Bilkent University
Department of Computer Engineering

Senior Design Project

T2302

Yicem

Design Project Final Report

Group Members:

Ahmet Alperen Yılmazyıldız 22002712

Ömer Burak Doğan 21902410

Yağız Özkarahan 22002276

Ege Çenberci 21801618

Bilgehan Sandıkcı 21902354

Supervisor: Özgür Ulusoy

Innovation Expert: Haluk Altunel

Course Instructors: Atakan Erdem, Mert Bıçakçı

13.05.2024

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

Table of Contents

1. Introduction

In today's world, food waste has become a significant concern. According to the Food and Agriculture Organization of the United Nations (FAO), approximately 13.8% of the food produced is wasted across various stages [1]. This wasted food not only represents lost resources used in production and transportation but also contributes to greenhouse gas emissions during decomposition. While restaurants and cafes strive to predict customer demand accurately, there's always a chance of perfectly good food going unsold at the end of the day.

Yicem emerges as a solution to address this very issue. It's a mobile application platform designed to bridge the gap between sellers with surplus perishable items and customers seeking great deals. By providing a platform for selling these soon-to-expire foods at discounted prices, Yicem aims to significantly reduce food waste while offering cost-conscious consumers the opportunity to purchase delicious and fresh food at a fraction of the original price.

Imagine a scenario where you're on your way home from work, feeling a rumble in your stomach but hesitant to spend a fortune on dinner. With Yicem, you can browse through nearby listings on your phone and discover exciting offers on fresh food from restaurants, cafes, or bakeries in your area. You can snag a delicious salad at a discount or enjoy a pastry for a fraction of the usual price, all while contributing to a more sustainable future by preventing food waste.

1.1 Purpose of the system

Yicem operates with a two-fold purpose, addressing both environmental and economic concerns. The primary objective is to significantly reduce food waste. By providing a platform for selling surplus perishable items at discounted prices, Yicem aims to divert perfectly edible food from landfills and promote its consumption before it expires. This not only reduces the environmental impact associated with food waste disposal but also conserves valuable resources used in food production and transportation.

Secondly, Yicem strives to create a win-win situation for both sellers and customers. Sellers can generate additional revenue by offloading surplus inventory at discounted prices, contributing to increased profit margins and reduced waste disposal costs. For customers, Yicem offers the opportunity to purchase delicious and fresh food at significant discounts,

fostering a more affordable dining experience. This increased accessibility encourages a wider customer base, potentially leading to an overall boost in sales for participating sellers.

1.2 Design goals

1.2.1 Scalability

The initial platform of the application is planned to be the campus of Bilkent university. But the plan is to grow the platform over time and open it up to other campuses in Ankara, and eventually other cities too. Considering this, it is essential to design the system to handle an increasing number of users and businesses.

- Initially, the application should support up to around 12.000 users (number of students in Bilkent). That is, the database of the application should be expected to have large enough space for storing this many users.
- Space cost of business and admin accounts is negligible compared to customers.
- As an estimation, the application should be able to accommodate around %20 monthly increase in scale (number of active users and businesses) without a decrease in performance.
- The system should be designed to allow horizontal scaling in elements such as server and database. For example, if the need arises, the system should be able to allow the addition of more servers without disrupting the existing server.

1.2.2 Performance

As it is the case with mobile apps, providing a seamless user experience should be a key requirement. Having slow response times and frequent crashes could deter potential users away from the application. To address these issues, there will be certain constraints:

- Upon opening, the application should ideally load within 3-5 seconds. Intermediate loading times shouldn’t exceed 3 seconds either.
- During peak hours (around the general closing hours of restaurants/cafes in this case, which is roughly 17.30 in Bilkent for example) the application should be expected to support at least 500 users simultaneously. This number can be larger as the platform grows (see scalability).

1.2.3 Usability

A clean and intuitive UI design is crucial for a mobile application. The application will follow certain constraints for this purpose:

- The most common functionalities should be available right away in the homepage (such as the list of currently open businesses).
- Navigation should not be convoluted. Each core functionality should be reachable from the main page within 3-4 touches.
- The labels and icons for functionalities should be intuitive. For example, the label/icon to indicate a functionality should be selected from the most commonly used options to indicate a similar functionality in other popular applications.
- The application should support Turkish and English at the very least. As the platform grows, different languages should be able to be added easily.

1.2.4 Marketability

For Yicem to grow larger and spread to more locations, we need to be able to communicate well with both the customers and businesses. For this goal, some key things that need to be done are:

- Develop a strong brand identity and compelling marketing materials to promote Yicem to both sellers and buyers.
- Highlight the benefits of using Yicem, including its contribution to reducing food waste, providing affordable deals, and supporting sustainability initiatives.
- Collaborate with local businesses, community organizations, and environmental groups to increase awareness and drive adoption of the Yicem platform in target markets.

1.2.5 Maintainability

- Design the Yicem system with clean, modular code to facilitate ease of maintenance and future updates.
- Implement coding best practices and adhere to established coding standards to ensure readability and maintainability of the codebase.
- Regularly review and refactor code as necessary to address technical debt and improve code maintainability over time.

- Implement unit testing frameworks to ensure code functionality and facilitate easier bug identification and resolution.
- Utilize version control systems to track code changes, allowing for easy rollbacks and collaboration among developers.

1.3 Definitions, acronyms, and abbreviations

- **Yicem:** The mobile application platform being developed in this project.
- **Yicem Seller App:** A mobile application specifically designed for sellers to list and manage their surplus perishable items on the Yicem platform.
- **Yicem App:** A mobile application designed for customers to browse and purchase discounted perishable food items from nearby sellers.
- **React Native:** A JavaScript framework used to develop cross-platform mobile applications (usable on both iOS and Android) with a single codebase.
- **Java:** A high-level programming language used to develop the backend server of the Yicem application.
- **Java Spring Framework:** An open-source framework that provides a comprehensive platform for developing Java applications, used in this project for the backend server.
- **MongoDB:** A NoSQL document database used to store application data in the Yicem project.
- **Docker Containers:** A software development method that packages code and its dependencies together in a standardized unit, simplifying deployment and ensuring consistency across environments. Used to deploy the Yicem backend server on the cloud server.
- **DigitalOcean Droplet:** Droplet is a web service by DigitalOcean that provides secure, resizable virtual computers in the cloud. Used to host the Yicem backend server.
- **REST APIs (Application Programming Interfaces):** A web development standard that defines how applications request and exchange data. Used to establish communication between the Yicem frontend apps and the backend server.
- **JWT (JSON Web Token):** An open standard (RFC 7519) for creating access tokens that contain claims (e.g., user information, permissions) encoded in a JSON object. Used for authentication and authorization in the Yicem application.
- **Controller-Service-Repository Pattern:** A software architectural pattern for backend development that separates concerns between:
    - Controllers - Handle user interactions and API requests.
    - Services - Encapsulate core business logic and interact with repositories.

    - Repositories - Provide an abstraction layer for accessing and manipulating data (e.g., from MongoDB in this case).
- **VCS (Version Control System):** A system that records changes to a file or set of files over time, allowing for tracking, rollback, and collaboration. Github is the VCS used in this project.
- **JSON (JavaScript Object Notation):** A lightweight data interchange format commonly used for transmitting data in web applications. Used as the payload format in HTTP requests within the Yicem application.

1.4 Overview

Yicem is a mobile application platform designed to combat food waste and create a win-win situation for both sellers and customers. By providing a platform for sellers to sell surplus perishable items at discounted prices, Yicem aims to significantly reduce the environmental and economic impact of food waste.

Food waste is a significant global issue, with estimates suggesting that roughly one-third of all food produced is lost or wasted each year [1]. This translates to a staggering amount of wasted resources and contributes to greenhouse gas emissions. While restaurants and cafes strive to predict customer demand accurately, there's always a chance of perfectly good food going unsold at the end of the day.

Yicem tackles this challenge by connecting sellers with surplus perishable items to value-conscious customers seeking great deals. Sellers, such as restaurants, cafes, and bakeries, can utilize the Yicem Seller app to list their surplus food items nearing expiry at discounted prices. Customers can then browse the Yicem app to discover these deals on delicious and fresh food options available nearby.

Yicem offers a multitude of benefits for both sellers and customers:

- Reduced Food Waste: By facilitating the sale of surplus food before it expires, Yicem significantly reduces food waste, contributing to a more sustainable future.
- Increased Sales & Revenue: Sellers can generate additional revenue by offloading surplus inventory at discounted prices, helping to improve their profit margins and reduce waste disposal costs.
- Affordable Deals for Customers: Customers can access delicious and fresh food at significantly lower prices, promoting a more affordable dining experience.

Yicem leverages a combination of technologies to deliver a robust and user-friendly platform:

- Frontend Apps: Developed using React Native, the Yicem Seller and Yicem apps provide a smooth and intuitive user experience across various mobile devices and operating systems.
- Backend Server: The backend server, implemented in Java using the Spring Framework, handles data storage, processing, and communication with the frontend apps.
- Database: MongoDB serves as the primary database for storing application data.
- Deployment & Scalability: Docker containers ensure easy deployment and scaling of the backend server on the cloud using DigitalOcean Droplet.
- Communication & Security: REST APIs facilitate communication between the frontend apps and the backend server. JWT tokens are utilized for secure authentication and authorization.

2. Requirements Details

2.1. Functional Requirements

- There should be 2 different applications, one specifically catered for the buyers, and one for the business owners/sellers.
- The business owners must be verified by the admins before they can start to use the application at its full capacity.
- The buyer users will be verified via email authentication.
- The buyer user will be able to see all the businesses that have products to sell.
- The buyer user will see the active discounted offers on the home page.
- The buyer user will be able to see the registered businesses on a map.
- Upon selecting a seller, or via selecting a discounted offer, the buyer user will be able to see all the offers of the selected seller and make reservations for the product they wish to buy.
- On a specific seller’s page, the buyer user will be able to see the opening and closing times of the business.
- On a specific seller’s page, the buyer user will be able to see the user reviews left for the seller.

- The buyer user will be able to see their past purchases and leave reviews for these purchases.
- The buyer user will see the general food types that are being sold by a specific seller. Some examples for the food types are desserts with milk, desserts with sherbet, donuts, biscuits, cookies, etc.
- The buyer users and the seller users will be able to change their account passwords.
- The buyer users will be notified of new offers from the businesses in their proximity.
- The buyer users will be able to favorite a business. The buyer users will be notified of their favorite business' offers regardless of their proximity.
- The buyer users will be able to unlock achievements with different milestones they hit (x amount of purchases made), which will be displayed in their profiles.
- The seller users will be able to put their products up for sale at a desired price.
- The seller users' sale activity will be logged. The logged statistics will be accessible by the seller user to adjust sale prices or times.
- The seller users will be given suggestions by examining their sales statistics.
- The seller users will be able to approve a transaction as complete after a product has been sold to the buyer that had reserved it.

2.2. Non-Functional Requirements

2.2.1. Scalability

The initial platform of the application is planned to be the campus of Bilkent university. But the plan is to grow the platform over time and open it up to other campuses in Ankara, and eventually other cities too. Considering this, it is essential to design the system to handle an increasing number of users and businesses.

- Initially, the application should support up to around 12.000 users (number of students in Bilkent). That is, the database of the application should be expected to have large enough space for storing this many users.
    - Space cost of business and admin accounts is negligible compared to customers.
- As an estimation, the application should be able to accommodate around %20 monthly increase in scale (number of active users and businesses) without a decrease in performance.

- The system should be designed to allow horizontal scaling in elements such as server and database. For example, if the need arises, the system should be able to allow the addition of more servers without disrupting the existing server.

2.2.2. Performance

As it is the case with mobile apps, providing a seamless user experience should be a key requirement. Having slow response times and frequent crashes could deter potential users away from the application. To address these issues, there will be certain constraints:

- Upon opening, the application should ideally load within 3-5 seconds. Intermediate loading times shouldn't exceed 3 seconds either.
- During peak hours (around the general closing hours of restaurants/cafes in this case, which is roughly 17.30 in Bilkent for example) the application should be expected to support at least 500 users simultaneously. This number can be larger as the platform grows (see scalability).

2.2.3. Reliability

- In the case of a server failure or a shutdown, the system should recover without loss of data.
- In-app payments should be made in an atomic way; that is, outer factors should not result in the loss of a value. In the case of a system failure or shutdown in the middle of a transaction, the system should abort the process and not do any transaction. It is not acceptable to have a scenario where the user loses money because of a system failure.

2.3.4. Security

- Authentication and authorization methods should comply with the industry standards. Only the most up-to-date and verified A&A technologies are accepted to be used.
- Sensitive data should be encrypted and payment info should be invisible to all parties, including the user themselves (for example, the user should only be able to see the last 4 digits of their credit card after they have entered it). This ensures that payment info is secure even if a malicious source enters the user's account.
    - Secure third-party payment frameworks (such as iyzico) can be used to handle transactions.

2.3.5. Usability

- A clean and intuitive UI design is crucial for a mobile application. The application will follow certain constraints for this purpose:
- The most common functionalities should be available right away in the homepage (such as the list of currently open businesses).
- Navigation should not be convoluted. Each core functionality should be reachable from the main page within 3-4 touches.
- The labels and icons for functionalities should be intuitive. For example, the label/icon to indicate a functionality should be selected from the most commonly used options to indicate a similar functionality in other popular applications.
- The application should support Turkish and English at the very least. As the platform grows, different languages should be able to be added easily.

3. Proposed software architecture

3.1 Overview

In this section, we will explain the final architectural structure and design details of Yicem. We will make the subsystems and their purposes clear and explain the components within each subsystem and their purposes.

Yicem uses a client-server pattern. The client side consists of a Frontend Layer which is implemented in React Native framework. The server side contains Backend Layer, Database Layer. Backend Layer is implemented in Java Spring Boot framework.

When a user interacts with the client side interface, it sends REST requests containing necessary values to the server. The server processes these requests and communicates with the MongoDB database to manipulate the data as required. The connection between the server and the MongoDB database is established through Spring Data MongoDB. This integration allows for efficient communication and data transaction between the backend server and the database.

Both the Backend and Database Layers of Yicem are deployed in the DigitalOcean cloud servers. This deployment ensures that Yicem is accessible and responsive to users' needs 24/7.

3.2 Subsystem Decomposition

Yicem follows a client-server architecture consisting of Frontend Layer, Backend Layer, Database Layer and integration of a Third Party API. The system is built up to these four primary components. Each one of these components serves distinct purposes in application.

The client layer has three sub-layers: the customer mobile application, the business mobile application, and the admin web application. The customer mobile application module provides user interfaces for customers to interact with the Yicem application. The business mobile app provides user interfaces for business owners to interact with Yicem, while the admin web application provides user interfaces for admins to interact with Yicem.

Backend Layer serves as an intermediate layer between Frontend Layer and the Database Layer which applies the logic of the application while handling the requests from the Frontend Layer. Backend Layer consists of the services AuthService, BuyerService, SellerService, OfferService and ReservationService.

MongoDB is the Database Layer of Yicem and its connection with Backend Layer is established by Spring Data MongoDB library of Java Spring Boot framework. All application data, including user information, offers, locations, etc. is stored and managed within this MongoDB database. Yicem also uses Google Maps API as a third party integration to provide map functionality to users.

The backend and data management modules are hosted on DigitalOcean Cloud servers also known as Droplets. We refer to them as Droplets in this report.

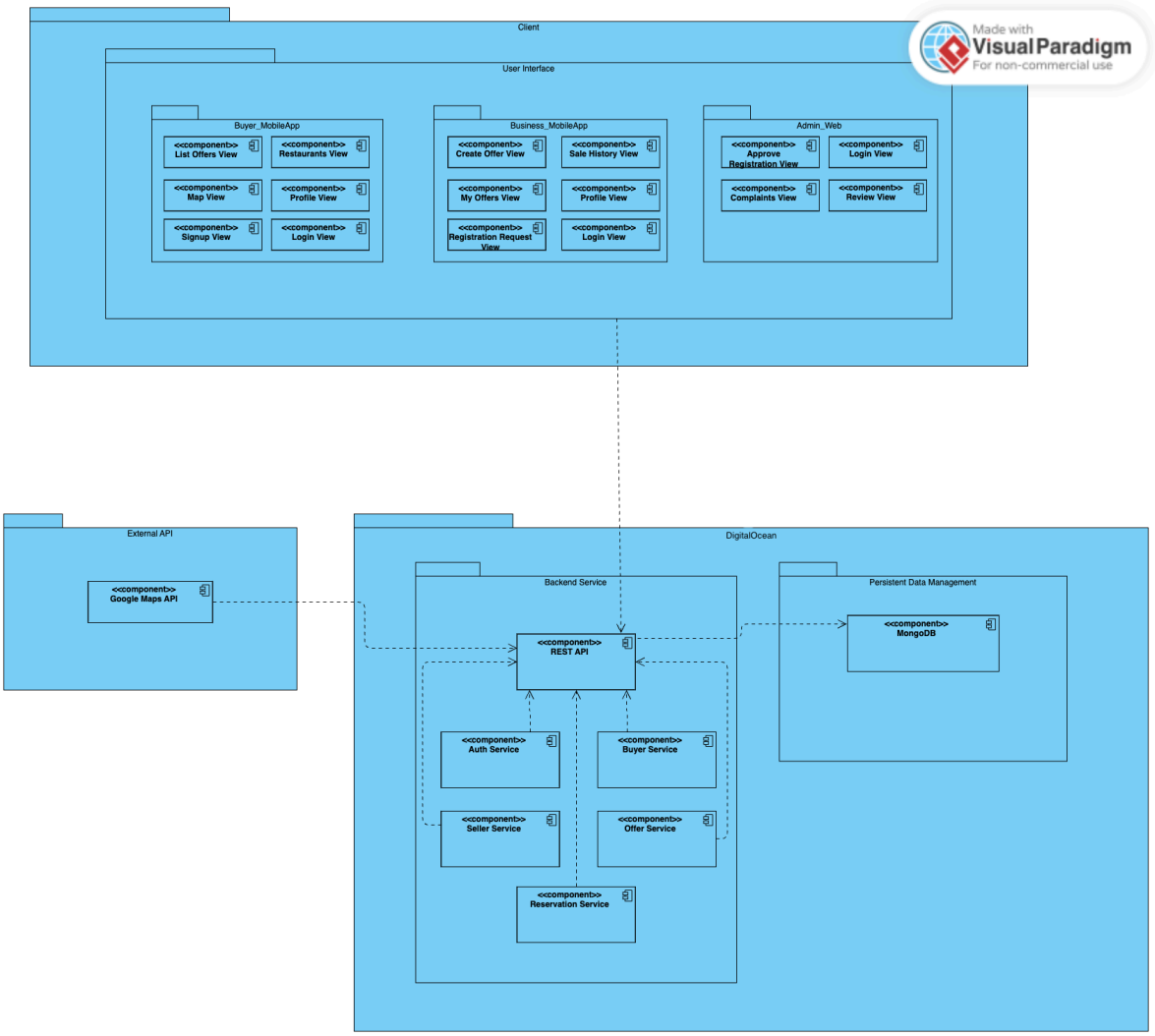

Figure 1: Subsystem Decomposition of Yicem (High resolution of diagram can be seen here)

3.3 Hardware/Software Mapping

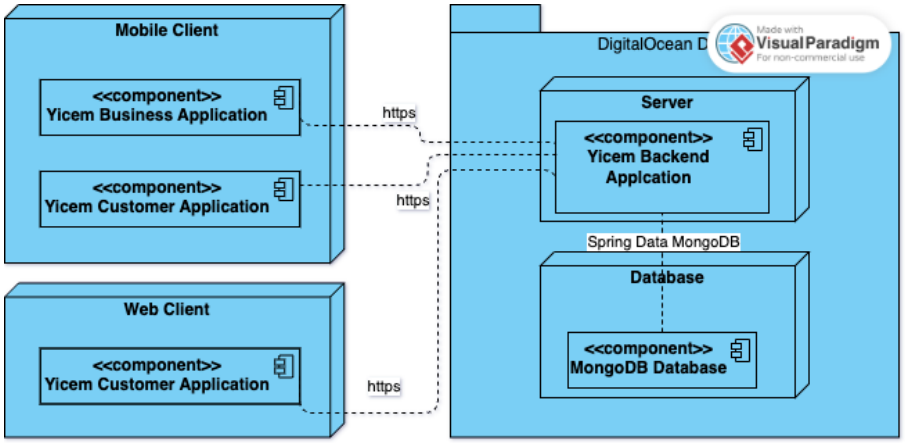

Figure 2: Hardware/Software Mapping of Yicem

Yicem has three different clients in total. There are two mobile clients, one for each user type. The mobile clients are for customer and business users, and the web client is for the admins. The clients communicate with the application's backend deployed on a Droplet through HTTPS messages and REST API. The DigitalOcean deployment contains the MongoDB database deployment and the backend logic. The backend server connects with the MongoDB database using Spring Data MongoDB library of Spring Boot Framework.

3.4 Persistent Data Management

Yicem application doesn't require the storage of immense amounts of data; instead, it requires a flexible database. We decided to use MongoDB as our database because MongoDB provides flexibility, scalability, and ease of development. Since MongoDB is a NoSQL database, it is very flexible and can accommodate application changes if and when needed. MongoDB does not require a schema to store information, which makes the development process easy and flexible. If a change in the currently stored information or a new data type is needed, thanks to the schemaless nature of MongoDB, we can make changes to our database quickly and easily. Moreover, we can add more servers to our cluster via MongoDB when our application's demand grows, increasing the system's scalability without any technical worries.

Our database consists of multiple collections, such as "buyers", "sellers", "users", "offers", etc. and each collection is designed to handle distinct application-related data. For instance, the "users" collection contains data about registered users' personal information, and login credentials. Similarly, the "offers" collection manages information regarding the offers that businesses created. Which have attributes such as offerName, description, price, itemCount, etc. We will have a collection for each of our models that we show in our final class diagram. Attributes of these collections also can be seen from the class diagram.

In brief, our data management system is developed for flexibility and scalability. Furthermore, robust security protections are established to secure the confidentiality and integrity of our users' information.

3.5 Access Control and Security

To maintain security throughout our application and authorize users as needed, we have elected to use JSON Web Tokens (JWTs). Upon successful login or sign-up, a JWT is assigned to the user and stored in the user's local device. The JWT token is checked when the user tries to access any page that requires authorization. The JWTs have expiration dates, which adds another layer of protection. JWTs can contain user-specific data. Since we have two separate applications for customers and businesses, there is no need to implement role-based access. However, the extra information storage functionality of the JWTs can be utilized if need be.

Moreover, no sensitive data is stored locally or in the system database. The passwords are encrypted and then stored in the database. Upon login, the user password is sent in the SHA-1 encrypted format, and the server decrypts the password it stores in the database and encrypts in the SHA-1 format to check if the received password value is the same as the newly encrypted one. Since there is no way to decrypt SHA-1 encryption other than by checking all possible values, the security of the password against attackers is guaranteed if the user has chosen a good enough password.

3.6 Subsystem Services

3.6.1 Client

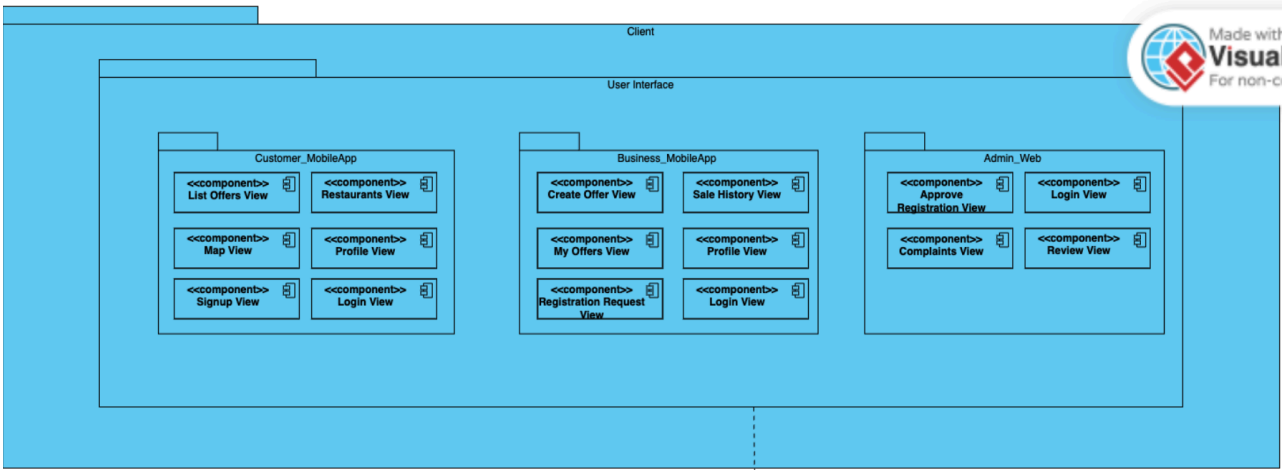

Figure 3: Client module of Yicem

3.6.1.1 Customer Mobile App

List Offers View: This view includes the list of offers made by businesses nearby where users can see those offers in detail and make reservations.

Restaurants View: This view contains a list of restaurants who are registered in the Yicem application where users can navigate to that restaurant page.

Map View: This view includes a google maps screen where each business is pointed out on the map.

Profile View: In this view customers can view their past purchases and edit their profile information.

Signup View: This view contains the signup page.

Login View: This view contains the login screen where users enter their email and password.

3.6.1.1 Business Mobile App

Create Offer View: In this view businesses can create their offers.

My Offers View: In this view businesses can view all of their active offers.

Sale History View: In this view businesses can see their past offers that are sold.

Profile View: In this view businesses can see and edit their profile information.

Registration Request View: In this view businesses can send their registration request to sign up to application.

Login View: This view contains the login screen where users enter their email and password.

3.6.1.1 Admin Web App

Approve Registration View: Admins can approve or reject registration requests made by businesses in this view.

Complaints View: Admins can view the complaints made by customers about businesses.

Review View: In this view admins can view and delete reviews made by customers about their purchases.

Login View: This view contains the login screen.

3.6.2 DigitalOcean Server

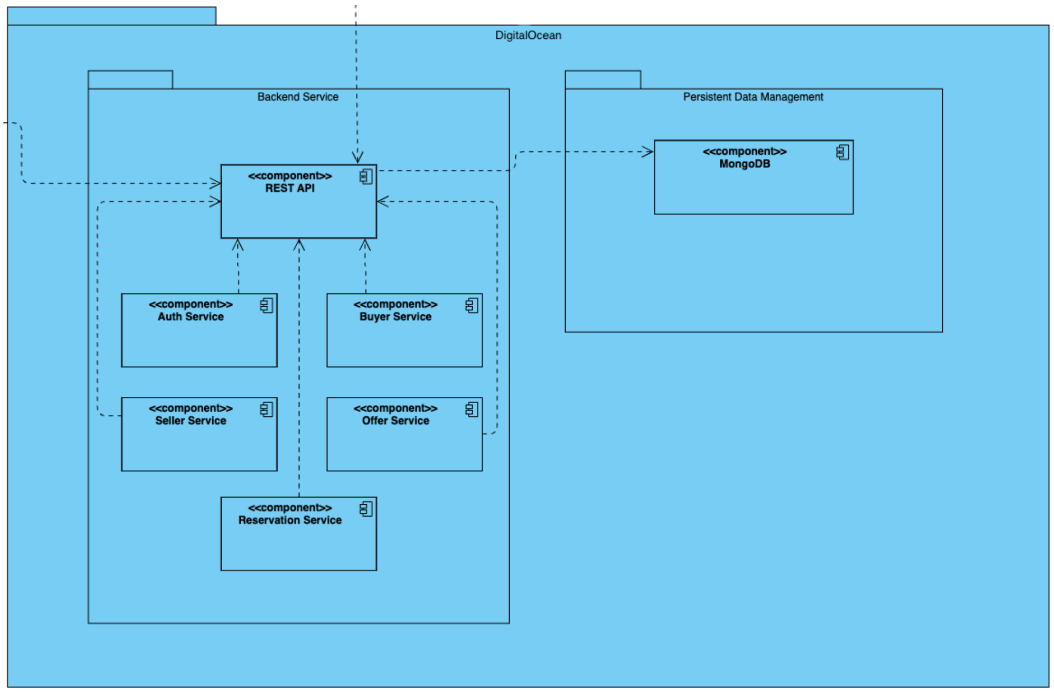

Figure 4: DigitalOcean module of Yicem

3.6.2.1 Backend Services

REST API: Rest API provides communication between frontend and backend.

Auth Service: This service handles authorization operations such as sign up, login and log out.

Buyer Service: This service handles customers' operations such as making reservations, leaving a review on their purchases, making a complaint and changing their profile information.

Seller Service: This service handles businesses' operations such as creating and editing offers, viewing their active and past offers and editing their business information.

Offer Service: This service handles operations regarding an offer. Helper methods for creating and editing and offer will be in offer service.

Reservation Service: This service handles operations regarding reservation made to an offer.

3.6.2.2 Persistent Data Management

MongoDB: MongoDB is the database of the application, enabling storing information regarding customers and businesses.

3.6.3 External API

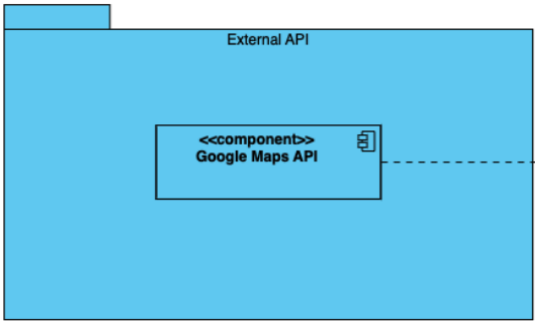

Figure 5: External API module of Yicem

Google Maps API: This API provides map and location services.

3.7 Class Diagram

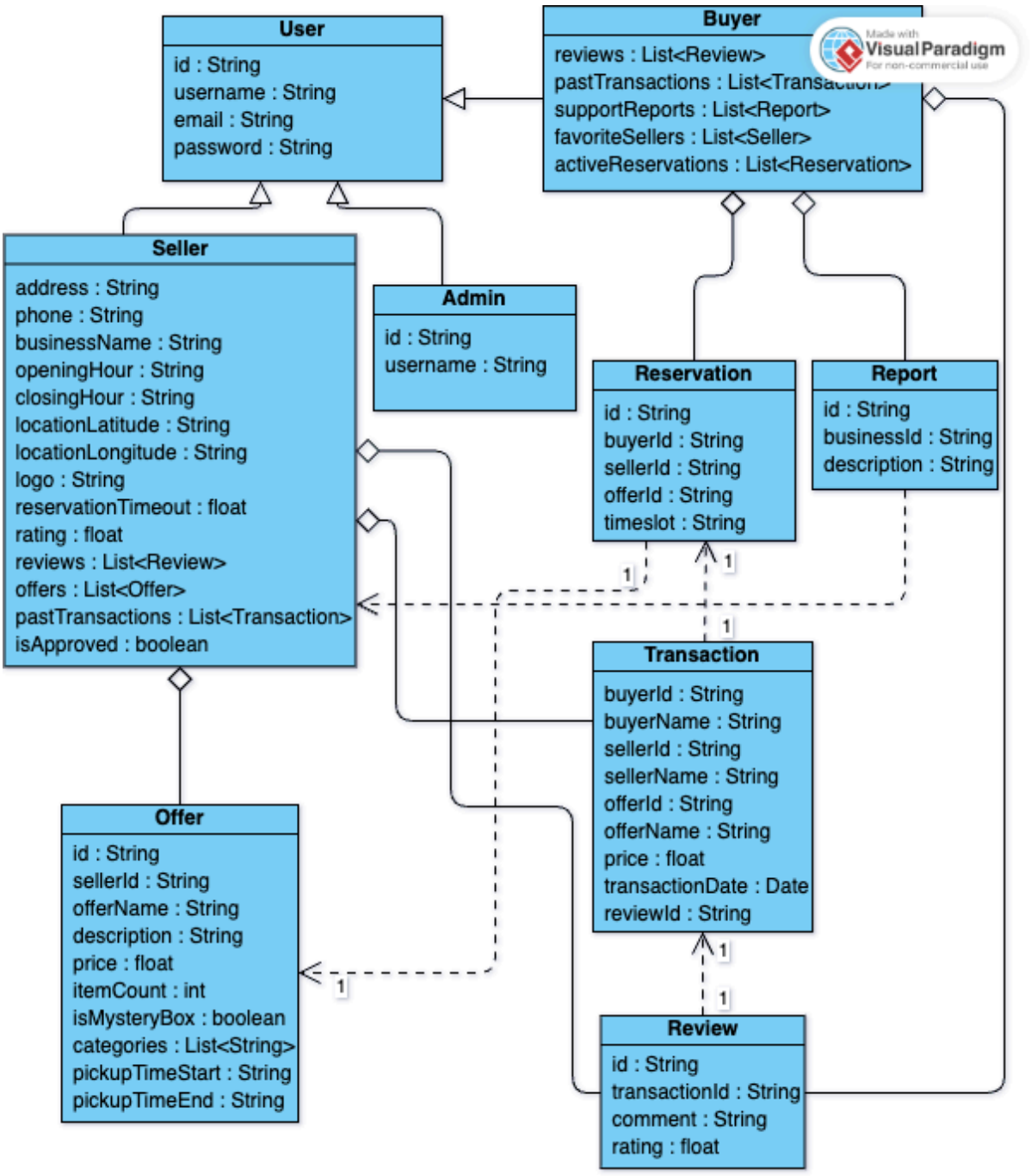

Figure 6: Class Diagram of Yicem

4. Development/Implementation Details

4.1 Frontend

In our system, we developed two separate mobile applications—one for customers and the other for venues—along with a web application for administrators. To build the mobile applications, we leveraged the React Native framework, a widely-used JavaScript framework for cross-platform development. JavaScript served as our primary frontend development language. React Native enabled us to create both iOS and Android applications using a single codebase. We opted for distinct codebases for our customer and venue applications. For the admin web application, we utilized the React framework, which is similar to React Native but tailored specifically for web applications. Each application had its own codebase. Additionally, in the frontend, we integrated the Google Maps API to enhance the mapping functionality within the customer app.

4.2 Backend

In our backend development, we relied on Java Spring Boot as our primary framework. MongoDB served as our chosen NoSQL database management system. Docker played a crucial role in containerizing both our database and backend services, making our deployment and development processes. By containerizing our application, deploying to DigitalOcean became much simpler. We merely uploaded our Docker containers on DigitalOcean's Droplets, which acted as our servers.

In our development workflow, we embraced a secure development approach. Instead of pushing all newly written code directly to the main branch, we adopted a careful testing and updating process. Each team member tested and approved incoming changes from other team members before integrating them into the codebase. GitHub served as our version control system and platform for managing tasks and codebase changes. We maintained a "main" branch solely for hosting thoroughly tested and production-ready code.

To organize our tasks in both backend and frontend development, we utilized GitHub's issue system. When implementing a new feature based on these issues, we created branches referencing the respective issue numbers. For instance, if an issue numbered 32 titled "bug fixes" required attention, the changes would be made in a new branch named "/33-bug-fixes". Once the tasks related to the issue were completed, a new pull request

would be initiated from "/33-bug-fixes" to the "main" branch for merging. This systematic approach allowed us to easily trace which branch corresponded to which issue.

5. Test Cases

5.1 Functional tests

In this category, there are two types of tests: integration and component. Component tests generally test one aspect or segment of one application, while integration tests can span across multiple segments, screens and even applications (since Yicem has two mobile apps for customer and business). In some cases there can be overlap between categories (a component test can make use of another component, or an integration test might focus more on one segment than the others), but the tests were put in the seemingly most fitting categories in general.

Test ID	F01	Category	Functional Integration	Severity	Critical
Objective	Verify login functionality				
Steps	1. Open either of the business or customer apps (repeat the test for both) 2. Open the login page 3. Check that the email and password fields are present and editable a. Check that the password field hides the input while it is being entered 4. Enter an invalid email address and password and press the login button a. Verify that the login attempt is rejected 5. Leave any (or both) of the email and password fields empty and press the login button a. Verify that the login attempt is rejected 6. Enter a valid email address and password, previously signed up to the system and press the login button				
Expected	After logging in with a valid email address and a password, in the customer app, the user should be successfully logged in to the system and redirected to their home page. In the business app, user should be successfully logged in to the system and redirected to either their “Offers” page if they are verified, or the information page if they are not yet verified				

Date-Result	- Date: 5/05/2024 - Outcome: Input fields working correctly - **Issues: Business is not directed to pending approvals page** - Result: Success

Test ID	F02	Category	Functional Component	Severity	Critical
Objective	Verify register functionality in the customer app				
Steps	1. In the customer app, press the “Don’t have an account? Sign up” text in the login page 2. Check the input fields in the register page a. Check that all three of username, email and password fields exist and are editable 3. Enter a username, email and password 4. Press the register button				
Expected	A verification link will be sent to the entered email, if it is indeed a valid email. If it is not a valid email, or an already registered email, an error notification will be sent to the user.				
Date-Result	- Date: 5/05/2024 - Outcome: Input fields working correctly, but there is no mail verification at this state. - Result: Partial Success				

Test ID	F03	Category	Functional Component	Severity	Major
Objective	Verify input fields in the register page				
Steps	1. Repeat the steps of test F02, but leave any/all of the fields (username, email, password) empty in the register page 2. Press register button				

Expected	User should be asked to enter valid information in the empty fields
Date-Result	- Date: 5/05/2024 - Outcome: valid information is asked for side cases as input verification - Result: Success

Test ID	F04	Category	Functional Component	Severity	Critical
Objective	Verifying the functionality of the business cards in the home page and the businesses page				
Steps	1. Open the customer app and login 2. In the home page or the businesses page, check the business cards a. Check that the business name and icon is present in all business cards b. Check that the open businesses with available offers are present with an “Open” label and a bright, opaque color c. Check that the open businesses with no available offers are present with an “No offers available right now” label and a semi transparent color d. Check that the closed businesses are present with a “Closed: opens at ….” label and a gray color e. If the card is located in the home page, check that the distance of the businesses to the user are presented on the card i. Check that the distances are correct f. Check that the businesses are pressable i. Check that the user is being redirected to the business’s page, upon pressing				
Expected	Cards are being displayed correctly, with valid information, and the user is redirected correctly upon pressing a business card				
Date-Result	- Date: 5/05/2024 - Outcome: Businesses are displayed correctly with relevant information and statuses. The cards are functioning correctly. - **Issue: no distance is displayed (this feature was removed during implementation)** - Result: Partial Success				

Test ID	F05	Category	Functional Component	Severity	Minor
Objective	Verify the functionality of the search bar in the businesses page				
Steps	1. Open the customer app and login 2. Navigate to the businesses page 3. Enter a business name in the search bar a. Check that the businesses are being filtered in real time as each letter is being typed b. Check that the displayed businesses are filtered according to the search query.				
Expected	Businesses displayed should match the search query.				
Date-Result	- Date: 5/05/2024 - Outcome: Businesses are filtered in real time according to search query. - Result: Success				

Test ID	F06	Category	Functional Integration	Severity	Major
Objective	Verify the functionality of the navigation bar in both applications				
Steps	1. Open the customer app and login 2. Check that there are four icons in the navigation bar, each corresponding to a page 3. Verify that the user is navigated to the home page upon pressing home icon 4. Verify that the user is navigated to the businesses page upon pressing the coffee icon 5. Verify that the user is navigated to the map page upon pressing map icon 6. Verify that the user is navigated to their profile page upon pressing person icon 7. Check that the icon of the active page is displayed with a border				

	and a different color 8. Open the business app and login 9. Check that there are four icons in the navigation bar, each corresponding to a page 10. Verify that the user is navigated to the offers page upon pressing the coffee icon 11. Verify that the user is navigated to the history page upon pressing the clock icon 12. Verify that the user is navigated to their profile page upon pressing the person icon 13. Verify that the user is navigated to the analysis page upon pressing the chart icon 14. Check that the icon of the active page is highlighted
Expected	The navigation icons are displayed correctly and they are redirecting the user to their corresponding pages, for both applications
Date-Result	- Date: 5/05/2024 - Outcome: Navigation bars are working correctly and directing users to the relevant pages when pressed. - Result: Success

Test ID	F07	Category	Functional Component	Severity	Major
Objective	Verify that the attributes of a business are being displayed on the business's page				
Steps	1. Open the customer app and login 2. Enter the page of any business (by pressing their card in either the home page or the businesses page) 3. Check their fields a. Verify the availability of the fields: business name, open hours, address, rating, "View Reviews" text, "Add to your favorites" text, report button, business icon, "Go Back" icon (left arrow) b. Check that these fields are displayed in an organized way, without overlapping on each other				
Expected	All attributes, fields, and buttons of the business are displayed correctly on the business page				

Date-Result	- Date: 5/05/2024 - Outcome: All attributes, fields, and buttons of the business are displayed correctly. - Result: Success

Test ID	F08	Category	Functional Component	Severity	Major
Objective	Verify the validity of buttons/pressables in a business page				
Steps	1. Open the customer app and login 2. Enter the page of any business (by pressing their card in either the home page or the businesses page) 3. Press the report button a. Check that a “Report business” pop-up is displayed with a text field and a button i. Verify that the pop-up disappears upon pressing outside 4. Press the “View Reviews” text a. Verify that the user is navigated to the reviews page of the business 5. Press the heart icon (favorite) a. Verify that the heart changes color (white to transparent or transparent to white) upon pressing 6. Press the left-arrow icon a. Verify that the user is navigated to the previous page				
Expected	All buttons/pressables are functioning correctly				
Date-Result	- Date: 5/05/2024 - Outcome: All buttons/pressables are functioning correctly. - Result: Success				

Test ID	F09	Category	Functional Component	Severity	Minor

Objective	Verify that the behaviors of business pages change according to the current state of business

Steps	1. Open the customer app and login
	2. Enter the page of an open business with available offers
	  a. Verify the business header is displayed in a bright color and available offers are listed
	3. Enter the page of a closed business
	  a. Verify the business header is grayed out and there are no available offers
	4. Enter the page of an open business with no available offers
	  a. Verify the business header is displayed in a bright color but there are no available offers listed

Expected	Business page changes its display correctly according to its current state
Date-Result	- Date: 5/05/2024 - Outcome: Business page is displayed and offers are listed according to current status of the business. - Result: Success

Test ID	F10	Category	Functional Integration	Severity	Critical
Objective	Verify the functionality of making a reservation				

Steps	1. Open the customer app and login
	2. Enter the page of an open business with available offers
	3. Press one of the available offers
	  a. Check that a pop-up for pick up time appears
	  i. Check that the pop-up disappears upon pressing outside
	4. Pick a time for pickup by checking one of the available checkboxes
	  a. Verify that only one checkbox can be selected
	5. Press the “Make a reservation” button				

Expected	User is informed that they have made a reservation successfully. Their reservation should be now visible from their homepage				
Date-Result	- Date: 5/05/2024				

	- Outcome: Reservation making components are functioning correctly: After reservation is made, it is visible from the homepage. - Result: Success

Test ID	F11	Category	Functional Component	Severity	Major
Objective	Reject empty reservation attempt				
Steps	1. Repeat the steps of test F10 but skip the fourth step (leave the checkbox empty)				
Expected	User is informed that they need to check one of the available fields				
Date-Result	- Date: 06/05/2024 - Outcome: Reservation without selecting a pick-up time is blocked and the user is notified with a pop-up alert. - Result: Success				

Test ID	F12	Category	Functional Integration	Severity	Major
Objective	Verify the functionality of leaving a review				
Steps	1. Open the customer app and login 2. Press the “Leave a Review” button from either the past purchases page, or the latest purchase visible in the profile page a. If the user has no past purchases, conduct this test after having a past purchase 3. Check that a review pop-up appears with a text field, rating field and a submit button a. Verify that the pop-up disappears upon pressing outside 4. Write a review in the text field 5. Add a rating 6. Press the submit button				

Expected	User is informed that they have submitted their review and that it will be evaluated before possibly appearing on the business's page
Date-Result	- Date: 06/05/2024 - Outcome: All the steps of the test are met. The user is shown a pop up field to rate and leave a review for the business. The user is notified that their review has been successfully submitted after pressing the submit button. - Result: Success

Test ID	F13	Category	Functional Component	Severity	Minor
Objective	Verify the fields of leaving a review				
Steps	1. Repeat the steps of test F12 but leave both the rating and text fields empty				
Expected	User should be notified that they need to at least enter a review or a rating				
Date-Result	- Date: 06/05/2024 - Outcome: Review submission is not permitted with empty fields, the user is alerted with a pop up message that they need to select a rating. After selecting a rating and trying to submit, the submission is blocked and the user is alerted with a pop up message to write a review. - Result: Success				

Test ID	F14	Category	Functional Component	Severity	Major
Objective	Verify the display of the user's profile page				

Steps	1. Open the customer app and login 2. Go the the profile page of the user, from the navigation bar 3. Check that the user's username and email are presented at the top of the page 4. Check that the user's most recent purchase is presented, with a "View All Past Purchases" button below a. Check that the button is pressable and leads to the past purchases page of the user 5. Check the buttons "Favorite businesses", "Change Password", "Help" and "Logout" are laid out at the bottom of the page a. Check that all of these buttons are pressable
Expected	All of the necessary information and functionalities related to the user's profile page are displayed correctly
Date-Result	- Date: 06/05/2024 - Outcome: The username and the email of the user is presented in the correct place in the profile screen.The "View All Past Purchases", "Favorite businesses", "Change Password", "Help" and "Logout" buttons are all functional and direct the user to the correct screens, or show the correct pop-up screens. - Result: Success

Test ID	F15	Category	Functional Component	Severity	Major
Objective	Verify the functionalities of favorite businesses page				
Steps	1. Open the customer app and login 2. Go the the profile page of the user, from the navigation bar 3. Press the "Favorite Businesses" button to be navigated to the favorite businesses page 4. Verify that the user's favorite businesses are displayed 5. Verify that all businesses are pressable a. Verify that the user is redirected to the business's page upon pressing the business's card				
Expected	Favorite businesses of the user are displayed and functioning correctly				

Date-Result	- Date: 06/05/2024 - Outcome: The user's favorite businesses are listed in the "Favorite Businesses" screen, and each business' card is pressable and directs the user to the specific business' screen. - Result: Success

Test ID	F16	Category	Functional Component	Severity	Minor
Objective	Verify the fields of past purchases				
Steps	1. Open the customer app and login 2. From the user's profile page, press the "View all past purchases" button to go to the past purchases page 3. Verify that the user's past purchases are presented 4. Check the fields of any past purchase a. Verify that the name of the business is present b. Verify that the name of the purchased item is present c. Verify that the price of the purchased item is present d. Verify that the date of the purchase is present e. Verify that a review button is present i. Verify that the button has the text "Edit review" instead, if the user has already left a review 5. Apply step 3 to the single latest review presented in the profile page				
Expected	Past purchases of the user are presented with all of the fields				
Date-Result	- Date: 06/05/2024 - Outcome: User's past purchases are presented in the past purchases screen, with the relevant information displayed as described correctly. However, the condition of Step "4.e.i" is not met since we had elected to discard the editing of reviews functionality. The "Edit review" button is replaced by a grayed out "Review Submitted" button. - Result: Partial Success				

Test ID	F17	Category	Functional	Severity	Major

			Integration		
Objective	Verify change password functionality				
Steps	1. Open either the customer app or the business app (repeat for both) 2. Press the “Change password” button in the profile page 3. Fill all of the fields a. Enter user’s old password b. Enter a new password c. Enter the new password again 4. Press the change password button a. Check that if the entered old password was incorrect, the attempt is rejected b. Check that if all fields were correct, the user is notified that their password has been changed 5. If step 4 is successful, log out and log in again with the new password				
Expected	User’s password is successfully changed upon entering the fields correctly, otherwise the attempt is rejected				
Date-Result	- Date: 06/05/2024 - Outcome: After pressing the change password button, a pop-up field with 3 inputs (old password, new password, and new password repeat) are presented to the user. The user is notified if their old password is entered incorrectly, or when the new password and the new password repeat fields do not match. When information is entered correctly, the user is notified of successful password change, but is not logged out. - Result: Partial success				

Test ID	F18	Category	Functional Integration	Severity	Critical
Objective	Verify logout functionality				
Steps	1. Open either the customer app or the business app (repeat for both) 2. Press the “Logout” button from the profile page 3. Check that the user is redirected to the login page				

Expected	User is logged out and cannot enter the application again without entering their credentials
Date-Result	- Date: 06/05/2024 - Outcome: Both user types are logged out of their account upon pressing the logout button and cannot enter the application without their credentials. - Result: Success

Test ID	F19	Category	Functional Component	Severity	Major
Objective	Verify the functionalities of the map page				
Steps	1. Open the customer app and login 2. Enter the map page, from the navigation bar 3. Verify that the user's current location is correctly displayed on the map 4. Verify that the map is interactable a. Check that the location changes on the map by pressing and dragging across the screen b. Check that the map can be zoomed in and out 5. Check that the businesses in the system are marked on the map, in places corresponding to their real locations a. Check that their markers are pressable b. Verify that a dialog opens when their marker is pressed i. Verify that the dialog is closed upon pressing elsewhere ii. Verify that the user is redirected to the business's page upon pressing the dialog again iii. Verify that the business's name and open/closed status is written on the dialog				
Expected	The user's location and the businesses in the system are presented on an interactable map				
Date-Result	- Date: 06/05/2024 - Outcome: The user's location is correctly displayed on the map screen, and the map can be moved and resized. The businesses are presented with markers and are pressable to reveal the business' name and open/closed status above the marker. The business' information field disappears when the user presses				

	elsewhere on the map. - Result: Success

Test ID	F20	Category	Functional Component	Severity	Critical
Objective	Verify business registration				
Steps	1. Open the business app 2. In the business login page, press the “Create registration request” button 3. Fill the forms business name, email, phone number, password and business address with valid information 4. Press the send application request button 5. Verify that the user is redirected to a page with the disclaimer “Your request is being processed and evaluated” 6. Press the “Change Info” button a. Verify that the user is redirected to the previous page for entering business information 7. Press the “Logout” button a. Verify that the user is redirected to the login page				
Expected	The user’s request is sent and the user is redirected to an information page, from where they can either edit their information or log out				
Date-Result	- Date: 06/05/2024 - Outcome: The user is redirected to a page with the disclaimer “Your request is being processed and evaluated…”, and a button that redirects the user to the login screen. Since the design has been changed since the test case was written, the steps 6 and 7 are irrelevant. - Result: Partial Success				

Test ID	F21	Category	Functional Component	Severity	Major
Objective	Verify business registration fields				

Steps	1. Go through the steps of F20 but enter an invalid input in one or more of the fields, or leave one or more fields empty. a. Enter a weak password (less than 8 characters, does not contain both letters and numbers) i. Check that the password guidelines are displayed near the password field b. Enter an invalid phone number (too short or too long, contains non-numeric characters)
Expected	The registration request attempt is rejected and the user is informed of which fields they should fix
Date-Result	- Date: 6/05/2024 - Outcome: Invalid or insufficient fields are rejected upon submitting, but password guidelines are not displayed while typing password - Result: Partial Success

Test ID	F22	Category	Functional Component	Severity	Major
Objective	Verify the contents of the offers page				
Steps	1. Open the business app and login 2. Go to the offers page of the business 3. Verify that the current offers of the business are presented in the page, each offer inside a card a. If the business has no offers yet, verify that there are no cards in the page 4. Check the contents of each offer card a. Verify that each offer card has a name for its offer b. Verify that each offer card has a description for its offer c. Verify that each offer card has a price for its offer d. Verify that each offer card has an available count number for its offer e. Verify that each offer card has a button that shows the list of the people who is queued for this offer i. Verify each of these people have a pressable “Mark as sold” button next to them				
Expected	The offer page displays all available offers and their attributes				

Date-Result	- Date: 6/05/2024 - Outcome: Offer page is displayed with all available offers and their attributes. All steps passed. - Result: Success

Test ID	F23	Category	Functional Integration	Severity	Critical
Objective	Verify the functionality of adding an offer				
Steps	1. Open the business app and login 2. In the offers page, press the “Add offer” button 3. Check that a pop-up for offer fields appears a. Verify that the pop-up has all of the following fields: title, description, price, total items b. Verify that the pop-up is closed upon pressing outside of it 4. Fill the fields in the pop-up with valid information 5. Press the “Save” button 6. Verify that the new offer is added to the offers page				
Expected	User is able to add a new offer				
Date-Result	- Date: 6/05/2024 - Outcome: All necessary fields are presented, and offer is successfully added - Result: Success				

Test ID	F24	Category	Functional Component	Severity	Minor
Objective	Verify the fields of adding an offer				
Steps	1. Repeat test F23 but leave at least one of the fields empty or invalid a. Enter a negative price b. Enter a negative number of total items c. Enter a non-numeric value in either the price or total items				

	field
Expected	The attempt should be rejected, and the user should be notified of the reason
Date-Result	- Date: 6/05/2024 - Outcome: Attempt is rejected, user is notified that they need to enter a valid price or item count - Result: Success

Test ID	F25	Category	Functional Integration	Severity	Major
Objective	Verify the functionality of editing an offer				
Steps	1. Open the business app and login 2. Press the edit icon (pencil) on one of the offers in the offers page 3. Check that a pop-up for offer fields appears a. Verify that the pop-up has all of the following fields: edit title, edit description, edit price b. Verify that the pop-up is closed upon pressing outside of it 4. Change one or more of the fields in the pop-up, with valid information 5. Press the save button 6. Verify that the offer is now presented with the changed information				
Expected	User is able to edit the information of any offer				
Date-Result	- Date: 6/05/2024 - Outcome: User is successfully able to edit offer information, and the offer is then presented with the updated information.				

Test ID	F26	Category	Functional Integration	Severity	Major

Objective	Verify the functionality of marking an offer as sold
Steps	1. Open the business app and login 2. Press the “mark as sold” button for one of the people on one of the offers in the offers page 3. Verify that the available count of the offer is reduced by one 4. Verify that the person who picked up this offer disappears from the list
Expected	An offer can be marked as sold to someone successfully
Date-Result	- Date: 6/05/2024 - Outcome: Offer can be sold successfully. Although in our current implementation, there is no “Mark as sold” button but instead a check icon button, which serves the same purpose - Result: Success

Test ID	F27	Category	Functional Component	Severity	Major
Objective	Verify the contents and functionalities of the history page				
Steps	1. Open the business app and login 2. From the navigation bar go to the history page 3. Check that the past sales of the user are displayed with each sale in a separate card a. Check the existence of the following fields: star rating, price, sale date and time 4. Verify that the card expands upon pressing the arrow icon on its upper corner a. Check that the customer name and comment (or a text indicating no comment was made) is visible when the card is expanded 5. Verify that the card shrinks back upon pressing the arrow icon again				
Expected	The history page correctly displays the previous sales with their corresponding attributes				
Date-Result	- Date: 6/05/2024				

	- Outcome: Previous sales, with all of their corresponding attributes, are correctly displayed - Result: Success

Test ID	F28	Category	Functional Component	Severity	Major
Objective	Verify the contents and functionalities of the profile page				
Steps	1. Open the business app and login 2. Go to the profile page, from the navigation bar 3. Check that all of the following fields are present: business name, email, password, phone number, address a. Check that these fields correctly display the current attributes of the business (except the password field, which should hide its current value) b. Check that each of these fields have a pressable pencil icon next to them, for editing i. Press the pencil icon in any of these fields ii. Verify that the field becomes editable and a “Save” button appears 1. If it is the password field, verify that there is an additional “Enter old password” field iii. Change the value of the field and press the save button 1. If it is the password field, make sure it displays an error message if the old password is wrong, or the new password is invalid iv. Verify that the value of the field is now changed 4. Check that the profile picture of the business is present and valid a. Verify that there is a button for changing the profile picture b. Verify that the user is prompted to choose a picture from their gallery upon pressing the button c. Verify that the user can choose a picture from their gallery and upload it as their profile picture. i. If there is an error because the picture has an invalid attribute (wrong image format, size too large etc.) make sure that the reason of the error is displayed				
Expected	The profile page correctly displays all of its fields and they are functioning correctly (editable, saveable, responsive).				

Date-Result	- Date: 6/05/2024 - Outcome: All fields are displayed and function correctly - Result: Success

Test ID	F29	Category	Functional Component	Severity	Major
Objective	Verify the contents and functionalities of the analytics page				
Steps	1. Open the business app and login 2. Go to the analytics page, from the navigation bar 3. Check that a component of sliding cards with the title “Most popular items this month” is present a. Check that this component has a card with the title of the most popular item this month, and a description of how many of it has been sold this month b. Check that the next card is presented when the user swipes right, except for the last card c. Check that the previous card is presented when the user swipes left, except for the first card 4. Check that a “Sales chart” is present at the bottom of the page a. Check that this component has a drop down list of options such as “Monthly sales, daily sales, hourly sales” etc. b. Verify that a chart is presented, with the number of sales as the y-axis and the selected option (month, day, hour) as the x-axis.				
Expected	Contents of the analytics page are displayed and are functioning correctly				
Date-Result	- Date: 6/05/2024 - Outcome: Analytics page was not prepared by the time this test was conducted. As it is now, it won’t be included in the first iteration of Yicem. - Result: Failure				

Test ID	F30	Category	Functional	Severity	Critical

			Integration		
Objective	Verifying that a newly added offer is present in the customer application				
Steps	1. Open the business app and login as one of the valid business accounts 2. Create a new offer from the offers page 3. After the offer is successfully created and present, log out or close the application 4. Open the customer app and login with a valid customer account 5. In the customer app, find the business that created the new offering (search the business name in the businesses page) 6. Enter the business' page 7. Verify that the offer added in step 2 is present in the "Available offers" of the business				
Expected	A new offer added from the business application by a business is correctly visible and interactable from the customer application by a customer				
Date-Result	- Date: 6/05/2024 - Outcome: Customers can see the newly added offer instantly. - Result: Success				

Test ID	F31	Category	Functional Integration	Severity	Major
Objective	Verifying that a customer can see the changes in an offer of the business				
Steps	1. Open the customer app and log in as a valid customer 2. Enter the page of a business with at least one available offer 3. Take note of the name, description and price of one of the offers 4. Open the business app and log in as the business picked in step 2 5. Pick the offer noted in step 3 and edit one or more of these fields: name, description, price 6. Open the customer app and enter the same business page again 7. Check the offer noted in step 3. 8. Compare the current fields of the offer with the ones noted in step 3 a. Verify that the fields have changed correctly				
Expected	When a business makes changes on an offer from the business				

	application, the customer can see the changes from the customer application
Date-Result	- Date: 6/05/2024 - Outcome: Customers can successfully view all the edited fields. They just need to refresh the page. - Result: Success

Test ID	F32	Category	Functional Integration	Severity	Major
Objective	Verifying that a customer can see the changes in the profile of the business				
Steps	1. Open the customer app and log in as a valid customer 2. Enter the page of a business 3. Take note of the attributes of the business, such as its name, open hours, address, profile picture 4. Open the business app and log in as the business picked in step 2 5. Enter the profile page 6. Change one or more of these fields: business name, address, profile picture, open hours 7. Open the customer app and enter the same business page again 8. Compare the current fields of the business with the ones noted in step 3 a. Verify that the fields have changed correctly				
Expected	When a business makes changes on their profile, such as their name, address, profile picture etc., the customer can see these changes from the customer app				
Date-Result	- Date: 6/05/2024 - Outcome: Customers can see the edits made on the business's profile, such as address, name, profile picture, map location etc. - Result: Success				

Test ID	F33	Category	Functional Integration	Severity	Critical

Objective	Verifying the approval of a business
Steps	1. In the business app, register as a business by following the instructions of the test F20 2. Open the admin website and enter with a valid admin account 3. Find the application of the business in the pending applications segment and approve it 4. Open the business app again and go to the login page 5. Login with the credentials of the approved business 6. Verify that the business can now enter the application
Expected	After approving the business, they are no longer redirected to the “Your registration is being evaluated…” page and can enter the application
Date-Result	- Date: 12/05/2024 - Outcome: Registered business can be seen from the admin webpage and it can be approved. Business cannot login before approval but can login successfully after. - Result: Success

Test ID	F34	Category	Functional Integration	Severity	Major
Objective	Verifying the review functionality between apps				
Steps	1. Open the customer app and login as a customer 2. Leave a review on one of the past purchases, following the instructions in F12 3. Open the admin website and login as an admin 4. From the pending reviews section, verify that the review that was left in step 2 is present a. Approve the review 5. Open the business app and login as the business was reviewed in step 2 6. From the history segment, find the past purchase that was reviewed a. Expand the details of the purchase b. Verify that the review left in step 2 is visible with its rating				
Expected	The review that was left on the customer app is visible in the admin app, and after being approved in the admin app, it is visible in the business app				

Date-Result	- Date: 6/05/2024 - Outcome: Review functionality works correctly on both the customer and the business's end, but we made a change on the admin's end. Reviews are automatically sent, and visible, without the need for approval. Admins however, can delete a review when they see fit. - Result: Success

Test ID	F35	Category	Functional Integration	Severity	Major
Objective	Verifying the cancellation of a reservation across the customer and business apps				
Steps	1. Open the customer app and login as a customer 2. Make a reservation by following the steps of F10 3. Open the business app and login as the business which is the owner of the reserved offer 4. Verify that the reservation that was made in part 2 by the user is visible 5. Open the customer app again, with the credentials of the customer in step 1 6. Cancel the reservation that was made in part 2 7. Open the business app again, with the credentials of the business in step 3 8. Verify that the reservation is no longer visible				
Expected	A reservation disappears from the business page after it is canceled				
Date-Result	- Date: 12/05/2024 - Outcome: Canceling a reservation logic works correctly on both customer and business side. It is not visible after a customer canceled a reservation. - Result: Success				

Test ID	F36	Category	Functional Integration	Severity	Major

Objective	Verifying a complete transaction between the customer and the business

Steps	1. Open the customer app and login as a customer
	2. Make a reservation by following the steps of F10
	3. Open the business app and login as the business which is the owner of the reserved offer
	4. Mark the offer as sold for the customer in step 1
	5. Open the customer app again, with the credentials of the customer in step 1
	  a. Verify that the purchase is now in the “Past purchases” part in the user’s profile page
	6. Open the business app again, with the credentials of the business in step 3
	  a. Verify that the purchase is now in the “History” page

Expected	A reservation is displayed as a past purchase after the transaction is done, in both the customer and business applications
Date-Result	- Date: 12/05/2024 - Outcome: A reservation turns into a transaction successfully after the customer marks it as sold and it can be seen from both customer and business's history. - Result: Success

5.2 Non-functional Tests

In this category, there are multiple types of tests: security, usability, compatibility, installation, documentation, reliability and performance.

Test ID	N01	Category	Non-Functional Security	Severity	Major
Objective	Testing for password security				
Steps	1. Open the register or change password fields in one of the customer or business applications (repeat the test for both applications and both fields) 2. Enter a weak password (less than 6 characters, does not contain both letters and numbers) a. Check that the password guidelines are displayed near the password field b. Check that the password attempt is rejected if the password does not conform to the guidelines i. Verify that the user is notified of which guideline they did not conform to				
Expected	User is enforced to use a sufficiently complex and long password				
Date-Result	- Date: 12/05/2024 - Outcome: Application only enforce character count requirement. But other password guidelines are not enforced while typing password. - Result: Partial Success				

Test ID	N02	Category	Non-Functional Security	Severity	Critical
Objective	Testing for security of requests				
Steps	1. Choose a method to display the packets/requests that are sent over the web while using the application, such as a packet sniffer, browser console etc. 2. Start watching the requests while using either the business or customer application (repeat the test for both) a. Verify that user's credentials are not visible in the request				

	while logging in b. Verify that user's credentials are not visible in the request header in any of the subsequent actions after logging in
Expected	User's credentials are hidden and they can't be traced from their requests, while using the application
Date-Result	- Date: 12/05/2024 - Outcome: User's credentials are encrypted in their JWT token and it cannot be retrieved from a bare request. It can only be decrypted using the JWT secret which is stored in the backend server. - Result: Success

Test ID	N03	Category	Non-Functional Security	Severity	Major
Objective	Testing for API security				
Steps	1. For each API endpoint, send a valid request but without an authorization token in the request header a. Verify that for the endpoints that require an authorization token, the attempt is rejected i. For each of these endpoints, also send a request with a valid authorization token but with a different role (for example, a customer token for a business API) 1. Verify that the attempt is still rejected				
Expected	Protected endpoints can't be reached without a valid authorization token with a valid role				
Date-Result	- Date: 12/05/2024 - Outcome: Protected endpoints cannot be reached without a valid JWT token and unauthorized attempts are rejected successfully. - Result: Success				

Test ID	N04	Category	Non-Functional Compatibility	Severity	Major

Objective	Test that the applications are compatible for different operating systems
Steps	1. Test both of the customer and business applications on an iOS phone and an Android phone 2. Verify that both of the applications render and display correctly on both operating systems 3. Verify that both of the applications function correctly on both operating systems, by testing the key functionalities
Expected	The applications should be compatible with its display and functionality for both iOS and Android phones
Date-Result	- Date: 12/05/2024 - Outcome: Key functionalities of the applications are tested on both iOS and Android mobile phones. Applications render and display correctly on these systems. Both of the applications run successfully on both iOS and Android operating systems. - Result: Success

Test ID	N05	Category	Non-Functional Compatibility	Severity	Major
Objective	Test that the applications are compatible for different hardware, same operating system				
Steps	1. Test both of the customer and business applications on multiple phones with the same operating system but different hardware and different screen sizes 2. Verify that both of the applications display correctly and the component sizes/placements scale according to the screen size 3. Verify that both of the applications function correctly on varying hardware, by testing the key functionalities a. If the application has problems for one of the devices, note the device with its specifications				
Expected	The applications should be compatible with its display and functionality for different hardware. If there is a problem with a device, evaluate the age, specifications, model etc. of the device to determine whether the problem is acceptable or not.				

Date-Result	- Date: 07/05/2024 - Outcome: Key functionalities of the applications are tested on the same operating system for different phones. Applications render and display correctly on these systems. Both of the applications run successfully on different models of Android phones. - Result: Success

Test ID	N06	Category	Non-Functional Usability	Severity	Major
Objective	Test the usability of the applications from end users' perspective				
Steps	1. Gather two user groups: one group consisting of people who have prior experience with the application and how to use it (such as the developers), and other group consisting of people with no prior knowledge or experience with the application 2. Make both of the groups use the application and complete a given set of key tasks 3. Observe the experiences of both of the groups a. Measure the time it takes to do key tasks for each group b. Take a note of the areas in which the inexperienced group is struggling c. Gather feedback from both of the groups on their experiences, about what to improve, what to change etc.				
Expected	Neither group should struggle too much with understanding the application and both groups should be able to complete the tasks. If the inexperienced group struggles a lot in some areas, consider making necessary changes in certain areas according to their feedback, and conduct the test again				
Date-Result	- Date: 07/05/2024 - Outcome: Application tested by one of developers's parents. Both of the parents completed the set of tasks without struggling. - Result: Success				

Test ID	N07	Category	Non-Functional Usability	Severity	Minor
Objective	Evaluate the style, accessibility, understandability and readability of the contents in the applications				
Steps	1. Open either the customer or the business application (repeat for both) 2. Navigate through the pages and subpages. For each page: a. Verify that all content is easily readable, with assessing the color contrasts, fonts, text sizes, and overall style b. Verify that all content is presented in a clear and organized style, with appropriate headings, lists, components etc. c. Check that the style and organization has consistency across different pages in the same application				
Expected	Each application should follow a consistent style and be easily accessible, understandable and readable				
Date-Result	- Date: 07/05/2024 - Outcome: Both of the applications provided consistent text style and font. Texts in the applications are readable and understandable. - Result: Success				

Test ID	N08	Category	Non-Functional Usability	Severity	Minor
Objective	Evaluate the ease of navigation in the customer and business applications				
Steps	1. Open either of customer or business applications (repeat for both) 2. Keep track of how many presses it takes to reach each type of page and subpage from the homepage 3. Keep track of how many presses it takes to complete each common action (making a reservation, changing your password, leaving a review etc.) from its starting page				
Expected	To ensure ease of navigation, each page and subpage should be reachable from the homepage within 3-4 presses, and each common action should be completable in 4-5 presses from its starting page				

Date-Result	- Date: 07/05/2024 - Outcome: The navigation throughout the application, usually takes 3 presses (4 at most). The actions could typically be completed in 4-5 clicks, depending on the task at hand. - Result: Success

Test ID	N09	Category	Non-Functional Performance	Severity	Major
Objective	Test response and loading times under normal load in customer and business applications				
Steps	1. Open either customer or business application (repeat for both) at a time when the servers are not overloaded 2. Browse through the pages and subpages and execute common functionalities (such as making a reservation, leaving a review, logging in/out, creating an offer etc.) a. Measure the loading time while browsing through different pages b. Measure the loading times and response times while executing common functionalities				
Expected	Time it takes to fully load a page should not exceed 3 seconds and the loading times for common functionalities should not exceed 5 seconds				
Date-Result	- Date: 08/05/2024 - Outcome: The time to load a page typically takes around 2 seconds, while the time for backend response usually ranges from 3 to 4 seconds. - Result: Success				

Test ID	N10	Category	Non-Functional Performance	Severity	Major
Objective	Test if the application can handle concurrent requests of the same type				

Steps	1. Open either customer or business application (repeat for both) in multiple devices 2. Do actions that would result in a request being sent to the server, and a response being sent from the server (such as logging in, leaving a review, making a reservation, sending a report, creating an offer, changing the business profile picture etc.) a. Now do the exact same action at the same time on multiple devices b. Verify that the action is completed without any errors or inconsistencies
Expected	The application can handle multiple requests of the same type concurrently without any errors or inconsistencies
Date-Result	- Date: 08/05/2024 - Outcome: The application handles multiple requests without error. - Result: Success

Test ID	N11	Category	Non-Functional Reliability	Severity	Major
Objective	Test reservation consistency in the case of a loss of connection				
Steps	1. Open the customer app and login as a customer 2. Choose a business with available offers 3. Make a reservation from the chosen business, but while loading the reservation, deliberately turn off the internet connection of the device 4. Turn the internet connection back on 5. From the customer app, check if the reservation is made 6. Open the business app and login as the business chosen in step 2 7. Check if the reservation is made				
Expected	The state of the reservation in both apps should be consistent: either the reservation is made or not made in both the customer and the business				
Date-Result	- Date: 08/05/2024 - Outcome: When the internet connection is interrupted in the middle of reservation, both customer and business application does not				

	see the reservation. - Result: Success

Test ID	N12	Category	Non-Functional Reliability	Severity	Major
Objective	Test the application behavior in the case of a loss of connection				
Steps	1. Open either one of the customer app or the business app and login (repeat the test for both) 2. Navigate through the application and perform actions that require internet connection (can be as simple as trying to load a new page or pressing a functional button) 3. Turn off the internet connection 4. Continue with the actions in step 2 5. Turn the internet connection back on 6. Continue with the actions in step 2				
Expected	The application should provide error messages in the case of a loss of connection, and should go back to functioning normally after the connection is re-established				
Date-Result	- Date: 08/05/2024 - Outcome: The proper error message is displayed. - Result: Success				

Test ID	N13	Category	Non-Functional Reliability	Severity	Major
Objective	Test the application behavior in the case of a server outage				
Steps	1. Open either one of the customer app or the business app and login (repeat the test for both) 2. Navigate through the application and perform actions that require the presence of the server (any action that would result in a request being sent to server) 3. Shut down the server				

	4. Continue with the actions in step 2 5. Turn the server back on 6. Continue with the actions in step 2
Expected	The application should notify the user that there is a server outage, when the server is offline. When the server is back on, the user should be able to continue using the application normally
Date-Result	- Date: 08/05/2024 - Outcome: The proper error message is displayed. Users were able to use the application after the server was back online. - Result: Success

Test ID	N14	Category	Non-Functional Installation	Severity	Critical
Objective	Test the installation process in Android and iOS				
Steps	*(this test is meant to be done after the app is published)* 1. In an Android phone, search the application from the Google Play Store 2. Verify that the app is listed in the search results 3. Press the “Install” button a. Wait for the app to download and install 4. Open the app and verify that it launches without any issues 5. Repeat steps 1-4 with an iPhone and App Store				
Expected	The app is successfully installed and launched in both Android and iOS				
Date-Result	- Date: 07/05/2024 - Outcome: Yicem is not published yet in Google Play Store and App Store. Therefore the test is not applicable. - Result: Failure				

Test ID	N15	Category	Non-Functional Documentation	Severity	Minor
Objective	Reviewing the application’s documentation				
Steps	1. Read all the provided documentation of the application, such as user manuals, guides etc. 2. Verify that it is clear and easily understandable 3. Verify that it covers all of the necessary functionalities and use cases 4. Verify that the documentation covers necessary troubleshooting guidelines, for possible problems that the user can encounter 5. Optionally, ask a person who has no prior knowledge with the application to also read the documentation and get their feedback				
Expected	Documentation should be clear, understandable, comprehensive, and helpful; both from the developers’ and end users’ perspective				
Date-Result	- Date: 09/05/2024 - Outcome: The user guide is in process of creation, therefore, this test is not applicable right now. However, a user guide is prepared considering these criterias. - Result: Partially Failed				

6. Maintenance Plan and Details

As it is now, Yicem is a finished and working app with all of its major functionalities, but it has not been published to app stores to download and use yet. This will be the next step in Yicem's path, and this task may require considerable amounts of maintenance, as more and more people and businesses begin to use the system and it grows in scale. We already have an admin board (admin website) prepared for the maintenance team to use. This board will be used for functionalities such as checking user's feedback, reports; deleting users (business or customer) when needed, removing harmful business reviews if necessary, checking business information and approving pending business registration requests. With these functionalities (and any possible functionality in the admin board that may be added later), the management team will ensure that the general system of Yicem works as intended.

We will also keep doing tests, working on fixing any newfound bugs, and adding new functionalities to provide the best user experience for everyone. For assigning and managing tasks, we plan to continue using the issue tracking system of Github (also known as Github Projects), although as the project grows on scale, we may migrate to a more enterprise-level issue tracking system. After the app is published, we will release these fixes and new features as updates in app stores.

One thing we might need to consider as the app grows on scale is the database of our system. Our application is deployed on a free-tier Mongo database right now, as the data we keep is not nearly large enough to cause any problems at this stage, but of course that might change with increasing number of users. In this case, we can easily update to one of the paid-tier Mongo databases to get more space, without causing any problems in our architecture.

Another important thing that we need to maintain is the packages and libraries in the application frontend. Unlike the backend server, our frontend code uses many public libraries associated with React native and Expo, which are used for building specific components and general styling purposes. As convenient as it is, these libraries may get updated or deprecated over time, which can possibly cause problems in how the application behaves or looks. Therefore, it will be our duty to keep the libraries up to date, migrate to a new library when necessary, and test there is no loss of functionality.

7. Other Project Elements

7.1 Consideration of Various Factors in Engineering Design

7.1.1 Public Health Considerations

As the core idea of Yicem consists of repurposing the leftover food in cafes, bakeries and restaurants, one important question that can be asked is the healthiness of the leftover food. It may be acceptable for a business to sell a product that is not entirely fresh (for example, a pastry made in the morning and sold in the evening) as it poses no threat to consume, but we cannot accept a scenario where a business sells food that has gone bad, or is not fit for human consumption and could be dangerous or unhealthy to eat. Although, we as developers cannot directly control the behavior of the businesses and completely prevent a business from ever doing something like this, we did take precautionary actions for this situation. The users will be able to report a business and explain their reason or the problem they've faced, from the business' page in the application. User reports will be reviewed by the developer team and we will take necessary actions depending on the situation, such as warning the business about the situation in mild cases, or removing the business from the application in more severe cases. Apart from that, another health consideration is the possibility of allergens in the products that are sold. For this case, we will provide a description part while creating an offer in the business application, so the businesses will be able to list any allergens or similar necessary information about the products they're offering.

7.1.2 Public Security and Safety Considerations

As Yicem requires the customers and the businesses to register to our system, we will be necessarily holding personal and private information, such as the users' emails and passwords. It is important for this information to be kept secure, so we will use industry-standard encryption methods to ensure they are not reachable by any third party, as explained in more detail in previous sections of this report. Moreover, we ourselves will not share any of this information with any third party either, as we value people's security and safety. Moreover, since Yicem is a mobile application, it requires some permissions from the user's phone, such as storage permission to change profile photos with a picture from their gallery and location permissions for the map functionality of the application. Yicem will keep the permissions it requires to a minimum, and will not access anything that is not necessary for any functionality. Moreover, the user will be in control of which accesses they're granting.

For example, the user may not allow the application to access their location if they do not wish to use the map functionality of Yicem.

7.1.3 Public Welfare Considerations

Yicem does not have any negative effects on public welfare in a direct or indirect way, so we did not need to take any precautionary action in this area. In fact, Yicem aims to improve public welfare by enabling people to access food in an easy and affordable way.

7.1.4 Global Considerations

Yicem is planned to launch on university campuses in Turkey, starting locally in Bilkent and later across Ankara. Depending on the level of success, it is within our plans to grow larger and serve in more cities across Turkey without confining Yicem within university campuses. However, currently it is not in our short-term or long-term plans to serve globally.

7.1.5 Cultural Considerations

We plan Yicem to be launched initially in university campuses starting from Bilkent, and we know that university campuses are culturally diverse environments. For university students from different nations to use our application easily, we plan to launch Yicem in English, as the vast majority of our initial user base (Bilkent students and personnel) will know English. Adding Turkish language support will be our next priority. Furthermore, as Yicem grows larger and becomes available in more areas of Turkey, we may need to add new language options to reach even more people. For that reason, we will make the design flexible for adding new language options.

7.1.6 Environmental Considerations

One of our main motivations for the idea of Yicem was sustainability, specifically by reducing food waste. Food waste is a major environmental issue, and our goal is to fight this issue within our capabilities by creating Yicem. Therefore sustainability, in the category of reducing food waste, is a major consideration for us and is one of the reasons why we chose to do this project.

7.1.7 Economic Considerations

Another main motivation for the idea of Yicem was allowing people access to food in a more affordable way, especially for the university students who have limited budgets. We

believe Yicem will be economically beneficial for all of its user types: as the businesses will be able to sell their leftovers instead of throwing it away, and customers will be able to pick up food for cheaper prices. So the economic benefit of our users is a major consideration for us, and for this reason we do not plan to put any feature of Yicem behind a paywall. As Yicem grows, we may need to use in-app advertisements for monetizing the application, but that will not be included in the initial release.

Factor	**Effect Level (out of 10)**	**Short Explanation**
Public Health	**7**	We acknowledge the health risks that might arise from selling leftover food, and we will take precautions accordingly
Public Security and Safety	**6**	We aim to minimize the amount of private/personal data we will take, and we will keep this data safe and secure
Public Welfare	**3**	We do not think the app has any potential negative effects on public welfare, we believe Yicem will have positive effects on this area
Global	**0**	Yicem will be based on Turkey for the foreseeable future
Cultural	**4**	We could not find any major cultural factors except the language options
Environmental	**10**	One of the main goals of Yicem is reducing food waste to improve sustainability
Economic	**10**	One of the main goals of Yicem is providing economic benefit for both our customers and businesses

Table 1: Factors and their effects

7.2 Ethics and Professional Responsibilities

As developers of Yicem, the major area in professional responsibility for us was being responsible towards each other. With weekly meetings and frequent communication, we made sure that we shared the tasks and workload among each other fairly, and we explained and demonstrated our contributions. We made sure to disclose to each other when any one of us was going to be unavailable for a duration of time, for any personal reason or in a particularly busy week for some members; so we were able to distribute tasks accordingly to not halt the progress of the project. For our professional responsibilities to the software industry in general, we made sure to not use any non-publicly available (copyrighted, private etc.) content in our application. The entire project was built on either free products published by software companies, or community packages and libraries that were open to everyone.

For our ethical responsibilities, we had many things to consider, which were explained in detail in section 7.1 of this report. During the development, we always kept these factors in mind in our decisions, and acted accordingly in order to create an app that is ethical, fair and safe to use for everyone. To give some specific examples, these decisions include things such as: keeping passwords encrypted in our database, allowing the users to report businesses, not allowing the users to edit their reviews (as some harmful agents might abuse this functionality to edit a previously approved review into something harmful or profane) and so on. Also, we made sure to ask for user's consent in functionalities where we need to reach their device content, such as their location in maps pages or their gallery in profile picture picking functionality. With this, the user is free to not disclose their gallery or map information to us if they don't wish to use these functionalities.

7.3 Teamwork Details

7.3.1 Contributing and functioning effectively on the team

For functional and effective teamwork, we had several applications and a culture among the teammates. Firstly, we ensured that all important decisions were made together. We had regular weekly meetings, mostly on Mondays, to inform each other about the updates and plans. We aimed to distribute the upcoming workload, check each other's progress, and solve problems we faced in these sessions. For our online meetings, we used Zoom. On the other hand, as a group, we frequently gathered in the University library to work on some important points that needed the attention of multiple teammates. In these

sessions, we aimed to engage in face-to-face communication, which we considered more productive, especially for these kinds of tasks.

For bigger tasks, especially the ones that required peer-to-peer communication, we distributed our work packages to groups of two teammates to improve our ability to double-check, organize, and help each other with their problems.

7.3.2 Helping creating a collaborative and inclusive environment

We use Zoom to conduct our online meetings, we are also actively using WhatsApp for planning our meetings and other smaller tasks. We use the GitHub Projects Tool to open issues about our work packages. Each teammate creates the tasks that he is planning to work on while we assign tasks to each other as well. While communicating and organizing our workload, we give utmost importance to effective and clear communication between teammates

7.3.3 Taking lead role and sharing leadership on the team

Our structure as a team is rather democratic. Everyone is responsible for their work package/task in the GitHub Projects Tool as we use it. Leadership roles are distributed for work packages among the teammates. Our team mates are proactive and respectful to each other in situations that require a figure to do the regulation, so the opinions of everyone are considered in these kinds of situations.

WP#	Work package title	Leader	Members involved
WP1	Project Specification Report	Ege Çenberci	All Members
WP2	Analysis and Requirements Report	Ahmet Alperen Yılmazyıldız	All Members
WP3	Frontend Development	Yağız Özkarahan	Yağız Özkarahan, Bilgehan Sandıkcı
WP4	Backend Development	Ömer Burak Doğan	Ege Çenberci, Ahmet Alperen

			Yılmazyıldız, Ömer Burak Doğan
WP5	Demo and Presentation	Ahmet Alperen Yılmazyıldız	All Members
WP6	Detailed Design Report	Bilgehan Sandıkcı	All Members
WP7	Final Report	Yağız Özkarahan	All Members
WP8	Beta Testing & App Launch	Ömer Burak Doğan	All Members
WP9	Final Demo	Ege Çenberci	All Members

Table 2: List of work packages

7.4 New Knowledge Acquired and Applied

One notable aspect of our project group was the collective lack of extensive experience or specialized knowledge required for a project of this scale, particularly within the specific areas that everyone has been assigned to. Of course, the backend team had previously worked on building a backend, and the frontend team had frontend knowledge and experience, but none of us worked on these roles for a long duration in a full scale mobile application project. For this reason, all of us had to learn how to do a lot of things from scratch. The backend team had to learn how to build a functioning server with a deployed database, and they learned the specifics and details of Java Spring and Mongo, and also the details of authentication/authorization. The frontend team, despite doing React web projects before, had never done a mobile application; so they had to learn libraries and tools such as React Native, Expo, and the specifics of making a mobile application that is responsive and live. Although this lack of previous experience slowed down our progress for a while, especially during the midterm period, in the end we managed to learn and apply newfound knowledge, and thus created two fully functioning mobile applications.

8. Conclusion and Future Work

The Yicem project was able to achieve most of its original goals. We implemented an application allowing businesses to sell surplus items at discounted prices within Bilkent University. However, we could not meet all our deadlines, such as deploying the application to the Google Play Store.

We encountered many engineering challenges during our project and gained vital experience working as a team. We have observed first-hand how easy it is for mistakes to slip in between the cracks, an error in communication to be made, or how problematic a lack of decisiveness can be. However, our senior project experience taught us how to set up systems and feedback mechanisms to minimize the probability of such errors occurring.

As for the future of Yicem, firstly, the tasks of the missed deadlines should be finished, such as completing the app's deployment on the app store. Moreover, during implementation, our team had to make some feature changes to Yicem, like the Analytics page. For example, a goal that could be set to improve Yicem further would be to generate an advisory tool within the Analytics page that uses an LLM to give the businesses curated suggestions to improve sales. Another more straightforward addition could be implementing profile pictures to buyer user profiles. Currently, Yicem is not a perfect application, and features that were discarded to save time are good starting points for improving Yicem.

9. References

[1] FAO, “Turkey’s National Strategy Document On Prevention, Reduction And Monitoring Of Food Loss And Waste And Its Action Plan.” *Food And Agriculture Organization of the United Nations*, pages 9, 12, 2020.