proper API documentation.

Many adjustments were made in the development of the CI pipeline.