Software Requirements Specification

for

Puff Lab Application

Version 1.0

Prepared by

Group Name: Codezilla

Faziana Binti Mashor	**83865**	**83865@siswa.unimas.my**
Delneeza Anak Dismon	**87143**	**87143@siswa.unimas.my**
Khairul Wafi Bin Mazelan	**84230**	**84230@siswa.unimas.my**
Caleigh Susan anak Jeffry	**82476**	**82476@siswa.unimas.my**
Bethsheba Sewing	**82469**	**82469@siswa.unimas.my**

Instructor: Ts. Nurfauza Binti Jali

Course: TMA3084_SoftwareEngineeringLaboratory

Lab Section: Group 1 and Group 2

Teaching Assistant: Ts. Nurfauza Binti Jali

Date: 2nd December 2024

Contents

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Draft Type and Number	Full Name	Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded.	00/00/00

1 Introduction

Codezilla and Puff Lab have reached an agreement for our team to produce a mobile application called the Puff Lab App. Our key objective for the mobile application development is to facilitate Puff Lab's employees in managing and tracking customer orders. This application will enable staff to access customer reviews regarding their overall experience. This feature will provide Puff Lab with essential data regarding user satisfaction, preferences, and areas for improvement. This mobile application will enable users to explore various flavours, personalise orders, and complete purchases effortlessly. The information, requirements, and guidelines necessary for the Puff Lab Application are included in the Software Requirement Specification (SRS) report.

1.1 Document Purpose

This document aids our group in establishing the groundwork for the development team and enables the team to produce the appropriate product as requested by our client. Listing the software specifications for the Puff Lab Application is the objective. It is intended to provide a comprehensive description of the operational elements, functionalities, and features that are essential for the successful development of this mobile application.

This Software Requirements Specification (SRS) offers a concise summary of the system that will be develop for the benefit of both our client and the application's user. This document functions as a plan of action, guaranteeing that the Puff Lab Application's objectives and features are understood by users and clients.

1.2 Product Scope

Puff Lab App is a mobile application intended to streamline the ordering and management processes for cream puff and beverage. The app provides users with a range of intuitive features, starting with sign up and sign in functionalities for account access. Upon logging in, customers can navigate to menu dashboard where it displays

four different categories. Customers can explore products within these categories, add product to their cart, and proceed to order summary to review their selections. Users obtain order confirmation alerts through both the application and email. Additional features include real-time order status tracking, user account management including detail updates or account deletion, order history viewing, the option to leave reviews and feedback post-order completion, and contact options for communication via call, email, or social media platforms such as WhatsApp and Instagram.

Puff Lab App offers administrators a comprehensive backend interface for efficient business management. Administrators able to log in and it will feature a robust dashboard that delivers real-time insights into the business's performance. It provides visualized data on total sales and orders monthly, along with identifying the best-selling drinks and cream puff flavors to support strategic decision-making. They can efficiently manage the product menu by adding, updating, or deleting products, and oversee the order list. The application also features the transaction history, and allow administrators interact with customer feedback by viewing, replying to, or removing reviews. Administrators can obtain customer profile information from the feedback area.

1.3 Intended Audience and Document Overview

The present document, denoted as the Software Requirement Specification (SRS), is designed to be perused by our client, the administrators of Puff Lab App and our instructor, Madam Nurfauza Binti Jali, for Software Engineering Laboratory course. This document, known as the Software Requirement Specification (SRS), includes information about the Puff Lab mobile application, including an overview, specific requirements, non-functional requirements, additional requirements, and appendices. As indicated by the Table of Contents, the Software Requirement Specification is divided into five primary components.

It is suggested to our client to read this document in the given order as stated below:

1. Introduction

2. Overall Description

3. Specific Requirements

4. Non-functional Requirements

5. Appendices

It is recommended that our instructor peruse this Software Requirements Specification (SRS) in its entirety. The subsequent section contains a broad description of this document and a synopsis of the product's characteristics. The following part detailing the need definition is designed for the developers, aiming to clarify the pre-requisites of this mobile application in an understandable manner. In conclusion, these sections present a thorough and cohesive representation of the web application.

1.4 Definitions, Acronyms and Abbreviations

Acronyms and Abbreviations	Definition
SRS	Software Requirement Specification
CRUD	Create, Read, Update, Delete
UNIMAS	University Malaysia Sarawak
GPS	Global Positioning System
RAM	random-access memory
UI	User Interface
CSS	Cascading Style Sheets
FAQ	Frequently Asked Questions
API	Application Programming Interface
PCI-DSS	Payment Card Industry Data Security Standard
SMTP	Simple Mail Transfer Protocol

MySQL	Structured Query Language
UI	User Interface
UX	User Experience

1.5 Document Conventions

Font and size:

- All text contained in this document is using Arial font size 12.
- Section tiles are using Arial font size 18.
- Subsection tiles are using Arial font size 14.

Spacing and margins:

- Document will be single-spaced and maintain 1” margins.

Section and subsection titles:

- Section and subsection titles follow the prescribed template to maintain a structured and organized document.

Highlighting:

- No special highlighting is used in this document.

1.6 References and Acknowledgments

Understanding Payment Card Industry Data Security Standard (PCI DSS) | Controller’s Office. (n.d.). https://controller.ucsf.edu/how-to-guides/accounting-reporting/understanding-payment-card-industry-data-security-standard-pci

2 Overall Description

2.1 Product Perspective

Puff Lab is a popular dessert shop originating from Sarawak, Malaysia. Known for its creative and delicious cream puffs, it has gained significant popularity, especially among the younger generation, particularly students from Universiti Malaysia Sarawak (UNIMAS). Beyond its original location, Puff Lab has expanded to other cities, making its unique offerings accessible to a wider audience. The shop's success can be attributed to its commitment to quality and innovation. By using high-quality ingredients and crafting visually appealing presentations, Puff Lab offers a unique and delightful dessert experience. Moreover, the shop supports local businesses by sourcing ingredients from local suppliers, contributing to the local economy and creating employment opportunities.

How Puff Lab interact with its environment:

1. Customers: Customers are the primary users of the product. They visit the physical store to purchase cream puffs or place orders through online platforms.
2. Suppliers: Puff Lab sources its ingredients from various suppliers, including those providing fresh cream, pastry flour, sugar, eggs etc.
3. Staff: The shop employs staff to handle various tasks, such as baking, serving customers, managing inventory, and handling online orders.

~~2.2~~ Product Functionality

Major functions of a Puff Lab Application:

Users

- User Account
  - Allow users to create manage accounts.
  - Store customer information.
  - User can save their preferred payment methods and delivery address.

- Product Catalog
  - Display list of pastries, including images, descriptions, and pricing information.

- Allows users to filter and search products based on various criteria such as price range and category.

- Order Placement
  - Enable users to select and add item to their cart.
  - Calculate the total order amount, including delivery fees.
  - Securely process payments through integrated payment gateways.

- Order Management:
  - Allow users to view order history and details.
  - Provide order cancellation.

Admin

- Admin Login
  - Login account using email and passwords.

- Product Catalog management
  - Add. Edit and remove products including details like name, description, price, images and flavours.
  - Managing products categories.

- Order Management
  - Processing and fulfilling orders, managing returns and refunds, and generating invoices.
  - Providing customers with order updates.
  - Handling order cancellations and refunds.

- Customer support management
  - Handling customer inquiries and complaints.
  - Providing customer support through various channels.

2.3 Users and Characteristics

Users and characteristics of Puff Lab Application:

- Customers
  - Casual Users: A casual customer who place orders infrequently.
  - Frequent Users: Regular customers who place orders multiple times a week or month.
  - Characteristics: Young adult between the age 18-35 who often seeking trendy and unique pastries and enjoy eating sweet delicacies.

- Puff Lab Store
  - Cashiers: Staff members responsible for taking orders and processing payments.
  - Kitchen Staff: Staff members responsible for preparing and baking pastries.
  - Characteristics: Staff members is familiar with inventory managements software and online platforms.

- Delivery Personnel
  - Delivery Driver: Staff members responsible for delivering orders to customers.
  - Characteristics: Ability to use GPS navigation, mobile apps, and other delivery tools.

2.4 Operating Environment

The operating environment for the Puff Lab's Application is as listed below.

- Smartphone: Puff Lab's Applications are designed to run on smartphones and tablets, primarily on Ios (iPhone, iPad) and Android operating systems.
- Stable internet connection: A stable internet connection is essential for network-based applications and software updates.

2.5 Design and Implementation Constraints

The development of the Puff Lab application was constrained by limitations in both memory capacity and storage space. Insufficient RAM can significantly impact performance, causing crashes and slow load times. To overcome this issue, we optimized memory usage by reducing unnecessary object creation and addressing memory leaks. Next is the issues of storage space, where limited storage space can restrict the application's functionality and the amount of data it can handle. To address these issues, we implemented efficient data storage strategies. The third constraint is the potential for software vulnerabilities, which can expose the application to security risks. These vulnerabilities could lead to unauthorized access, data breaches, or malicious attacks, compromising the integrity and confidentiality of the application and its data. To resolve this, a regular updates of software components and libraries are essential for the system.

Furthermore, the next constraints are the User Interface (UI) limitations. Different devices have different screen sizes and resolutions, which can impact the layout and responsiveness of the Puff Lab application's interface. To ensure user can access this application in any devices, implement a responsive design technique to create a single UI that adapts to various screen sizes. This involves using fluid grids, flexible images, and CSS media queries to dynamically adjust the layout. Lastly, considering the diverse user base, it's crucial to ensure accurate translation of all text elements, including menu items, product descriptions, and error messages. This involves implementing support for multiple languages and allowing users to easily switch between them.

2.6 User Documentation

To enhance user with a seamless user experience when using the Puff Lab’s application, user manuals, on-line help system and tutorials will be provided along with the software. This user documentation will guide users through the application’s features and functionalities, ensuring a smooth and efficient experience.

1. User manual – A comprehensive user manual will be available in digital formats. This user manual includes the installation and setup instructions, basic navigation and interface usage, and Frequently Asked Questions (FAQs)

2. Online Help System – In-App help system will be integrated into the application. Users can access this help by clicking on specific elements withing the interface.
3. Tutorials – Short video tutorials will be available on Puff Lab's social media platforms to guide users on how to effectively use the application. These tutorials video will demonstrate the key features and functionalities of the application and allowing uses to learn at their own pace.

Assumptions and Dependencies

Some major assumptions that might significantly affect the design of a Puff Lab Application are user familiarity, it is assumed that users possess a fundamental understanding of mobile applications and online shopping platforms. This familiarity ensures smooth user interaction with the application and if user is not familiar with the application, they may encounter various challenges that can hinder their experience and adoption. Next, the application must accommodate a diverse range of devices, particularly those running on iOS and Android operating systems. Having a stable internet connection is also another assumption and it is the first requirement for user to access the application. For instance, a stable connection is needed for user to browse product, place an order, and payment processing. Additionally, the integration of a payment gateway will significantly impact the application's design. The application's backend must integrate with the payment gateway's API to securely transmit payment information, process transactions, and receive transaction status.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

TO DO: The least you can do for this section is to describe in words the different User Interfaces and the different screens that will be available to the user. Those who will be able to provide optional Graphical User Interface screenshots, will be rewarded by extra marks.

Figure 1: Homepage of Puff Lab Application

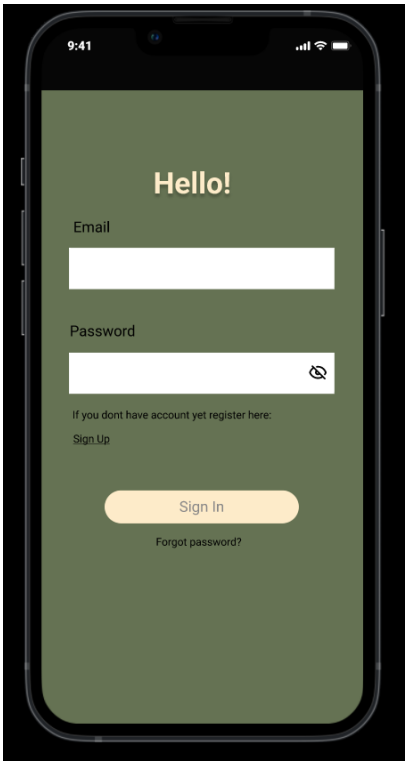


Figure 2: Customer Log in Page

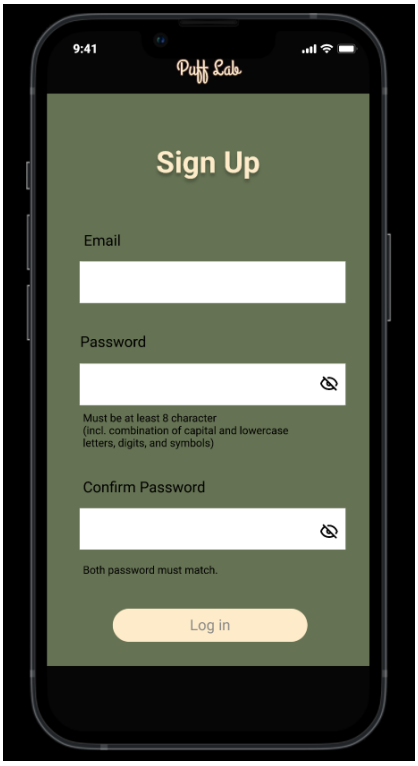


Figure 3: Customer Sign up Page

Figure 4: Puff Lab Main Menu Page

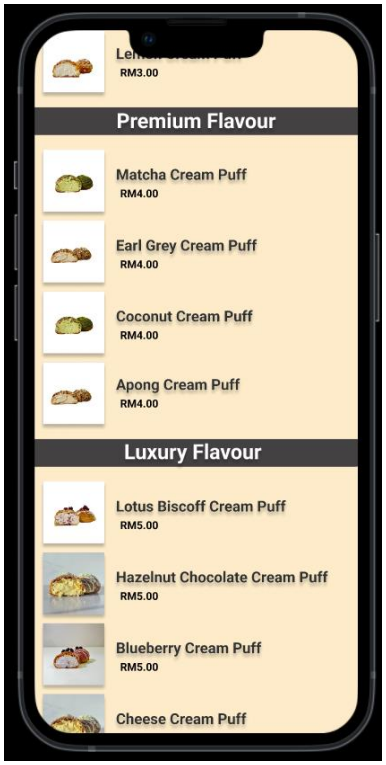


Figure 5: Puff Lab's List Menu

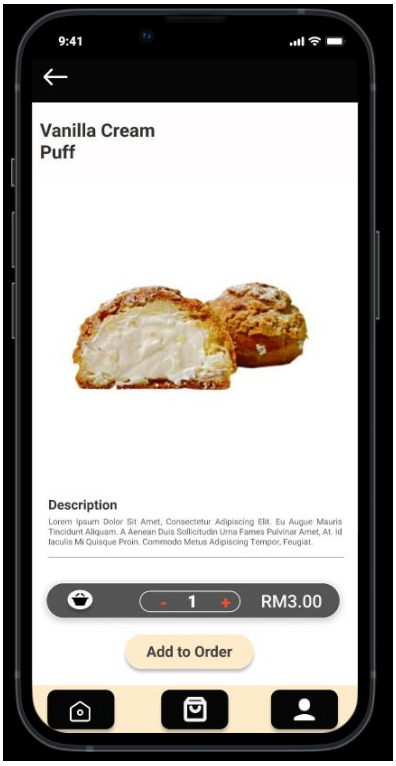


Figure 6: Place Order Page

Figure 7: Order Summary Page

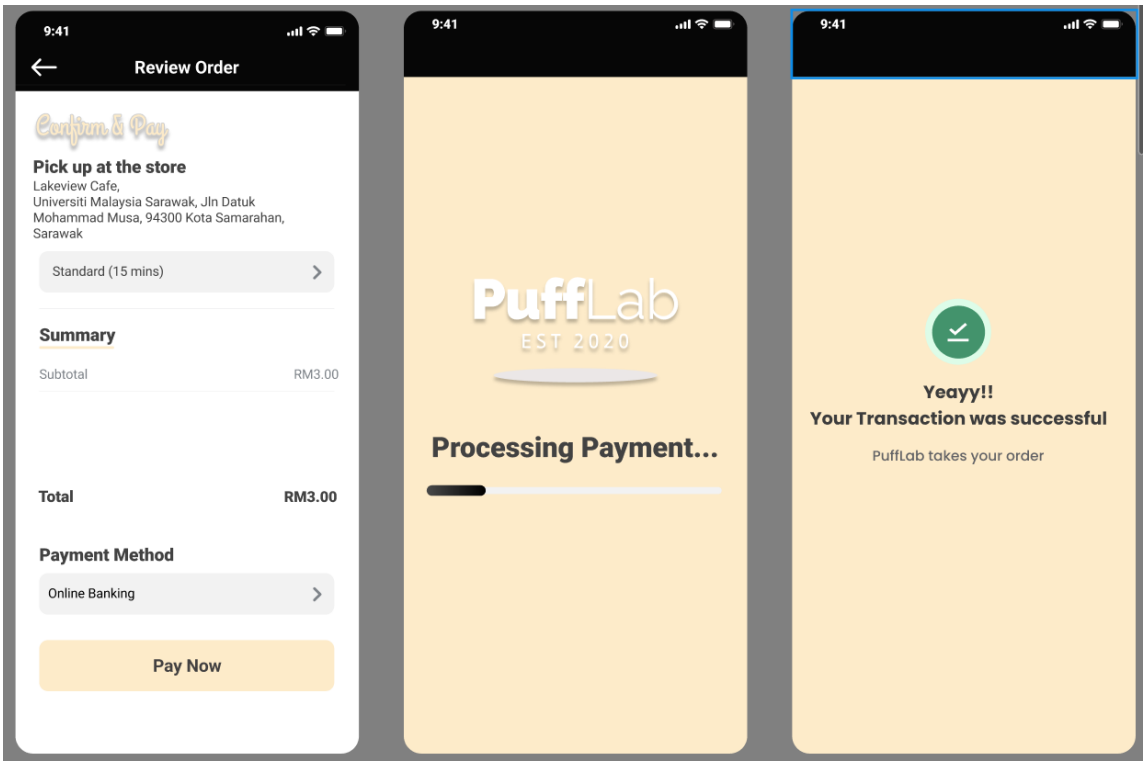


Figure 8: Payment Process Page

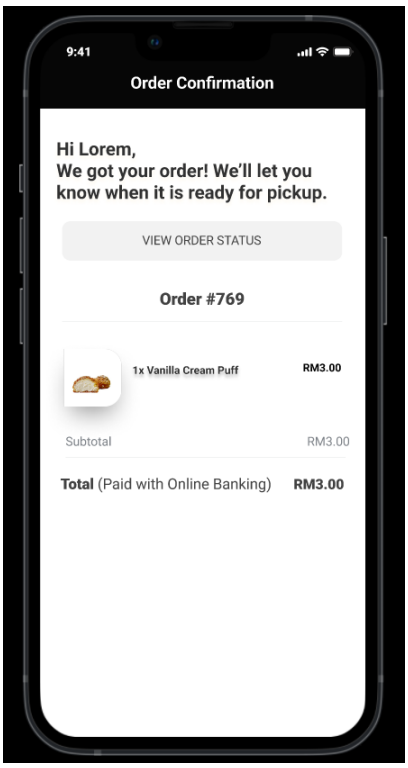


Figure 9: Order Confirmation Page

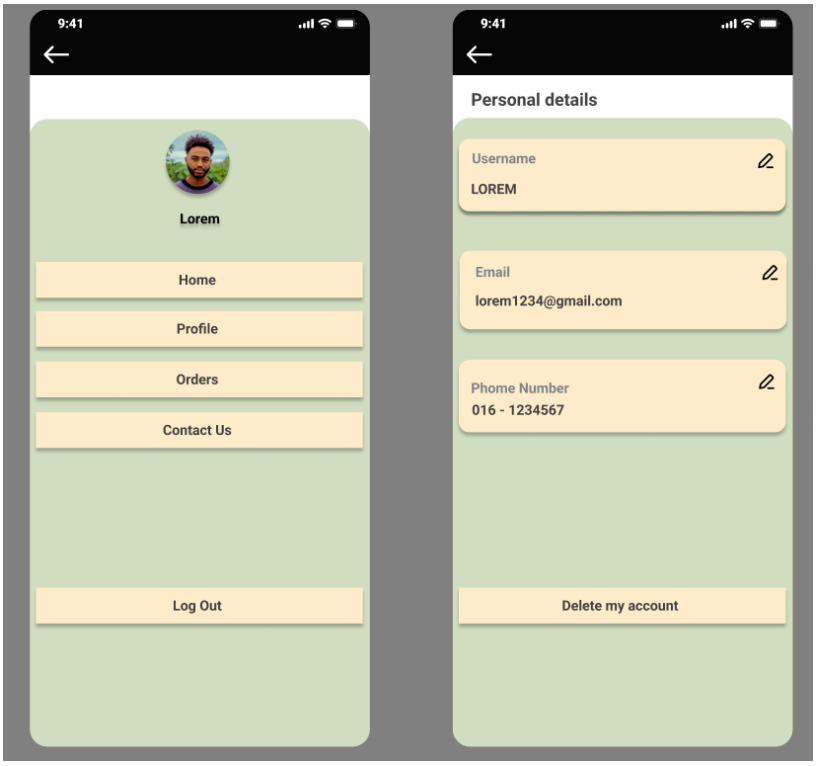


Figure 10: Customer Profile Management Page

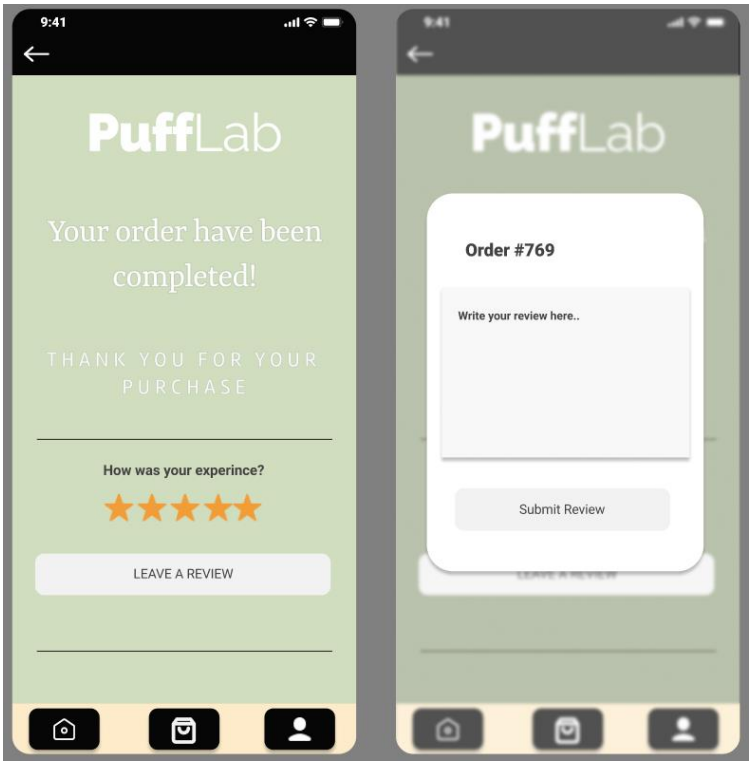


Figure 11: Customer Leave Review and Feedback Page

Figure 12: Contact Page

Figure 13: Admin Log in Page

Figure 14: Admin Dashboard Page

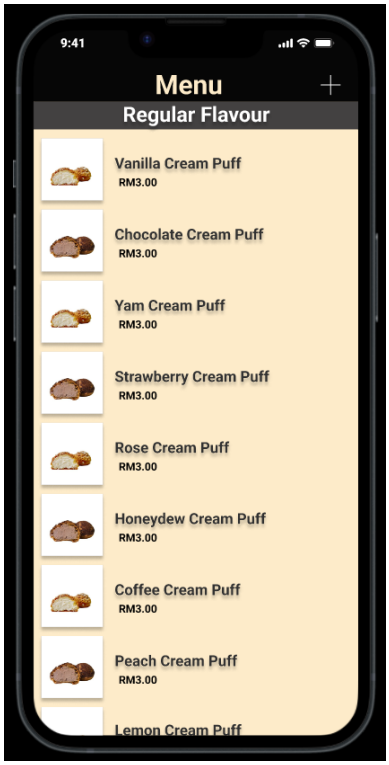


Figure 15: View Menu Page

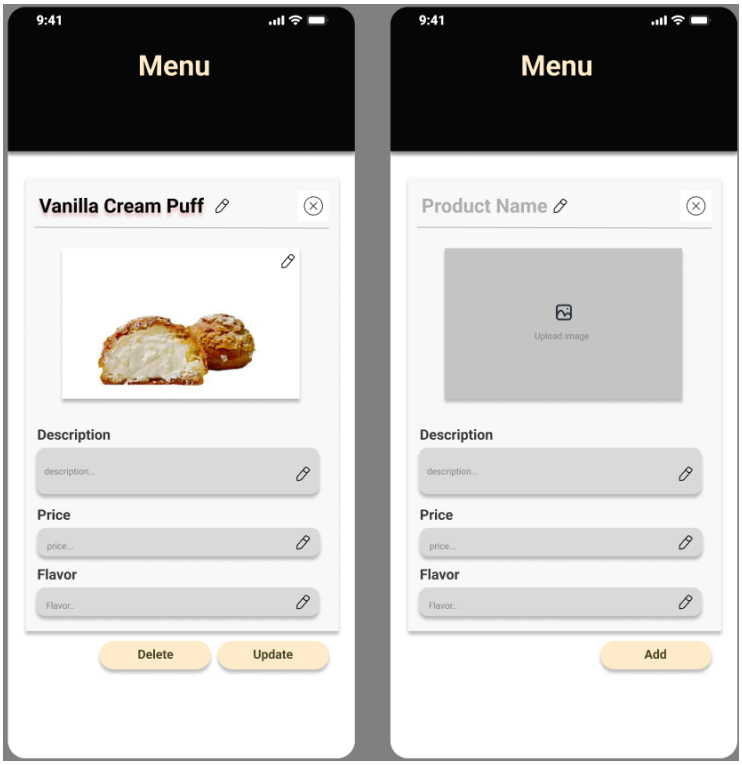


Figure 16: Edit Menu Page

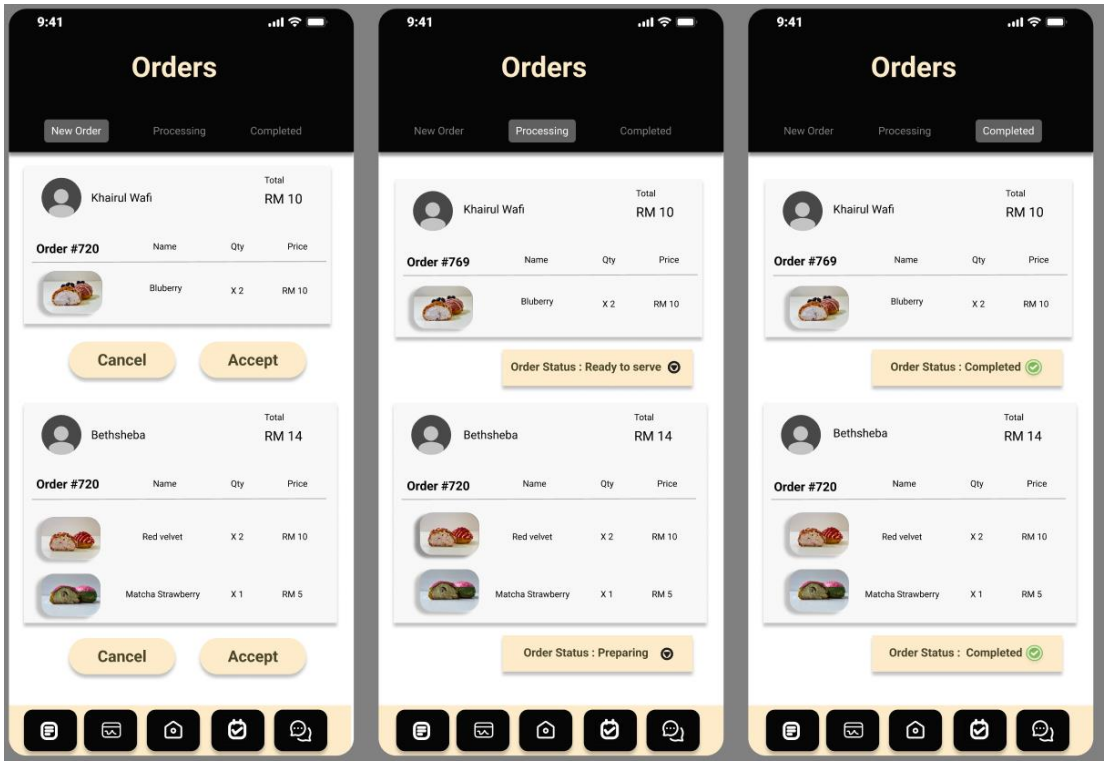


Figure 17: Order Management (New Order, Processing, Completed) Page

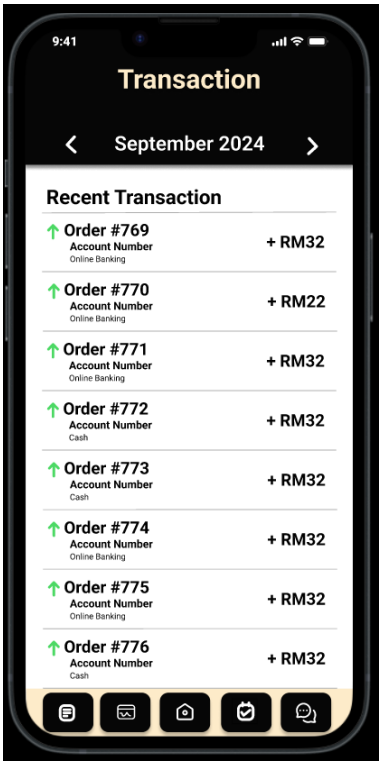


Figure 18: Transaction History Page

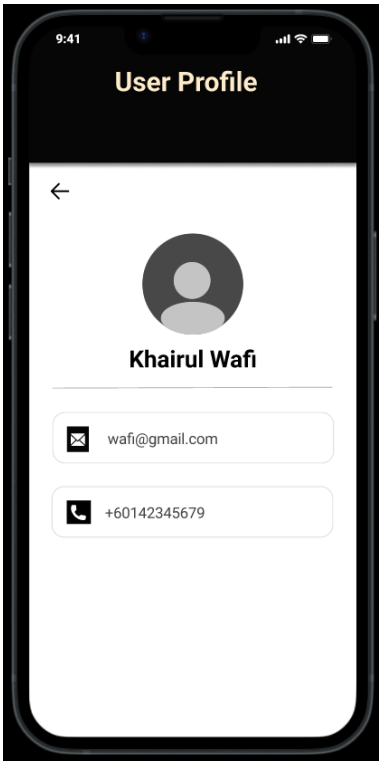


Figure 19: View Profile Page

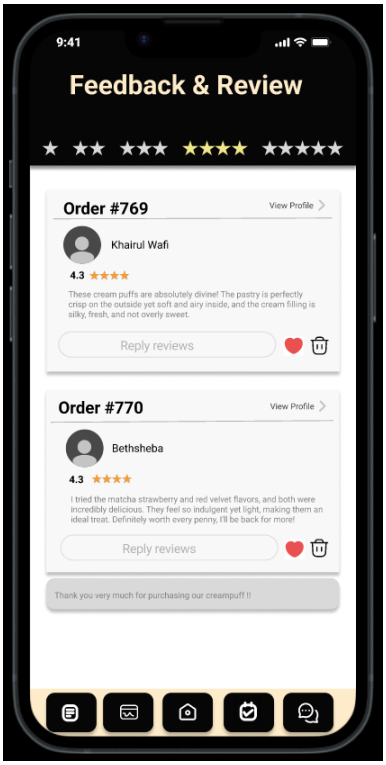


Figure 20: Feedback & Review Management Page

3.1.2 Hardware Interfaces

The Puff Lab Application is a mobile-based system that is built using technologies such as Flutter since these are technologies which are used in creating applications that can function properly in both Android and iOS devices. MySQL manages the user accounts, order information, products, and transactions through this application integrated into it. Some of these are APIs for payment processing security and integration with communication services for customer concerns or feedback among others. Android and iOS platforms are integrated into the application to allow for compatibility with the operating ecosystem to promote efficiency in use. It also requires enough storage and RAM to guarantee optimal performance when in use.

3.1.3 Software Interfaces

The Puff Lab Application has been developed using mobile development technologies suitable for Android and iOS devices. Some of the frameworks used to achieve coherence are frameworks like Flutter to guarantee the establishment of a coherent and responsive

experience between interfaces. The backend is achieved through code written to the server side of the program using server-side scripting languages and the database management using MySQL to handle the data of the users, orders and products.

The application supports third-party API for payment options and sending of emails. They state that these software components facilitate smooth interactions that involve placing orders, having accounts, and receiving transaction updates all in a safe means which ensures data integrity.

3.1.4 Communications Interfaces

The Puff Lab Application works through client-server model and allows users to send messages with doubts and request information to the server. Other activities like orders, user's profile updates are developed and maintained using encrypted procedures to ensure proper security of data.

For notification and communication, the application employs SMTP in sending out email notifications for order status change or any question that the users may have. It also synchronizes with social networks such as WhatsApp and Instagram to offer users other means of approaching them with queries, complaints, or comments. The application of the high encryptions guarantees that all the data transmitted gets protected in the course of the communication.

3.2 Functional Requirements

3.2.1 User Register Account

3.2.1.1 Stimulus/*Response Sequence*

Use Case	Register Account
Short Description	To allow new user to create account by providing their needed personal information and login credentials.
Actor(s)	Customer
Pre-condition(s)	1. The customer has downloaded and opened the mobile app.

Post-condition(s)	1. The customer 's account is successfully registered and stored in the database. 2. The customer can log in using the created login credentials.
Main Flow	1. The customer taps on the "Sign In" button on the login page. 2. The app displays the registration form. 3. The customer enters required details such as name, email, password, and username. 4. The app validates the input and sends the data to the server. 5. The server confirms successful registration. 6. The app displays a confirmation message and redirects the customer to the login page.
Alternative Flow(s)	N/A
Exception Flow(s)	N/A

3.2.1.2 Direct Functional Requirements

Requirement	**Description**
REQ-1	The mobile app shall provide a registration form for new users to input their details.
REQ-2	The mobile app shall validate user input.
REQ-3	The mobile app shall integrate with third-party authentication providers such as Gmail.

3.2.2 Admin Log In

3.2.2.1 Stimulus/Response Sequence

Use Case	Log In
Short Description	The admin logs into the system using a secure username and password.
Actor(s)	Admin

Pre-condition(s)	1. Admin input the correct username and password.
Post-condition(s)	1. Admin is authenticated and redirected to the dashboard.
Main Flow	1. Admin navigates to the login page. 2. Admin enters valid credentials such as username and password. 3. System validates the credentials. 4. Admin is granted access and redirected to the admin dashboard.
Alternative Flow(s)	N/A
Exception Flow(s)	Admin enters invalid credentials. 1. System displays an error message and prompts to retry. Customer forgets passwords. 1. System redirects to the password recovery process.

3.2.2.2 Direct Functional Requirements

Requirement	**Description**
REQ-1	Validate admin credentials securely.
REQ-2	Redirect to the admin dashboard upon successful login.

3.2.3 Customer Log in

3.2.3.1 Stimulus/Response Sequence

Use Case	Log In
Short Description	The customer logs into the app to access their account and features.
Actor(s)	Customer
Pre-condition(s)	1. Customer account exists in the system.

Post-condition(s)	1. Customer is authenticated and redirected to the homepage.
Main Flow	1. Customer navigates to the login page. 2. Customer enters valid credentials such as email or username and password. 3. System validates the credentials. 4. Customer is granted access and redirected to their account homepage.
Alternative Flow(s)	N/A
Exception Flow(s)	Customer enters invalid credentials. 1. System displays an error message and prompts to retry. Customer forgets passwords. 1. System redirects to the password recovery process.

3.2.3.2 Direct Functional Requirements

Requirement	Description
REQ-1	Validate customer credentials securely.
REQ-2	Provide a password recovery option.

3.2.4 Customer Manage Profile

3.2.4.1 Stimulus/Response Sequence

Use Case	Manage Profile
Short Description	To view and edit their profile information, including personal details and account settings.
Actor(s)	Customer
Pre-condition(s)	1. The customer is logged into the app.
Post-	1. Profile updates are saved, and the customer receives a

condition(s)	confirmation.
Main Flow	1. Customer navigates to the "Profile" section. 2. Updates profile information such as name, email, phone number. 3. System validates the inputs. 4. Changes are saved, and a confirmation message is displayed by the apps.
Alternative Flow(s)	N/A
Exception Flow(s)	Invalid input detected. 1. System notifies the customer and prompts for corrections.

3.2.4.2 Direct Functional Requirements

Requirement	Description
REQ-1	Validate input fields.

3.2.5 Customer View Products

3.2.5.1 Stimulus/Response Sequence

Use Case	View Products
Short Description	To allows customers view the available products details in the menu catalogue which include images, descriptions and pricing information.
Actor(s)	Customer
Pre-condition(s)	1. Customer is logged into the app. 2. Products are available.
Post-condition(s)	1. Customer has viewed the desired product(s) details.
Main Flow	1. Customer navigates to the product catalogue. 2. System displays the products details, including image,

	description, price, reviews and availability.
Alternative Flow(s)	N/A
Exception Flow(s)	N/A

3.2.5.2 Direct Functional Requirements

Requirement	**Description**
REQ-1	System shall allow customers to tap on a product to view its details.
REQ-2	Display detailed information for each product, including image, price, review and description.

3.2.6 Customer Place Order

3.2.6.1 Stimulus/Response Sequence

Use Case	Place Order
Short Description	To order the desired product before proceeding to payment process.
Actor(s)	Customer
Pre-condition(s)	1. Customer is logged into the app. 2. Customer has customized or selected products.
Post-condition(s)	1. Order is submitted and the system will display payment method.
Main Flow	1. Customer customizes or selects a product. 2. Customer reviews order details. 3. Customer submits the order. 4. System generates an order summary. 5. Customer proceeds to make the payment.
Alternative Flow(s)	N/A
Exception	N/A

Flow(s)	

3.2.6.2 Direct Functional Requirements

Requirement	**Description**
REQ-1	System must allow customers to review their order before submission.

3.2.7 Customer Make Payment

3.2.7.1 Stimulus/Response Sequence

Use Case	Make Payment
Short Description	To enable customer to pay for their orders using various payment methods.
Actor(s)	Customer
Pre-condition(s)	1. Customer is logged into the app. 2. Order is ready for payment.
Post-condition(s)	1. Payment is successfully processed, and the order status is updated.
Main Flow	1. Customer selects a payment method. 2. System redirects to the payment gateway. 3. Customer completes the payment process. 4. System confirms payment and update the order status.
Alternative Flow(s)	N/A
Exception Flow(s)	N/A

3.2.7.2 Direct Functional Requirements

Requirement	**Description**

REQ-1	System must support multiple payment methods such as cash and online banking.
REQ-2	System must securely handle payment information and comply with relevant standards.
REQ-3	System must update the order status upon successful payment completion.

3.2.8 Order Confirmation

3.2.8.1 Stimulus/Response Sequence

Use Case	Confirm Order
Short Description	To notifies users that their order has been processed, and the details has been confirmed.
Actor(s)	Customer, System
Pre-condition(s)	1. Customer has completed the payment process. 2. Order details are finalized.
Post-condition(s)	1. Order confirmation details are displayed and sent to the customer via email.
Main Flow	1. System verifies payment status. 2. System generates a confirmation summary with order details. 3. Customer reviews the confirmation details. 4. System sends an email with the order summary.
Alternative Flow(s)	N/A
Exception Flow(s)	N/A

3.2.8.2 Direct Functional Requirements

Requirement	**Description**
REQ-1	System must verify payment status before generating order

	confirmation.
REQ-2	System must display a detailed order summary upon confirmation.
REQ-3	System must send an email notification confirming order details after a successful transaction.

3.2.9 Order Update

3.2.9.1 Stimulus/Response Sequence

Use Case	Update Order
Short Description	To update customer's order status.
Actor(s)	Admin
Pre-condition(s)	1. Admin is logged into the app. 2. Order is confirmed but not yet processed.
Post-condition(s)	1. Updated order status is saved. 2. System displayed the order status.
Main Flow	1. Admin navigates to the order details. 2. Admin navigates to the order details. 3. System displays available status options: Preparing, Ready to Serve, Completed. 4. Admin selects a new status and confirms. 5. System saves changes and updates the order status.
Alternative Flow(s)	N/A
Exception Flow(s)	N/A

3.2.9.2 Direct Functional Requirements

Requirement	**Description**

REQ-1	System must allow admins to update the status of orders.
REQ-2	System must provide status options: Preparing, Ready to Serve, Completed.

3.2.10 Customer Feedback and Review

3.2.10.1 Stimulus/Response Sequence

Use Case	Give Feedback and Review
Short Description	To enables customers to provide feedback and reviews for completed orders.
Actor(s)	Customer
Pre-condition(s)	1. Customer is logged into the app. 2. Customer has received their order. 3. Feedback can only be submitted within 7 days of the order completion.
Post-condition(s)	1. Feedback and review are recorded in the system.
Main Flow	1. Customer receives order confirmation. 2. System displays a feedback form. 3. Customer submits their feedback and review. 4. System stores the review submission.
Alternative Flow(s)	Customer skips the feedback and reviews after order conformation: 1. Customer navigates to the order history. 2. Customer selects the option to provide feedback/review. 3. System displays a feedback form. 4. Customer submits their feedback and review. 5. System acknowledges and stores the submission.
Exception Flow(s)	N/A

3.2.10.2 Direct Functional Requirements

Requirement	Description
REQ-1	Display a feedback prompt after order confirmation.
REQ-2	Allow customers to provide feedback/review from the order history.

3.2.11 Admin Dashboard

3.2.11.1 Stimulus/Response Sequence

Use Case	-
Short Description	Provides the overview of the apps to monitor total sales and orders.
Actor(s)	Admin
Pre-condition(s)	1. Admin is logged into the app.
Post-condition(s)	1. Admin has access to real-time insights of the total sales and orders.
Main Flow	1. Admin logs into the dashboard. 2. Dashboard displays key metrics. 3. System updates the dashboard with real-time data.
Alternative Flow(s)	N/A
Exception Flow(s)	N/A

3.2.11.2 Direct Functional Requirements

Requirement	Description
REQ-1	Ensure real-time updates for critical information.

3.2.12 Admin View Orders

3.2.12.1 Stimulus/Response Sequence

Use Case	View Order
Short Description	To view a list of customer orders with details that include new order and completed order.
Actor(s)	Admin
Pre-condition(s)	Admin is logged into the app. Orders exist in the database.
Post-condition(s)	Admin has viewed or filtered the list of orders.
Main Flow	Admin navigates to the "Orders" section. System displays a list of all orders with its details that divided into three sections which are New Order, Processing and Completed Order. Admin selects an order for detailed viewing.
Alternative Flow(s)	N/A
Exception Flow(s)	N/A

3.2.12.2 Direct Functional Requirements

Requirement	**Description**
REQ-1	Display all order with relevant details.

3.2.13 Admin Add/Update Products

3.2.13.1 Stimulus/Response Sequence

Use Case	Update Products
Short Description	To add new products or update existing products including prices and description.
Actor(s)	Admin

Pre-condition(s)	1. Admin is logged into the app. 2. The products are available in the menu catalogue.
Post-condition(s)	1. Products details are updated. 2. System will display the updated products in the app.
Main Flow	1. Admin navigates to the "Menu" section. 2. Admin selects an item to edit or chooses to add a new one. 3. Admin updates the items by filling the new details including name, prices and description. 4. System validates inputs and saves changes.
Alternative Flow(s)	N/A
Exception Flow(s)	N/A

3.2.13.2 Direct Functional Requirements

Requirement	Description
REQ-1	System allows admins to add and edit items.

3.2.14 Admin Manage Feedback and Reviews

3.2.14.1 Stimulus/Response Sequence

Use Case	Manage Feedback and Reviews
Short Description	To view customer reviews and complaints, respond to them, or take corrective action where necessary.
Actor(s)	Admin
Pre-condition(s)	1. Admin is logged into the app. 2. Customer feedback and reviews exist in the system.
Post-condition(s)	1. Admin responds to customer reviews by replying, liking, or deleting the review if necessary.
Main Flow	1. Admin navigates to the "Feedback & Review" section.

	2. System displays a list of feedback and reviews. 3. Admin view customer's feedback and review. 4. Admin responds to the feedback by liking, replying or deleting the posts if necessary.
Alternative Flow(s)	N/A
Exception Flow(s)	N/A

3.2.14.2 Direct Functional Requirements

Requirement	**Description**
REQ-1	Display a list of feedback and reviews with filtering options.
REQ-2	Allow admins to respond to complaints and log their responses.
REQ-3	Notify customers when their complaint is resolved or responded to.

3.2.15 Admin View User's Profile

3.2.15.1 Stimulus/Response Sequence

Use Case	View Profile
Short Description	To view the detailed profiles of users includes their username, email and contact details.
Actor(s)	Admin
Pre-condition(s)	1. Admin is logged into the app. 2. Admin has necessary permissions to access user profiles.
Post-condition(s)	1. Admin successfully views the user's profile details.
Main Flow	1. Admin navigates to the Feedback and Review section of the app.

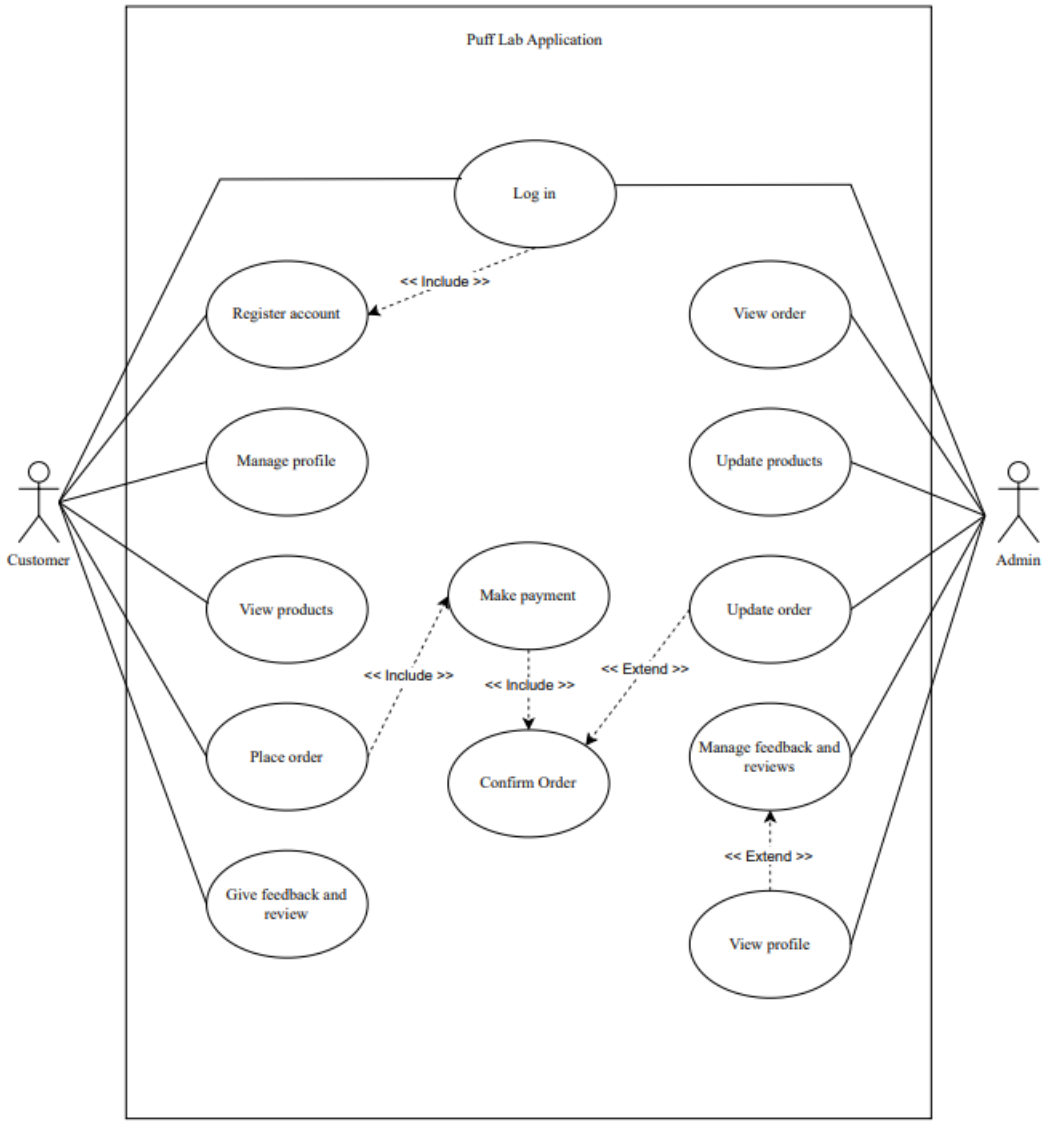
	2. Admin selects a specific feedback entry from the list. 3. System retrieves and displays the profile details of the user who submitted the feedback. 4. Admin reviews the profile, including name, email, contact details.
Alternative Flow(s)	N/A
Exception Flow(s)	N/A

3.2.15.2 Direct Functional Requirements

Requirement	**Description**
REQ-1	System must allow admins to view user profiles from the Feedback and Review section.

3.3 Behaviour Requirements

3.3.1 Use Case View



The use case diagram for the Puff Lab Application represents the interactions between two primary actors, Customer and Admin, and the system functionalities.

Actors:

1. **Customer:** Interacts with functionalities related to placing orders, managing their profile, and providing feedback.
2. **Admin:** Manages system operations such as updating products, orders, and viewing customer profiles.

Use Cases:

1. **Customer's Actions:**
    a. **Log in:** Required for accessing all features.
    b. **Register account:** Included in the login process for new customers.
    c. **Manage profile:** Allows customers to edit their personal details.
    d. **View products:** Displays available products in the menu catalogue.
    e. **Place order:** Enables customers to order products.
    f. **Make payment:** Making payment before completing an order.
    g. **Confirm order:** Receive an order confirmation form the system.
    h. **Give feedback and review:** Allows customers to provide reviews after order confirmation.
2. **Admin's Actions:**
    a. **Log in:** Grants access to admin features.
    b. **View order:** Checks order details placed by customers.
    c. **Update products:** Manages product inventory.
    d. **Update order:** Tracks and updates the status of customer orders.
    e. **Manage feedback and reviews:** Handles user reviews.
    f. **View profile:** Allows admins to access detailed user information.

Relationships:

- **<<Include>>:**
    - "Register account" is included in "Log in."
    - “Place order” is included in “Make payment”.
    - "Make payment" is included in "Place order."
- **<<Extend>>:**

- "Manage feedback and reviews" extends "Confirm order."
- "View profile" extends "Manage feedback and reviews."

4 Other Non-functional Requirements

4.1 Performance Requirements

Performance requirements are crucial to define the standards for how the PuffLab App should perform under various condition. It's to ensuring the deliverable meets both functional and user experience. The key performance requirements for the app we focused on response time, system capacity, availability, transaction throughput and data synchronization.

4.1.1 Response Time for User Action

Response time is the time required by the application to process a user's command and produce a response. For example, when a customer orders food or looks at the menu, the application should provide some sort of response or transition to another page as response quickly time as should possible be with less no or delay. The response time should be 2 seconds or less, the so that the users do not experience lag or delay, which in turns make the user experience better. A slow app can frustrate user easily and causing them to leave. Providing quick feedback would fulfill customer satisfaction.

4.1.2 Ordering Process Capacity

This requirement addresses the capacity of the app to manage a large volume of traffic, for example, when there are many customers active at the same time especially during rush hours for instance lunch or dinner hours. The app able to manage at least 50 orders at the same time without any challenges like slow response, errors or crashes. This entails the back-end servers and database of the application as well as the ability of the infrastructure to handle multiple requests at the same time. PuffLab Bakery is a very popular brand and receives many customers especially during the busy hours. If the system fails to process the orders concurrently then the customers may experience delays which may lead to complaints and loss of business. This performance measurement is to ensure that the app is capable of handling increasing customer base.

4.1.3 Easy Navigation

The application has a user-friendly interface which makes it easy for the user to browse through the different sections of the application like ordering, browsing through products and checking out. The layout should be simple with easy to find buttons and functions and enough buttons should be provided between which users of all levels of technical knowledge can easily get around and perform tasks with ease.

4.1.4 Data Synchronization Output

When changes are made by the admin (e.g., menu updates, price changes, or new product additions), these need to be reflected in real-time across the user interface. The application will ensure that any changes that are done are updated in a period that is less than five seconds. This involves the updating of the database, and all devices connected to the system so that customers have the most recent information on available items and their costs. Delay in data synchronizing could cause a problem whereby customers may purchase things that are not within the store or that have been marked as out of stock.

4.1.5 System Features and Accuracy

The application provides core functions like ordering, taking payments for orders, writing product comments and controlling an availability of items perform accurately and consistently. To illustrate, it is important to emphasize that order information gets captured and processed properly, payment transactions are successfully completed, and all availability item balances are updated in real time. Besides, the system should not contain any errors related with receiving and processing product comments and accurately convey changes on the user interface and on the administrator interface. Issues related core features could lead to customer dissatisfaction and might lost customer trust.

4.2 Safety and Security Requirements

4.2.1 Safety Requirements

- **Safety Payment Processing**
  To avoid mistakes or financial loss, be sure that every payment transaction is handled securely. To avoid misunderstanding and potential financial loss, the app should make use of reliable payment channels and make sure that the user is informed in a clear and concise manner of any payment failure.

- **Handling Orders**
  To prevent inaccurate purchases from being delivered or money from being charged, the PuffLab application ensure that customers' orders are processed accurately. This is essential to avoid financial losses or customers dissatisfaction. Regularly conduct testing procedures to ensure that the order processing system functions correctly.

- **Clear Error Message**
  Whenever a process, like the execution of a transaction or the entered of an input goes wrong, the PuffLab application offers its users a user-friendly error messages to provide help. Such as in situation when user didn't fulfill criteria during creation password, system will prompt user to create new strong password. This prompt is to prevent any confusion and ensure that users can deal with issues immediately rather than be left hanging that may cause frustration or even risky behaviors such as repeated payment attempts.

4.2.2 Security Requirements

- **Strong Password Creation**
  To protect user accounts and prevent unauthorized access, a strong password policy is essential. Enforcing strong passwords (e.g., combination of capital and lowercase letters, digits, and symbols) lowers the possibility that accounts will be

compromised by simple guesswork or brute force assaults. This fundamental security precaution aids in protecting critical company data and user data. Creating a strong password is a security measure because it directly influences the app's ability maintain data integrity.

- **Data Encryption**

  Data encryption ensuring protect the integrity and confidentiality of data between the Pufflab user's browser and the server. This will safeguard data both during transmission and while stored on the server (e.g., passwords, payment details, and personal information). In addition, this app would also need to comply with the PCI-DSS standards to be able to handle payment data securely to minimize chances of breach and maintain the trust of the user.

- **User Authentication and Access Control**

  PuffLab application system making sure that only user (admin and customers) with permission can access app functionalities. Enforcing robust authentication procedures will stop unwanted access to private information or features. the app ensures that users only access features relevant to their role, such as customers being able to place orders and admins managing inventory.

4.3 Software Quality Attributes

4.3.1 Reliability

Reliability on PuffLab Application refers to the capacity on consistently carry out its intended tasks. Users may place orders, pay, and check order statuses without any errors or disruptions when they use the app. The apps should be designed to perform minimal downtime and smooth performance.

How to achieve:

- Perform continuous testing to ensure that the app perform in high loads and it's to identify and fix the issue before deployment.

- Perform real time testing to quickly identify the problems and address any system failure if occur then immediately fix it.

4.3.2 Usability

Usability refers to how convenience and intuitively for user to explore and navigate the PuffLab Application, by focusing on provide an enjoyable, efficient, and straightforward experience when performing all functionalities with various features. It's also a user-friendly app especially for a new user to get familiar with basic functionalities.

How to achieve:

- Conduct a user-testing session that focus on User-Centered Design during the development process. Gather and analyze feedback from user to improve user Interface (UI) and User Experience (UX).
- Prioritize the features that focus only on what's important and make the design simple, so the user can understand it's features and navigate the app easily.
- Research and analyze which feature(s) are used the most to make the experience better.

4.3.3 Maintainability

PuffLab Application system's maintainability ensure it can adapt with future changes or be optimized for better performance. It's can easily be update, fixed and modified over the time if occur any issue. The new features that need to be update would not affect the current entire system, so it'll not give any problems for future developers to understands and maintain it.

How to achieve :

- Use modular programming technique to ensure some functionalities (e.g., payment processing and order system) independently.

- Make sure to documentation all fixed and updated features during testing phase so future developers can easily address what has been changed.

4.3.4 Flexibility

The ability of PuffLab application to expand its functionality or shifting needs without requiring a complete modification or significance change is called flexibility. This characteristic crucial in context to adapt any rapid changing to satisfied user needs. (e.g., easy **customization** of update and deleted menu items, prices. Also enabling Puff Lab app to add new features without extensive development work.

How to achieve:

- To add new features, PuffL ab application use a plug-in architecture, which allows for easy extension or modification of the application without interfering core functionality. It's also ensuring it can handle user traffic and additional features.

4.3.5 Testability

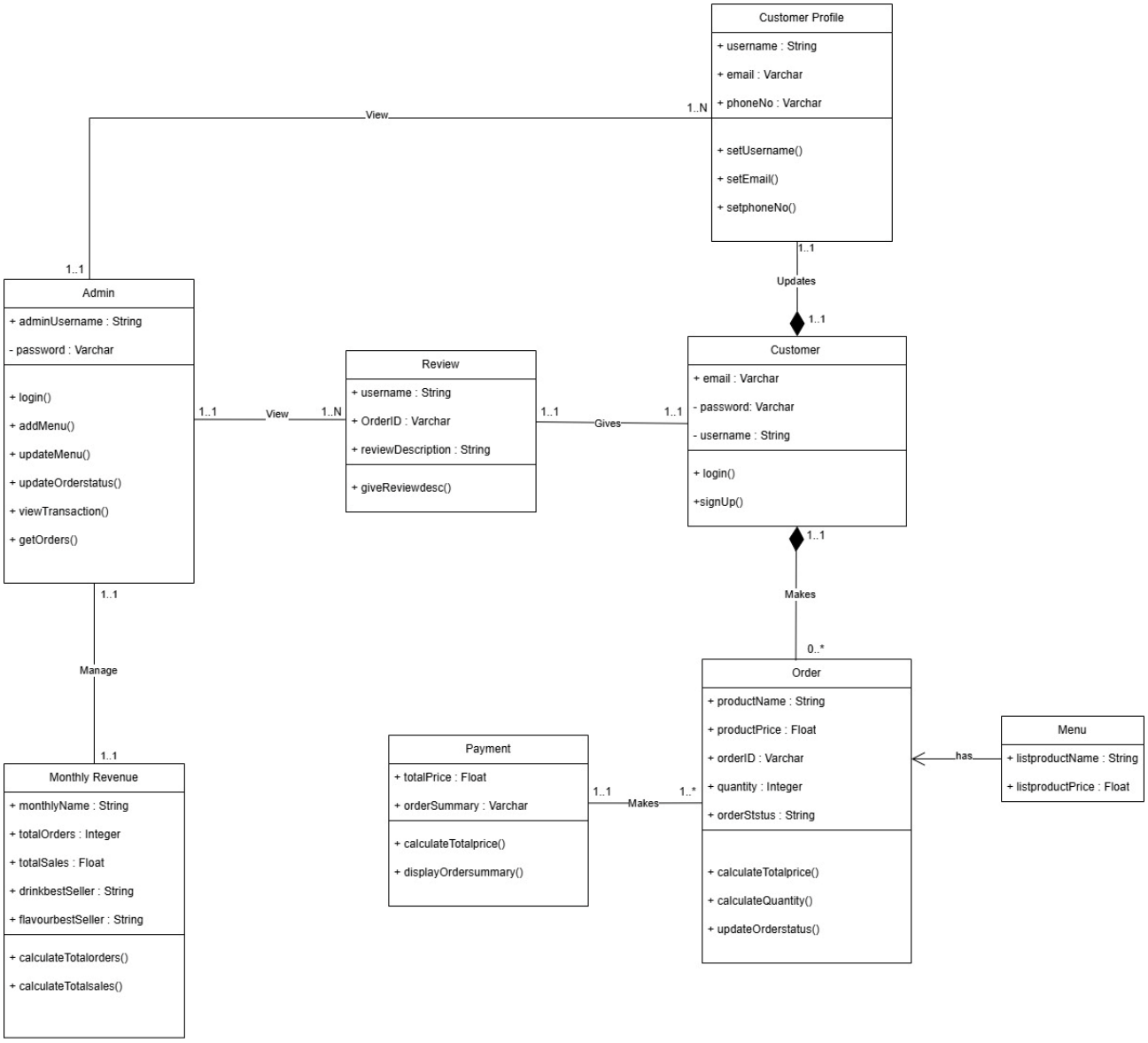
Testability focuses on the extent to which the app can be easily tested for correctness, performance, and reliability. In general, high testability improves the developers process to find bugs earlier and keep the level of the app's quality throughout its life cycle beyond standards.

How to achieve:

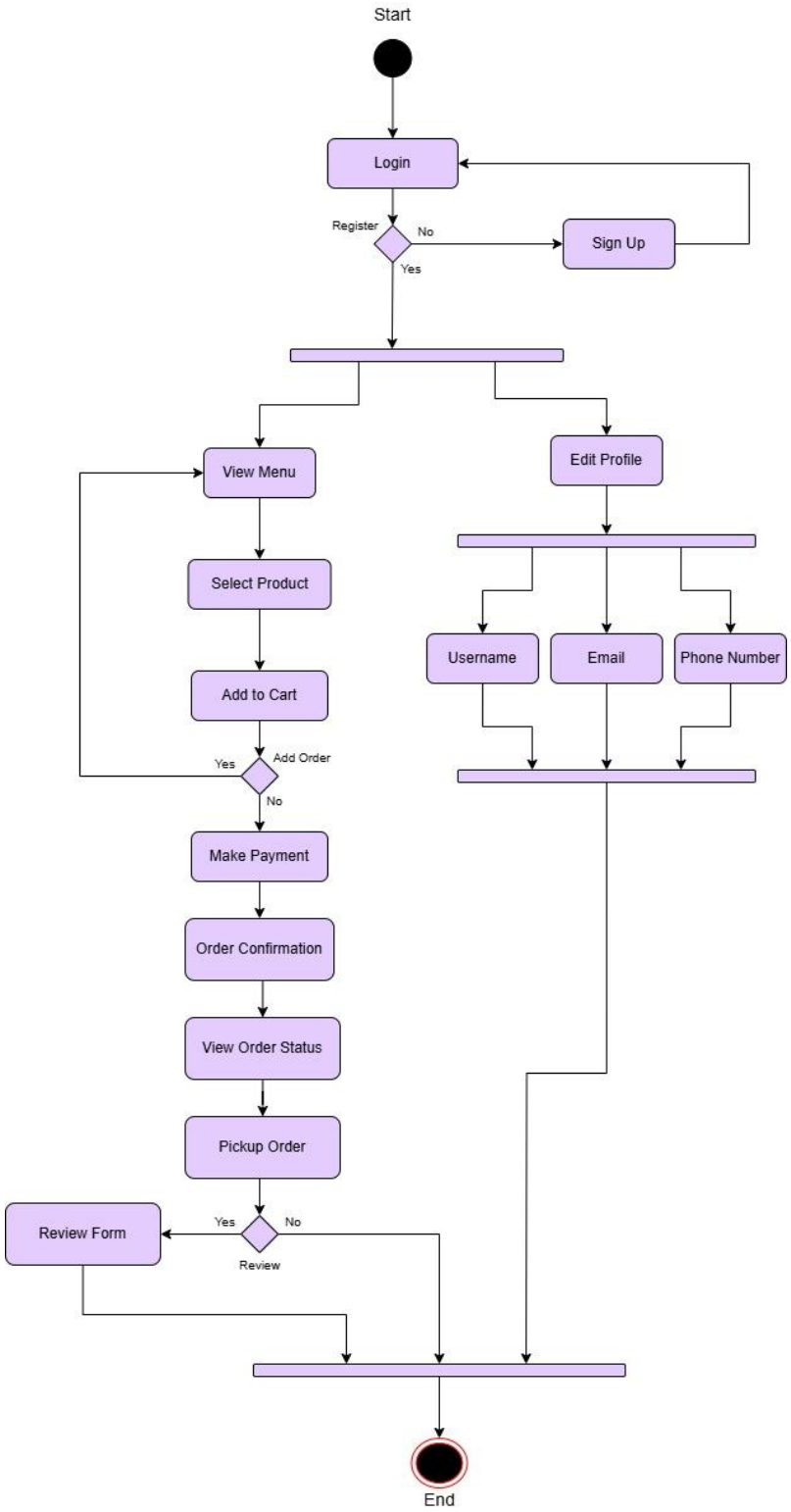
- Conduct continuous manual testing for core functionality or complex flows that might be difficult but to maintain app's performance.
- Using continuous Integration help a lot when there is new update or code be implements it's automatically tested.

5 Other Requirements

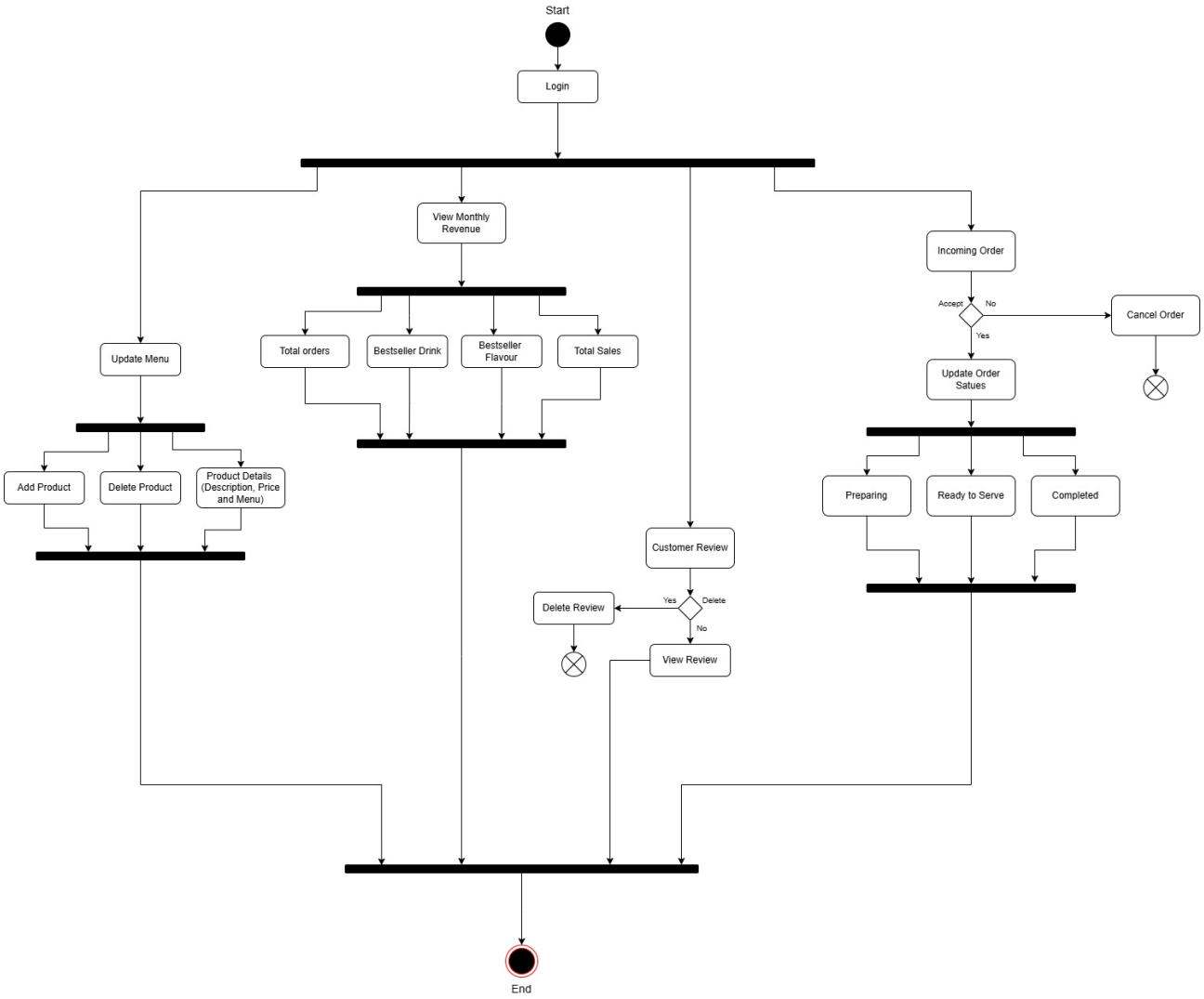
5.1 Class Diagram



5.2 Activity Diagram (User)



5.3 Activity Diagram (Admin)



Appendix A – Data Dictionary

1. Customer

1.1 Sign Up/Log in

Field Name	Data Type	Size	Description	Example
Customer Email	Varchar	30	Must be in email format	Wafi@gmail.com
Customer Password	Varchar	20	Password must be 8 length and unique.	Wafi1234!

1.2 Menu

Field Name	Data Type	Size	Description	Example
Menu Categories	String	50	Overall Menu	Regular/Premium/Luxury Flavour and Drinks
Products Name	String	50	List of all products	Vanilla/Matcha/Cranberry Cream Puff, Yam Latte
Prices	Float	10	Price of products	RM5.00
Products Picture	Varchar	255	Picture of products' menu	Yamlatte.jpg

1.3 Order

Field Name	Data Type	Size	Description	Example
Products Name	String	50	Selected products to order	Vanilla cream Puff
Prices	Float	10	Price of selected products to order	RM3.00
Quantity	Integer	100	Number of quantity products customer want to order	3
Order ID	Varchar	100	Customer will get order number to wait for their turn	#769
Order Status	String	50	Status of customer order	“Your order is being prepared”
Estimate Time Order	Time	8	Estimate time for order ready to pickup	06:00 PM

1.4 Payment

Field Name	Data Type	Size	Description	Example
Total price	Float	10	Total prices of all order	RM10.00
Receipt	Varchar/Image	255	Payment summary	Order summary

1.5 User Profile

Field Name	Data Type	Size	Description	Example
Username	String	80	Customer name	Lorem
Customer Email	Varchar	200	Customer registered email	Lorem@gmail.com
Customer Phone Number	Varchar	15	Customer personal number	016-1234567

1.6 Customer Reviews

Field Name	Data Type	Size	Description	Example
Review Description	String	200	Customer leave review based on order experience	Creampuff is delicious and creamy

2. Admin

2.1 Admin Log In

Field Name	Data Type	Size	Description	Example
Admin Username	Varchar	30	Admin entered the username to be navigate to admin dashboard	PuffLab_min1
Customer Password	Varchar	20	Password must be 8 length and unique.	Admin123!

2.2 Monthly Revenue

Field Name	Data Type	Size	Description	Example
Total Orders	Integer	350	Total order for every month	128
Drink Bestseller	String	50	Monthly customer favourite drink	Blue Lagon
Flavour Bestseller	String	50	Monthly bestseller creampuff flavour	Cranberry
Total Sales	Float	10	Monthly sales for all items	RM3928.00

2.3 Add/Update Menu

Field Name	Data Type	Size	Description	Example
Products Name	String	50	List of all products	Apong Cream Puff, Matcha Latte
Products Description	String	100	Description of product	Cheesy and full of cream inside
Prices	Float	10	Price of product	RM5.00

Products Picture	Varchar	255	Picture of products’ menu	Yamlatte.jpg
Products Flavour	String	15	Products' filling	Chocolate

2.4 Update Status order

Field Name	Data Type	Size	Description	Example
New Order Request	String	10	Admin can choose to accept or cancel the customer order	Cancel, Reject
Update Order Status	String	20	Admin can notify customer, for their order status	Ready to Serve, Preparing
Order Complete	String	10	Admin click button complete when order has been pickup by customer	Complete

Appendix B - Group Log

Project Meeting Minutes	
1. Meeting title:	Software Development for Puff Lab Application
Date & time started:	19 Nov 2024 (Tuesday)
Location (*optional*)**:**	Petary, UNIMAS
Project name:	Puff Lab Application
2. Attendees:	
Caleigh Susan Anak Jeffry	Project Manager / Planning Manager
Bethsheba Sewing	Customer Interface Manager
Delneeza Anak Dismon	Implementation Manager
Faziana Binti Mashor	Quality Manager / Test Manager
Khairul Wafi Bin Mazelan	Process Manager / Support Manager
3. Agenda item 1: Planning Phase	Discuss the project scope including features, functionalities and limitations of the system.
3.1 **Action items:**	1. Define the project scope [Assigned task: Delneeza Anak Dismon] 2. Identify features and functionalities [Assigned task: Faziana Binti Mashor] Deadline: 30 Nov 2025
4. Agenda item 2: Development Phase	Discuss about the project design and interface.
4.1 **Action items:**	1. Design the user interface (UI) and user experience (UX) for the software. [Assigned task: Bethseba Sewing] 2. Develop the prototype using figma. [Assigned task:

Project Meeting Minutes	
1. Meeting title:	Software Development for Puff Lab Application
	Caleigh Susan Anak Jeffry] Deadline: 1 December 2024
5. Agenda item 3: Functional and Non-Functional Requirement Analysis	Define the system requirements and the non-requirements of the system.
5.1 **Action items:**	1. Outline functional and non-functional Requirements. [Assigned Task: Khairul Wafi bin Mazelan]
6. Project updates:	Milestones Achieved: Completed the initial stage and designing the application. Next Steps: Developing the application.
8. Next meeting:	Meeting with Puff Lab's staff on Application Design
Date & time:	10 December 2025
Proposed agenda items:	Review of existing design
9. Meeting adjourned at:	2:30 pm
10. Minutes prepared by:	Faziana Binti Mashor (83865)
11. Minutes approved by:	Caleigh Susan Anak Jeffry

Group Activities

1. Meeting with Puff Lab's staff regarding the development of a Puff Lab's Application.
   Date: 23 October 2024
   Time: 12:00 PM
   Place: Puff Lab Store at Lakeview, UNIMAS

2. Group meeting for Software Development Puff Lab's Application.
   Date: 19 November 2024

Time: 1:00 PM

Place: Petary, UNIMAS