TQS: Product specification report

André Butuc [103530], Artur Correia [102477], Daniel Carvalho [77036]
v2023-06-05

1 Introduction

1.1 Overview of the project

This project was developed within the scope of the course *Software Testing and Quality Control* to put into practice the Quality Assurance, Test-Driven Development and CI/CD strategies learned in the practical classes.

In this assignment we developed an application named DropMate whose goal is to manage a network of associated collection points (ACP) and parcel lockers and provide a withdrawal B2B service to e-Stores.

The full-stack application DropMate consists of a REST API and a web application to be used by internal administrators and associated collection points operators. In addition, we developed an e-Store web instance of an online flower shop called Floral Fiesta as a proof-of-concept demonstrator of the full-stack application.

1.2 Limitations

Due to time constraints some features were not implemented or were partially implemented. As future work, we would improve the design of the user interface to make it more intuitive and improve our notification pipeline. Additionally, we would develop Cucumber tests to properly validate our user interface and user story implementation.

2 Product concept

2.1 Vision statement

Nowadays, pick-up points services are closely related to the main-stream and well-known courier services, serving as sub-systems. However, this fact limits potential partnerships with small stores and shops that are on efficient locations for pick-ups but are not seen as relevant to the big courier services. We are committed to develop a flexible solution that allows physical stores to approach us and make pick-up deals as well as allow online stores to solicit our services. This way we are thriving to achieve an efficient network of pick-ups that bring the purchased products closer and faster to the end users.

2.2 Actors and Use Cases

While doing the requirements gathering for the project, three main actors were identified for the systems to be developed:

- **E-Store Client:** The user of the Floral Fiesta E-Store. This user will browse the E-Store catalogue of products, being able to place orders and track their status.
- **DropMate Admin:** An administrator for the DropMate platform. This administrator will be able to manage both the network of ACP partnered with DropMate, as well as the network of E-Stores that use DropMate as a pickup points platform. They will also be able to check operational statistics of the platform.
- **ACP Operator:** An operator for one of the ACP's partnered with DropMate. They will use a special UI developed for their ACP, in which they can manage the parcels being delivered to their pickup point.

For each of these main actors, simple narrative-driven use cases were created. These use cases were used to decide the functionalities developed and implemented for the Floral Fiesta, DropMate and ACP systems, as well as the basis on which the user stories and epics for development were defined.

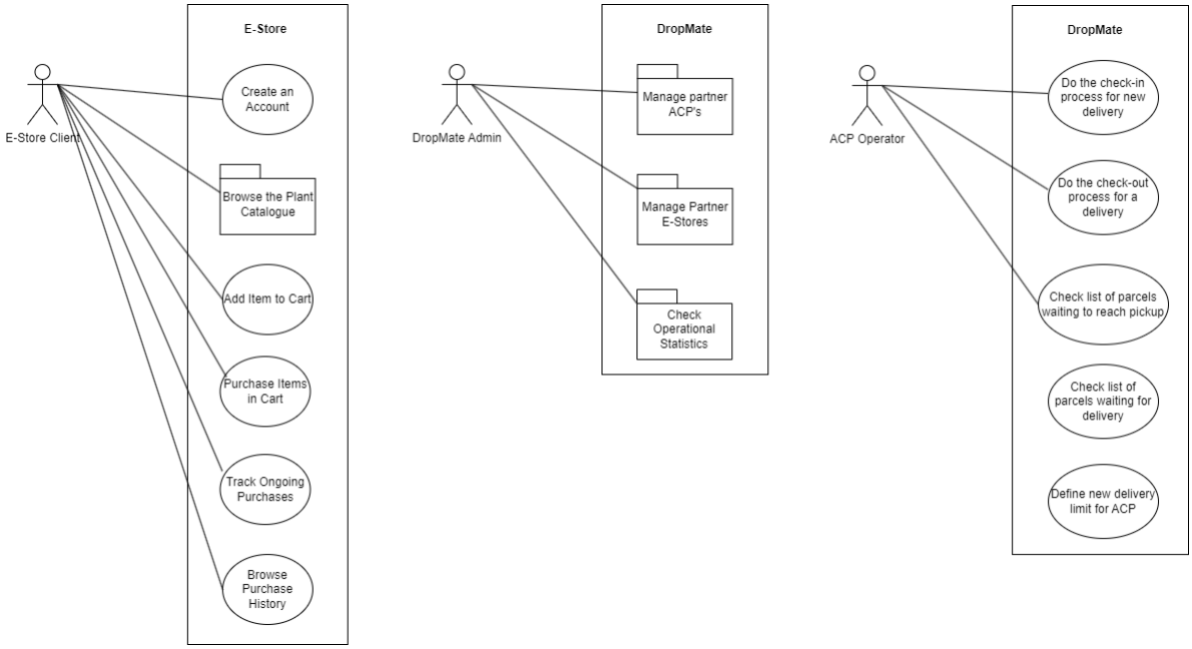

Figure 1 – Use case diagram for the project implementation.

The use cases for the E-Store client are further detailed on Table 1.

Table 1 – Details of the identified use cases for the E-Store client.

Use Case	Description
Create an Account	The E-Store Client creates an account in the platform, in order to be able to buy new items, track their shipping status, or check the history of shipments made.
Search for a Specific Plant	Search for a specific plant available in the store.
Browse a Category of Plants	Browse the plants associated with a given Plant Family (Category). The categories should be present in a tab on the home page of the store.
Add Item to Cart	After finding a plant that is of interest to the user, they add the plant to the shopping cart. They can add as many instances of the plant as possible. The items will appear in the shopping cart.
Purchase Items in Cart	After browsing the store and finding all of the items of interest, the user will complete the buying check-out in the shopping cart. Afterwards, they must input their addressing and payment information, and choose one of the delivery pick-up points currently available for the store. Once the purchase is complete, a shipment ID is generated, that will allow the user to identify as the owner of the package when picking it up in the ACP.
Track Ongoing Purchases	After placing an order, users will be able to check the current status of their orders in a dedicated page. Orders currently being sent to the chosen pick-up delivery point will appear with the state “sending to pickup”. Orders already checked in at the ACP will appear with the state “waiting for pickup”. Once the user physically picks up the shipment, using the ID code generated at purchase, the order will disappear from this tracking page.
Browse Purchase History	After an order is shipped and picked up, the orders and its details become available in the history tab of the user account.

The use cases for the DropMate Admin are further detailed on Table 2.

Table 2 - Details of the identified use cases for the DropMate Administrator.

Use Case	Description
Browse the ACP’s Registered in the Platform	The DropMate administrator can browse the list of ACP partnered up with the service, as well as their details.
Review a Request for a new ACP	The admin can review the request by a new store to partner with DropMate as a new pick-up point. They can either accept this request, adding the pick-up point to the database, or reject it.
Remove an ACP	The admin can remove an ACP from the platform.
Browse the E-Stores Associated with the Service	The admin can browse the list of partner e-stores using DropMate’s services to manage pick-up points.
Check the Number of Deliveries by an ACP	The admin can see the number of parcels processed by each ACP. This includes the number of completed shipments,

	shipments waiting pickup, and shipments yet to be delivered to each ACP. With this, the system admin can check which pickup points are overcrowded and at their activity limit, and which points have seen less request traffic.
Browse the Parcels Waiting for Delivery	The system admin can browse the full list of parcels being delivered to a pickup point.
Browse the Parcels Waiting Pickup	The admin can brose the full list of parcels waiting for pickup by their buyer on an ACP.

The use cases for the ACP Operator are further detailed on Table 3.

Table 3 - Details of the identified use cases for the ACP Operator.

Use Case	**Description**
Do the Check-In Process for new Delivery	When a parcel reaches the ACP, the Operator does the check-in process, notifying the DropMate service that the delivery has reached the pick-up-point, using the ID for the parcel. The state of the shipment should be changed to “waiting for pickup” in the DropMate service, and this change should be reflected in the page tracking the ongoing purchases of the Client, on the E-Store website.
Do the Check-Out Process for a Delivery	Once a client comes up to pick-up their delivery, the ACP Operator can initiate the check-out process. In case the client supplies the correct code corresponding to the parcel, registered on the DropMate backend, the delivery is concluded. In case the code is incorrect, the ACP Operator must stop the check-out process.
Check List of Parcels Waiting to Reach Pickup	The ACP Operator can check the full list of parcels already shipped by their store, but that haven’t yet reached the ACP. They should be able to check their details, such as shipment date, expected delivery date, ID, among others.
Check List of Parcels Waiting for Delivery	The ACP Operator can check the full list of parcels waiting for delivery, that is, the parcels that have reached the ACP but that haven’t yet been picked up by their client. They should be able to check their details, such as shipment date, the date when the order was received, ID, among others.
Define new Delivery Limit for ACP	Each ACP will define a number maximum parcels waiting delivery at a time, with this number being managed by the ACP Operator. If an ACP has reached it’s limit, the DropMate backend will not allow a partner store to send shipments to this ACP, until it’s orders have been picked up.

2.3 Project epics and priorities

In order to aid on the development of all the required systems for this project, the team defined 3 project epics that were used to guide the organization of the user stories on the backlog management platform:

- Development of the Floral-Fiesta E-Store – This epic encompasses all of the user stories required for the development of the E-Store interface. This includes not only the UI, but the backend supporting the store, and which will interact with the DropMate backend via an Http Client.
- Development of the DropMate Admin UI – Contains the user stories related to the UI for the DropMate administrator. This also includes the tasks for the development of the DropMate backend methods related to the information presented and use cases created on this UI.
- Development of the ACP Operator UI – Contains the user stories related to the UI for an ACP operator. This also includes the tasks for the development of the DropMate backend methods related to the information presented and use cases created on this UI.

As previously stated, user stories were created based on the use cases described on the previous section. These user stories followed the traditional guidelines for the writing of user stories, including the identification of the actor, what they will do when interacting with the system, and the results of their interaction with said system. Besides, acceptance criteria was written for each story. This helped in guiding the development of the project, both when it comes to the backend and frontend implementation, but also the related tests.
Figure 2 presents an example of a user story. The rest of the user stories written for the project can be found here.

Use case	User stories	Acceptance criteria
Do the Check-In Process for new Delivery	**ACP Operator checks in a new parcel at the ACP** **As** an ACP operator, **I want to** be able to check in new deliveries for the pick-up point, **So that I** can notify the DropMate service that the delivery has reached the pick-up point **And** update the state of the shipment for the corresponding client on the E-Store website.	Scenario 1 Given that I'm a logged in Operator And that I am on the "ACP Management" page And that "Test Parcel" is on the "List of Parcels Waiting Delivery" table Then I should be able to see the "Parcel Check-In" form And when I complete the form for the parcel "Test Parcel" Then "Test Parcel" should appear on the "List of Parcels Waiting Pickup" table And "Test Parcel" should disappear from the "List of Parcels Waiting Delivery" table • The operator should be able to input the parcel code on the form. • The system validates the ID against the database of pending deliveries and returns an error message if the ID is not found. • The operator is prompted to confirm the check-in process. • Upon confirmation, the DropMate service is notified of the delivery status change to "waiting for pickup". • The E-Store website is updated to reflect the new status of the shipment for the corresponding client.

Figure 2 – Example of an user story written for the "Do the check-In process for new delivery" use case.

When it comes to prioritizing work, the first user stories to be completed where the ones directly related with the flow and business logic of the process, such as the tasks related to the buying of new parcels by an E-Store client on the store service, and the related check-in and check-out operations done by the ACP Operator when receiving and delivering these packages. However, all of the created user stories and use cases were planned to be ready by the time of the first release (when the project was presented in class).

3 Domain model

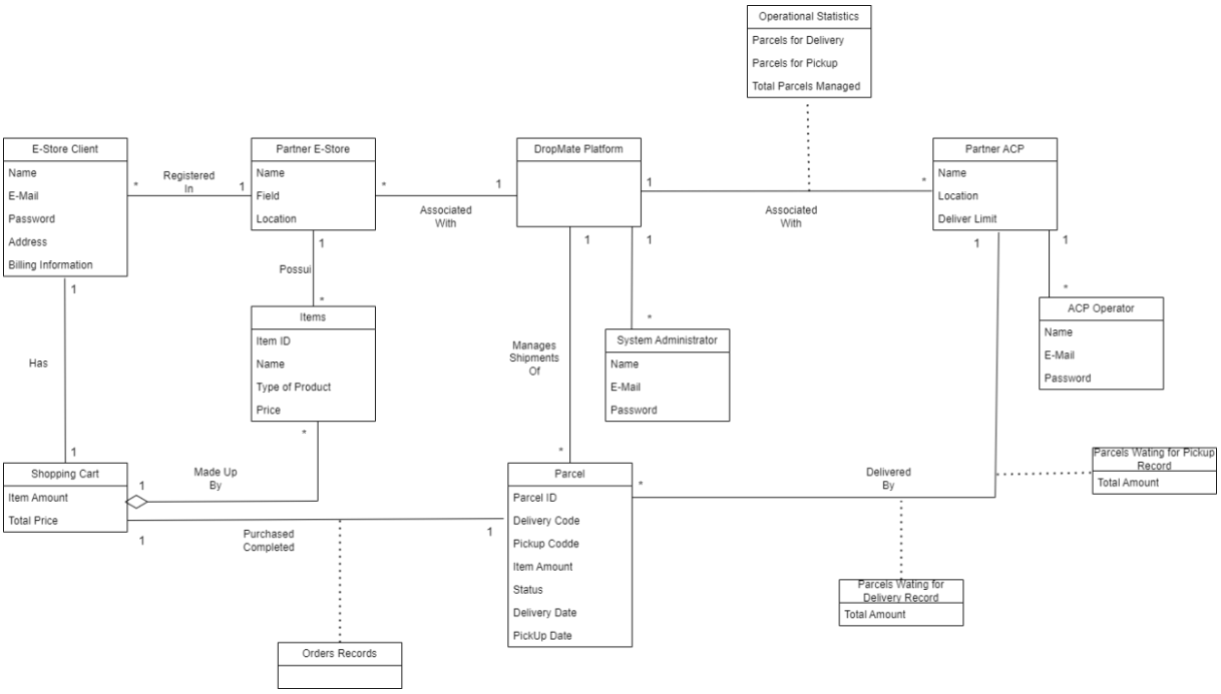

Figure 3 – Domain Model of the system.

The central entities comprising the information domain model of the system are the **Partner E-Store**, **DropMate Platform**, and **Partner ACP**, which represent the E-stores partnered to the DropMate service, the DropMate service itself, and the partner ACPs serving as associated pick-up points, respectively.

Associated with the E-Store is the **E-Store Client**, representing the clients with an active account on the platform of an e-store. These clients can browse the catalogue of **Items** offered by the E-store, adding the items they desire to buy to their personal **Shopping Cart**. The items have a name, a type of product and a price, as well as an ID identifying them in the E-Store database. The Shopping Cart has an amount for each item, and a calculated total price. Once clients are ready to buy the items on their cart, they can place an order, that is sent by the store to the DropMate platform, represented by the **Parcel** entity. Associated with this entity is the ID of the ACP delivering the order, which was chosen by the client, as well as generated delivery codes and pickup codes by the DropMate platform. The pickup codes are sent to the clients, so they can pickup their order at the ACP; the delivery codes are sent to the ACP operators, so they can verify if the orders bought by the courier correspond to the orders placed on the service. Parcels have a **status**, an enum that can belong to any of the following three states:

- IN DELIVERY, describing parcels currently being sent to the ACP;
- WAITING FOR PICKUP, describing parcels received by the ACP, but that haven't yet been picked up by their client;
- DELIVERED, describing parcels already picked up by their clients;

The relationship between the Parcel and Shopping Cart entities also creates a **Order Record** entity, that is stored on the side of the store, so clients can check the current status of their parcels, and view their order history.

Associated with the DropMate platform is also the **System Administrator**, corresponding to the DropMate Admin actor previously identified.

The partner stores and ACP are described by their name and location, with the partner ACP also having another very important attribute used in the DropMate business logic – their maximum **delivery limit**, which describes the maximum number of active packages they can deal with at a given time. If they reach this limit, the ACP will not be available to accept new orders placed by the E-Store Clients. The relationship between a Partner ACP and a Parcel also gives way to the creation of two relationship entities – the **Parcels In Delivery Record** and the **Parcels Waiting for Pickup Record**. The relationship with the DropMate platform, meanwhile, creates the **Operational Statistics** association entity, which contains the overall statistics for this pickup point, such as the current number of parcels in the delivery state, the current number of parcels in the waiting for pickup state, and the total number of parcels serviced by the pickup point.

Finally, the **ACP Operator** is the entity describing the previously identified ACP Operator actor.

4 Architecture notebook

4.1 Key requirements and constrains

The main purpose of DropMate service is to integrate with other business partners. To achieve this, the service must provide a scalable API capable of handling requests from a growing community of partners.

DropMate service should include a responsive management page for performing administrative tasks. This page will expose a significant amount of information, including parcels, partners, and other common data. It should handle data effectively, for example, by implementing pagination.

Each tracked parcel will have three distinct codes to change its status. These codes will be disclosed at the appropriate moment. When a code is entered, the DropMate service should verify its validity to ensure accurate status updates.

To assist operator workers who only need to update the order status of a parcel, the DropMate service should provide an easy-to-use dashboard. This dashboard should simplify the process of updating the status, making it efficient and user-friendly.

4.2 Architetural view

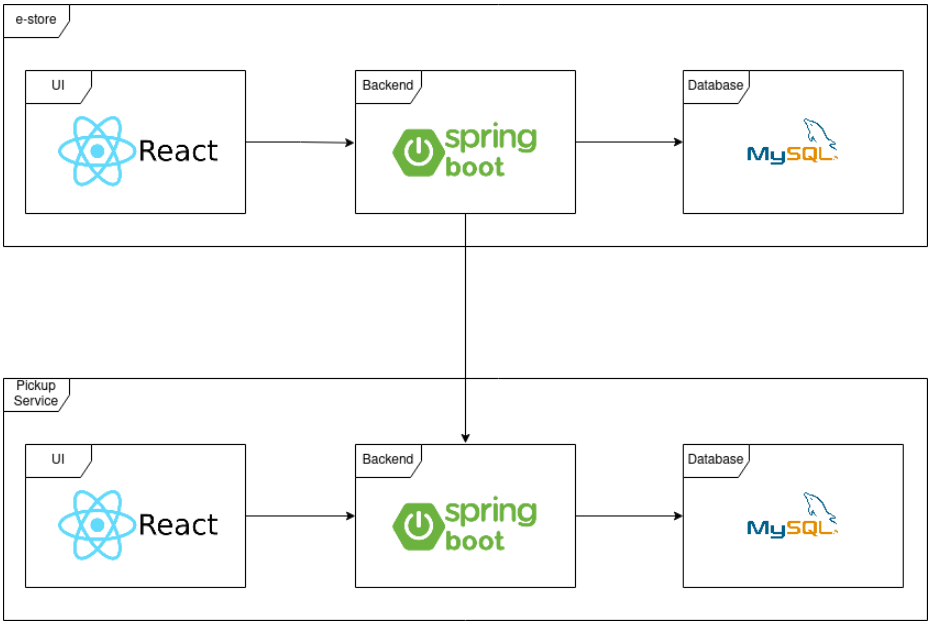

Figure 4 – Overview of the project architecture for the E-Store and for the DropMate Pickup services.

The solution comprises two systems: the e-store platform and the DropMate pickup service system.

The first system revolves around the e-store platform, primarily functioning as a web application. For its implementation, we have opted for the ReactJS framework for the front-end, accompanied by a simple Spring Boot backend and a MySQL database. The MySQL database will be responsible for storing client information and maintaining the product catalog.

The second system is dedicated to the DropMate pickup service. Like the e-store platform, it is designed as a web application and features two distinct views: one for the system administrator and another for workers responsible for updating order statuses. The front-end of this system will also be developed using ReactJS, while the back end will utilize Spring Boot and a MySQL database to manage and store the orders.

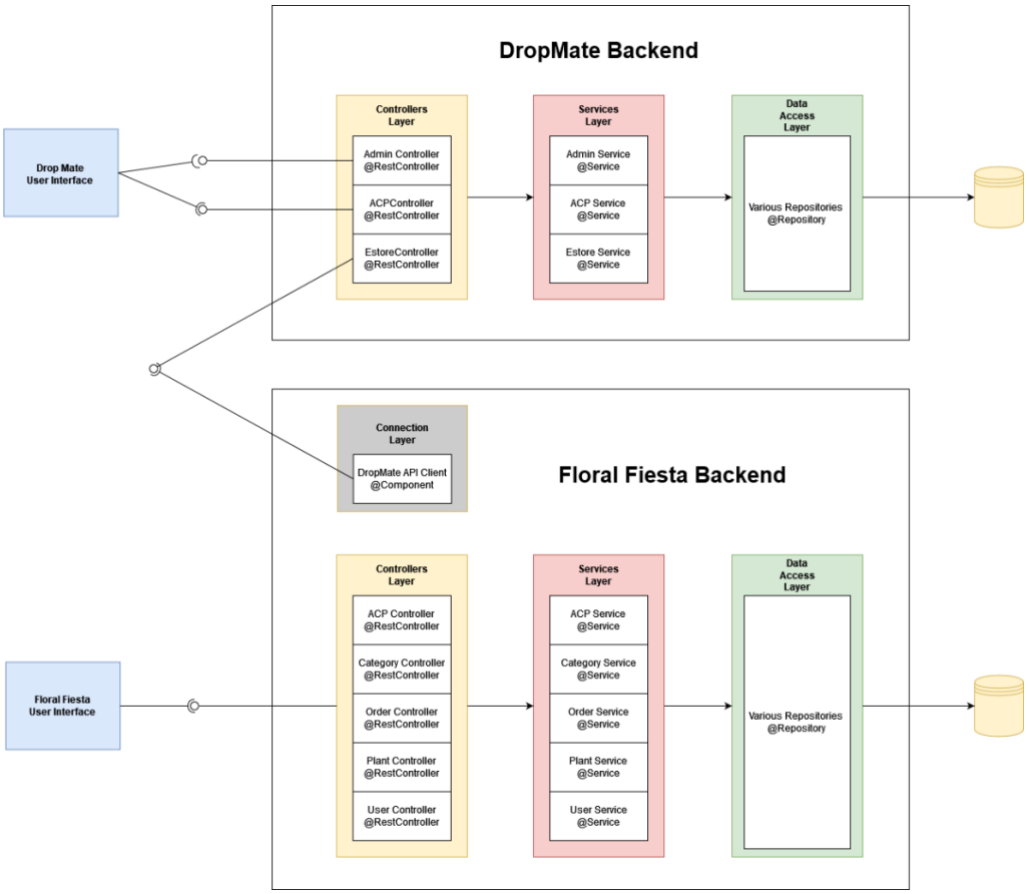

Figure 5 – Overview of the backend of both the Floral Fiesta E-Store and the DropMate services.

The DropMate service is structured with three distinct controllers and corresponding services for different user roles: Administrators, ACP Operators, and E-store partners. One of these controllers, the e-store controller, facilitates the integration between the DropMate service and the implemented e-store system.

Within the e-store system, the DropMate API Client component handles all the services related to DropMate functionality. The other controllers cater to specific use cases within the e-store: the ACP controller handles requests for available ACPs (Automated Collection Points) processed by the DropMate service. The category controller manages the various product categories in the store. Additionally, the plant controller is responsible for the catalog of products, with each product being referred to as a "plant" within the system. The user controller handles user-specific requests, such as login and authentication. Finally, the order controller manages all order-related operations and communicates with the DropMate API Client component to retrieve relevant information from the DropMate service.

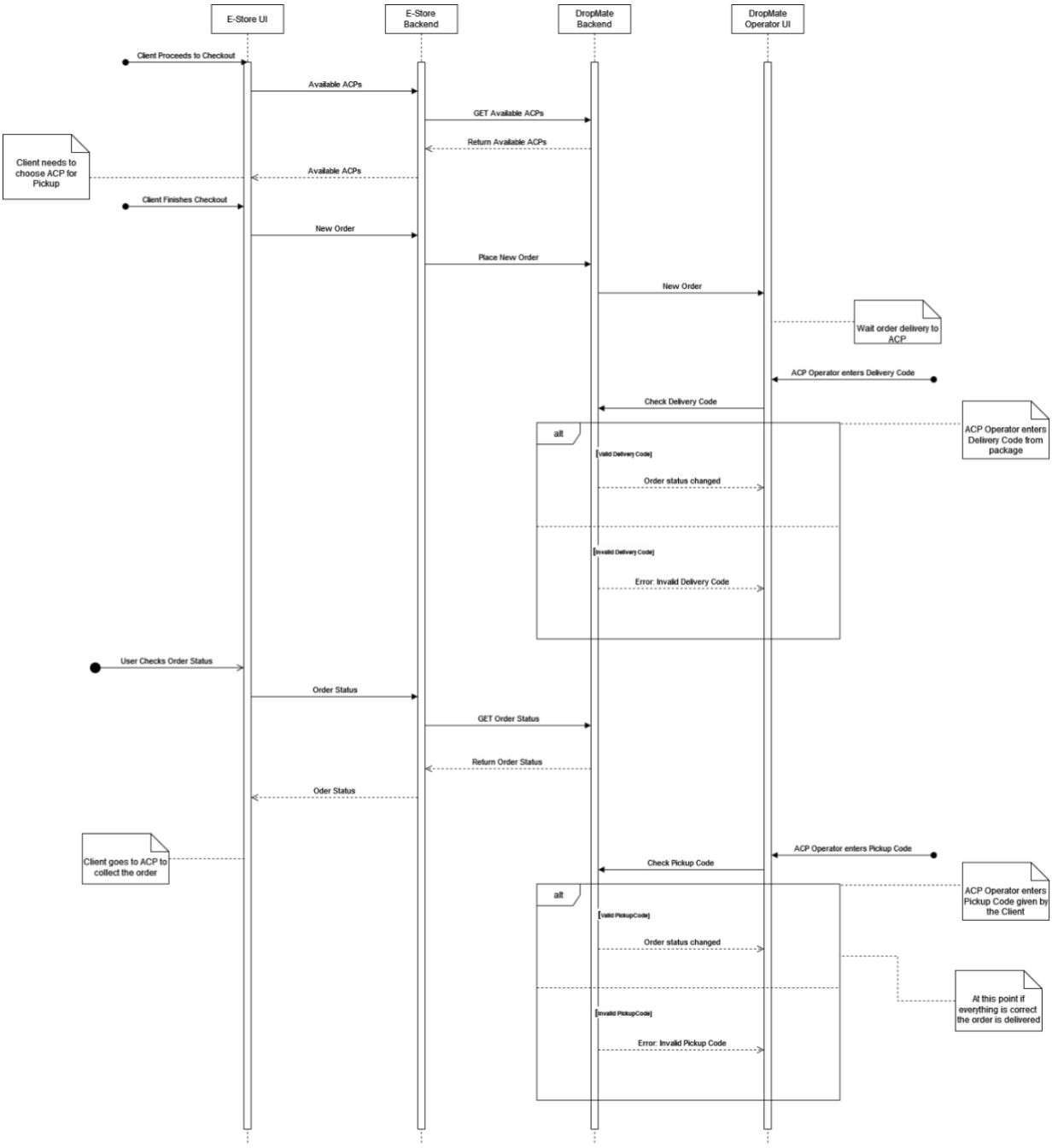

Figure 6 – Interaction diagram for the process regarding the buying of new orders by a client of the e-store, and the checking of a order status by that same client. For a higher definition view, the figure is available here.

During the checkout process on the e-store, when a client proceeds to checkout, the e-store backend will establish communication with the DropMate service to request the available Automated Collection Points (ACPs). Once the client selects an ACP for pickup and completes the checkout, the order details will be communicated to the DropMate service and saved there. Subsequently, the client can track the order's status through the Tracking Page on the e-store service. Whenever the Tracking Page is accessed, the e-store will contact the DropMate service to retrieve the current order status.

On the side of the ACP Operator at the chosen ACP location, when an order arrives, it will come with a delivery code that needs to be manually entered and validated by the DropMate service backend. Upon successful validation, the order's status will be updated from "In Delivery" to "Waiting for Pickup".

When the client physically visits the ACP for pickup, they will be required to provide a Pickup Code to the ACP operator for validation. If the provided code is correct and valid, the order will be considered delivered, and the status will be updated to "Delivered".

4.3 Deployment architecture

For the deployment of our solution, we opted to use a service-oriented deployment through Microsoft Azure services and GitHub pages. Therefore, instead of creating docker containers to encapsulate each subsystem, we created two “App Services” for both of our APIs, created an SQL Azure Server and SQL Azure Database instance, and deployed both of our React Web Apps with GitHub Pages.

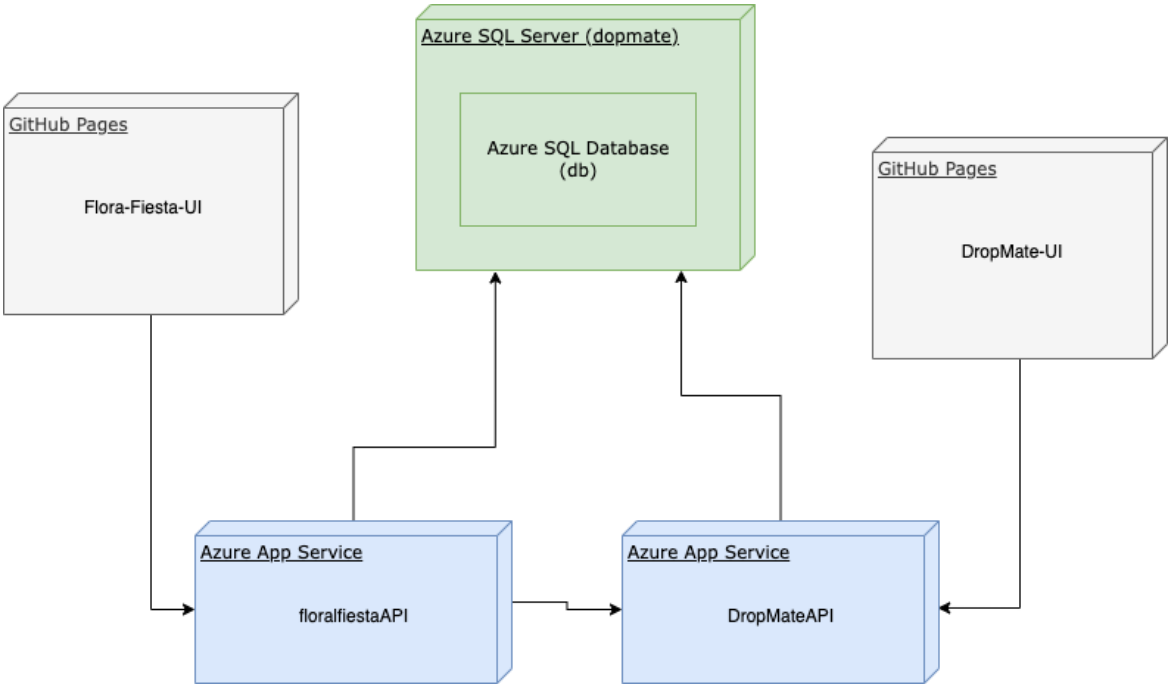

Figure 7 – Deployment diagram.

This deployment strategy allowed us to implement a modularized CI/CD pipeline for every component of our infrastructure.

5 API for developers

The Floral Fiesta API provides methods for a developer to retrieve information about the plants and categories in the Floral Fiesta database as well as register and login users, create and browse orders and ACPs. A more detailed documentation is accessible online, hosted by Postman, at this link.

The DropMate API, meanwhile, provides the REST endpoints accessing the business logic and the processes managed by the DropMate platform. This includes de endpoints accessed to create the UI for the DropMate system administrator, the endpoints accessed to create the UI for the ACP operators, as well as the endpoints contacted by the partnered E-Stores, such as the case of the Floral Fiesta, to manage their pick-up processes. The detailed documentation is accessible online, hosted by Postman, at this link.

6 References and resources

Cloud Monitoring Observability - https://learn.microsoft.com/en-us/azure/cloud-adoption-framework/manage/monitor/observability

Spring Data JPA Reference Documentation - https://docs.spring.io/spring-data/jpa/docs/current/reference/html/

“How to Change the Default Port in Spring Boot” - https://www.baeldung.com/spring-boot-change-port

“Enabling Cross Origin Requests for a RESTful Web Service” - https://spring.io/guides/gs/rest-service-cors/

SQL IDENTITY_INSERT Documentation - https://learn.microsoft.com/en-us/sql/t-sql/statements/set-identity-insert-transact-sql?view=sql-server-ver16

JPA List Mapping - https://www.javatpoint.com/jpa-list-mapping

Persisting Maps with Hibernate - https://www.baeldung.com/hibernate-persisting-maps

“Deploying React apps to GitHub Pages” - https://blog.logrocket.com/deploying-react-apps-github-pages/#setting-up-the-react-application

“How can I allow unknown users to access my SQL (Azure) DB?” - https://stackoverflow.com/questions/13406663/how-can-i-allow-unknown-users-to-access-my-sql-azure-db

Jackson ObjectMapper - https://www.baeldung.com/jackson-object-mapper-tutorial