

ONLINE FOOD PRE-ORDERING SYSTEM

A PROJECT REPORT

Submitted by

DHANUSHYA V

(Reg. No.: 24MCR017)

KARTHIK SANTHOSH S

(Reg. No.: 24MCR050)

KIRUTHIGA M

(Reg. No.:24MCR057)

*in partial fulfilment of the requirements
for the award of the degree
of*

MASTER OF COMPUTER APPLICATIONS

DEPARTMENT OF COMPUTER APPLICATIONS



KONGU ENGINEERING COLLEGE

(Autonomous)

PERUNDURAI, ERODE – 638 060

DECEMBER -2024

DEPARTMENT OF COMPUTER APPLICATIONS**KONGU ENGINEERING COLLEGE****(Autonomous)****PERUNDURAI, ERODE – 638 060****DECEMBER-2024****BONAFIDE CERTIFICATE**

This is to certify that the project report entitled “**ONLINE FOOD PRE ORDERING SYSTEM**” is the bonafide record of project work done by **DHANUSHYA V (24MCR017)**, **KARTHIK SANTHOSH S(24MCR050)** and **KIRUTHIGA M (24MCR057)** in partial fulfilment of the requirements for the award of the Degree of Master of Computer Applications of Anna University, Chennai during the year 2024-2025.

SUPERVISOR**HEAD OF THE DEPARTMENT****Date:****(Signature with seal)**

Submitted for the end semester viva-voce examination held on _____

INTERNAL EXAMINER**EXTERNAL EXAMINER**

DECLARATION

I affirm that the project report entitled “**ONLINE FOOD PRE-ORDERING SYSTEM**” being submitted in partial fulfilment of the requirements for the award of Master of Computer Applications is the original work carried out by us. It has not formed the part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

DATE:

DHANUSHYA V

(Reg. No: 24MCR089)

KARTHIK SANTHOSH S

(Reg. No: 24MCR084)

KIRUTHIGA M

(Reg. No. 24MCR080)

I certify that the declaration made by the above candidates is true to the best of my knowledge.

Date:

Name and Signature of the Supervisor

[Dr. K. CHITRA]

ABSTRACT

The online food pre-ordering system (FoodQ) is a modern solution aimed at transforming the dining experience for students, faculty, and staff within the college campus. The platform facilitates seamless pre-ordering of meals, eliminating long queues and ensuring prompt service. Users can explore an extensive menu of food and beverages, view detailed nutritional information, and make secure online payments through an intuitive interface accessible via mobile phones or computers.

FoodQ emphasizes user convenience and inclusivity by offering meal customization options that cater to individual dietary preferences and nutritional needs. Users can adjust spice levels, modify ingredients, and select accompaniments, delivering a personalized dining experience tailored to diverse tastes. This feature empowers individuals to make healthier food choices and enjoy meals that align with their preferences.

To maintain operational efficiency, the system restricts its services to users within a 3 km radius of the campus, ensuring that only students, faculty, and staff on-site can place orders. This location-based functionality streamlines the food preparation and delivery process, guaranteeing timely and fresh meal service. Additionally, the platform provides tools for vendors to manage orders efficiently, reduce errors, and enhance productivity.

By integrating advanced features and promoting a hassle-free dining environment, FoodQ revolutionizes food court operations while aligning with the institution's commitment to innovation. This platform enhances user satisfaction, saves time, and establishes the food court as a vital component of a vibrant campus experience. Furthermore, it fosters a sense of community by addressing the diverse needs of the college population and ensures sustainability through optimized resource management. The system serves as a model for integrating technology into daily operations, demonstrating how digital solutions can enhance quality of life.

ACKNOWLEDGEMENT

We respect and thank our correspondent **THIRU.A.K.ILANGO, B.Com.,M.B.A.,LLB.,**and our Principal **Dr.V.BALUSAMY B.E(Hons)., M.Tech, PhD.** Kongu Engineering College, Perundurai for providing us with the facilities offered.

We convey our gratitude and heartfelt thanks to our Head of the Department **Dr.A.TAMILARASI MSc., M.Phil., PhD** Department of Computer Applications, Kongu Engineering College, for her perfect guidance and support that made this work to be completed successfully.

We also like to express our gratitude and sincere thanks to our project coordinator **Dr.T.KAVITHA MCA., M.Phil., PhD** Assistant Professor (Sr.G), Department of Computer Applications, Kongu Engineering college who have motivated us in all aspects for completing the project in scheduled time.

We would like to express our gratitude and sincere thanks to our project guide **Dr. K.Chitra MCA., M.Phil., PhD** Assistant Professor (Sr.G), Department of Computer Applications, Kongu Engineering College for giving her valuable guidance and suggestions which helped us in the successful completion of the project.

We owe a great deal of gratitude to our parents for helping overwhelm in all proceedings. We bow our heart with heartfelt thanks to all those who thought us their warm services to succeed and achieve our work.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No.
	ABSTRACT	iv
	ACKOWLEGEMENT	v
	LIST OF FIGURES	viii
	LIST OF ABBREVIATIONS	ix
1	INTRODUCTION	1
	1.1 ABOUT THE PROJECT	1
	1.2 EXISTING SYSTEM	2
	1.3 DRAWBACKS OF EXISTING SYSTEM	2
	1.4 PROPOSED SYSTEM	3
	1.5 ADVANTAGES OF PROPOSED SYSTEM	3
	1.6 ANTICIPATION OF CONCLUSIONS	4
2	SYSTEM ANALYSIS	5
	2.1 IDENTIFICATION OF NEED	5
	2.2 FEASIBILITY STUDY	5
	2.2.1 Technical Feasibility	6
	2.2.2 Operational Feasibility	6
	2.2.3 Economic Feasibility	6
	2.3 SOFTWARE REQUIREMENTS SPECIFICATION	6
	2.3.1 Hardware Requirements	9
	2.3.2 Software Requirements	9
	2.4 Software Description	10
3	SYSTEM DESIGN	12
	3.1 MODULE DESCRIPTION	12
	3.2 DATAFLOW DIAGRAM	14

	3.3 DATABASE DESIGN	17
	3.4 TABLE DESIGN	18
	3.5 INPUT DESIGN	19
	3.6 USER AUTHENTICATION	20
4	IMPLEMENTATION	23
	4.1 SYSTEM IMPLEMENTATION	23
	4.2 STANDARDIZATION OF THE CODING	24
	4.3 ERROR HANDLING	24
5	TESTING AND RESULTS	25
	5.1 SYSTEM TESTING	25
	5.1.1 Unit Testing	25
	5.1.2 Integration Testing	26
	5.1.3 Validation Testing	26
	5.5 RESULT	29
6	CONCLUSION AND FUTURE ENHANCEMENT	30
	6.1 CONCLUSION	30
	6.2 FUTURE ENHANCEMENT	31
	APPENDICES	32
	A. SAMPLE CODING	32
	B. SCREENSHOTS	39
	REFERENCES	51

LIST OF FIGURES

FIGURE No.	FIGURE NAME	PAGE No.
3.1	Data Flow Diagram Level 0	14
3.2	Data Flow Diagram Level 1	14
3.3	Data Flow Diagram Level 2	15
3.4	Class Based Diagram	16
3.5	Authentication Table Design	18
3.6	Menu Table Design	18
3.7	Order Table Design	19
3.8	Favourite Table Design	19
3.9	User Login Page	21
3.10	User Sign-Up-Page	22
5.1	Sign-up Page with user already exists	27
5.2	Login Page	28
B.1	Sign-Up page	39
B.2	Login	40
B.3	Menu Page	41
B.4	Home Page	42
B.5	Menu Details page	43
B.6	Cart page	44
B.7	Bill page	45
B.8	Payment page	46
B.9	Order page for user	47
B.10	Profile page	48
B.11	Admin page	49
B.12	Order page for admin	50

LIST OF ABBREVIATIONS

UI	User Interface
DFD	Data Flow Diagram
DOM	Document Object Model
API	Application Programming Interface
JSX	JavaScript XML
SDK	Software Development Kit
JSON	JavaScript Object Notation
DB	Data Base

CHAPTER 1

INTRODUCTION

1.1 ABOUT THE PROJECT

The online food pre-ordering system (FoodQ) for our college food court is designed to enhance and simplify the dining experience for students, faculty, and staff. By enabling users to pre-order their meals through the platform, it eliminates the hassle of waiting in long queues, ensuring prompt and efficient service. Users can browse a diverse menu of food and beverage options, select their desired items, and make secure online payments from their mobile phones or computers. Additionally, users have the flexibility to cancel their orders within 2 minutes of placing them, adding convenience and reducing potential errors. This system saves time and enhances the overall convenience of ordering meals, allowing individuals to focus on their commitments without worrying about delays at the food court.

The system provides detailed nutritional information for each dish, empowering users to make informed dietary choices. This feature caters to a wide range of needs, whether maintaining a balanced diet, adhering to specific guidelines, or exploring healthier options. Additionally, the support for meal customization allows users to personalize their orders by adjusting spice levels, modifying ingredients, or selecting accompaniments. These options accommodate the diverse preferences of the college community, providing a tailored dining experience that suits individual tastes.

To ensure operational efficiency, the platform restricts orders to users within a 3 km radius of the campus. This strategic limitation ensures that only students, faculty, and staff present on campus can place orders. By focusing on users within the campus vicinity, the system streamlines the entire process, ensuring meals are prepared fresh and delivered promptly.

By integrating these features, the system optimizes food court operations and fosters a seamless, stress-free dining experience. It reflects the institution's commitment to innovation and to addressing the evolving needs of its community, making the food court a cornerstone of a thriving campus environment.

1.2 EXISTING SYSTEM

The existing canteen system relies heavily on manual processes for food ordering, leading to various inefficiencies and errors in day-to-day operations. Users are required to physically visit the canteen to place their orders, resulting in long queues, unnecessary delays, and significant time spent waiting for their turn. These prolonged wait times often create communication gaps between users and staff, leading to misunderstandings or mistakes in processing orders. Furthermore, the absence of an online platform forces the canteen manager to rely on manual methods to maintain and track user details and order records. This outdated approach not only increases the risk of data loss but also makes it difficult to retrieve or analyze information when needed, hindering decision-making and operational improvements. The lack of digital organization contributes to chaotic workflows, as staff struggle to keep up with order volumes and customer expectations. This traditional system places undue pressure on both canteen staff and customers, ultimately resulting in a decline in service quality and overall dining efficiency. These challenges underscore the urgent need for a more modern and streamlined solution to enhance the canteen experience for all stakeholders.

1.3 Drawbacks of the Existing System

- **Long Wait Times:** High demand during peak hours results in prolonged waiting periods for meals.
- **Lack of Nutritional Information:** The absence of nutritional details prevents users from making informed and healthy choices.
- **Limited Menu Visibility:** Poorly displayed or inaccessible menus restrict customers from exploring all available food options.
- **Inconvenience in Meal Planning:** Difficulty in pre-ordering or customizing meals disrupts effective meal planning and healthy eating habits.
- **Vulnerability of Paper Records:** Paper-based records are prone to loss, damage, or misplacement, putting crucial customer data, order histories, and financial records at risk.

1.4 PROPOSED SYSTEM

The online food pre-ordering system is a cutting-edge platform designed to modernize and simplify the dining experience for students, faculty, and staff within the college campus. It enables users to pre-order their meals conveniently through a user-friendly interface, eliminating the need to wait in long queues and ensuring faster service. The system allows users to browse detailed menus with descriptions and pricing, making it easier to select their meals. For added convenience, users can save their favorite items, allowing for quick reordering in the future. A key feature of the platform is the ability to customize orders based on individual preferences, ensuring a personalized dining experience that caters to varied tastes and dietary needs.

The location-based functionality of the platform restricts its services to users within the college campus, ensuring that all orders are processed and delivered promptly without delays. This feature helps maintain service efficiency and ensures the canteen can effectively manage order volumes. For vendors, the system provides tools to manage orders seamlessly, organize workflows, and streamline operations, reducing errors and increasing productivity.

By integrating these features, the online food pre-ordering system aims to improve the overall efficiency of the canteen, enhance user satisfaction, and provide a modern dining solution tailored to the specific needs of the college community.

1.5 ADVANTAGES OF THE PROPOSED SYSTEM

- **Reduced Waiting Times:** Pre-ordering enables users to avoid long lines and receive their meals faster.
- **Customization:** Users can personalize their food and beverage orders to suit individual tastes and dietary requirements.
- **Time Flexibility for Students:** The system allows students to place orders at their convenience, accommodating their busy schedules.
- **Enhanced User Satisfaction:** Streamlined operations and a user-friendly interface create a seamless and enjoyable dining experience.
- **Personalization:** Customizable meal options cater to individual preferences and dietary requirements, further enhancing user satisfaction.

- **Location-Based Services:** Restricting the system to users within the college campus ensures prompt order processing and optimized resource management.
- **Favorite Items:** Users can save their favorite meals for convenient reordering in the future.
- **Enhanced Payment Experience:** Integration of secure payment options allows for seamless transactions, reducing the reliance on cash and simplifying order confirmation.

1.6 ANTICIPATION OF CONCLUSIONS

The online food pre-ordering system transforms the college dining experience by eliminating long waits and offering a user-friendly interface for convenient pre-ordering. With detailed menus, customizable options, and the ability to save favorite items, it ensures a personalized and efficient dining experience. Customization options ensure a personalized dining experience tailored to individual tastes and dietary needs.

Location-based functionality ensures prompt processing and delivery within the campus, while vendors benefit from tools that streamline operations and increase productivity. By enhancing the canteen's efficiency, the system improves overall service quality and user satisfaction. The inclusion of nutritional information encourages healthier food choices, empowering users to maintain balanced diets. Feedback mechanisms allow continuous improvement by addressing user concerns and preferences. The system's secure online payment feature adds a layer of convenience, ensuring hassle-free transactions. Ultimately, this innovative platform integrates technology to create a smarter, more enjoyable campus dining experience.

SUMMARY

This chapter explains the limitations of the current dining system and introduces the proposed online food pre-ordering system with its benefits. The project focuses on creating a user-friendly platform for pre-ordering meals in the college food court. It provided practical experience in designing interfaces using React Native and managing databases with Firebase. Features like online payments, customizable orders, and analytics were implemented. System analysis, along with hardware and software requirements, is covered in Chapter 2.

CHAPTER 2

SYSTEM ANALYSIS

2.1 IDENTIFICATION OF NEED

The current food court system is inefficient, with long wait times and no online ordering options. Students, faculty, and staff often experience delays during peak hours, making the dining experience frustrating. There is no easy access to nutritional information, limiting users' ability to make healthy choices. The absence of customizable meal options results in a less personalized experience. The manual order-taking process increases the likelihood of errors and miscommunication. Users face limited flexibility in meal planning, especially during busy periods.

A digital platform is needed to streamline orders, reduce wait times, and enable meal customization. The system should also provide nutritional information and real-time order tracking. Additionally, incorporating a feedback mechanism would ensure continuous improvement. An online pre-ordering system would significantly enhance both operational efficiency and customer satisfaction.

2.2 FEASIBILITY STUDY

A feasibility study is an analysis conducted to determine whether a proposed project or system is practical and likely to succeed. It examines several key factors, including technical, operational, financial, legal, and social aspects, to assess the project's overall viability. The purpose of a feasibility study is to establish whether the project can be realistically completed within the available time, budget, and resources, and whether it will achieve its intended goals. In essence, it helps decision-makers understand the potential risks and benefits of the project before proceeding.

Three considerations involved in feasibility are:

- Technical Feasibility
- Operational Feasibility
- Economic Feasibility

2.2.1 TECHNICAL FEASIBILITY

The system is developed using React Native, JavaScript, and Firebase, offering scalable and secure solutions for pre-ordering, payments, and real-time data management. React Native ensures cross-platform compatibility, enabling the app to work seamlessly on both Android and iOS devices. Firebase provides efficient backend support with features like database management, authentication, and cloud hosting. These technologies ensure a robust, user-friendly platform that meets the functional requirements of the college community.

2.2.2 OPERATIONAL FEASIBILITY

The system integrates effectively with existing canteen operations, focusing on streamlining meal ordering and delivery processes. By offering features like pre-ordering and meal customization, the platform enhances convenience and reduces waiting times. The absence of customer feedback and rating features simplifies implementation, making the system easier to adopt and manage while maintaining its primary focus on efficiency and user convenience.

2.2.3 ECONOMIC FEASIBILITY

The system is financially viable, as it reduces operational costs associated with manual order management and minimizes errors. Faster service and efficient menu management can increase customer throughput, potentially boosting revenue for the canteen. The use of cost-effective tools like Firebase for backend operations further optimizes the budget, making the system a practical investment that delivers long-term value to both the canteen and its users.

2.3 SOFTWARE REQUIREMENT SPECIFICATION

The Software Requirements Specification (SRS) for our mini-project, College Food Court Food Pre-Ordering System, defines the functional and non-functional requirements for creating an efficient, user-friendly platform. It addresses issues like long wait times and lack of customization, ensuring the system enhances convenience, improves operations, and delivers timely service for the college community.

FUNCTIONAL REQUIREMENTS

1. User Registration & Authentication:

- Users must be able to create an account using their email or social media credentials.
- The system will ensure secure login functionality for all users.

2. Menu Management:

- Users can browse the food and beverage menu with detailed descriptions and pricing.
- Nutritional information for each menu item will be displayed, enabling informed dietary choices.
- Vendors have the ability to update, delete, or add new menu items.
- Users can view the estimated preparation time for their selected food items.

3. Order Placement:

- Users can add items to their cart and customize their orders by adjusting spice levels, modifying ingredients, or selecting accompaniments.
- Orders will include options for specifying quantities, customizations, and meal preferences.

4. Payment Integration:

- Users can complete their orders through secure payment gateways, including options like credit/debit cards, UPI, and digital wallets.

5. Admin Dashboard:

- Vendors will have access to an admin dashboard to view and manage incoming orders.
- The dashboard allows vendors to analyze order trends and adjust the menu to match customer preferences effectively.

NON-FUNCTIONAL REQUIREMENTS

1. Performance:

- The system should support up to 1000 concurrent users without performance degradation.
- Orders must be processed in real time to avoid delays.

2. Security:

- The platform will employ HTTPS for secure data transmission.
- Payment transactions will use secure, encrypted gateways like Google Pay (GPay) and PhonePe.

3. Scalability:

- The system will be designed to handle increased traffic efficiently during peak hours, such as lunch breaks.

4. Usability:

- The user interface will be intuitive, user-friendly, and accessible for all members of the college community, requiring minimal training.

5. Backup and Recovery:

- Data backup mechanisms will ensure the safety of user data.
- Disaster recovery systems will prevent data loss in case of unexpected failures.

6. Reliability:

- The system should ensure high availability with minimal downtime, even during peak usage periods.
- Regular system monitoring will be in place to detect and resolve issues promptly.

2.3.1 HARDWARE REQUIREMENTS

1. PROCESSOR:

- Apple M1 chip (best for macOS development) or Intel i3 (basic but sufficient for light tasks).

2. RAM:

- 8 GB minimum for smooth multitasking and development.

3. WEB SERVER:

- Dedicated server or cloud service (e.g., Firebase) for hosting and scaling.

2.3.2 SOFTWARE REQUIREMENTS

Frontend:

- IDE : Visual Studio Code
- Languages : JavaScript, React Native
- Browsers : Google Chrome, Mozilla Firefox (for testing)
- Development Tools : Expo (for React Native)

Backend:

- Languages : JavaScript
- Web Server : Node.js
- Database : Firebase (real-time database & authentication)
- Version Control : Git
- API Integration : Payment gateways (e.g., Google Pay, PhonePe)
- Package Manager : npm (for managing dependencies)

2.4 SOFTWARE DESCRIPTION

FRONT END

React Native is a widely recognized open-source framework developed by Meta (formerly Facebook) that allows developers to build mobile applications for both Android and iOS platforms using a single codebase. It is based on React, a JavaScript library popular for creating user interfaces, but it extends its capabilities to mobile development by providing native-like performance and appearance.

Key Features of React Native:

1. Cross-Platform Development

Write one codebase for both Android and iOS platforms, saving time and effort. Ideal for startups and enterprises to optimize resources effectively.

2. Hot Reloading

View code changes in real-time without restarting the app. Speeds up development and debugging, enabling faster iterations.

3. Native-Like Performance

Uses native components to deliver near-native app performance. Ensures smooth animations and interactions without relying on WebView.

4. Reusable Components

Breaks the app into modular, reusable components, reducing duplication. Simplifies maintenance and allows component sharing across projects.

5. Ecosystem

Offers extensive tools, plugins, and third-party integrations. Supported by an active developer community for enhanced development.

BACK END

Firebase, developed by Google, is a versatile platform for web and mobile app development. It offers services like Realtime Database for live data syncing, Authentication for secure user login, and Cloud Firestore for scalable data storage. Firebase Hosting provides fast, secure web hosting with a global CDN, while Firebase Analytics tracks user behaviour to enhance app performance and engagement.

Key features of Firebase:

Realtime Database: Real-time data sync across clients.

Cloud Firestore: Scalable NoSQL database for data storage.

Authentication: Easy user sign-in with multiple methods.

Firebase Hosting: Fast, secure web app hosting with CDN.

Cloud Functions: Run server-side code in response to events.

Cloud Storage: Scalable file storage for media and documents.

Analytics: Track user behavior and app performance.

Push Notifications: Send notifications across platforms.

Crashlytics: Real-time crash reporting for bug fixes.

SUMMARY

This chapter outlines the feasibility study for the Online Food Pre-Ordering System, emphasizing its advantages over the traditional manual ordering process. It examines the economic, operational, and technical feasibility, including the necessary hardware and software requirements. The study also addresses critical aspects such as system performance, security, and reliability. Detailed system design and implementation will be explored further in Chapter 3.

CHAPTER 3

SYSTEM DESIGN

3.1 MODULE DESCRIPTION

A module description provides detailed information about the module and its supported components, which is accessible in different manners. In this application, it contains many sub- modules.

Register

The Register Module allows new users (students or admins) to create an account by providing essential details such as name, email, password, and user role. After successful registration, users can log in to access the app's features.

Menu Module

The Menu Module presents a wide range of food items with detailed nutritional information, enabling users to make healthier and informed decisions. It includes customization features, such as the ability to modify ice and sugar levels to suit individual preferences. Users can effortlessly add items to their cart with a simple and intuitive interface.

Cart Module

The Cart Module manages the items users select while shopping. Users can view the contents of their cart, adjust quantities, remove items, and proceed to checkout. It is integrated with the payment system to process the final order once the user decides to place it.

Profile Module

The Profile Module lets users manage their account details, including personal information. Students can update their preferences, while admins can edit their credentials and view system-related activity.

Favourites Module

The Favourites Module allows users to save their preferred food items for quicker access in the future. They can mark items as favourites from the menu, providing them with a personalized selection for easy reordering. These modules work together to provide a seamless and efficient user experience within the FoodQ app.

Bill Module

The Bill Module allows users to view a detailed breakdown of their orders, including cost, taxes, and discounts. It supports secure online payments with multiple options, ensuring a seamless transaction experience. Users can easily review, finalize their orders, and track payment status. This integration enhances convenience and efficiency, providing a smooth and confident ordering process.

Payment Module

The Payment Module generates a QR code with the total amount of the order, allowing users to make payments securely. After reviewing their order, users can scan the QR code through their preferred payment app. The code includes the total cost, taxes, and any applicable discounts, streamlining the payment process. This feature provides a fast, contactless, and secure way to complete transactions, enhancing the overall user experience.

Admin Module

The Admin Module provides a centralized platform for food court administrators to manage the menu in the FoodQ app. Admins can add, update, or delete menu items, including details like name, description, price, and nutritional information, ensuring the menu is accurate and up-to-date. This streamlined system simplifies menu management, helping administrators maintain an efficient and user-friendly food ordering experience.

3.2 DATAFLOW DIAGRAM

The Data Flow Diagram provides information about the inputs and outputs of each entity and process itself.

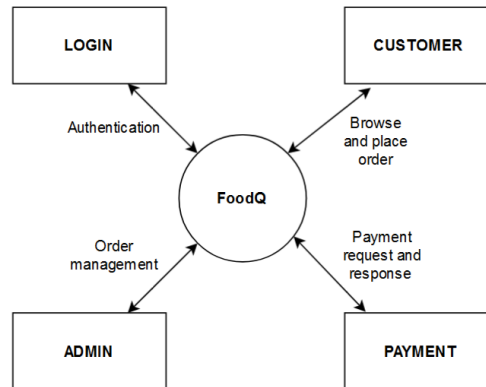


Figure 3.1 Dataflow Diagram Level 0

In Figure 3.1 Admin can view customer, view the products in the app and also manage the orders. Customer can view the products and they can place their orders.

LEVEL 1

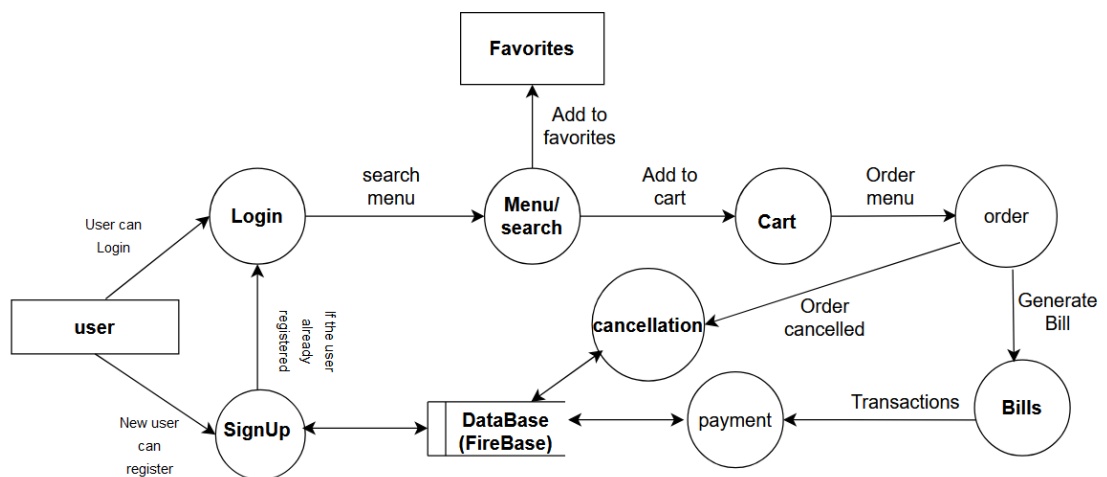


Figure 3.2 Dataflow Diagram Level 1

In Figure 3.2, the user can log in to the system using their credentials to access the menu and browse the available food items. After exploring the menu and reviewing detailed nutritional information, the user can add their preferred items to the cart for easy management. If the user decides to proceed with the purchase, they can confirm the order through a secure payment gateway, ensuring a safe and convenient transaction. Additionally, the system provides the flexibility to cancel the order within a 120-second window after confirmation, offering users a quick option to make changes if needed. This feature enhances user satisfaction by accommodating last-minute adjustments.

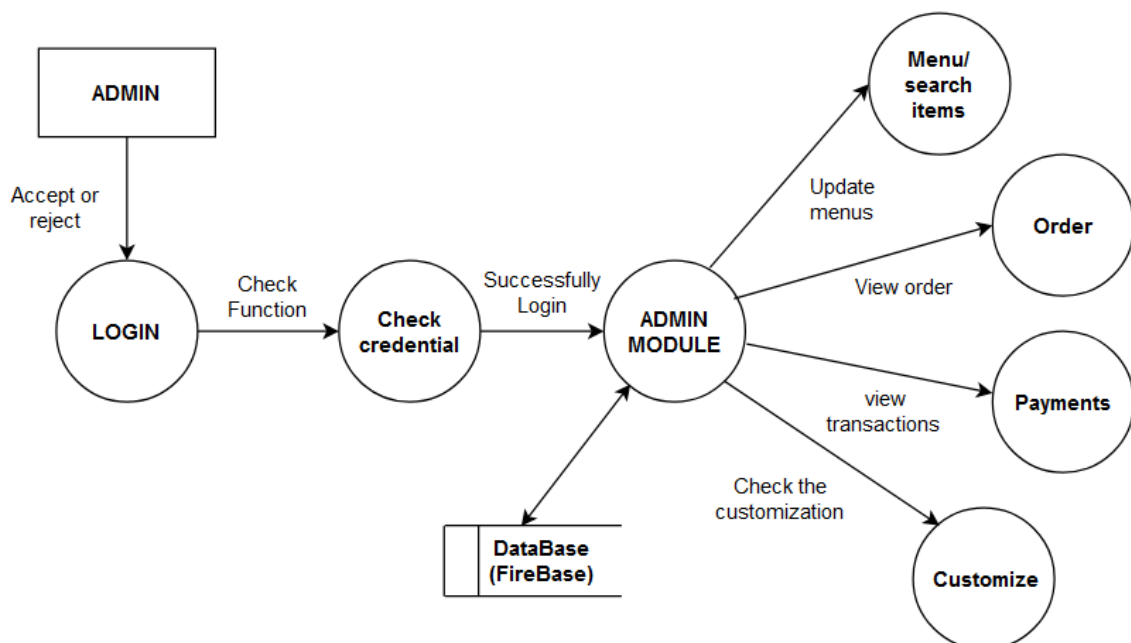
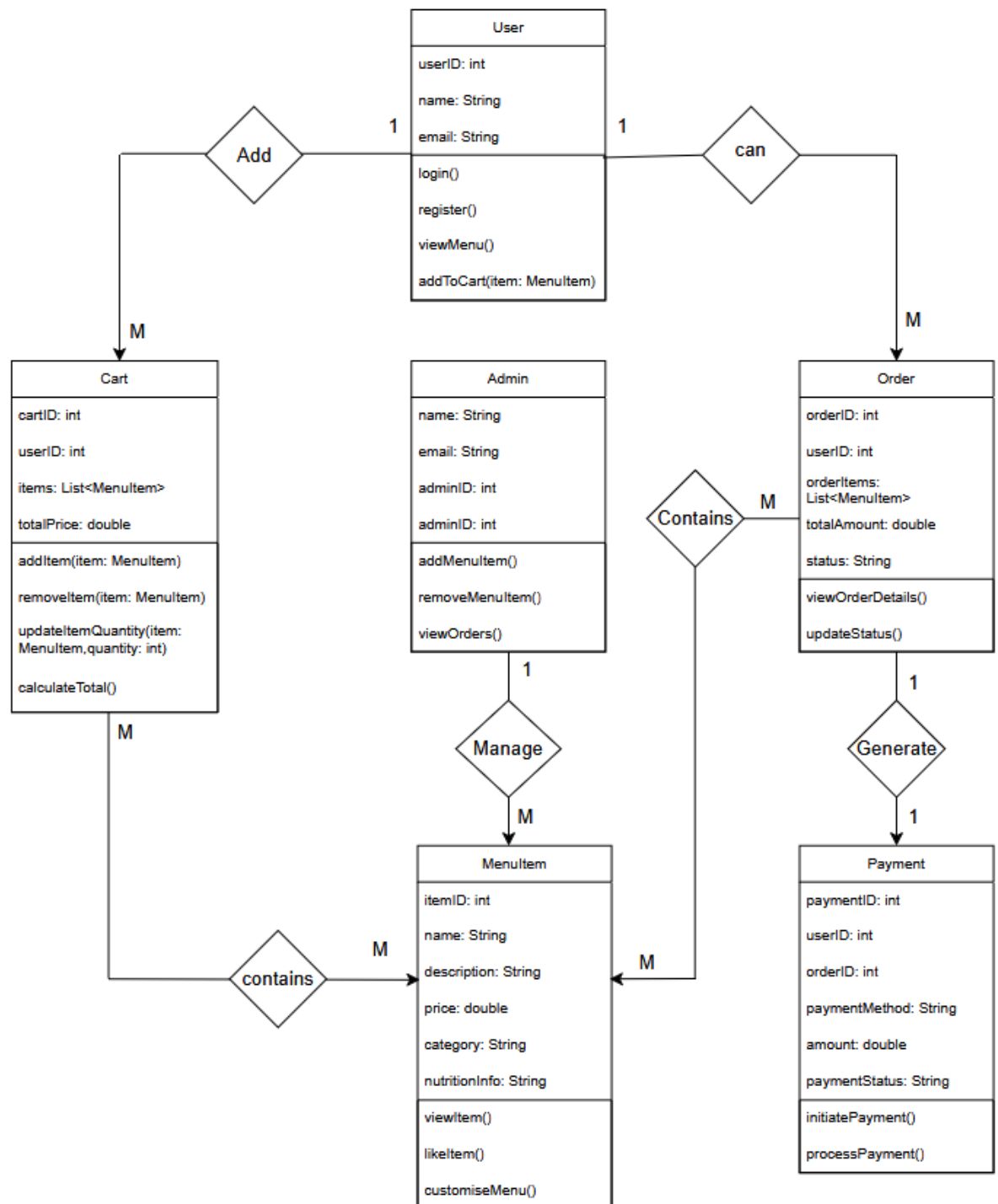


Figure 3.3 Dataflow Diagram Level

In Figure 3.3 In Figure 3.3, the Admin has a dedicated interface to log in and manage the overall functionality of the food pre-ordering system. After logging in, the Admin can efficiently maintain the list of items displayed to users by adding new products or editing existing ones, such as updating prices, descriptions, or nutritional information. The Admin also oversees the orders placed by customers, ensuring they are processed accurately and on time.

CLASS DIAGRAM:**Figure 3.4 Class based diagram**

3.3 DATABASE DESIGN:

Database design involves organizing and structuring data to ensure efficient storage, retrieval, and management. The process starts with identifying what data needs to be stored and determining the relationships between different data elements. A well-designed database aligns the structure with the application's requirements and ensures optimal performance. The core focus is on defining the logical data model that supports the application's functionality while maintaining scalability and reliability.

Data Integration:

Data integration ensures that information from multiple sources is combined seamlessly into a unified view. In a centralized database system, data from various files or modules is accessed as if it resides in one location, regardless of its actual physical storage. This coordination reduces redundancy, ensures consistent data across all systems, and simplifies updates, as a single change reflects across all dependent applications. Centralized integration enhances data accuracy and minimizes storage requirements.

Data Independence

Data independence refers to isolating the application's operations from changes in the physical storage structure or data organization. It ensures that developers can modify the database schema—such as adding new attributes or restructuring data—without requiring changes to the application code. Similarly, application updates can occur without altering the underlying data storage. This adaptability is crucial for systems that grow or evolve over time, as it minimizes disruptions and facilitates maintenance.

3.4 TABLE DESIGN

Field Name	Data Type	Size	Constraints
user_id	varchar	-	Primary key
user_name	varchar	15	unique
password	varchar	10	not null
name	varchar	20	not null
email	varchar	15	not null

Table 3.5 auth_user

In this table 3.4 user details such as user ID, name, email, and password are stored for customers (students, faculty, and staff) who are registering to use the food pre-ordering system.

Field Name	Data Type	Size	Constraints
product_id	integer	-	Primary key
product_name	varchar	15	not null
category	varchar	30	not null
sub_category	varchar	30	not null
price	integer	-	not null
image	varchar	30	not null
description	varchar	200	not null

Table 3.6 food_menu

This table stores product details such as product ID, name, category, sub-category, price, and description. These products represent food items available for pre-order.

Field Name	Data Type	Size	Constraints
order_id	integer	-	Primary key
user_id	varchar	15	foreign key
amount	varchar	-	not null
Order_time	timestamp	-	not null
status	varchar	20	not null

Table 3.7 orders

In the above table 3.6 details such as order ID, user ID, food, total amount, order_time and the status of the order are stored.

Field Name	Data Type	Size	Constraints
favorite_id	integer	-	Primary key
User_id	varchar	15	foreign key
Product_id	integer	-	foreign key

Table 3.8 favorites

In the above table 3.7 details such as user ID, food item ID, and the status of the item (whether it's added to the favorites) are stored, allowing users to personalize their experience by saving their favorite food items for quicker future orders.

3.5 INPUT DESIGN

Input design plays a critical role in the development of any system, as it focuses on converting user inputs into a format that the system can effectively process. In the context of the Online Food Pre-Ordering System, input design is pivotal in ensuring that the system functions smoothly and without errors. A well-thought-out input design minimizes potential issues such as incorrect data, user confusion, or system failures, ultimately allowing the system to operate more efficiently. The input forms within the system are carefully crafted to be intuitive and user-friendly, ensuring that users can easily select their meal preferences, customize their orders, and provide all necessary

details such as delivery information, special requests, and payment preferences. To further enhance the reliability of the system, validation mechanisms are built into the input process, ensuring that all inputs are correct before they are processed. This prevents errors during the ordering process and ensures that only valid orders are submitted. The system is designed to utilize cost-effective, freely available technologies, making it easy to implement while still meeting the diverse needs of users. Despite its affordability, the system does not compromise on performance or functionality, ensuring that all user requirements are met with the highest standards of efficiency and reliability.

3.6 USER AUTHENTICATION

The online food pre-ordering system ensures secure access through a robust user authentication process, which is critical for protecting user data and maintaining the integrity of the platform. The login and sign-up processes are designed to validate user credentials, allowing only authorized users to access the system's features, such as placing orders and customizing their meal preferences. Proper validation mechanisms are in place to verify the information provided by users, preventing unauthorized access.

- **Figure 3.9** Figure 3.8 illustrates the User Login Page, where users must enter a valid username and password combination to gain access to the platform. This page is equipped with security features, including password strength requirements and encryption, to ensure that user credentials remain protected.
- **Figure 3.10** displays the User Sign-Up Page, where new users can register for the platform by providing required credentials. This includes a valid institutional email address to verify their affiliation with the institution. The registration process ensures that only authorized users can access the system. Upon successful registration, users can proceed to log in and access the platform's features.

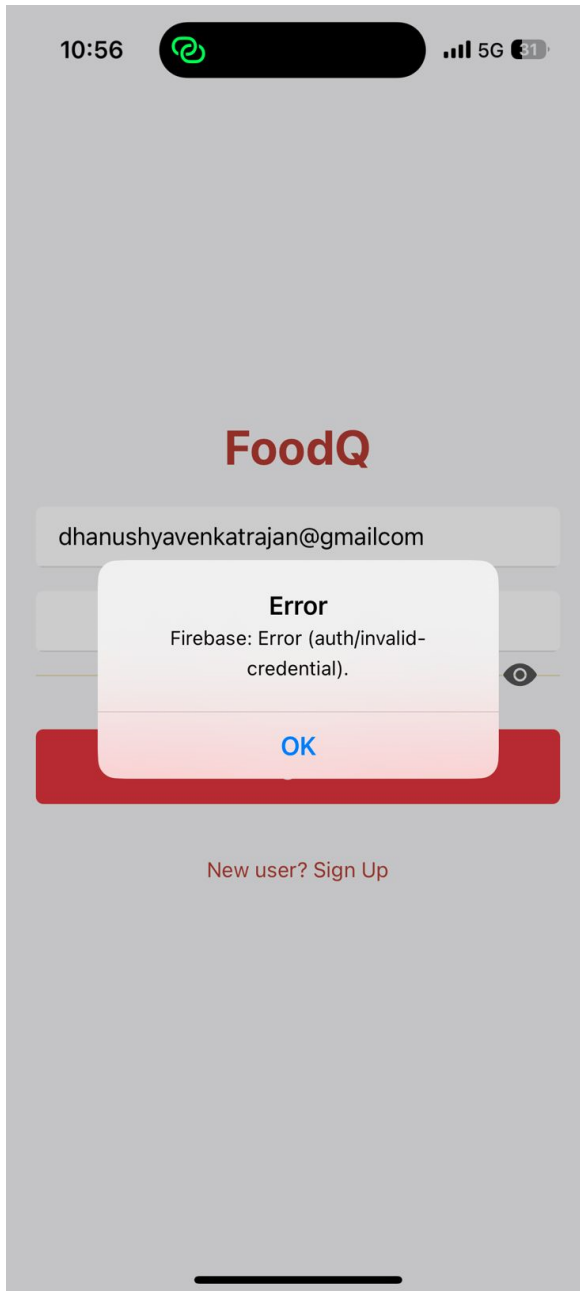


Figure 3.9: User Login Page

The user login page requires users to enter their username and password.

- The username must be in a proper format and in lowercase.
 - If the fields are left empty, the user cannot log in or place orders.
 - The password must contain at least 6 digits or characters.
- In case of an invalid format or missing credentials, the system will display an error message, as seen in the figure.

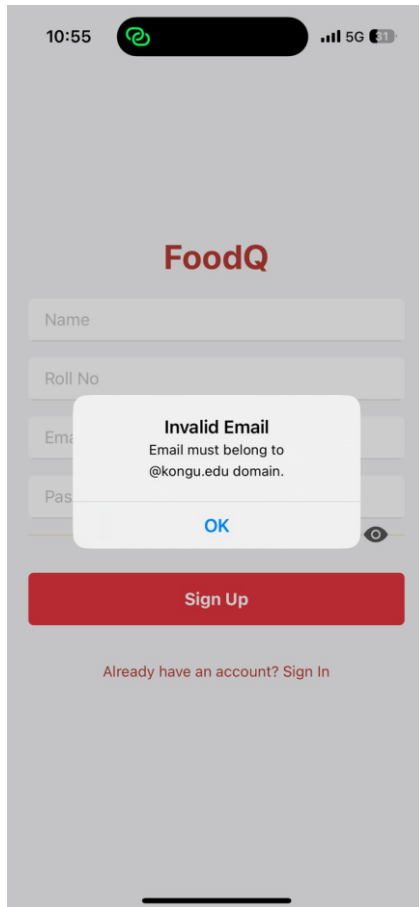


Figure 3.10: User Sign-Up Page

The user sign-up page requires input for Name, Roll No, Email, and Password.

- The email must belong to a valid @kongu.edu domain.
- If the email does not follow the required domain format, the system will display an Invalid Email error, preventing sign-up.
- This ensures only authorized users can create an account to access the system.
- All fields are mandatory to complete the sign-up process.

SUMMARY

This chapter covers the Input and Output design, Admin and User modules, and includes the Data Flow Diagram, Class Diagram, and database design. It outlines key system functions and data integrity, with implementation details in Chapter 4.

CHAPTER 4

IMPLEMENTATION

4.1 SYSTEM IMPLEMENTATION

Implementation is the stage in the project where the theoretical design is turned into a working system. The most critical stage is achieving a successful system and giving confidence in the new system for the users, that it will work efficiently and effectively. It involves careful planning, investing of the current system, and its constraints on implementation, design of methods to achieve the changeover methods. The implementation process begins with preparing a plan for the implementation of the system. According to this plan, the activities are to be carried out in these plans; discussion has been made regarding the equipment, resources, and how to test activities.

The coding step translates a detailed design representation into a programming language realization. Programming languages are vehicles for communication between humans and computers. Programming language characteristics and coding styles can profoundly affect software quality and maintainability. The coding is done with the following characteristics in

- Ease of design to code translation.
- Code efficiency.
- Memory efficiency and maintainability.

The user should be very careful while implementing a project to ensure what they have planned is properly implemented. The user should not change the purpose of project while implementing. The user should not go in a roundabout way to achieve a solution; it should be direct, crisp and clear and up to the point.

4.2 STANDARDIZATION OF CODING

React Native follows certain rules and maintains a consistent coding style. Indentation is used to indicate the start and end of control structures, clearly specifying the sections of code between them. Consistent naming conventions for variables are followed throughout the code, and data is properly described. Comment lines are used to help developers understand the code. These comments improve the understanding of the code without affecting its core functionality. The code is written to enhance readability while maintaining its essential function.

4.3 ERROR HANDLING

An exception can be defined as an abnormal condition in a program that disrupts the normal flow of execution. When an exception occurs, the program halts, preventing the further execution of code. In React Native, an exception is an error that React Native cannot handle by default. However, React Native provides methods to handle exceptions, allowing the rest of the code to continue running without disruption. This technique ensures that errors are managed gracefully, and the application remains functional.

SUMMARY

In this chapter, system implementation focuses on how the Online Food Pre-Ordering System is developed, ensuring it is functional and meets quality standards. We explored writing standardized, readable code with clear comments for better understanding and handling exceptions effectively. The system ensures smooth operation, even when errors occur. Testing and results for the project are discussed in Chapter 5.

CHAPTER 5

TESTING AND RESULTS

5.1 SYSTEM TESTING

Software testing serves as the final assessment of specifications, designs, and coding and is a crucial component of software quality assurance. The system is tested throughout the testing phase utilizing diverse test data. The preparation of test data is essential to the system testing process. The system under study is tested after the test data preparation. Once the source code is complete, relevant data structures should be documented. The finished project must go through testing and validation, when errors are explicitly targeted and attempted to be found.

The project developer is always in charge of testing each of the program's separate units, or modules. Developers frequently also perform integration testing, which is the testing phase that results in the creation of the complete program structure.

This project has undergone the following testing procedures to ensure its correctness

- Unit testing
- Integration Testing
- Validation Testing

5.1.1 UNIT TESTING

- **Testing Each Unit:** Unit testing verifies that each unit of the food pre-ordering system functions as expected.
- **Developer Responsibility:** The web developer performs unit testing during the application development process.
- **Independent Module Testing:** Each module is tested independently to locate errors before integration into the full system.
- **Error Detection:** Unit testing helps identify coding and logic errors within individual modules.

- **Program Testing:** Unit testing is also known as Program Testing, as it focuses on testing the program's functionality.
- **Early-Stage Testing:** Testing is carried out during the programming stage, allowing developers to address issues early.
- **Efficiency:** Early detection and resolution of issues lead to a more efficient and robust food pre-ordering system.

5.1.2 INTEGRATION TESTING

- **Purpose:** Integration testing ensures that individual modules of the food pre-ordering system work together as a cohesive unit.
- **Test Data:** Instead of manual test data, automatic data generated from various modules is used to test the integration.
- **Interface Testing:** The focus is on testing how the modules interact with each other and the proposed system.
- **Identifying Issues:** Integration testing helps identify problems in the interfaces between modules.
- **Sequential Integration:** The modules are integrated one by one and tested to ensure they function properly when combined.
- **System Interaction:** This testing checks how the modules communicate and ensure that data flows correctly between them in the food pre-ordering system.

5.1.3 VALIDATION TESTING

Validation testing can be defined in many ways, but a simple definition is that can be reasonably expected by the customer. After the validation test has been conducted, one of two possible conditions exists.

The functions or performance characteristics confirm to specification and are accepted. A deviation from the specification is uncovered and a deficiency list is created.

The proposed system under consideration has been tested by using validation testing and found to be working satisfactorily. For example, in this project validation

testing is performed against the user module. This module is tested with the following valid and invalid inputs for the fields.

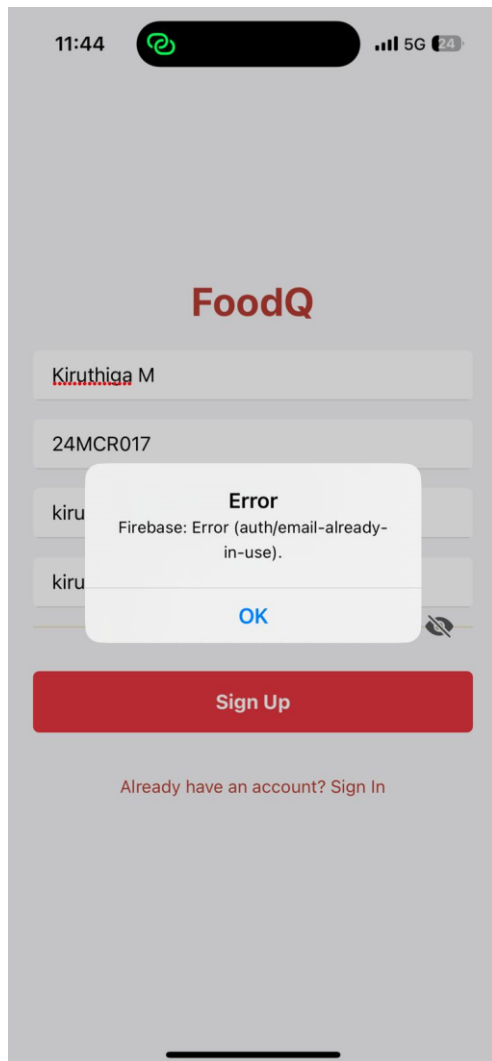


Figure 5.1 Sign-up Page with user already exists

In Figure 5.1, sign-up page which gives the error message that mail already in use. So, the user needs to give new username or new password to create account.

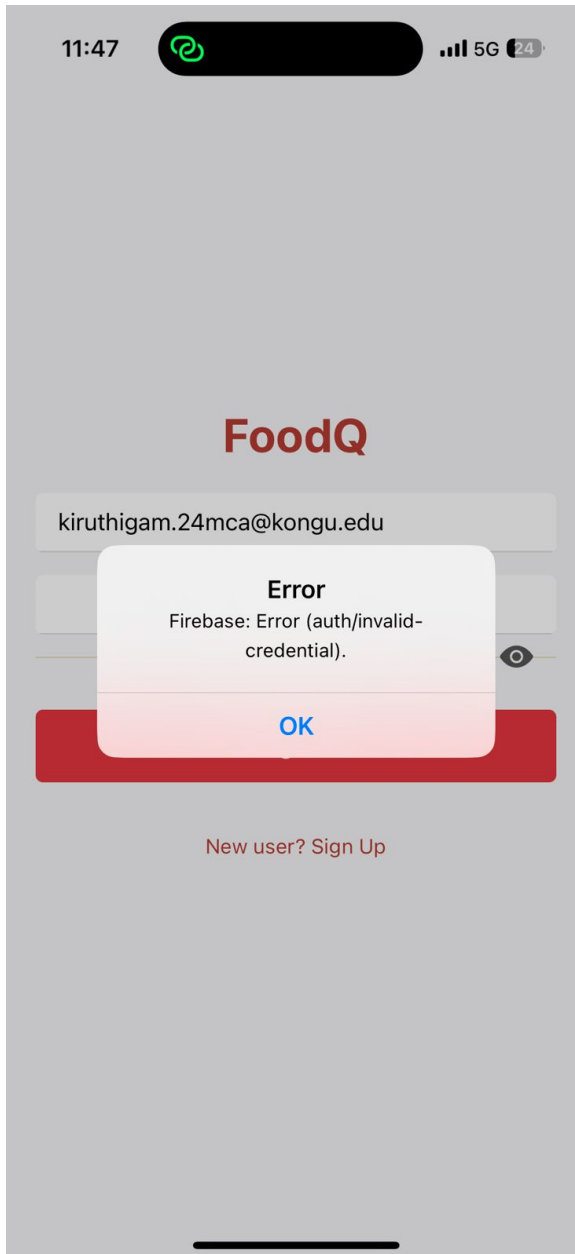


Figure 5.2 Login Page with Invalid username or password

In Figure 5.2, when the user tries to login with wrong username or password, it will show the error message as Invalid credential! Please try again.

5.5 RESULT

This section discusses the use, benefits, and technology behind the Online Food Pre-Ordering System, along with the outcomes achieved. The application streamlines the food ordering process for users, allowing them to pre-order meals, view nutritional information, and make payments conveniently. It is built using React Native for mobile app development and Firebase for database management, ensuring a smooth user experience and secure data handling.

The project provided valuable insights into designing mobile applications, creating responsive layouts, and managing databases with Firebase. Additionally, it enhanced our understanding of the software development life cycle, from development to testing and deployment. The primary goal of this system is to improve operational efficiency for food vendors while offering a seamless experience for customers. The user interface is designed to be intuitive and easy to navigate, ensuring a smooth and enjoyable experience for users, making it a great application to develop and implement in the real world.

SUMMARY

In this chapter, system testing and its results for the Online Food Pre-Ordering System are discussed. The testing involves evaluating the entire application to ensure all components function properly. End-to-end testing is conducted to verify that all user scenarios, from placing an order to making a payment, work as intended. The challenges faced during the development of the system are addressed, and the solutions implemented to overcome these challenges are highlighted. The conclusion and future work for the project are covered in Chapter 6.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1 CONCLUSION

The project "ONLINE FOOD PRE-ORDERING SYSTEM" has been meticulously designed and developed to meet the specified requirements, ensuring it addresses all necessary functionalities. The project provides a comprehensive understanding of its purpose and operation, offering users a simple and effective way to place their food orders in advance. The code behind this system is structured clearly, following best practices, which makes it not only efficient but also easy to maintain and extend in the future.

Thorough testing has been conducted with various sample data, ensuring the system delivers accurate results consistently. Based on a detailed analysis of the positive features and the constraints of the current component, it can be concluded that this product is a highly efficient and user-friendly, GUI-based application. The system operates smoothly and satisfies all the user requirements effectively. It offers great flexibility, as it can be easily integrated into other platforms or systems, allowing for further scalability and adaptability.

This application has been designed to significantly reduce the time and effort for users, providing a seamless and optimized experience. It ensures that users can place orders conveniently and monitor them in real-time. Looking ahead, there are plans to expand this application by developing a web version that will cater to more diverse user needs and allow integration with multiple branches, broadening its scope and usability.

The new system is a marked improvement over the existing manual processes, addressing key issues such as inefficiency and human error. It provides users with a quick and intuitive solution that enables them to place, modify, or cancel their orders at any stage of the process. This streamlined approach not only enhances the user experience but also increases operational efficiency, making the entire process more effective for both users and service providers. As we move forward, additional features and improvements will continue to be implemented to ensure the system remains adaptable and valuable for its users.

6.2 FUTURE ENHANCEMENT

The Online Food Pre-Ordering System for our college food court is designed with flexibility to include future enhancements as needed. As the system evolves, additional features and modules can be integrated to further enhance the dining experience and improve customer engagement. These updates will strengthen the relationship between the users and the application, adapting to future needs and technological advancements. The system aims to continually improve its functionality, ensuring it meets the growing demands of the users and aligns with industry trends. With each update, we strive to create a more intuitive, efficient, and enjoyable food ordering experience. These enhancements will empower customers with more control over their orders and allow for greater convenience in managing their dining preferences.

Some potential future enhancements for the system include:

- Adding features like product ratings to allow customers to review meals.
- Allowing customers to provide feedback on their ordered meals, ensuring continuous improvement in service.

APPENDICES

A. SAMPLE CODING

HomeScreen code

```
import React, { useRef, useEffect, useState } from "react";
import { View, Text, Image, StyleSheet, Dimensions, FlatList, TouchableOpacity } from
"react-native";

const { width } = Dimensions.get("window");
const data = [
  {
    image: require("../assets/images/pizza.png"),
    title: "Pizza",
    description: "Deliciously cheesy and loaded with toppings, our pizzas are perfect for any
occasion.",
  },
  {
    image: require("../assets/images/coffee.png"),
    title: "Coffee",
    description: "Kickstart your day with our freshly brewed, aromatic coffee blends.",
  },
  {
    image: require("../assets/images/biryani.png"),
    title: "Biryani",
    description: "Savor the rich flavors of our authentic, aromatic biryani dishes.",
  },
  {
    image: require("../assets/images/burger.png"),
    title: "Burger",
    description: "Juicy and satisfying burgers with a variety of options to choose from.",
  },
]
```

```

    {
      image: require("../assets/images/sandwich.png"),
      title: "Sandwich",
      description: "Freshly made sandwiches with the perfect balance of flavors for a quick bite.",
    }
  ];

  const HomeScreen = ({navigation}) => {
    const flatListRef = useRef(null);
    const [currentIndex, setCurrentIndex] = useState(0);

    const scrollToNext = () => {
      const nextIndex = (currentIndex + 1) % data.length;
      flatListRef.current.scrollToIndex({ index: nextIndex, animated: true });
      setCurrentIndex(nextIndex);
    };

    useEffect(() => {
      const interval = setInterval(scrollToNext, 3000); // Auto-scroll every 3 seconds
      return () => clearInterval(interval); // Clear interval on unmount
    }, [currentIndex]);

    const renderItem = ({ item }) => (
      <View style={styles.slide}>
        <Image source={item.image} style={styles.image} />
        <Text style={styles.title}>{item.title}</Text>
        <Text style={styles.description}>{item.description}</Text>
      </View>
    );

    return (
      <View style={styles.container}>
        <Text style={styles.screenTitle}>"Welcome to FoodQ"</Text>
        <FlatList
          ref={flatListRef}

```

```

    data={data}
    renderItem={renderItem}
    keyExtractor={(item, index) => index.toString()}
    horizontal
    pagingEnabled
    showsHorizontalScrollIndicator={false}
    onScrollToIndexFailed={() => {}}
  />

  <TouchableOpacity onPress={() => navigation.navigate("Menu")}>
    <Text style={styles.skip}>
      Go To Menu
    </Text>
  </TouchableOpacity>
</View>
);
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: "#ffffff",
    alignItems: "center",
  },
  screenTitle: {
    fontSize: 30,
    fontWeight: "900",
    marginTop: 60,
    color: "#D24545",
  },
  slide: {
    width,
    justifyContent: "center",
    alignItems: "center",
    padding: 20,

```

```
},  
image: {  
  width: "60%",  
  height: "200",  
  resizeMode: "contain",  
  margin: 10,  
  borderRadius: 20,  
  backgroundColor: "#f5f6f7",  
},  
title: {  
  fontSize: 22,  
  fontWeight: "900",  
  marginBottom: 10,  
  color: "#333",  
},  
description: {  
  fontSize: 16,  
  textAlign: "center",  
  color: "#666",  
},  
skip: {  
  position: "absolute",  
  bottom: 40,  
  alignSelf: "center",  
  fontSize: 18,  
  fontWeight: "bold",  
  color: "#ffffff",  
  backgroundColor: "#D24545",  
  borderRadius: 20,  
  paddingVertical: 10,  
  width: "90%",  
  textAlign: "center",  
},
```

```

});
export default HomeScreen;
App.js

import React, { useState, useEffect } from 'react';

import { NavigationContainer } from '@react-navigation/native';

import { createNativeStackNavigator } from '@react-navigation/native-stack';

import { onAuthStateChanged } from 'firebase/auth';

import { auth } from '../firebase/firebaseConfig'; // Update the path

import AuthScreen from '../screens/AuthScreen';

import MainTabs from '../navigation/MainTabs';

import MenuDetailScreen from '../screens/MenuDetailScreen';

import AdminScreen from '../screens/AdminScreen';

import BillScreen from '../screens/BillScreen';

import CartScreen from '../screens/CartScreen'; // Ensure CartScreen is imported

import { ActivityIndicator, View, StyleSheet } from 'react-native'; // Import
ActivityIndicator for loading

import PaymentScreen from '../screens/PaymentScreen';

const Stack = createNativeStackNavigator();

const App = () => {

  const [user, setUser ] = useState(null);

  const [loading, setLoading] = useState(true);

  useEffect(() => {

    const unsubscribe = onAuthStateChanged(auth, (currentUser ) => {

      setUser (currentUser );

      setLoading(false);

    });
  });

```

```

    return unsubscribe; // Cleanup listener on unmount
  }, []);

  return (
    <NavigationContainer>

      <Stack.Navigator>

        {user ? (
          <>

            <Stack.Screen name="HomeTabs" component={MainTabs} options={{
headerShown: false }} />

            <Stack.Screen name="MenuDetail" component={MenuDetailScreen}
options={{ headerTintColor: "#D24545", headerBackTitle: "Back" }} />

            <Stack.Screen name="AdminScreen" component={AdminScreen} options={{
headerTintColor: "#D24545", headerBackTitle: "Back" }} />

            <Stack.Screen name="Cart" component={CartScreen} options={{ title: 'Your
Cart' }} />

            <Stack.Screen name="BillScreen" component={BillScreen} options={{ title:
'Bill Summary', headerTintColor: "#D24545" }} />

            <Stack.Screen name="PaymentScreen" component={PaymentScreen}
options={{headerTintColor: "#D24545",}} />

          </>

        ) : (

          <Stack.Screen name="Auth" component={AuthScreen} options={{
headerShown: false }} />

        )}

      </Stack.Navigator>

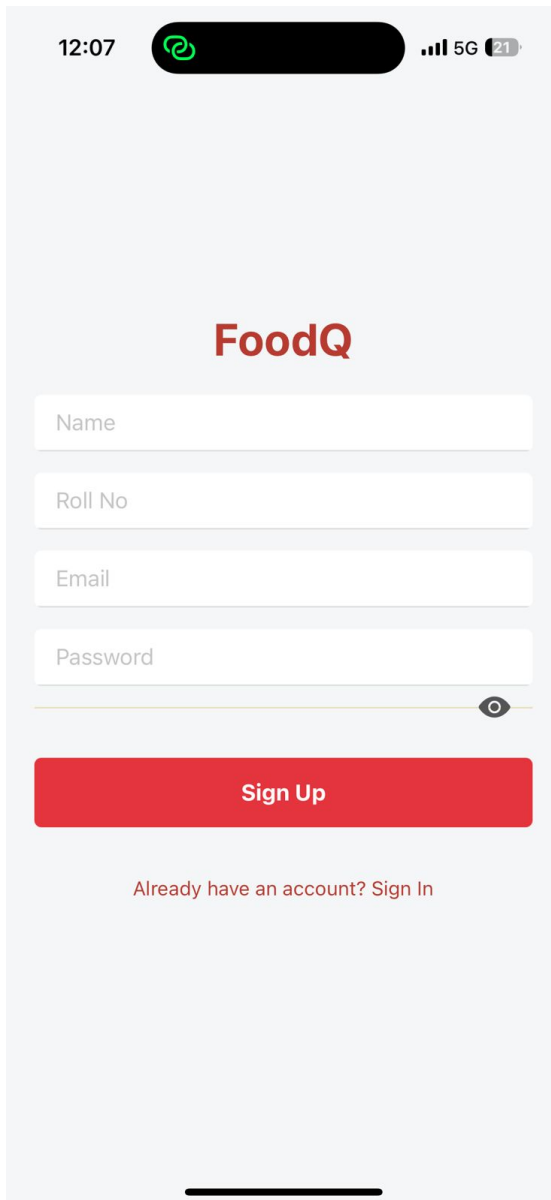
    </NavigationContainer>

  );
};

```

```
const styles = StyleSheet.create({  
  loadingContainer: {  
    flex: 1,  
    justifyContent: 'center',  
    alignItems: 'center',  
  },  
});  
  
export default App;
```

B. SCREENSHOTS



The screenshot shows a mobile application interface for 'FoodQ'. At the top, the status bar displays the time 12:07, a green circular icon, 5G signal strength, and a battery level of 21%. The app's logo 'FoodQ' is centered in a bold, dark red font. Below the logo are four white input fields with light gray borders, labeled 'Name', 'Roll No', 'Email', and 'Password'. A horizontal line with a small eye icon (for password visibility) is positioned below the 'Password' field. A prominent red button with the text 'Sign Up' in white is located below the input fields. At the bottom of the form, there is a link that reads 'Already have an account? Sign In' in a smaller, dark red font. The entire form is set against a light gray background.

Figure B.1 Sign-Up Page

In Figure B.1 When a new customer comes first time to the app, they have to do registration first. They have to fill all the details and click on sign up button.

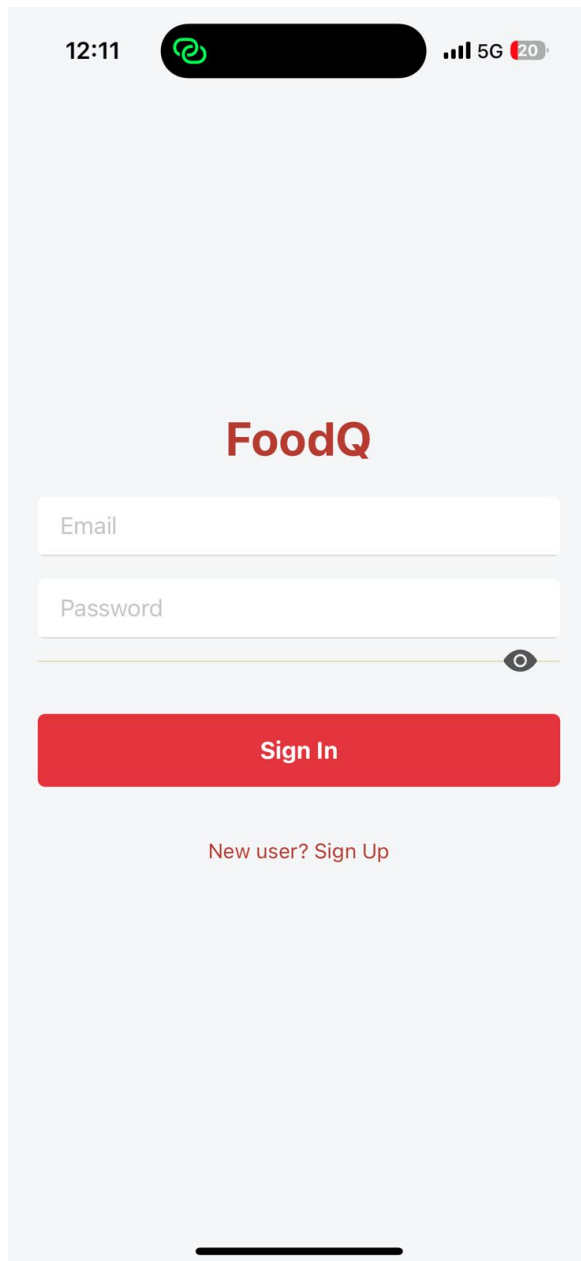


Figure B.2 Login Page

In Figure B.2 the customer can login into the app by entering their email and password they created during this sign-up process. After login customer can able to checkout and order products.

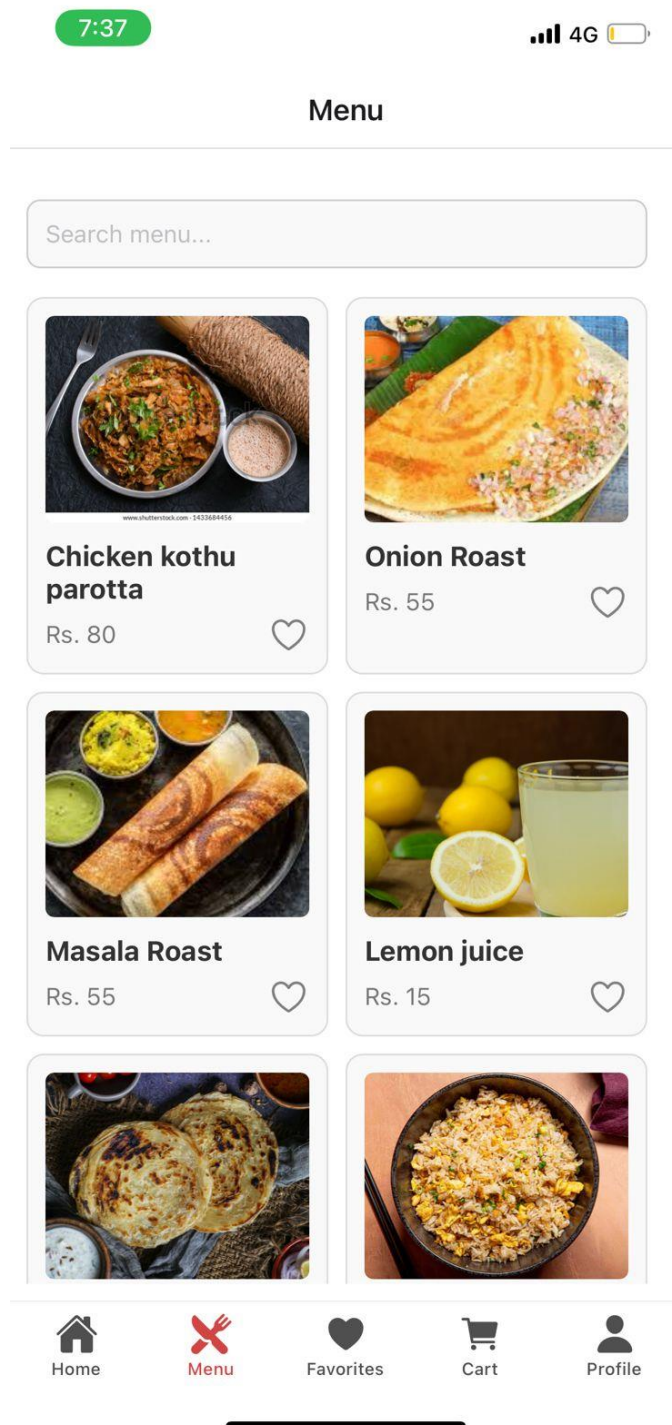


Figure B.3 Menu Page

In Figure B.3, after login, the user is directed to the Menu Page, showcasing menu items with images, descriptions, and a search bar for quick navigation. Users can explore and select items easily for their orders.

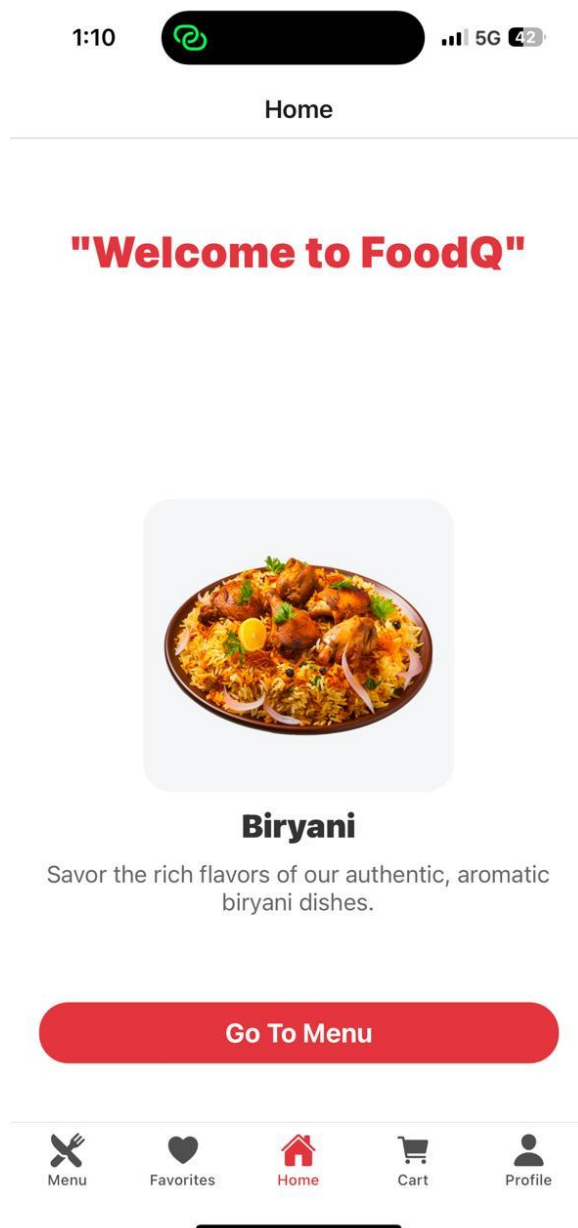


Figure B.4 Home Page

In Figure B.4, the Home Page showcases the top items prominently and includes a Go to Menu button, allowing users to navigate easily to the full menu for further selection.

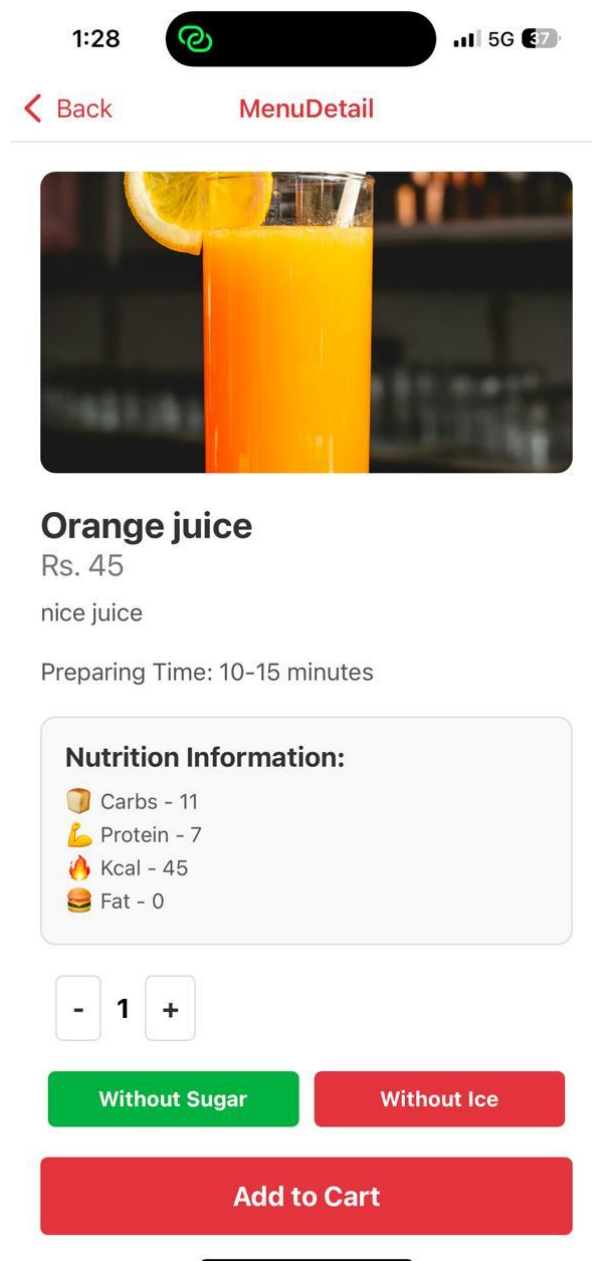


Figure B.5 Menu Details page

In Figure B.5, the Menu Details Page provides detailed information about a selected menu item, including customization options, price, nutritional value, and quantity selection, allowing users to tailor their order as needed.

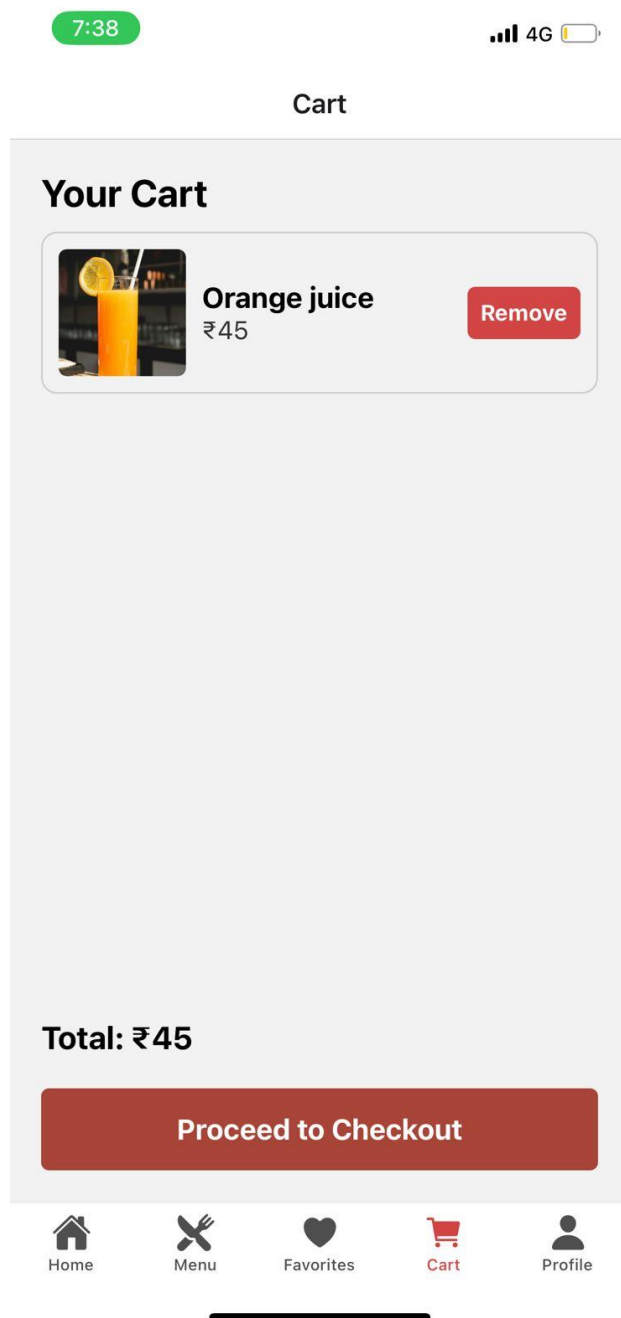


Figure B.6 Cart page

In Figure B.6, the Cart Page displays the selected items with their names, any customization options listed next to the item names, and the total amount for the order at the bottom.

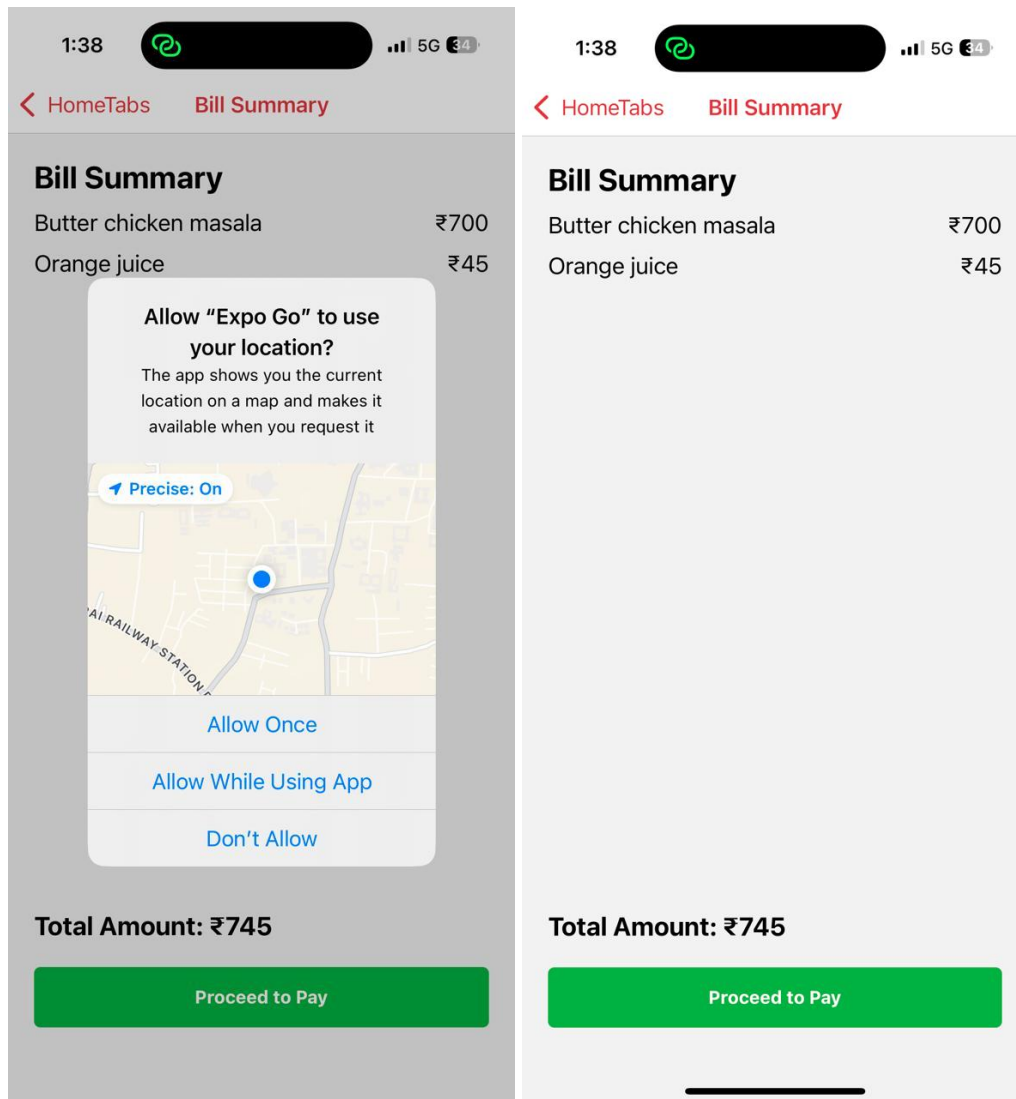


Figure B.7 Bill page

In the Bill Page, each item name is listed along with its corresponding amount, providing a clear breakdown of the total cost for the order. Additionally, the page prompts the user to allow location access to verify if they are within the college premises for delivery eligibility.

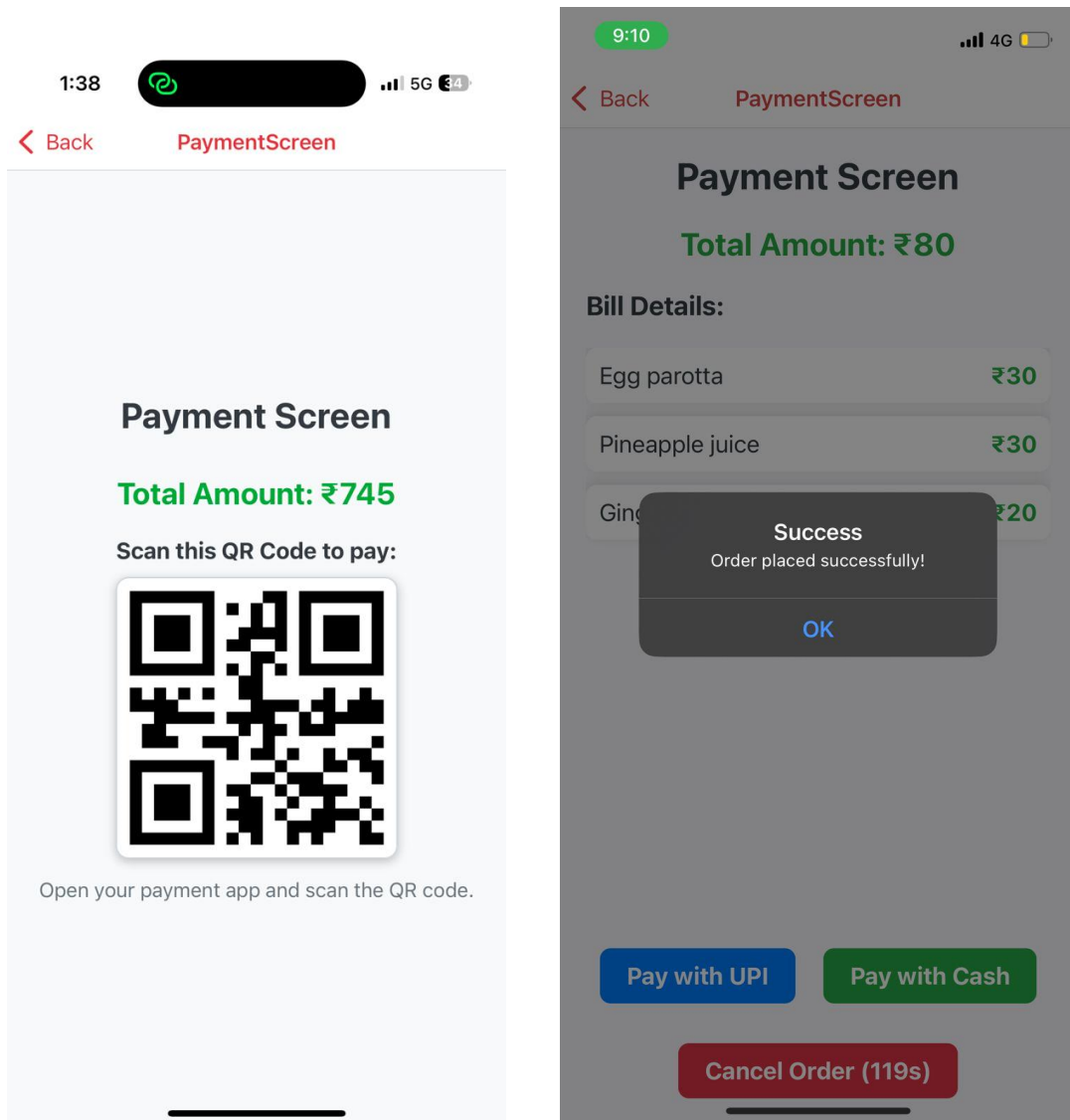


Figure B.8 Payment page

In Figure B.8, the Payment Page displays a QR code for users to complete their payment quickly by scanning it with a mobile payment app. Once the transaction is completed, the order is confirmed, but the user has the option to cancel the order within 2 minutes if needed, ensuring flexibility and convenience.

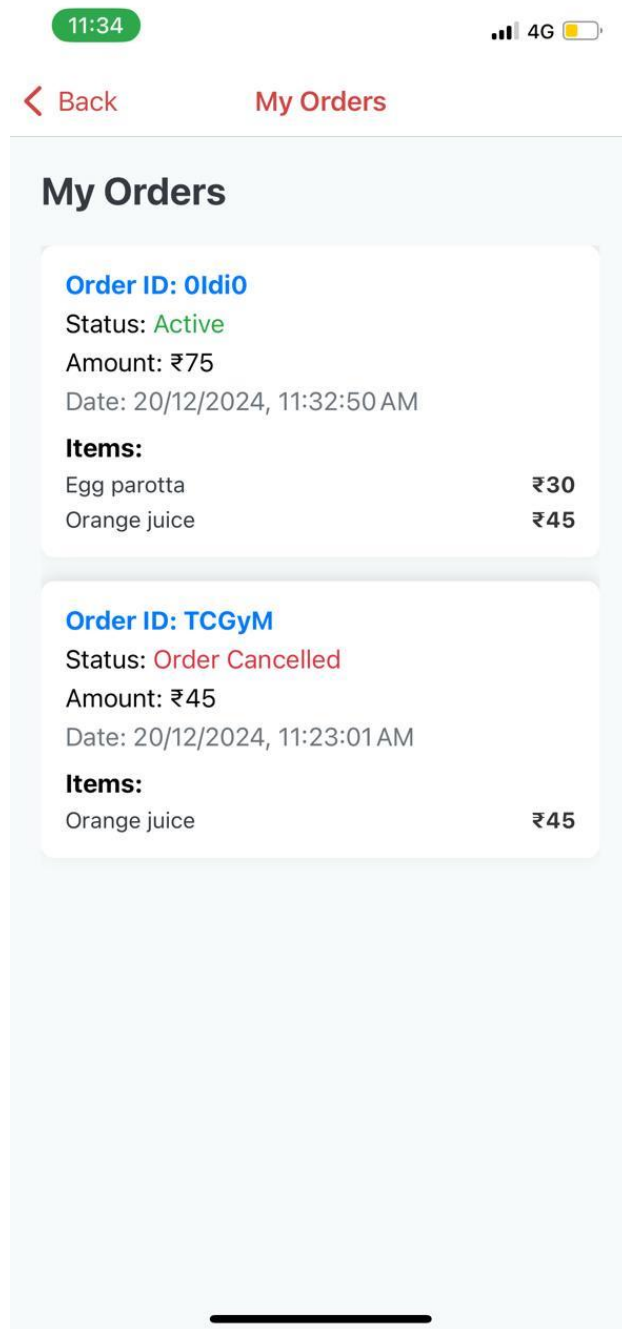


Figure B.9 Order page for user

In the Order Page, the user can view detailed information about their order, including the Order ID, Status, Amount, Date and Time of the order, and a list of the Items ordered. This page provides a comprehensive overview, allowing users to track their order's progress and review the details of their purchase.

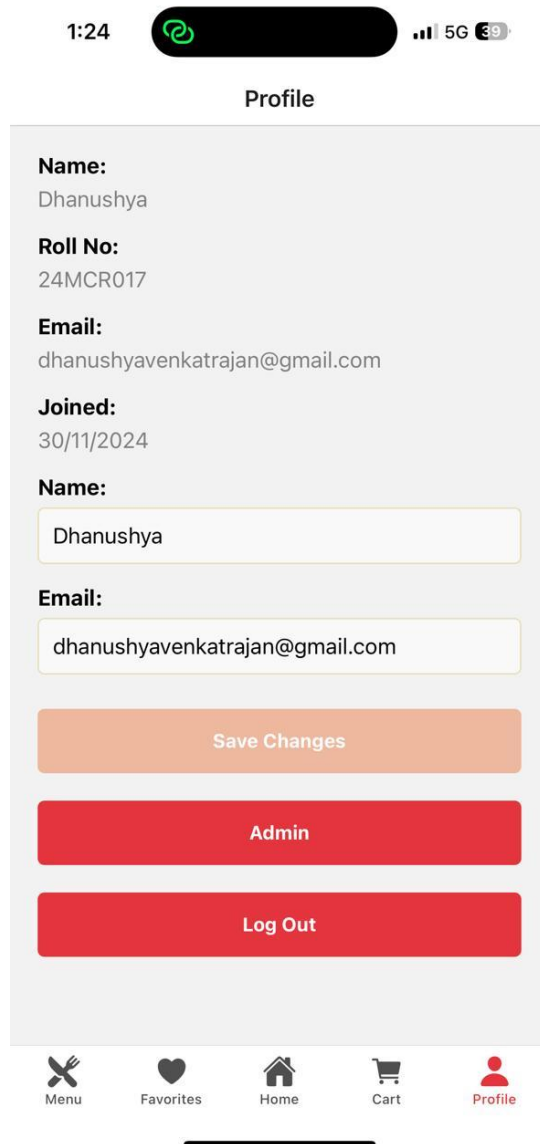


Figure B.10 Profile page

In the Profile Page, users can view and update their personal information, such as name, contact details, and account settings. The page also includes a Logout button for users to sign out. If the user is an admin, an Admin Button is displayed, allowing the admin to access the Admin Screen for managing the system.

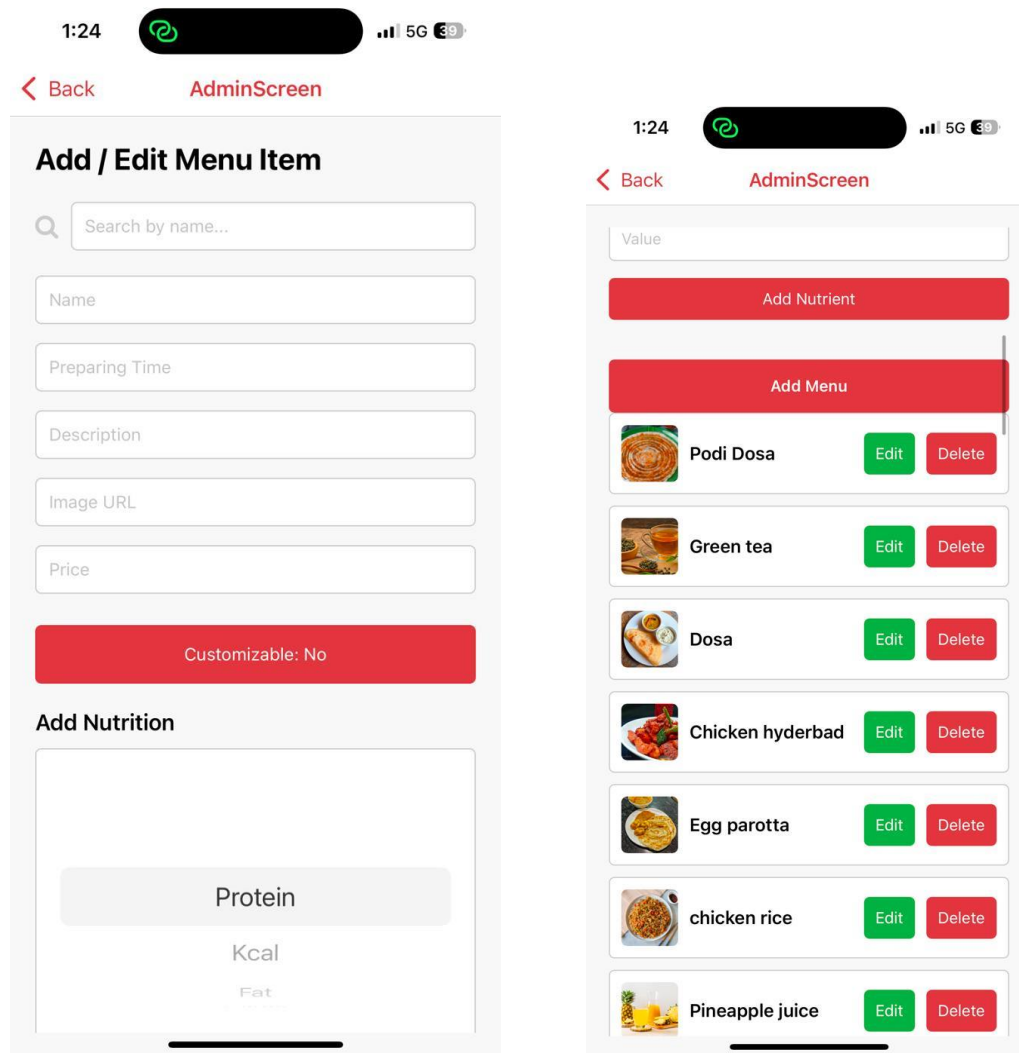


Figure B.11 Admin page

In the Admin Screen, admins have the ability to add, update, and delete menus, allowing them to manage the available items, prices, descriptions, and other details to ensure the menu is current and accurate.

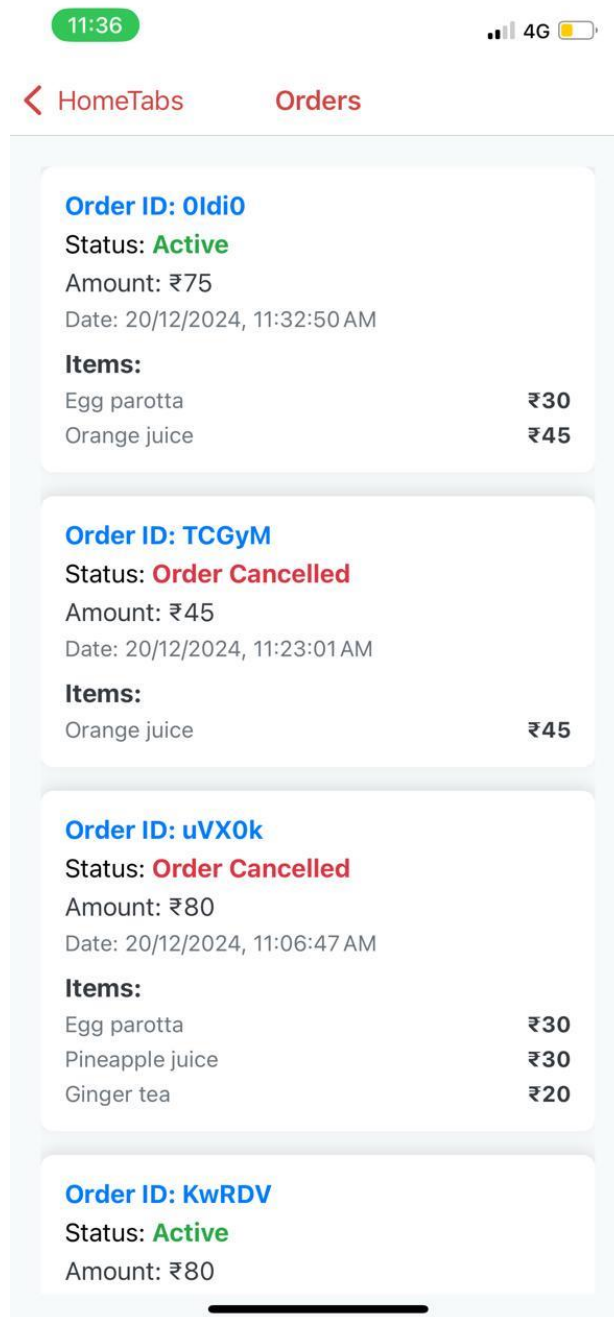


Figure B.12 Order page for admin.

In the Order Page for the Admin, they can view a detailed list of all customer orders, including the Order ID, Customer Information, Status of the order (e.g., pending, completed, cancelled), the Amount, and the Date and Time of the order. Additionally, the Admin can see the Items ordered, manage the status of each order, and take necessary actions to ensure timely processing and delivery. This page enables the Admin to efficiently monitor and manage all incoming orders.

REFERENCES

- [1] Eisenman, B. (2015). *Learning React Native*. O'Reilly Media.
- [2] Dabit, N. (2017). *React Native in Action*. Manning Publications.
- [3] Crockford, D. (2008). *JavaScript: The Good Parts*. O'Reilly Media.
- [4] Haverbeke, M. (2018). *Eloquent JavaScript: A Modern Introduction to Programming* (3rd ed.). No Starch Press.
- [5] Paul, A., & Nalwaya, A. (2018). *React Native for Mobile Development*. Packt Publishing.
- [6] Simpson, K. (2014). *You Don't Know JS: Scope & Closures*. O'Reilly Media.
- [7] https://www.researchgate.net/publication/380401936_ONLINE_FOOD_ORDERING_SYSTEM_PROJECT_REPORT
- [8] <https://zomato.com>
- [9] <https://www.freecodecamp.org/news/full-stack-react-firebase-tutorial/>
- [10] <https://reactnative.dev/docs/getting-started>
- [11] <https://www.nngroup.com/articles/>
- [12] <https://firebase.google.com/docs>