National College of Ireland

< Computing Project (BSHCSDE4) >

<Software Development>

<Academic Year 2022/2023>

<Jiashun Chen>

<x19134118>

< x19134118@student.ncirl.ie>

< Mealtime: Mobile App Development >

Technical Report

Contents

Executive Summary

The goal of this project is to help the catering industry increase profits by lowering commission fees charged by third-party platforms such as Just-eat or UberEATS. Catering owners are often faced with high commission fees, which can be a significant drain on their business. By creating a food ordering mobile app that eliminates the need for such platforms, the app can provide a cost-effective solution that reduces business costs for catering owners and provides greater control over their business management.

In addition to reducing commission fees, the app also includes a range of features that make it an ideal solution for catering businesses. One such feature is the use of cloud printing technology, which significantly reduces customer complaints by ensuring that no orders are lost or delayed. All orders are sent directly to kitchen printers without delay, and in the

event of an internet issue, the orders will be automatically reprinted. This feature adds an extra level of reliability to the system and helps to maintain a smooth and seamless ordering process.

Furthermore, the app provides catering owners with greater freedom to manage their business at their fingertips. The app allows catering owners to manage their menus, pricing, and promotions, as well as track their sales and customer data. The app also allows catering owners to communicate directly with their customers, providing a personalized and efficient service.

In summary, this food ordering mobile app provides a comprehensive solution to the catering industry, enabling catering owners to manage their businesses more efficiently, reduce costs, and enhance the customer experience by reducing errors and delays in the ordering process. It is an ideal solution for catering businesses looking to improve their profitability and customer satisfaction.

Introduction

1.1. Background

In today's fast-paced world, many customers prefer to order food online due to its convenience. However, the commission fees charged by ordering platforms can be exorbitantly high, resulting in customers paying significantly more than if they were to order in-store. As a result, catering owners face a challenge in reducing the cost-of-service commission and increasing their profits.

1.2. Aims

Our food ordering mobile app provides a cost-effective solution to customers, reducing the commission fees they pay, and enabling catering owners to increase their profits. By incorporating third-party payment systems and cloud printing thermal printers, we can ensure a seamless and reliable ordering experience for both customers and catering owners.

To address the details, our food ordering mobile app integrates third-party payment systems, such as Stripe, to process online payments. By using Stripe, with a low fee of 1.4% + €0.25, we can significantly reduce the service fees charged by platforms like Just-eat, which can charge up to 14% service fees.

1.3. Technology

To achieve the goals of this project, we will be using specific technologies that enable us to build a high-quality food ordering mobile app. For the backend, we have selected Spring Boot, a popular Java-based framework that allows us to build a restful API capable of handling CRUD operations on the PostgreSQL database. With Spring Boot, we can ensure that the app is scalable, reliable, and performs well, meeting the demands of catering owners and their customers. [1]

For the frontend, we will use React Native along with Expo, an open-source platform for creating universal native apps for Android, iOS, and the web with JavaScript and React. React Native provides us with a framework to create a user interface that is intuitive, responsive, and user-friendly. By using Expo, we can streamline the development process and reduce development time, enabling us to deliver the app to market quickly.[2]

Overall, our use of Spring Boot, React Native, and Expo will enable us to create a powerful, reliable, and user-friendly food ordering mobile app that meets the needs of both catering owners and their customers. With this technology stack, we can ensure that the app is scalable, performant, and accessible across multiple platforms, providing a seamless experience for all users.[3]

1.4. Structure

This document is a technical report for the development of a food ordering mobile app. The document is structured as follows:

Executive Summary: This section provides a summary of the project's objectives, technology, and structure.

Introduction: In this section, the background, aims, technology, and structure of the project are discussed.

System: This section focuses on the requirements, design, architecture, implementation, graphical user interface, testing, and evaluation of the system.

Conclusions: This section summarizes the key findings and outcomes of the project.

Further Development or Research: This section discusses possible future developments or research directions for the project.

References: This section lists all the sources cited in the document.

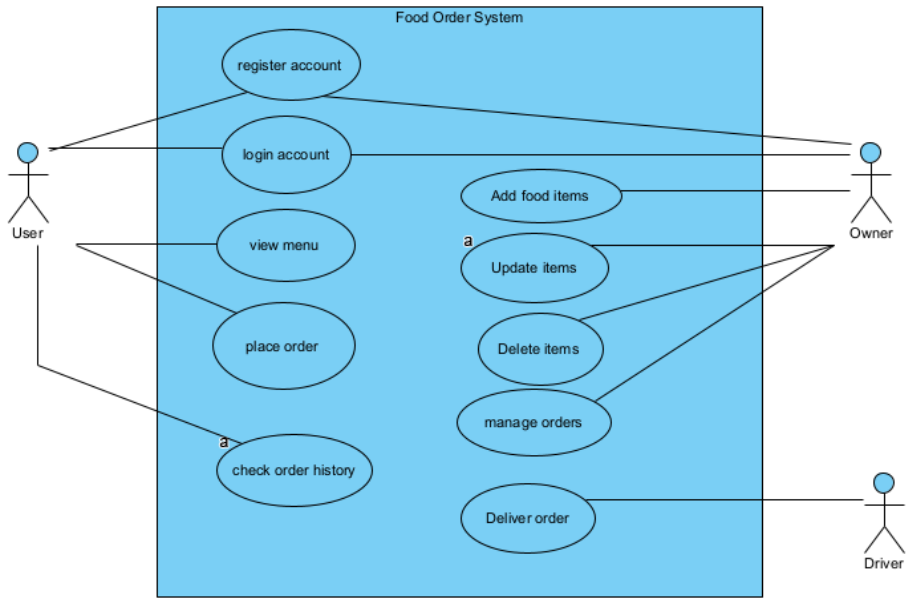
Appendices: This section includes the project proposal, ethics approval application (if required), reflective journals, invention disclosure form (if completed), and other materials used.

2.0 System

2.1. Requirements

2.1.1. Functional Requirements

2.1.1.1. Use Case Diagram



2.1.1.2. Requirement 1: User registration

2.1.1.3. Description & Priority

The requirement for users to be able to register an account has a high priority in the overall system. Without this functionality, users would not be able to access the features of the app or place orders. Priority: High

2.1.1.4. Use Case

UC1: Register account

Scope

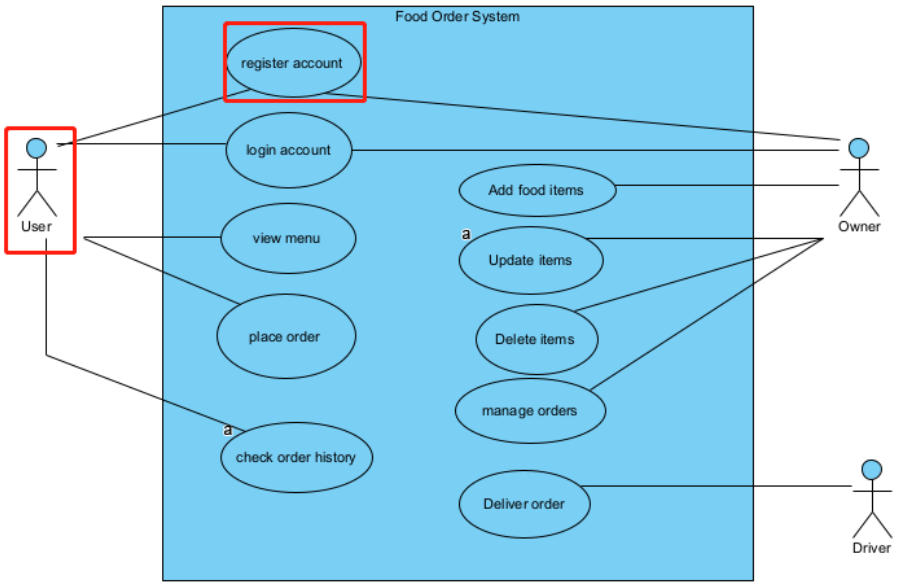
The scope of this use case is to register account for users.

Description

This use case describes the registration in system.

Use Case Diagram

Diagram should highlight actors and uses cases……..



Flow Description

Precondition

The user has opened the mobile app.

The user has not yet registered for an account with the system.

Activation

This use case starts when a user accessed the registration page.

Main flow

1. The user accesses the registration page on the mobile app.
2. The system presents a form for the customer to fill in personal information.
3. The user fills in the form with personal information and clicks the "Register" button.
4. The system validates the information entered by user, if all are validated, system creates an account for the customer and stores their information in the database. [A1 Form validation fail]

Alternate flow

A1: Form validation fail

1. The system shows error message.
2. The user corrects the input value.
3. Continue step 3 of main flow.

Exceptional flow

Termination

The user closes the app.

Post condition

The user has successfully registered for an account with the system and can now log in to account.

2.1.1.5. Requirement 2: User login

2.1.1.6. Description & Priority

Priority: High

This requirement allows users to log in to their account using their registered email and password. The login feature is essential to the overall system as it ensures the security of the user's account and their personal information. It also provides a personalized experience for the user, allowing them to view their order history, saved addresses, and check order cart.

2.1.1.7. Use Case

UC2: user login

Scope

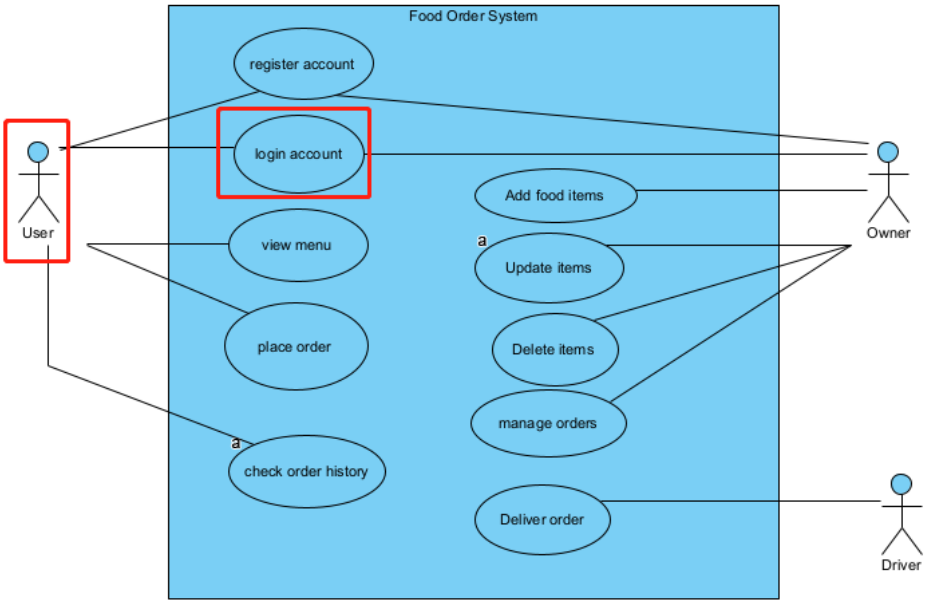
The scope of this use case is for users to login to their accounts

Description

This use case describes the process of a customer logging into their account in the food ordering mobile app.

Use Case Diagram

Diagram should highlight actors and uses cases……..



Flow Description

Precondition

The user must have already created an account.

The user must have access to the internet.

The user must have the app installed on their device.

Activation

This use case starts when a user accessed to the login page.

Main flow

1. The user opens the mobile app of the system.
2. The system presents a login screen with fields for the user's email and password. [A1 Password reset]
3. The user enters their email and password and clicks the "Login" button.
4. The system verifies the email and password combination.
5. If the email and password are correct, the system logs the customer into their account and displays the main screen of the app.
6. If the email and password are incorrect, [A2 Password incorrect]

Alternate flow

A1 : Password reset

1. The user clicks reset password button.
2. The system direct to reset page with email address field.
3. The user input registered email address and click “Reset”.
4. The system sends a password reset link to user’s email.
5. The user opens the link and enter new password, click “submit”.
6. The system updates user’s password in database.

7. The system shows password reset success message on the page.

A2: Password incorrect

1. The system shows email address and password incorrect.
2. The user input correct email address and password.
3. Continue step 3 of main flow.

Exceptional flow

Termination

The user closes the app.

Post condition

The user is logged into their account and can access their profile information, order history, and other features.

2.1.1.8. Requirement 3: Place order

2.1.1.9. Description & Priority

Priority: High

The "Place Order" requirement is a critical functional requirement for the food ordering mobile app as it is the primary action that customers will take when using the app. The priority for this requirement is high, as it directly impacts the app's usability and user satisfaction.

2.1.1.10. Use Case

UC3: Place order

Scope

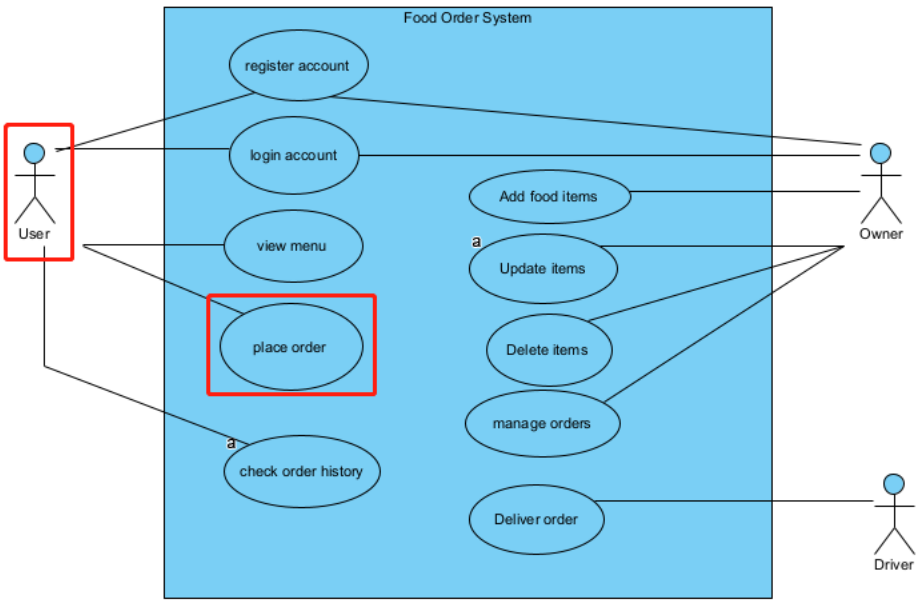
The scope of this use case is for users to place order from app.

Description

This use case describes the steps to place order for a user.

Use Case Diagram

Diagram should highlight actors and uses cases……..



Flow Description

Precondition

The user has downloaded and installed the mobile app on their device.

The user has created an account and is logged in to the mobile app.

The restaurant owner has listed its menu on the mobile app.

Activation

This use case starts when a user desires to order food.

Main flow

1. The user opens the mobile app and browses the menu.
2. The user selects the desired food items and adds them to shopping cart.
3. The user reviews the order and confirms the total price.
4. The user proceeds to the checkout page and enters the delivery address and payment information. [A1 Collection order]
5. The user reviews the order details and confirms the payment.
6. The mobile app sends the order to the system.
7. The system returns payment success message and clear the shopping cart.

Alternate flow

A1: Collection order

1. The user selects the collection order option.
2. The user reviews the order and confirm the total price.
3. Continue step 4 in the main flow.

Exceptional flow

Termination

The user closes the app.

Post condition

The customer receives the ordered food.

The restaurant receives the payment.

2.1.1.11. Requirement 4: Add food items

2.1.1.12. Description & Priority

Priority: High

The "add food item" requirement is a critical feature as it allows catering owners to add, edit, and remove food items from their menus. This functionality is essential as it allows business owners to keep their menus up-to-date and accurate, giving customers a clear and concise understanding of what is available for purchase. The priority for this requirement is high as it is integral to the overall functionality of the app.

2.1.1.13. Use Case

UC4: Add food items

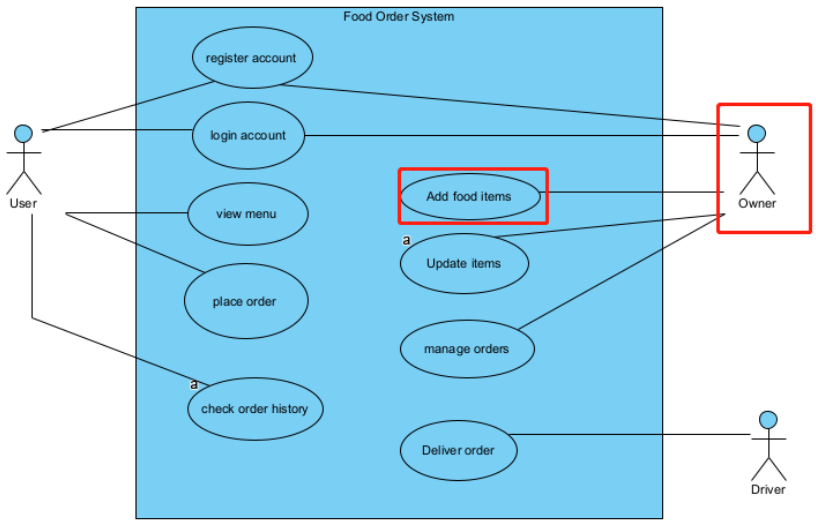
Scope

The scope of this use case is to add food items to online menu.

Description

This use case describes the steps to add food items by catering owners.

Use Case Diagram



Flow Description

Precondition

The restaurant owner must be logged in to system.

Activation

This use case starts when an owner desires to update menu.

Main flow

1. The restaurant owner selects the "Add Food Item" feature.
2. The system displays a form for adding a new food item.
3. The restaurant owner enters the details of the new food item, including the name, description, price, and category. [A1 Invalid information]
4. The restaurant owner uploads a photo of the new food item.
5. The restaurant owner submits the form.
6. The system adds the new food item to the database.
7. The system displays a list of food items.

Alternate flow

A1 : Invalid information

1. The system shows invalid or incomplete information message.
2. The user reviews the information and correct them.
3. Continue step 4 in the main flow.

Exceptional flow

Termination

Owner logout from the system.

Post condition

The new food item is added to the menu and is available for customers to order.

2.1.1.14. Requirement 5: Manage orders

2.1.1.15. Description & Priority

Priority: High

This functional requirement allows owners to view and process incoming orders, update the order status, and communicate with customers. It is critical to the success of the food ordering mobile app as it ensures that orders are processed efficiently and accurately, resulting in a positive customer experience.

2.1.1.16. Use Case

UC5: Manage orders

Scope

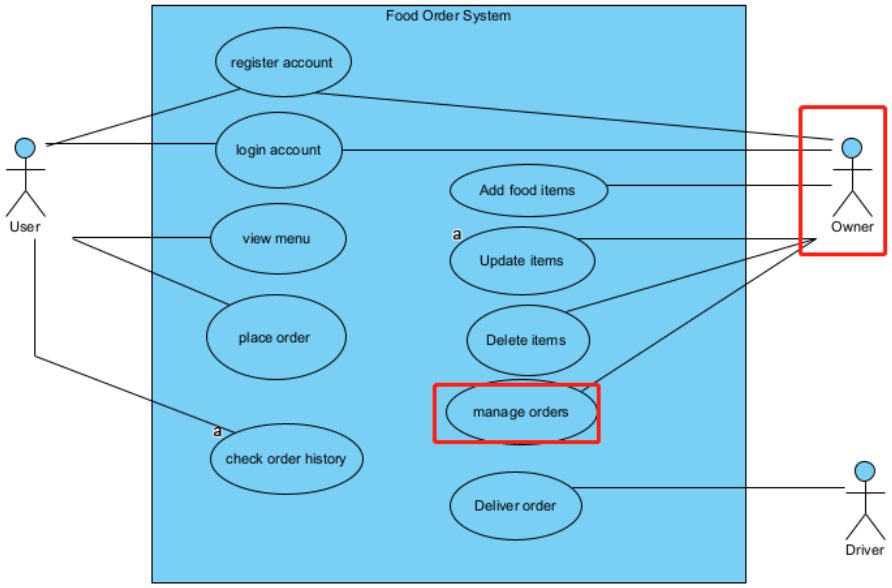
The scope of this use case is to manage orders from users.

Description

This use case describes the steps to manage online order.

Use Case Diagram

Diagram should highlight actors and uses cases……..



Flow Description

Precondition

The restaurant owner must be logged in to system.

Activation

This use case starts when delivery orders received.

Main flow

1. The system prints out the pending orders.
2. The orders send to kitchen for cooking. [A1 cancel order]
3. When it is ready for delivery, the owner handles food order package to driver.
4. The driver deliver order to the address provided by user.

Alternate flow

A1: cancel order

1. The user call to the owner and want to cancel the order.
2. The owner notices kitchen not to prepare the cancelled order.

3. The owner changes the status of order to be cancelled.

Exceptional flow

Termination

Owner logout from the system.

Post condition

The status of the order is cancelled in the system.

2.1.1.17. Requirement 6: Delete items

2.1.1.18. Description & Priority

Priority: medium

This requirement is essential as it allows business owners to keep their menus up-to-date and accurate, giving customers a clear understanding of what is available for purchase. Failing to remove discontinued or unavailable items can lead to customer dissatisfaction and order errors.

2.1.1.19. Use Case

UC6: Delete items

Scope

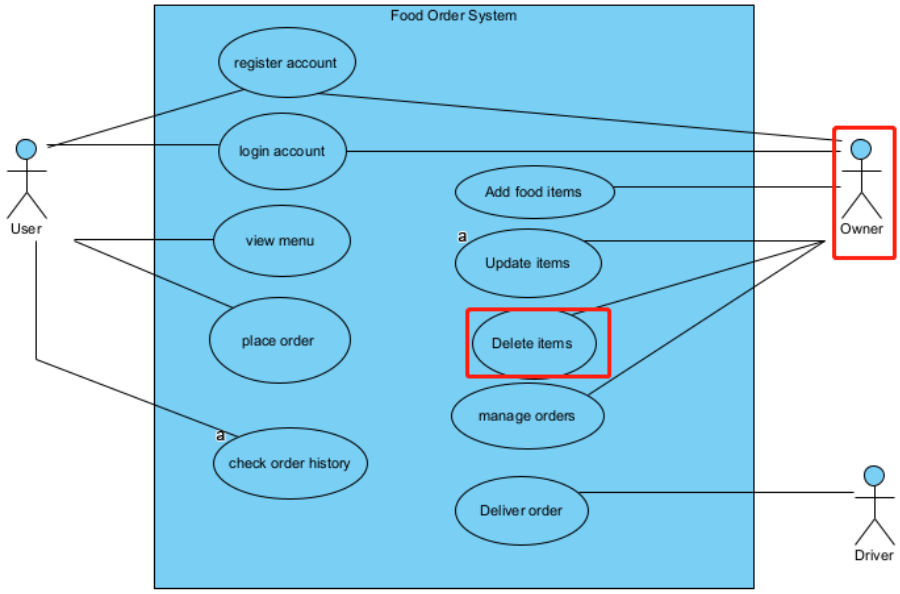
The scope of this use case is to delete unused food items from system.

Description

This use case describes the steps to delete food items by catering owners.

Use Case Diagram

Diagram should highlight actors and uses cases……..



Flow Description

Precondition

The restaurant owner must be logged in to system.

Activation

This use case starts when a food item not long used.

Main flow

1. The restaurant owner selects the Food Items to delete.
2. The restaurant owner clicks delete button. [A1 Archive items]
3. The system deletes the selected items from database and display delete success message.
4. The system displays a list of remaining food items.

Alternate flow

A1: Archive items

1. The system shows error: “items in use, can’t be delete”.
2. The owner chooses archive option to hide items.
3. The system archives items.

Exceptional flow

Termination

Owner logout from the system.

Post condition

The food item is archive and is unavailable for user to order.

2.1.2. Data Requirements

Data requirements are crucial for software development as an accurate and efficient database is the foundation of this project. Here is the list of data requirements below.

- Food Items: The food items are the primary data for this system that contains all available food items, including item prices, destinations, and images.
- User Information: The system stores essential information about User, including names, contact numbers, email addresses, encrypted password, and delivery addresses. The use of customer information should follow the GDPR principles.
- Order Information: The system stores all orders places by users, including order date, order id, customer details, order items, payment status, and so on.
- Restaurant Information: The system stores details about the restaurant, including name, address, contact information, open and close time.
- Payment Information: The system stores necessary information about transactions, including transaction id and payment status. Sensitive information of Credit card details will not be stored in the system.
- System Logs: The system should log all errors and events for maintenance and debug purposes.

2.1.3. User Requirements

User requirements are the expectations of the users while using the system. To benefit from the positive user experience, these user requirements should be achieved.

- Friendly UI: The system should have user-friendly interfaces with easy navigation and simple layout that allows users to search, add items and place orders.
- Food Items: The system should provide food items with details including price, descriptions, images, and allergy information.
- Update Shopping Cart: The user should be able to add or remove items, update quantities.
- Order History: The user should be allowed to view the history orders.
- Multiple Language: The system should support multiple languages, depending on the target users.
- Sensitive information: The system should forbid to store sensitive information about users.

2.1.4. Environmental Requirements

Environmental requirements refer to hardware, software and network must be in place to ensure the system operates properly.

- OS Compatibility: The system should be tested and compatible with major mobile platforms such as Android and iOS.
- Device Compactivity: The system should be available to range of devices with different hardware factors such as screen size and resolution.

- Performance: The system should be optimized for best performance, fast respond time, reducing loading time and delay time.

2.1.5. Usability Requirements

Usability requirements refer to the ease of use of the system.

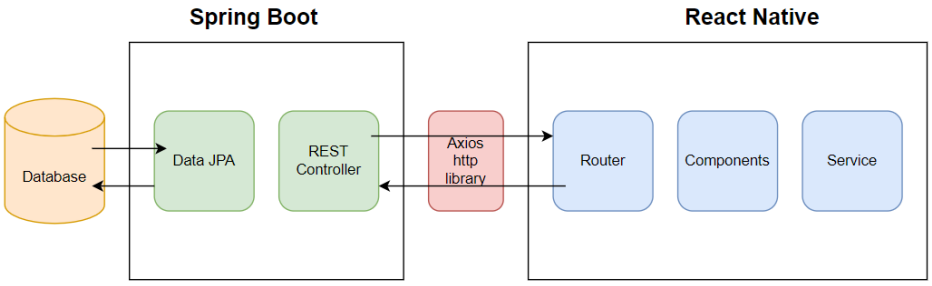
- The system must have user-friendly interfaces, and easy for users to navigate and place orders.
- The system should have fast load time and minimize delay time.
- The system should be designed for easy viewing with suitable font size and colours.

2.2. Design & Architecture

Client-server architecture is used to build a scalable and maintainable project for mobile application development.

- The server side is built using Spring Boot, a popular Java-based framework. With Spring Boot, a RESTful API is created for handling CRUD operations on the database.
- The client side is built with cross-platform framework React Native, including several screens such as home screen, item list screen, item details screen, and so on. The client retrieves data from backend, post requests to backend in a secure way.
- Expo tool helps to generate applications for iOS and Android platforms without the installation of Xcode and Android Studio. It reduces the burden of building applications for developers focusing on coding.

Architecture Diagram:



2.3. Implementation

Describe the main algorithms/classes/functions used in the code. Consider showing and explain interesting code snippets where appropriate.

- Crypt algorithm is used to hash passwords for security solution, avoiding password hacking.

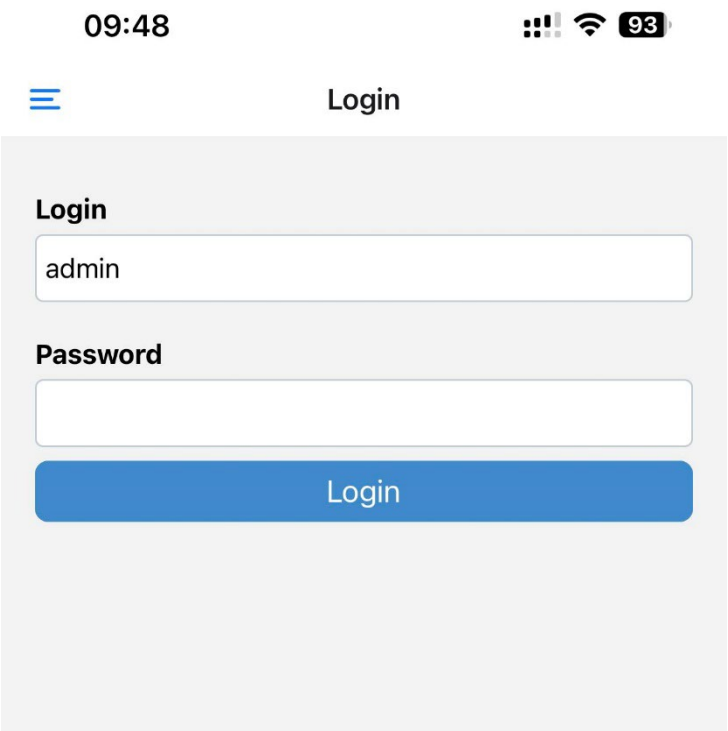
  code: import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

- Spring Boot provides excellent RESTful API to operate CRUD methods. In the API calls, GET request reads data from API, POST request creates new data, PUT request updates records with specific id, and DELETE request deletes records. For example, the code below retrieve data with an id.

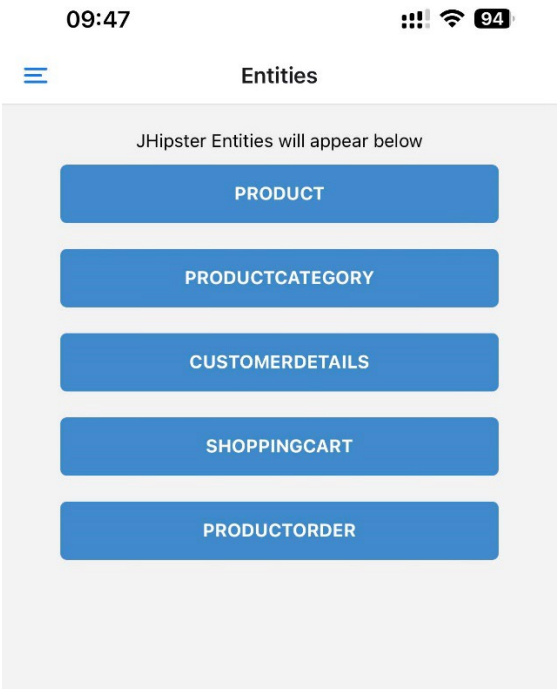
```
/**
 * {@code GET  /products/:id} : get the "id" product.
 *
 * @param id the id of the product to retrieve.
 * @return the {@link ResponseEntity} with status {@code 200 (OK)} and with body the product, or with status {@code 404 (Not Found)}.
 */
@GetMapping("/products/{id}")
public ResponseEntity<Product> getProduct(@PathVariable Long id) {
    log.debug("REST request to get Product : {}", id);
    Optional<Product> product = productService.findOne(id);
    return ResponseUtil.wrapOrNotFound(product);
}
```

2.4. Graphical User Interface (GUI)

Provide screenshots of key screens and explain what can be seen in each one.

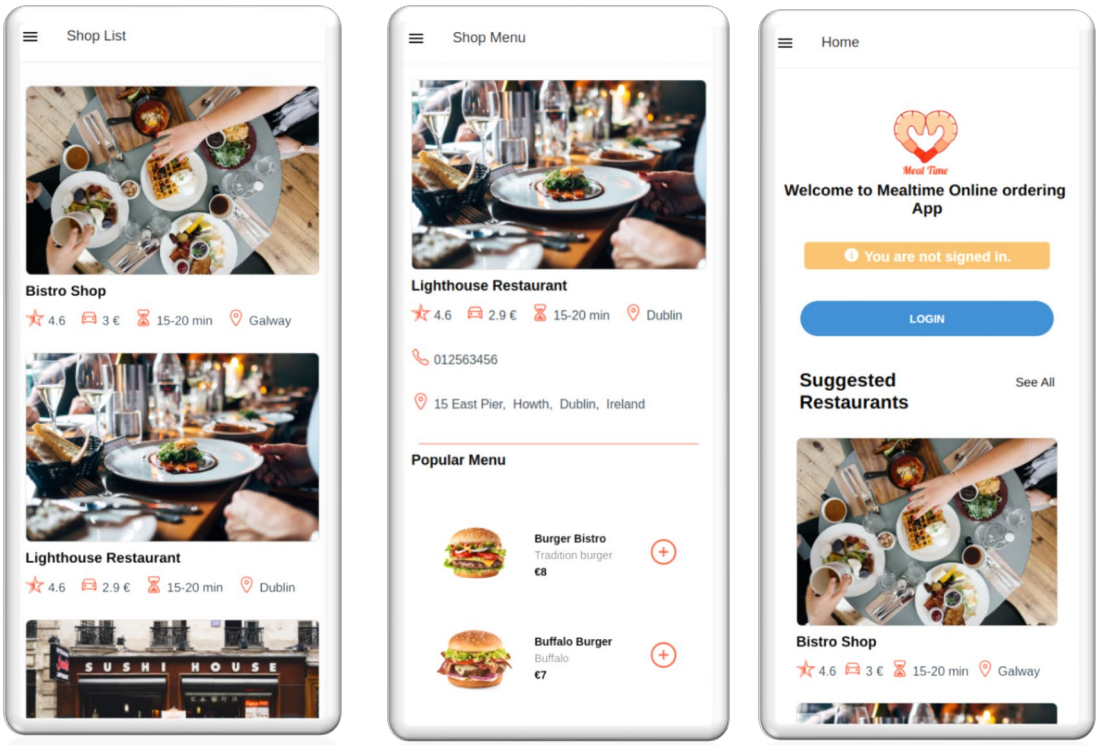


The above screenshot is Login screen from mobile application. User needs input correct username and password, then the system authenticates the user.



The above screen lists the entities used in the system. From this screen, CRUD operations can be operated without issue.

The below pictures show the UI for customers.



2.5. Testing

2.5.1 Backend:

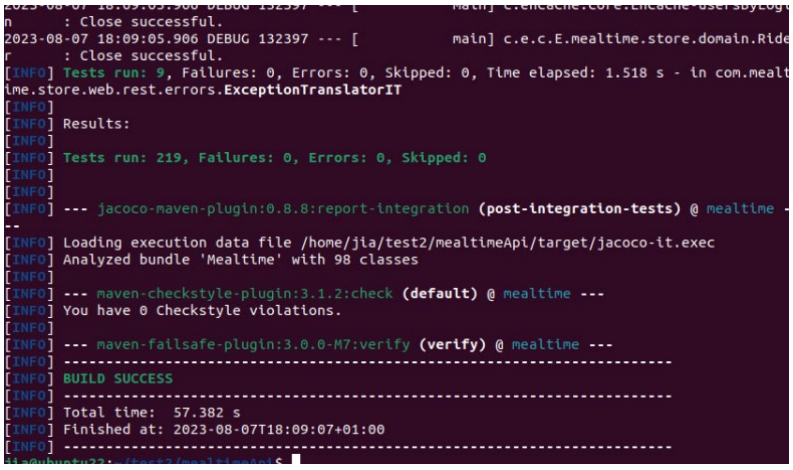
- Start Spring Boot application test, run the command below, to find any error or mismatch.

./mvnw verify

```
      Application 'mealtime' is running! Access URLs:
      Local:          http://localhost:8082/
      External:       http://192.168.1.206:8082/
      Profile(s):     [prod]
----------------------------------------------------------
cjia@ubuntu22:~/test2/mealtimeApi$ ./mvnw verify
Warning: JAVA_HOME environment variable is not set.
[INFO] Scanning for projects...
[INFO]
[INFO] --------------------< com.mealtime.store:mealtime >---------------------
[INFO] Building Mealtime 0.0.1-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-resources-plugin:3.3.0:copy-resources (default-resources) @ mealtime ---
[INFO] Copying 4 resources
[INFO] Copying 39 resources
[INFO]
[INFO] --- maven-resources-plugin:3.3.0:resources (default-resources) @ mealtime ---
[INFO] Copying 4 resources
[INFO] Copying 39 resources
[INFO]
[INFO] --- maven-enforcer-plugin:3.1.0:enforce (enforce-versions) @ mealtime ---
[INFO]
[INFO] --- maven-enforcer-plugin:3.1.0:enforce (enforce-dependencyConvergence) @ mealtime --
[WARNING]
```

```
[INFO]
[INFO] -------------------------------------------------------
[INFO]  T E S T S
[INFO] -------------------------------------------------------
2023-08-07 18:08:14.368 DEBUG   --- [           main] com.tngtech.archunit.ArchConfiguration
  : No configuration found in classpath at archunit.properties => Using default configuration
[INFO] Running com.mealtime.store.config.StaticResourcesWebConfigurerTest
[INFO] Tests run: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.012 s - in com.mealt
ime.store.config.StaticResourcesWebConfigurerTest
[INFO] Running com.mealtime.store.config.WebConfigurerTest
2023-08-07 18:08:16.136 DEBUG   --- [           main] org.jboss.logging
  : Logging Provider: org.jboss.logging.Log4j2LoggerProvider
2023-08-07 18:08:16.381 DEBUG   --- [           main] _.s.web.servlet.HandlerMapping.Mappings
  :
       c.m.s.c.WebConfigurerTestController:
       {GET [/api/test-cors]}: testCorsOnApiPath()
       {GET [/test/test-cors]}: testCorsOnOtherPath()
2023-08-07 18:08:16.923 DEBUG   --- [           main] _.s.web.servlet.HandlerMapping.Mappings
  :
       c.m.s.c.WebConfigurerTestController:
       {GET [/api/test-cors]}: testCorsOnApiPath()
       {GET [/test/test-cors]}: testCorsOnOtherPath()
2023-08-07 18:08:16.941 DEBUG   --- [           main] _.s.web.servlet.HandlerMapping.Mappings
  :
       c.m.s.c.WebConfigurerTestController:
       {GET [/api/test-cors]}: testCorsOnApiPath()
       {GET [/test/test-cors]}: testCorsOnOtherPath()
2023-08-07 18:08:16.955 DEBUG   --- [           main] _.s.web.servlet.HandlerMapping.Mappings
  :
       c.m.s.c.WebConfigurerTestController:
       {GET [/api/test-cors]}: testCorsOnApiPath()
       {GET [/test/test-cors]}: testCorsOnOtherPath()
[INFO] Tests run: 5, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.158 s - in com.mealt
ime.store.config.WebConfigurerTest
[INFO] Running com.mealtime.store.domain.CustomerDetailsTest
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.021 s - in com.mealt
```

```
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.936 s - in com.mealt
ime.store.TechnicalStructureTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 57, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- jacoco-maven-plugin:0.8.8:report (post-unit-test) @ mealtime ---
[INFO] Loading execution data file /home/jia/test2/mealtimeApi/target/jacoco.exec
[INFO] Analyzed bundle 'Mealtime' with 98 classes
[INFO]
[INFO] --- maven-jar-plugin:3.2.2:jar (default-jar) @ mealtime ---
[INFO] Building jar: /home/jia/test2/mealtimeApi/target/mealtime-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- spring-boot-maven-plugin:2.7.3:repackage (default) @ mealtime ---
[INFO] Replacing main artifact with repackaged archive
[INFO]
[INFO] --- modernizer-maven-plugin:2.4.0:modernizer (modernizer) @ mealtime ---
[INFO]
[INFO] --- jacoco-maven-plugin:0.8.8:prepare-agent-integration (pre-integration-tests) @ meal
time ---
[INFO] argLine set to -javaagent:/home/jia/.m2/repository/org/jacoco/org.jacoco.agent/0.8.8/o
rg.jacoco.agent-0.8.8-runtime.jar=destfile=/home/jia/test2/mealtimeApi/target/jacoco-it.exec
-Djava.security.egd=file:/dev/./urandom -Xmx1G
[INFO]
[INFO] --- maven-failsafe-plugin:3.0.0-M7:integration-test (integration-test) @ mealtime ---
[INFO] Using auto detected provider org.apache.maven.surefire.junitplatform.JUnitPlatformProv
ider
[INFO]
[INFO] -------------------------------------------------------
[INFO]  T E S T S
[INFO] -------------------------------------------------------
2023-08-07 18:08:22.475 DEBUG   --- [           main] com.tngtech.archunit.ArchConfiguration
```

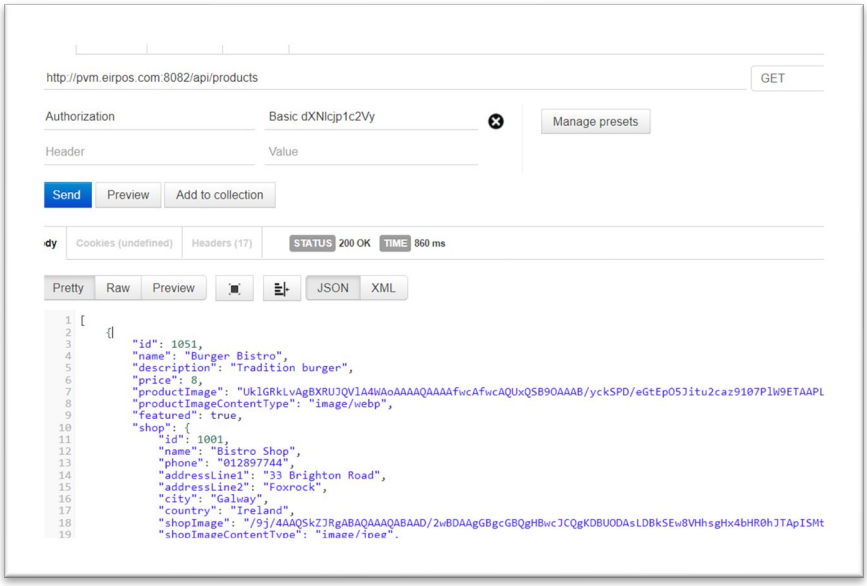


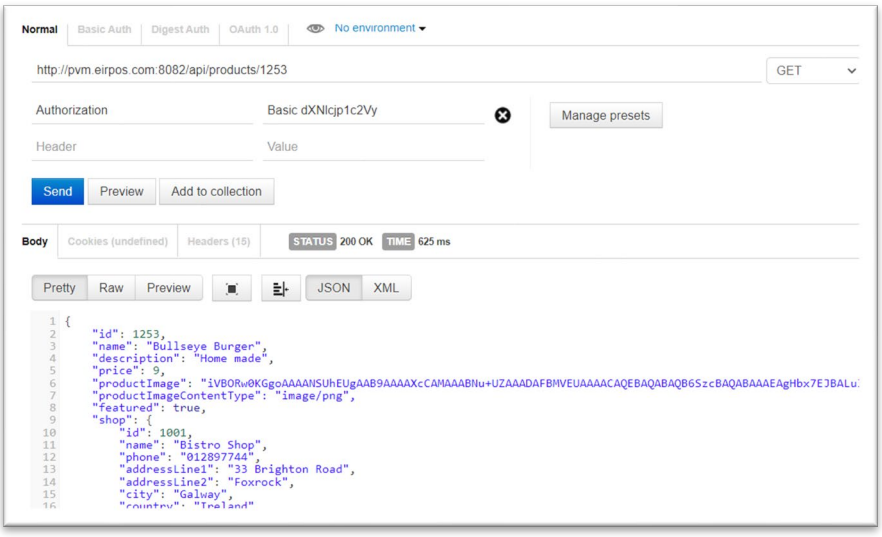
- Start client test by Jest, run command:

  Npm test

Screenshots of the result:

- Postman tool will be used to test the backend API with CRUD operations.



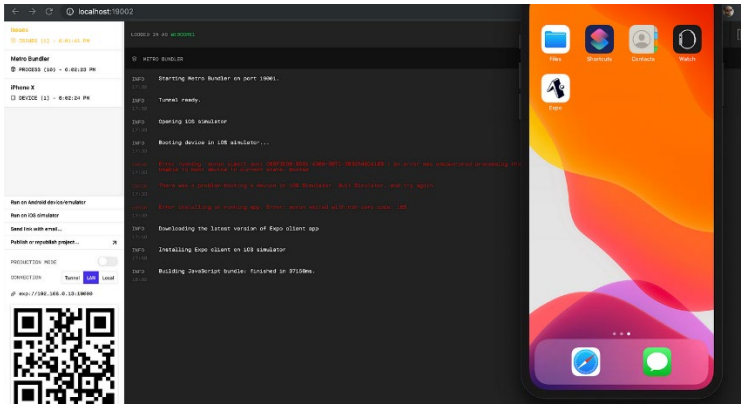


- Test React Native frontend end-to-end test, run command:

  Npm run test:e2e

  (this test not run as additional libraries required)

- Test by Expo go App on mobile phone.
  Run: npm start, then scan the QR Code to start the test on the phone.



2.6. Evaluation

- Functionality: The system should be evaluated to ensure that all functional requirements have been implemented correctly and are working as expected. Any bugs or errors should be resolved.
- Usability: The usability requirements should be evaluated to determine if the system is user-friendly and easy to use.
- Performance: The system's performance should be evaluated to ensure that it is operating at an optimal level. The response time and system load should be minimized.

- Security: The system's security should be evaluated to ensure that it is protected against potential threats. Any vulnerabilities such as data breaches should be removed.

3.0 Conclusions

Describe the advantages/disadvantages, strengths and limitations of the project

Advantages: [4]

- Using modern technology stack of Spring Boot and React Native helps to build robust and reliable platform for the system. [5]
- Client-server decouple architecture removes the limitation of adaption different frontend framework.
- The integration of third-party payment gateway services such as Stripe that provides secure and reliable payment solution.

Disadvantages:

- The users in poor network areas may face disruptions in the ordering process. Failures of order submitting may lead to user complaints.
- The user experience may be different depends on the types of devices.

Strengths:

- Using expo tool to generate cross-platform applications simplify the process of development significantly.

Limitations:

- Old generation without smart phones may not be able to access to the system.

4.0 Further Development or Research

With additional time and resources, which direction would this project take?

- Integration with social media platforms could allow users to share their favourite food which will bring more attention and business opportunities to the restaurants.
- Implementation of order tracking for delivery and making improvements on delivery services.

5.0 References

[1] 'Getting Started | Building a RESTful Web Service', *Getting Started | Building a RESTful Web Service*. https://spring.io (accessed Mar. 05, 2023).
[2] 'Introduction · React Native', Jan. 12, 2023. https://reactnative.dev/docs/getting-started (accessed Mar. 05, 2023).
[3] 'What Is React Native? Complex Guide for 2022'. https://www.netguru.com/glossary/react-native (accessed Mar. 05, 2023).
[4] 'Pros and Cons of Using Spring Boot', *Insights*, Jul. 04, 2021. https://bambooagile.eu/insights/pros-and-cons-of-using-spring-boot/ (accessed Mar. 05, 2023).
[5] 'Is React Native good? Advantages and disadvantages', *BinarApps*, Jan. 08, 2021. https://binarapps.com/is-react-native-good-advantages-and-disadvantages/ (accessed Mar. 05, 2023).

6.0 Appendices

This section should contain information that is supplementary to the main body of the report.

6.1. Project Proposal

Computer Project: Project Proposal

Food Order Mobile App Development

Software Development

Contents

Objectives

This project aims to help catering industry to make more profits by lower commission fees charging from 3rd party platforms such as Just-eat or UberEATS. It is not only releasing the stress of business cost for catering owners, but also gives more freedom to manage business at their fingertips. What's more, with the support of cloud printing technology, it helps to reduce customer complaints significantly as no loss of orders. All orders are sent to kitchen printers directly without delay and will be reprinted if any issue with the Internet occurs.

Background

Nowadays, more and more customers tend to make online orders for food because of its convenience. But ordering platforms charge much higher commissions, as a result, customers must pay much more than ordering in-store. With the rising living costs, it is challenging to lower the cost-of-service commission and gain more profit for catering owners. Firstly, third-party payments will be integrated into our system to process online payments. Stripe payment with as low as 1.4% + €0.25 fee is our first option to use, compared to the 14% service fee charged by the Just-eat platform.

With cloud printing thermal printers, order printing is much more stable and never misses an order anymore. The printer will resume the printing job after a power outage or network disconnection.

State of the Art

In the online food ordering market, there are big companies such as Just-eat, Uber-Eats, Deliveroo and flip-dish, which provide food ordering solutions for takeaways and restaurants. They are platforms with similar services that handle food orders. All these services are charged with high commissions that the owners of the catering industry couldn't afford it.

The biggest advantage of my project is the payment integrations. Our system will be integrated with Stripe payment with a low service charge. This solution will save a lot of money for customers.

What's more, we provide customization for individual customers to build their own brands.

Technical Approach

The Agile approach will be taken during the development, instead of the traditional waterfall management. Having a meeting with my supervisor weekly, we will discuss what has been done, and what is the plan to do next. We may have many requirements after discussion, but we will break them into individual tasks to achieve within a week or more. My work may be out of track, so, my supervisor may correct me promptly due to the benefits of the Agile approach.

The product backlogs will be created and some need to be completed with priority.

Technical Details

Java programming:

For the backend, a popular Java-based framework named Spring boot is used to build a restful API handling CRUD operation on the PostgreSQL database.

JavaScript language:

For the front end, the JavaScript framework React Native is used to build cross-platform mobile applications running both on Android and iOS devices.

Special Resources Required

Spring boot tutorial: https://www.javatpoint.com/spring-boot-tutorial

https://www.tutorialspoint.com/spring_boot/index.htm

https://www.jhipster.tech/

React Native tutorial: https://reactnative.dev/docs/getting-started

https://www.tutorialspoint.com/react_native/index.htm

https://www.youtube.com/watch?v=taPz40VmyzQ

… more

Project Plan

Stage 1 Research technology for development, Start date: 21-11-2022, End Date: 18-12-2022.

As mobile development on cross-platform is totally new to me, I have to search and find suitable technologies to learn by myself and use on my project.

Stage 2 Creating project proposal. Start date: 17-12-2022, End Date: 21-12-2022.

Stage 3 Setting up a development environment. Start date: 21-12-2022, End Date: 31-12-2022.

A Linux server is required to start, prefer Ubuntu20.04. A java environment is required to setup for Spring Boot to run; and NodeJS must be installed for React Native to start.

Stage 4.1 Design ERD diagram, Start date: 28-12-2022, End Date: 04-01-2023. An entity-relationship diagram is required to lay out the entities and attributes required before database creation. The relationship between different entities must be related correctly, for example, one2many, many2many, etc. The data types of attributes must be set up correctly according to project requirements.

Stage 4.2 Start build RESTful API for Backend, Start date: 05-01-2023, End Date: 28-02-2023.

I must learn from online tutorials to gain a basic understanding of Spring Boot; it is new to me as well. Spring Boot creates the link between Java classes and database entities, called object-relational mapping. So, there is no need to write SQL queries anymore. All data-related operations can be handled by the CRUD method from the Java classes.

After a RESTful API is created by Spring Boot, sample data has been imported for testing. Postman software is used to test the GET, UPDATE, POST, and DELETE methods. If all these four methods are all tested pass, this API is ready to use.

Stage 5 Start build React Native Front-end, Start date: 30-01-2023, End Date: 29-04-2023.

Stage 5.1 It will take a lot of time to learn React Native. After understanding the concepts such as state, props, and arrow function, then I can write the components to build the app.

Stage 5.2 Connect the RESTful API to fetch and display data in React Native app.

Stage 5.3 Build Home component for display shop information.

Stage 5.4 Build Dish components to list dishes.

Stage 5.5 Build Category components list dish categories.

Stage 5.6 Build Cart components for users to add dishes.

Stage 5.7 Build Checkout components with Stripe payment integration.

Stage 5.8 Build cloud printing function for receipt print.

Stage 5.9 Style the layout of the App and add images and Icons.

Stage 5.10 generate Android and iOS app packages.

Stage 5.11 Test Android and iOS apps.

Testing

For backend, Spring Boot framework has integrated with automation testing, just adding the testing dependency “spring-boot-starter-test”. Without automation testing, it will be very risky to release the new version of applications.

To test the CRUD functions in RESTful API, postman can be used to perform the GET, UPDATE, POST, and DELETE methods.

For frontend, there are many testing tools for React Native, for example, Jest is test tool to work out of the box.

Usually, when writing code, it is better to do unit tests to test every single function or class.

When designing a component, a component test is required to ensure the component respond correctly and render layout and elements properly.

Before releasing the mobile app, user interaction testing is carried out to test user clicks on different mobile devices.

6.2. Ethics Approval Application (only if required)

6.3. Reflective Journals

6.3.1. Reflection Semester 1

Supervision & Reflection Template

Student Name	Jiashun Chen
Student Number	X19134118
Course	Computing Project (BSHCSDE4)
Supervisor	Eugene McLaughlin

Month: November-2022

What? During the first meeting with my supervisor, we agreed to meet online every week to have better communication. I asked him how to master time management, because I had a full-time job and family life, and the time left for my study was very limited. He advised me to arrange every job well and strive for as much time as possible, and gave up non-essential activities, so, the project can be achieved according to requirements.
So What? As my project is building a cross-platform ordering app for catering shops, It requires a framework that write once and run any device. After research, there are two solutions, React Native and Flutter. As I have never known Dart programming, So, React Native could be my best option now. React Native is built on ReactJS which introduces new concepts like status and hooks that could be a challenge for me.

Now What? I need a learning plan for React Native and ReactJS to start this project. Without learning new technology, I can't go any further.
Student Signature | *Jiashun Chen*

Supervision & Reflection Template

Student Name	Jiashun Chen
Student Number	X19134118
Course	Computing Project (BSHCSDE4)
Supervisor	Eugene McLaughlin

Month: December-2022

What? This month I have learned some basic concepts of ReactJS from online tutorials to start my learning plan. And I have run a React demo app successfully. In the meantime, I started to search for some eBooks about Spring boot which is used widely by big tech companies. But I have learned rails on ruby for this semester, I have much experience and I feel much more confident to build a backend than Spring boot. The downside is that I

will not learn new technology this year. What's more, Spring boot is built on Java programming which has better career prospects, and I am very interested in it.
So What? My project will come with a backend and a frontend. The frontend is React Native I have chosen. But the database management system is not decided. The backend should be a RESTful API handling CRUD operation.
Now What? I need to do research to find a suitable backend framework with stability and easy implementation. I have learned Rails on Ruby during this semester, but Spring boot is much more popular recently. However, MongoDB is much more modern and quicker to start without any hassle.

Student Signature	Jiashun Chen

6.3.2. Reflection Semester 2

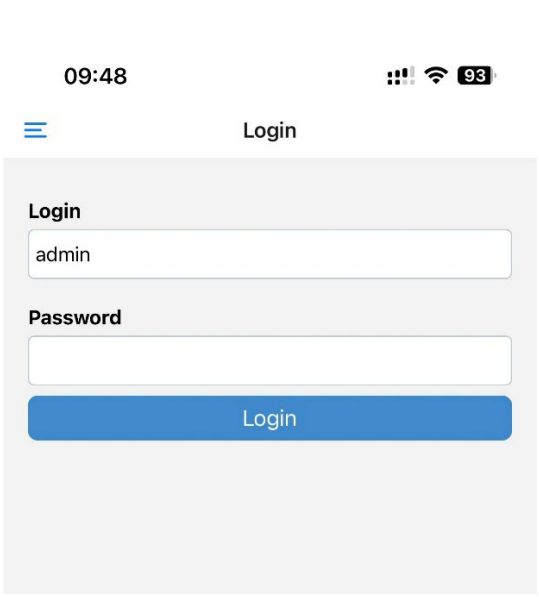
Supervision & Reflection Template

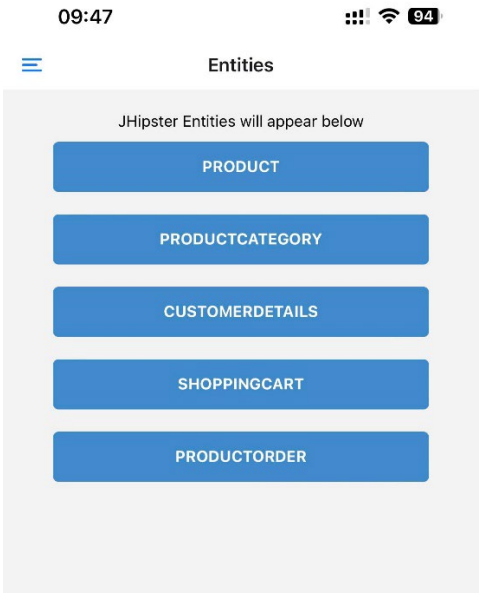
Student Name	Jiashun Chen
Student Number	X19134118
Course	Computing Project (BSHCSDE4)
Supervisor	Eugene McLaughlin

Month: January-2023

What?

During this month, I managed to set up the development process and achieved the goal that running demo apps on android and iOS devices. The difficult part is to set up both Android and iOS development environments on my computer. It is very complex to set up for Android, and impossible for iOS because I am using the windows system. So, after research, the Expo is the right tool that can generate apps for both platforms. Then I can download and install the app on an android phone and iPhone for testing. With Expo, it saved me massive time, so I have more time to focus on the design, coding, and UI.





So What?

1. I have learned that research is very important for this project and my learning. I should spend more time on research to find more possible solutions and choose the right one. It can expand my mind and gain more experience. As technology innovation grows fast and I should learn more new skills.
2. Expo is a convenient tool that I have to master for cross-platform mobile app development. I will spend more effort learning the workflow of app building.

3. As the database is the core of the app, a good database design will improve performance and save lots of hassles. I have designed the entities such as customers and food products, but it is not clear how to design the orders.
4. User privileges need to be set up in the database for administrators and users.
5. Google developer account is registered for app testing and publishing. Most testing will be operated on Android devices as the Android platform is more friendly to use.

Now What?

1. Thinking to register an Apple developer account for iOS testing, but the fee is expensive.
2. As I have only basic experience with the Spring Boot framework, I will dive into deeper to learn it for further development.
3. I will learn more about ReactJS state, hooks, and props as they are important in React app for data flow.
4. Do research to find a better UI framework for React Native to improve user experience.

Student Signature	Jiashun Chen

Supervision & Reflection Template

Student Name	Jiashun Chen
Student Number	X19134118
Course	Computing Project (BSHCSDE4)
Supervisor	Eugene McLaughlin

Month: February-2023

What?

During this month, I haven't done much on the project as there are other CAs and projects to finish. After researching how to design a better database, I learned a lot and realized my database design is not robust enough. More time will spend on database design because the database is the foundation of the application.

So What?

1. I have learned by myself how to code a React Native application following tutorials online. React Native is not difficult to learn according to the tutorial, but I have less experience with JavaScript and found the syntax of JavaScript hard to understand. For example, I was confused with the arrow function and took me a lot of time to understand it.
2. Storybook JS UI building tool is great to learn, it makes every UI as a component. It is used in this project. I need to spend more time to learn it.

Now What? Time management is still the biggest challenge for me. Having a full-time job and a few new CAs need to be finished on time, it is difficult to squeeze out extra time to continue the project. But I have to use my free time at weekend to continue learning new technology and implement it in my project.	
Student Signature	Jiashun Chen

Supervision & Reflection Template

Student Name	Jiashun Chen
Student Number	X19134118
Course	Computing Project (BSHCSDE4)
Supervisor	Eugene McLaughlin

Month: March-2023

What? During this month, I had some time to familiarize myself with the Expo tool by reading the official documentation. Expo is a great tool to simplify the development progress without installing any SDK tool. With the EAS-CLI commands, it is very easy to build and generate the APK file for the Android platform, and then download and install the APK file to an Android emulator. The application will be tested inside the emulator.
So What? 3. I have tested the demo application on the Android emulator but haven't tested it on the iOS platform as I haven't paid the Apple developer membership fee. 4. Instead of testing on an iOS simulator, Expo provides Expo GO on iPhone for testing which is quite useful.

Now What? As all tests are carried out locally, there may be some problems while switching to a public Network. For example, the database connection may be lost. So, a public database connection should be implemented, and the connection should be secure as well to avoid data breaches.

Student Signature	Jiashun Chen

Supervision & Reflection Template

Student Name	Jiashun Chen

Student Number	X19134118
Course	Computing Project (BSHCSDE4)
Supervisor	Eugene McLaughlin

Month: April-2023

What?

During this month, I implemented a public database connection with a subdomain URL for persistence data connection.

So What?

1. I started to read the document from the Stripe website and try to understand how the demo payment works. The integration with Stripe payment and React Native is new to me, and I need time to understand the principle of integration.
2. Cloud printing integration with Feie SAAS API is a great solution for kitchen order printing, but it may be difficult to develop as it involves printer hardware. I have registered Feie SAAS service and received an API token. The thermal printer is set up and connected to the Internet receiving orders.

Now What?

As semester 2 finished, and all CA finished as well, I will have more free time to learn the technologies and tools used in my project. I will check what I have done so far and try to improve the application.

Student Signature	Jiashun Chen

6.3.3. Reflection Semester 3

Supervision & Reflection Template

Student Name	Jiashun Chen
Student Number	X19134118
Course	Computing Project (BSHCSDE4)
Supervisor	Eugene McLaughlin

Month: May-2023

What?

This month, after careful consideration I realized the crucial role of databases in storing and retrieving data because new tables and properties are required to be added during the development. It is inefficient to modify and redesign the database. I found the importance of proper planning and foresight important when designing the database for as many aspects as possible.

So What?

5. I learned database design again through hands-on practice to fully understand the terminology and principles.
6. Study the normalization technique applied in a database to minimize redundancy and ensure the database is well-structured and efficient.
7. Consider the relationships between entities and choose the right ones, for example, one-to-one, one-to-many and many-to-many.

Now What? Although Spring Boot ORM (object-relational mapping) can help to access data easily, it provides simple APIs to use without writing any SQL query. It is strongly recommended to design the database from scratch and gain a solid understanding of the fundamentals of database designing and implementation.

Student Signature	Jiashun Chen

Supervision & Reflection Template

Student Name	Jiashun Chen
Student Number	X19134118
Course	Computing Project (BSHCSDE4)
Supervisor	Eugene McLaughlin

Month: June-2023

What? During this month, I spent more time learning and exploring React Native, a popular framework to build cross-platform mobile applications.
So What? 3. After learning I realized the benefits of reusable components in React Native. The component-based architecture enhances the productivity of the development and makes the code reusable and maintainable. 4. I explored React hooks that are significantly important and used widely. The most common hooks are useState, useEffect, useContext and useRef. I have learned these hooks by practising and understanding the different usage between them.
Now What? React Native is a powerful framework to build mobile applications. Next, I am going to learn the integration of third-party libraries, such as Stripe Payment, adding animation effects, and APIs for data queries.

Student Signature	Jiashun Chen

Supervision & Reflection Template

Student Name	Jiashun Chen
Student Number	X19134118
Course	Computing Project (BSHCSDE4)

Supervisor	Eugene McLaughlin

Month: July-2023

What?

In the final month, after a few times rebuilding the backend and database, the RESTful API became production which is built on the framework of Spring Boot and Java. The API is live and open to the Internet after getting a permanent sub-domain hosted by my home lab server. The API service is stable and responding requests without problems after testing in the Postman client.

So What?

1. Backup is very important. I have learned a lesson after upgrading the system and causing the whole project to fail to build. I had spent days trying to fix the problem but without luck. The solution is rebuilding from zero.
2. Keep studying new technologies. During the project, I found that the packages and libraries may become deprecated and cause the project failure of build. It is a very difficult task to maintain the packages updated and have minimum impact on the running project.
3. My project is built on NodeJS and requires massive packages which makes me confused. The Maven build tool is new to me as well. Learning by practice is the best way to familiar with them.

Now What?

Although I have done most work on the backend, the work on the front end seems more difficult to me. React Native is the framework to build the UI. It costs me a lot of time to adjust the structures or styles after refreshing pages again and again. After diving into the development of the front end, many more tools and third-party libraries are used in React Native projects, such as redux and saga. Now I am focusing on the front end and connecting React Native app with Spring Boot RESTful API.

Student Signature	Jiashun Chen