SMARTCANTEEN: TRANSFORMING CANTEEN ORDERING WITH A MOBILE APPLICATION

BY

KAVIRAJ MANIVEEL

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONOURS)

Faculty of Information and Communication Technology

(Kampar Campus)

JUNE 2024

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: SMARTCANTEEN: TRANSFORMING CANTEEN ORDERING WITH A MOBILE APPLICATION

Academic Session: June/2024

I KAVIRAJ MANIVEEL
(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

(Author's signature)

(Supervisor's signature)

Address:
No. 6, Jalan Abadi 1/14,
Abadi Heights, 47120,
Puchong, Selangor, Malaysia

Phan koo Yuen
Supervisor's name

Date: 13/9/2024

Date: 13/9/2024

Universiti Tunku Abdul Rahman			
Form Title : **Sample of Submission Sheet for FYP/Dissertation/Thesis**			
Form Number: **FM-IAD-004**	Rev No.: **0**	Effective Date: **21 JUNE 2011**	Page No.: **1 of 1**

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

UNIVERSITI TUNKU ABDUL RAHMAN

Date: 13/9/2024

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that Kaviraj Maniveel (ID No: 21ACB06054) has completed this final year project entitled : " **SMARTCANTEEN: TRANSFORMING CANTEEN ORDERING WITH A MOBILE APPLICATION "** under the supervision of Ts. Dr Phan Koo Yuen (Supervisor) from the Faculty of Information and Communication Technology_____.

I understand that University will upload softcopy of my final year project / dissertation/ thesis* in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,

____

(*Kaviraj Maniveel*)

DECLARATION OF ORIGINALITY

I declare that this report entitled "**METHODOLOGY, CONCEPT AND DESIGN OF A 2-MICRON CMOS DIGITAL BASED TEACHING CHIP USING FULL-CUSTOM DESIGN STYLE**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature :

Name : Kaviraj Maniveel

Date : 13/9/2024

ACKNOWLEDGEMENTS

I'd like to convey my heartfelt appreciation to various individuals and organisations whose invaluable contributions and unwavering support were critical to the successful completion of this project. First and foremost, I'd want to express my heartfelt gratitude to my excellent supervisor, Ts. Dr. Phan Koo Yuen, for his exceptional advice, useful insights, and consistent encouragement throughout this research. His extensive knowledge, critical input, and unshakable commitment to academic achievement have all played important roles in influencing the direction and quality of this work. I am also extremely grateful to the instructors and staff of the University Tunku Abdul Rahman (UTAR) for creating a positive learning atmosphere and giving access to required resources and facilities.

Their dedication to academic success has had a tremendous impact on my personal and professional development. In addition, I would like to thank my students and friends for their encouragement and support. Their willingness to engage in thought-provoking discussions and provide constructive feedback has enhanced my understanding and assisted me in overcoming numerous hurdles experienced during this assignment. Finally, I am grateful to my family for their unwavering love, support, and belief in my skills, which has driven my perseverance and ambition. Their compassion, understanding, and steadfast encouragement have proven crucial throughout this difficult path.

ABSTRACT

The traditional paper-based ordering system used in most Malaysian canteens has been unchanged for almost five decades, with extremely manual and time-consuming operations that cause bottlenecks and reduce productivity. Long lineups during peak hours, a lack of menu visibility, numerous order problems, and limited payment alternatives have all contributed to an unfavourable shopping experience for students, staff, and management. This project seeks to address these long-standing inefficiencies by creating a new mobile application, “SmartCanteen,” that uses cutting-edge technologies and user-centric design concepts to revolutionise the canteen ordering process. The proposed solution takes a complete approach, including features like remote pre-ordering, digitised menus with real-time availability updates, automated order processing to reduce errors, and interaction with a variety of electronic payment options. The “SmartCanteen” app aims to improve efficiency, sustainability, accessibility, and overall customer pleasure by streamlining operations, increasing transparency, and optimising staffing using data-driven insights. The project's methodology includes a thorough literature research to obtain knowledge about existing systems, technologies, and best practices. Following that, a user-centric design strategy is used, which includes active user interaction and iterative feedback loops to ensure alignment with organisational goals and preferences. The development phase focuses on iteratively refining prototypes based on user feedback, using technologies like Android Studio and Firebase to provide smooth integration and data management. The “SmartCanteen” project offers a big step towards modernising Malaysia's canteen ordering experience. By tackling the highlighted difficulties with an innovative and complete solution, it provides a creative way to streamlining operations, increasing customer happiness, and boosting operational efficiency in the field of campus dining experiences.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF LISTINGS

LIST OF ABBREVIATIONS

API	Application Programming Interface
COD	Cash on Delivery
IDE	Integrated Development Environment
IoT	Internet of Things
LCD	Liquid Crystal Display
MAC	Media Access Control
RAD	Rapid Application Development
RFID	Radio Frequency Identification
SDK	Software Development Kit
SOAP	Simple Object Access Protocol
SSL	Secure Sockets Layer
TCP/IP	Transmission Control Protocol/Internet Protocol
UI	User Interface
UML	Unified Modeling Language
UPI	Unified Payments Interface
WSDL	Web Service Description Language
XML	Extensible Markup Language
ZigBee	A wireless communication protocol

Chapter 1: Introduction

1.1 Background information

Traditional canteen ordering system.

The antiquated paper-based ordering system still followed in most conventional canteens in Malaysia has remained fundamentally unchanged for over 50 years and is characterized by highly manual, time-consuming processes that create bottlenecks [1]. During peak lunch hours, painfully long queues routinely form as hundreds of students and employees' line up one-by-one at the 1-2 counters available to verbally relay their food selections [2]. This strains operations, causing massive crowds and delays as customers wait up to 30 minutes just to place orders, impacting productivity. Additionally, the lack of menus or item availability details prior forces patrons to visit canteens unsure of actual offerings. At the solitary payments counter, cash-only policies hinder sales as students without cash cannot buy food. Reliance on paper order slips also causes errors like staff misreading 'teh tarik' as 'teh halia', angering customers who get wrong drinks [3]. Overall, myriad inefficiencies in the antiquated system creates a highly unsatisfactory purchasing experience while hindering sales, sustainability, and productivity. An intelligent digitized system can help transform this [4].

Mobile app for canteen ordering

With rising smartphone penetration across diverse demographic groups including students, working professionals, elderly citizens and more, intuitive mobile apps provide a viable solution to modernize conventional canteens and better serve the public [5]. A thoughtfully designed, user-centric mobile app can allow all customer segments including busy employees, budget-conscious students, visiting parents and customers to avoid tedious physical queues by seamlessly pre-ordering customized meals that they can collect as per individual convenience and needs. Digitizing menus, prices, hourly offerings, and real-time availability of items directly on the app provides patrons comprehensive visibility to make well-informed ordering decisions tailored to budget, taste preferences, dietary needs, and convenience prior to visiting the canteen [6]. Customers in a rush can save recurring favourite orders for one-click rapid reordering while features like scheduling future pickups, combo discounts, bill splitting make the process more personalized, affordable, and collaborative for groups. Integrating diverse electronic payment methods including credit/debit cards and digital wallets eliminates dependence on cash transactions alone, expanding the customer base [7]. Elderly visitors unfamiliar with mobile apps can order and pay in-person. Powerful data analytics on order histories and buying patterns can provide customer insights across age groups, occupations, timings to help optimize inventory management, production planning, dynamic staff allocation to improve cost and operational efficiency. Thoughtful accessibility options like text/icon sizes, colour contrasts facilitate usage across the wide spectrum of abilities and technological familiarity. Key features are remote pre-ordering, digitized menus, customized orders, diverse payments, and data analytics. This strategic solution will streamline operations, boost transparency, drive sales, and optimize staffing. The goal is an automated, user-centric platform that improves productivity, sustainability, accessibility, and satisfaction by better serving diverse customer's needs. In summary, a strategic mobile solution can help overhaul canteens to drive automated, insightful, and personalized ordering that serves the diverse needs of the heterogeneous customers far more effectively.

Key terms definitions:

1. User Interface (UI): The user interface is the visual and interactive part of a software that users interact with, including buttons, menus, and icons.

2. Real-time Updates: Real-time updates provide immediate and continuously updated information within a system as changes occur, such as order status or menu changes in a mobile app.

3. Accessibility Features: Accessibility features are functionalities designed to ensure that individuals with disabilities can effectively use a software, including options to adjust text size and support for screen readers.

4. User-centric: User-centric means focusing on what users need and prefer, ensuring products or services are designed to meet their requirements and make things easier for them

1.2 Problem Statement for the Existing Traditional Canteen System.

1. Inefficient ordering process:

Manual ordering at canteen counters is inefficient, time-consuming, and inconvenient for customers wishing to purchase meals during short break times [8]. The requirement to queue in extremely long, crowded single-file lines only to place food orders has resulted in increased wait times, delays, loss of productivity, and widespread frustration. With only 30–

60-minute windows to access the canteen counters before they close for lunch or tea breaks, many hungry customers have been forced to depart without meals due to the painfully slow pace of manual ordering [9]. This has resulted in people returning to work, school, or other activities without eating, reducing energy levels, appetite, and productivity. Implementing faster, parallelized ordering via online platforms or mobile apps could significantly improve the experience by allowing customers to place advance orders before going to the counter. This enables for simultaneous processing of many orders, avoiding clogged queues. Customers could quickly collect packaged food, saving valuable time.

2. Inadequate menu and inventory visibility:

Without access to current menu options, meal deals, and real-time food availability, customers frequently arrive at the canteen unaware of what is available. This lack of transparency into available dishes, offerings, and ready inventory has resulted in enormous disappointment and frustration as desired menu items are regularly and inevitably sold out or unavailable after consumers have already spent time travelling to the canteen [10]. Customers have been unable to reasonably analyse or plan potential orders ahead of time, or to ensure that their preferred foods are available during visits. Access to clearly visible, constantly updated menu listings and real-time inventory status via digital platforms will substantially empower customers with the necessary information to make smarter, more informed food shopping selections before visits.

3. Manual order errors:

The use of handwritten receipts at counters has resulted in frequent order fulfilment problems due to poor handwriting and misheard spoken instructions in noisy canteens [11]. Because of this reliance on error-prone manual processes, incorrect dish specifics have been recorded, and the public has received completely incorrect meal items when compared to what they had initially purchased. Customers have expressed extreme discontent, and food has been squandered because of communication mistakes. Automating orders through user-friendly digital interfaces would minimise the ambiguity associated with deciphering messy handwritten orders. Customers who enter selections straight into app or website menus will reduce errors and significantly increase order accuracy [12].

4. Limited Payment options and Inconvenient Transaction Processes:

The traditional canteen system's limited payment alternatives and time-consuming transaction processes cause customer annoyance and inefficiency. With cash being the only form of payment in many canteens, customers frequently have trouble making accurate change or dealing with coinage, resulting in delays and lengthier wait times during peak hours [13]. Furthermore, the lack of digital payment alternatives, such as credit/debit card payments or digital wallets, limits the options for clients who prefer cashless transactions or do not always carry cash. This constraint becomes increasingly difficult in today's digital environment, as contactless payments become more popular. Furthermore, the lack of interface with institutional or corporate payment systems causes delays in the checkout procedure for employees or students who use centralised accounts to make food purchases. Implementing a variety of payment options, such as contactless and connectivity with existing payment systems, would speed up transactions, minimise wait times, and boost overall customer satisfaction [14].

1.3 Motivation

The fundamental motivation driving this project is to develop an innovative mobile application that wholly transforms the antiquated canteen ordering experience, which currently hampers productivity and satisfaction for students, staff, and management alike. This aims to deliver an intuitive, seamless platform that makes purchasing food quick, convenient, and insightful.

Primary motivations include:

- Alleviating the massive productivity loss that stems from long canteen queues and delayed service by enabling customers to place advance food orders before reaching the counter. A streamlined process for pre-ordering meals will allow students and employees to skip lines entirely and simply collect their packaged food during short lunch breaks, saving precious time.
- Providing real-time visibility of daily menus with dish options and inventory availability of ingredients so customers can make informed purchasing choices based on updated offerings and stocks. This eliminates uncertainty around what foods can be ordered each day.
- Minimizing the constant errors in manual order taking that lead to wrong food items being prepared and resultant waste. Automating the ordering process via a digital interface can significantly improve accuracy.

- Offering diverse digital payment options beyond just cash for faster checkouts. Cashless modes like mobile wallets and card services optimize billing flows, are more convenient for users, and open sales opportunities.
- Leveraging order data analytics to forecast demand, gain customer insights, and optimize procurement, inventory, staffing to prevent shortages of food. Data driven operations alignment can enhance sustainability.

In summary, this project aims to completely transform outdated canteen purchasing with an innovative mobile solution that tackles nagging issues faced today. Intelligent automation will be the vehicle to upgrade speed, convenience, and service quality.

1.4 Project Scope

The scope of this project is the development of a mobile application called "SmartCanteen" to improve the food ordering experience for students and staff in a campus canteen. The final product will be a fully functional Android application built using the Java programming language and the Android Studio framework. The programme will allow users to explore digital menus, place customised meal orders, track orders in real time, and make secure payments using integrated payment options, with a primary focus on credit/debit card transactions via the Curlec by RazorPay payment gateway. The Firebase API will be used for backend services, including real-time order processing, communication between users and canteen providers, and order tracking. A vendor interface will be built, enabling canteen personnel to manage orders, amend menu items, and view order history. The application's goal is to speed the ordering process, shorten wait times, and increase overall user happiness in the college canteen environment. The project will offer numerous digital wallet choices in addition to credit/debit card payments, giving customers access to a wide selection of safe payment methods. This increased payment feature is intended to accommodate to a variety of user preferences and improve the overall convenience of the ordering process. The project will also cover the implementation of crucial features for vendors, such as user authentication, menu customisation, and basic data analytics, to ensure a comprehensive and user-centric solution for campus canteen management.

1.5 Project Objective

The objectives of this project are:

1. To develop an intuitive, user-friendly mobile application for canteen ordering that provides a seamless experience for all customers including youths, adults, and the elderly.

- Simple and clean interface with large buttons/icons for easy navigation with large tap targets for fingers when making selections and high contrast colours and minimal design elements to avoid visual clutter.
- Step-by-step process with minimal data inputs for quick ordering which allows progressive disclosure of order flow one step at a time and have default selections and prefilled data to minimize user inputs.
- Customizable settings like text size and colour schemes for visually impaired users and options to increase font size for low vision accessibility. Colour contrast and scheme settings to aid readability. Confirmation messages at each step to prevent incorrect submissions. It also has review summaries showing order details for user confirmation including pop-up prompts on unusual quantities to validate intention.

2. To facilitate seamless lunch order placements that allow efficient food preparation timelines.

- Customers can quickly browse menus and place orders with a few taps. The proposed app will have an intuitive category-based menu navigation and one-click ordering. It saves previous orders for repeat customers to re-order fast.
- Real-time order information for kitchen staff to prepare ingredients and food as orders come in. It will have dashboard views of pending orders with prep time indicators.
- Performance metrics to continuously improve order-to-table time over iterations. It has track key metrics like wait times, prep times and will analyse data to identify bottlenecks and improvement areas.

3. To integrate various digital payment options that allow for efficient, seamless checkout.

- Incorporate payment processing via credit/debit cards, mobile wallets, and campus dining accounts to enable seamless transactions for all users.
- Allow customers to securely store their payment credentials within the app for faster checkout on future orders.

- Provide digital receipts and user order histories to simplify tracking and personal records.

4. To determine user requirements by using online survey questionnaires.

- Design and distribute extensive questionnaires to identify diverse user needs and preferences for the proposed canteen ordering system.
- Collect quantitative and qualitative data on user demographics, ordering behaviours, and existing solution pain points to help with system design.
- Use open-ended questions to collect insights and novel ideas directly from potential users, thereby promoting a user-centric development strategy.

1.6 Impact, significance, and contribution

The project's aim is to greatly improve the campus dining experience for students and staffs by implementing a user-friendly mobile application. This solution resolves common problems like long wait times and inadequate order processing by optimizing the meal ordering process and increasing overall convenience and accessibility. By delivering real-time order information and personalized recommendations, the application helps users have a more efficient and pleasurable eating experience. To integrate various digital payment options that allow for efficient, seamless checkout.

The project's emphasis on inclusiveness guarantees that all members of the campus community may take use of its features, regardless of technology skill or accessibility requirements. The application responds to varied user preferences by providing customizable font size and colour schemes, resulting in a more inclusive and pleasant eating atmosphere. In a nutshell, the antiquated paper-based ordering system currently utilized in most conventional canteens across Malaysia is inefficient and hampers productivity and satisfaction for all stakeholders. The project aims to overhaul outdated paper ordering in Malaysian collage canteens which causes productivity losses from long queues, lack of menu visibility, order errors and food wastage. Overall, the project's unique approach to improving campus eating demonstrates a dedication to increasing the quality of life and general satisfaction of students and teachers, making it a vital and significant addition to campus life.

1.7 Report Organization

The report is organised into seven chapters. The first chapter is the introduction of the proposed project, which includes project background, problem statement, motivation, project scope, project objectives, the impact, significance, and the mobile application's unique characteristics. Chapter 2 is a thorough literature review that covers various food ordering application approaches and current systems. This research that has been done in this chapter is significant because the conclusion of this chapter decides the outcome of the project. Chapter 3 discusses the system approach, focusing on the Rapid Application Development (RAD) technique and demonstrating the system's architecture and operation via UML diagrams. In chapter 4, it has stated the user requirements, verification plan (testing), issues and challenges during implementation and the overall timeline of the whole project. In chapter 5 describes the implementation process, which includes the installation of Android Studio and Firebase with configuration details, and screenshots of each module have been documented. In chapter 6, each module will be tested and documented in the test case. After that, a survey has been conducted based on the functions obtained from the existing applications. The result of the survey has been analysed and discussed. Finally, Chapter 7 concludes the report by summarising the project's accomplishments, discussing its limits, and making recommendations for future “SmartCanteen” application enhancements.

1.8 Conclusion

In a nutshell, the antiquated paper-based ordering system currently utilized in most conventional canteens across Malaysia is inefficient and hampers productivity and satisfaction for all stakeholders. The project aims to overhaul outdated paper ordering in Malaysian canteens and restaurants which causes productivity losses from long queues, lack of menu visibility, order errors and food wastage. An innovative mobile app is proposed to transform the experience via automation. Key features are remote pre-ordering, digitized menus, customized orders, diverse payments, and data analytics. This strategic solution will streamline operations, boost transparency, drive sales, and optimize staffing. The goal is an automated, user-centric platform that improves productivity, sustainability, accessibility, and satisfaction by better serving diverse public needs. Further analysis of related technologies and existing systems will inform solution design. In the next chapter, we will delve deeper into this by analysing relevant research papers, articles, and reviews of existing or prior systems. This will allow us to thoroughly examine and assess various approaches to inform the optimal solution design.

Chapter 2: Literature Review

2.1 Introduction

The literature review aims to provide useful insights into the challenges, possibilities, and best practices for developing and implementing mobile solutions to improve the dining experience. It also establishes a theoretical foundation and contextual understanding that guides the “SmartCanteen” application's development and deployment, ensuring that its objectives are met efficiently.

2.2.1 Comparison of different existing food ordering techniques

A "Wireless Food Ordering System Based on Web Services" notion exemplifies a dramatic leap into the digital age, revolutionising how food services are accessed and experienced [15]. This unique system combines wireless connectivity and web services to provide a seamless and user-centric platform for ordering meals. Users can place orders remotely via mobile connectivity and an intuitive web interface, avoiding physical lineups and increasing convenience. The web services infrastructure ensures real-time interactions and safe data sharing between users, canteen staff, and the database. This not only simplifies the ordering procedure, but it also provides users with order tracking, personalised food customization, and secure online payments. This project was inspired by the "Wireless Food Ordering System Based on Web Services" described in [15], seeks to modernize food ordering experiences in a similar manner. The completed project, like the system described in the paper, uses wireless connectivity to establish a user-centric platform for meal ordering. This revolutionary method allows customers to place orders remotely using their mobile devices, eliminating the need for real lines, and increasing overall convenience. By integrating mobile application services infrastructure, the project ensures real-time interactions and secure data sharing among users, canteen staff, and the central cloud database. Furthermore, this project simplifies the ordering procedure while also providing extra features such as order tracking, personalized customization possibilities, and safe online payments, which replicate the capabilities described in the wireless system. By adopting this wireless connectivity strategy, we hope to transform the traditional food ordering experience, giving users with a seamless and quick way to access and enjoy their favourite meals.

Machine Learning Based Platform for Food Ordering Services has transcended traditional food ordering paradigms by tailoring choices to individual preferences [16]. The technology uses data-driven insights to curate menus and recommend dining selections, hence improving customer satisfaction and engagement. The recommendation engine gives a dynamic menu exploration that exactly fits every user's taste and dietary requirements by leveraging the enormous data library of user preferences and past decisions. The "Machine Learning Based Platform and Recommendation System for Food Ordering Services within Premises" represents of modern food services, delivering a genuinely personalised and forward-thinking culinary direction by fusing technical innovation and gourmet delight.

An "AI-based Food Ordering Application" represents a fundamental shift in the way people navigate and enjoy culinary experiences [17]. This application, based on the power of artificial intelligence, transforms standard meal ordering operations into a dynamic and personalised journey. Advanced algorithms are used by the application to understand user preferences, past orders, and real-time data, resulting in personalised recommendations and menu selections. This application aids in streamlining the ordering process. Using AI, users are presented with a holistic culinary experience that is tailored to their tastes and dietary requirements. The "AI-based Food Ordering Application" defies convention, ushering dining experiences into the world of intelligent gastronomy.

Although the above techniques are considered the most effective and safe method, however each of them has their own advantages and limitation as shown in the table 2.2.1.1.

Methods	**Advantages**	**Limitations**
Wireless Food Ordering System Based on Web Services Journal: X. Hongzhen, T. Bin and S. Wenlin, "Wireless Food Ordering System Based on Web Services," 2009 Second International Conference on Intelligent Computation Technology and Automation, Changsha, China, 2009, pp. 475- 478,	• Convenience and Mobility: Enables users to order remotely from wireless devices. • Real-time Interaction: Facilitates instant order processing and updates. • Data Centralization: Centralizes order	• Technological Barriers: Users unfamiliar with technology may face difficulties. • Technical Dependencies: System functionality relies on stable wireless connectivity. • Maintenance Requirements: Ongoing updates and

	data for streamlined management. • Reduced Queues: Minimizes physical presence in queues, enhancing efficiency. • Integration Possibilities: Can integrate with payment gateways and other systems.	maintenance are essential for seamless operation
Machine Learning Journal: G. M. Aditya et al., "Machine Learning Based Platform and Recommendation System for Food Ordering Services within Premises," IEEE Xplore, Oct. 01, 2021.	• Personalized Menus: Customizes menu options based on user behaviours and preferences. • Real-time Adaptation: Adjusts recommendations as user preferences evolve. • Enhanced User Engagement: Increases user satisfaction and interaction. Streamlined Decision Making: Facilitates quicker and informed menu choices.	• Data Dependency: Effective recommendations require substantial historical user data. • Resource Intensive: Complex machine learning models can demand significant computational resources. • Algorithm Complexity: Developing and finetuning accurate recommendation algorithms can be challenging. Initial Implementation Effort: Setting up the system initially requires expertise.

AI-based Food Ordering App Journal: Tejas Raibagi, Ashwin Vishwakarma, J. Naik, R. Chaudhari, and Guntis Kalme, “Orderista - AIbased Food Ordering Application,” International Conference on Artificial Intelligence	- Personalized Recommendations: Tailors menu suggestions based on user preferences and history. - Enhanced User Experience: Provides a dynamic and engaging ordering process. - Predictive Analysis: Anticipates user preferences, improving order accuracy.	- Complex Implementation: Development and integration of AI components can be intricate. - Data Dependency: Requires extensive data to provide accurate recommendations. - Initial Setup: Initial setup and training of AI models can be time consuming. - Technical Complexity: Users with limited technical knowledge might find it challenging

Table 2.2.1.1: Advantages and disadvantage of food ordering

2.2.2 Summary on types of food ordering application techniques

The landscape of ordering techniques in modern food services has evolved significantly because of technological innovations and changing user preferences. Three working techniques stand out in the context of a Canteen Ordering Application: AI-based Food Ordering, Machine Learning Platform with Recommendation System, and Wireless Food Ordering System based on Web Services. Each technique brings unique benefits and considerations to the table, catering to various user needs and improving overall efficiency

2.3 Review on Existing Food ordering applications

2.3.1 Wireless Food Ordering System Based on Web Services

In this reviewed study [15], Xu et al. proposed a wireless meal ordering system that smoothly incorporates web services and wireless communication technology. The system's major goal is to provide efficient and convenient food ordering operations for restaurants by providing both wired and wireless access to servers via desktop computers and mobile devices such as PDAs across an integrated wired/wireless local area network. The system architecture is a four-tiered, web-based client-server approach with six major components: a web server, database server, cash register, mobile context server, mobile users, and desktop users. The web server serves as the central hub, delivering essential information and exposing services to mobile devices and desktop computers using the Web Service Description Language (WSDL) and Simple Object Access Protocol (SOAP) protocols. The database server oversees storing and managing data on food items, orders, and customer information. The cash register component is responsible for cost computation and billing functions. The mobile context server also plays an important role in customising and optimising content for different mobile devices based on their capabilities, such as screen size and resources

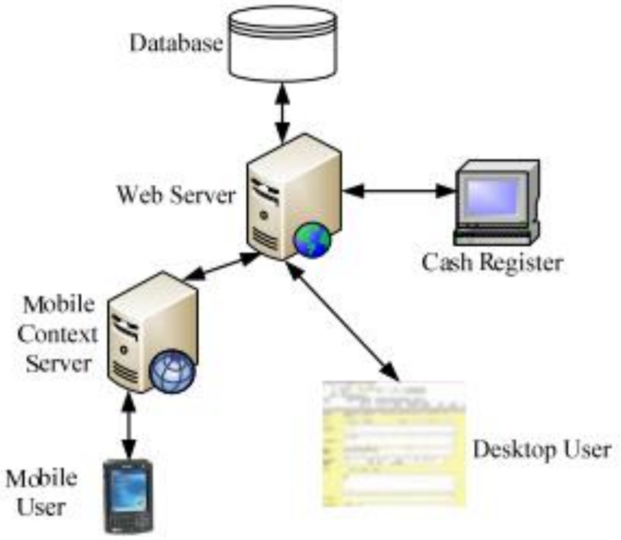


Figure 2.3.1.1: Overview of the system architecture, illustrating the interplay between the various components [15].

To achieve comprehensive security measures, the authors implemented web service security methods at two levels: platform/transport and application. At the platform level, the Secure Sockets Layer (SSL) protocol is used to enable end-to-end encryption while ensuring message integrity and confidentiality throughout transmission. On the application level, special SOAP headers are used for authentication and authorization, enabling the system to handle fine-grained access control and user management.

The wireless food ordering system is built using the Microsoft.NET framework and.NET Compact Framework. The server application is built with ASP.NET and C#, while the database backend is driven by Microsoft SQL Server 2000. The system consists of five major functional modules: system management, food management, client management, food ordering management, and finance management. These modules work together to perform activities like system configuration, menu administration, customer information storage, order processing, and financial operations.

Figure 5. A view of desktop PC

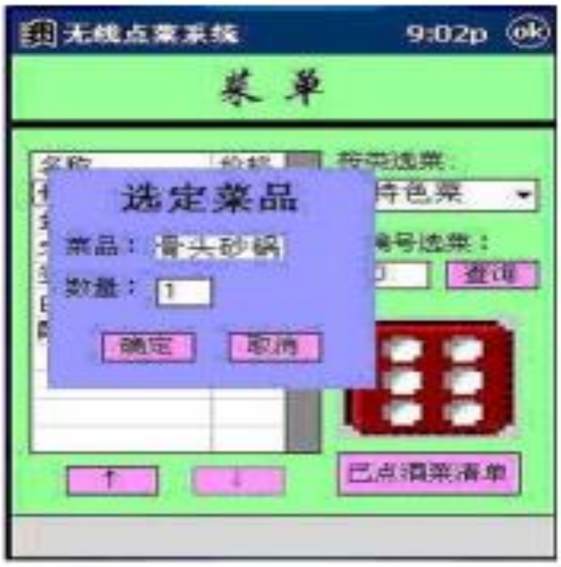


Figure 6. A view of PDA

Figure 2.3.1.2: User interfaces for desktop PC (Figure 5) and PDA users (Figure 6), respectively.

The authors used a variety of ways to solve the unique security concerns that mobile devices provide. These include data encryption via the WEP (Wired Equivalence Privacy) standard, individual user authentication techniques, and MAC address filtering to guarantee that only authorised devices can connect to the system. This paper presents a wireless food ordering system that uses online services and wireless communication technology to revolutionise the restaurant meal ordering procedure. The technology promises to improve efficiency, eliminate human errors, and improve the overall customer experience by providing wired and wireless access to servers from a variety of devices. The system's modular design and integration of strong security features make it scalable and adaptable to the changing needs of the restaurant business.

2.3.2 Presto Canteen Management System with an Android Application

In this reviewed study [18], Manikandan et al. offer a canteen management system called "Presto" that uses an ARM processor, Bluetooth module, thermal printer, Liquid Crystal Display (LCD), and an Android application. The primary goal of this system is to simplify the meal ordering procedure and shorten queue wait times for university students and staff. The introduction provides background information on the current technology revolution and its impact on our daily life. It emphasises the use of technology in canteen management systems to provide a more convenient ordering experience, minimising the need for physical queues at the cafeteria. The authors analyse previous work on automating canteen operations, citing research articles that used technologies such as RFID, ZigBee, Raspberry Pi, and the Internet of Things (IoT).

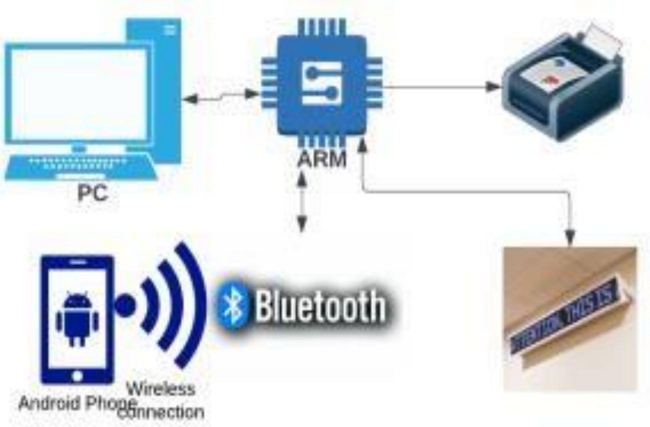


Figure 2.3.2.1: The block diagram of presto canteen management system [18]

The system architecture is shown, which includes an ARM7 32-bit processor, Bluetooth module, LCD display, thermal printer, and Android application. The Android app has several features, including a menu with prices and ratings, an ordering system, customisation choices, token collection, a payment gateway, order monitoring, a loyalty programme, a feedback system, social sharing, and push notifications. The authors provide a full explanation of the system's workflow, as seen in Figure 2.3.2.2. Customers can use the app to select food items from the menu, view their final bill, and pay. After a successful payment, the order is placed in the "waiting for payment" queue, and the payment status is displayed in the app and on the canteen cashier's computer. If the payment fails, clients can have their meal delivered or choose a table number. The programme prints the bill and delivers it to the food service staff, while also displaying a token number and order details on the LCD panel and mobile app.

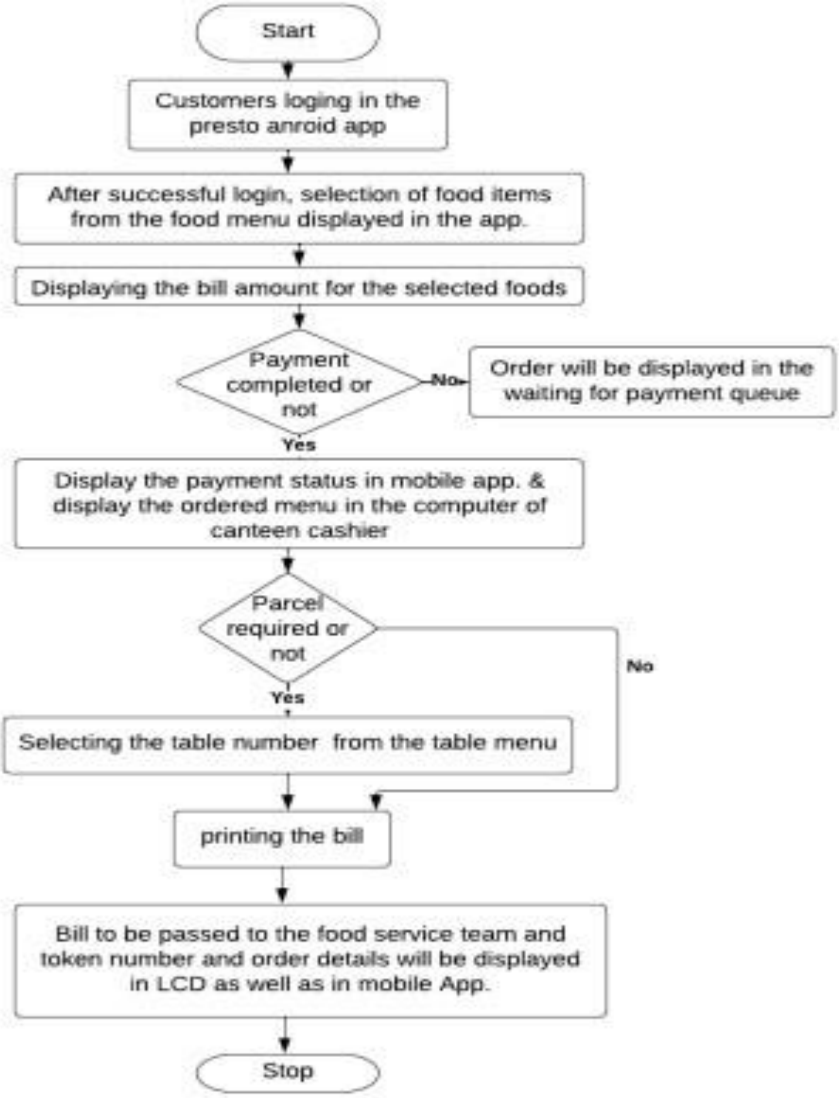

Start
Customers loging in the presto anroid app
After successful login, selection of food items from the food menu displayed in the app.
Displaying the bill amount for the selected foods
Payment completed or not
No
Order will be displayed in the waiting for payment queue
Yes
Display the payment status in mobile app. & display the ordered menu in the computer of canteen cashier
Parcel required or not
No
Yes
Selecting the table number from the table menu
printing the bill
Bill to be passed to the food service team and token number and order details will be displayed in LCD as well as in mobile App.
Stop

Figure 2.3.2.2: Workflow of the Proposed Canteen Management System [18]

The implementation section below focuses on the software element, namely the usage of Android Studio as the Integrated Development Environment (IDE) for developing the Android application. The authors go over the advantages of Android Studio, including its code editor with syntax highlighting, code completion, and error checking, as well as its robust debugger and device emulator.

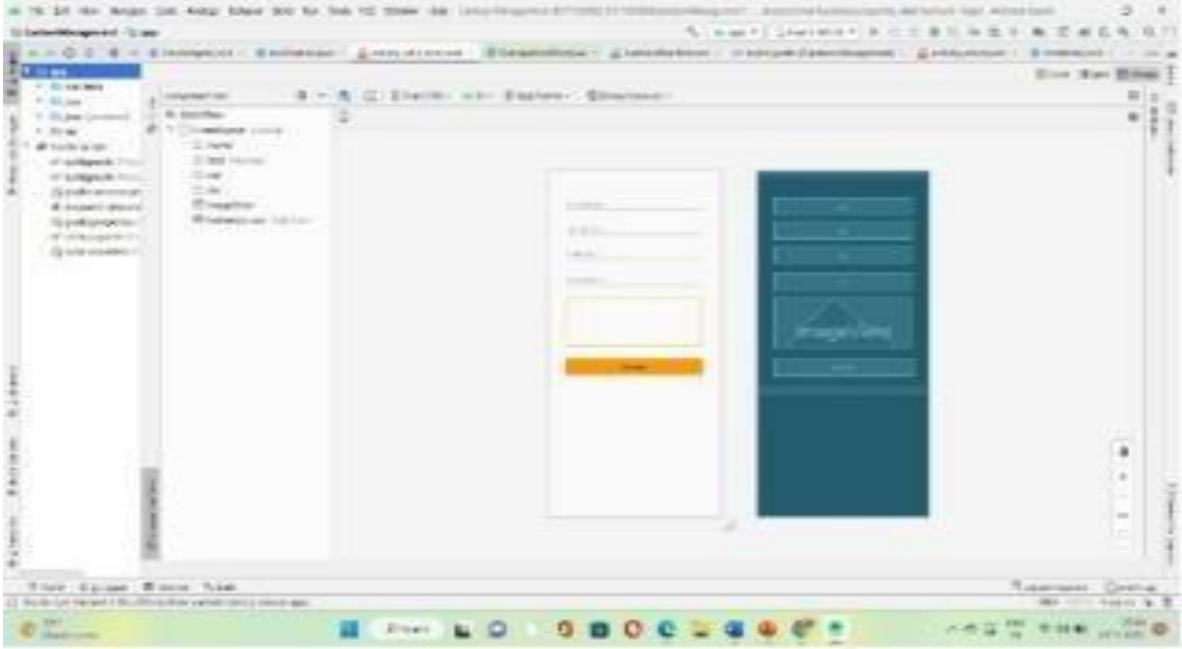


Figure 2.3.2.3: Software Implementation on Android Studio [18]

The hardware implementation is also documented, with figure 2.3.2.4 and figure 2.3.2.5 below depicting the interconnections of the various components, such as the ARM7 32-bit processor, Bluetooth module, LCD display, and thermal printer. The authors describe how to initialise the Bluetooth module with AT commands over the ARM7 processor's UART interface, display information on the LCD, and print data with a thermal printer.

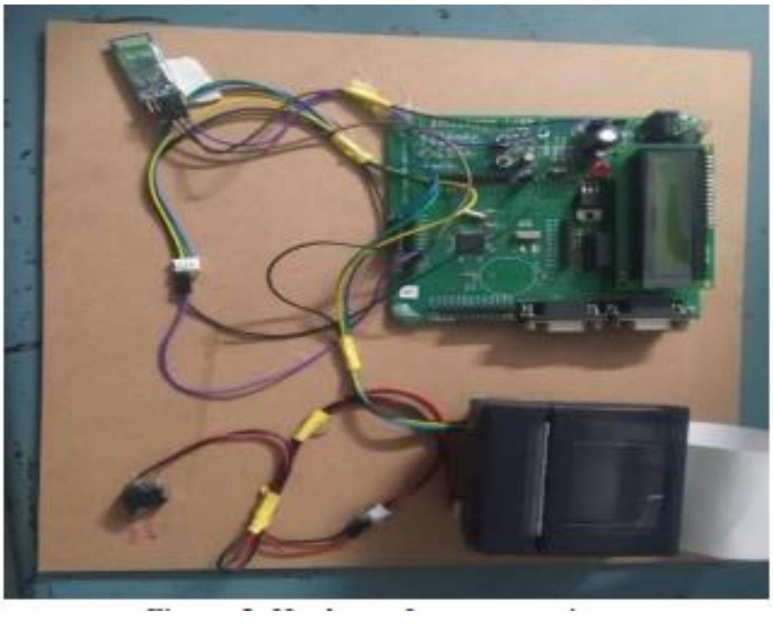


Figure 2.3.2.4: Hardware implementation [18]

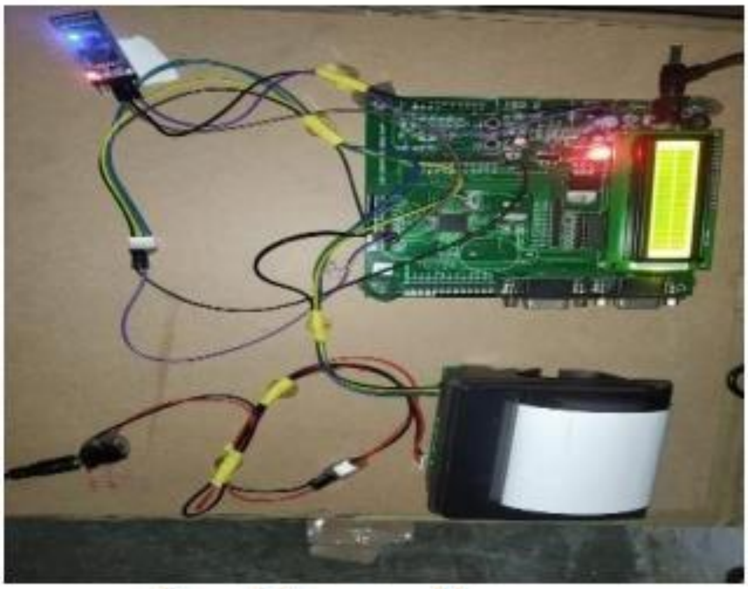
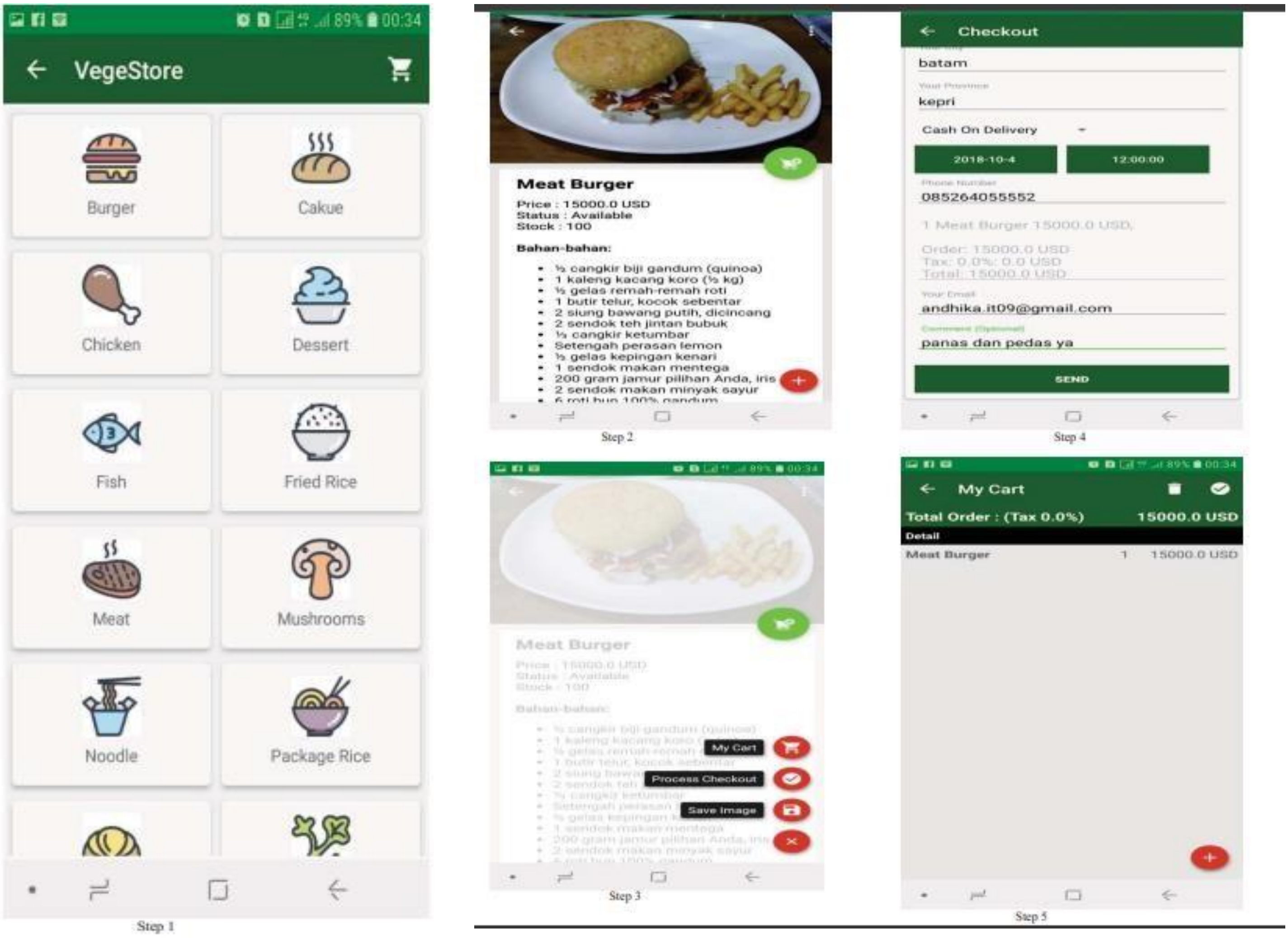


Figure 2.3.2.5: Hardware prototype and connections of the system [18]

The article continues by outlining the suggested system's advantages, such as the ease of ordering food via the Android app, time savings, and the ability to check food inventory and acquire insights into client preferences and ordering patterns. The authors emphasise the need of creating the system from the top down, which ensures a user-friendly and effective solution.

2.3.3 Vege Application! Using Mobile Application to Buy Vegetarian Food

The paper by [19] proposes an Android mobile application called "Vege Application" to promote and sell vegetarian food options in Batam, Indonesia. The rationale is that vegetarian diets can reduce risks of chronic diseases, but access to healthy vegetarian food is currently limited due to factors like lack of information and inability to easily purchase online. To address these problems, the authors suggest a comprehensive mobile application solution that takes advantage of Android technology. The application architecture is intended to create a user-friendly interface that allows users to browse and purchase vegetarian food items easily. The aim of the Vege Application is to increase access to vegetarian food by allowing users to conveniently browse, order, and buy vegetarian products directly through their smartphone app. The application functionality includes Browsing vegetarian food products, viewing product details like prices, stock levels, and availability, placing orders and purchases directly within the app, cash-on-delivery payment method to reduce buyer anxiety about online payment.



Meat Burger

Price : 15000.0 USD
Status : Available
Stock : 100

Bahan-bahan:

- ½ cangkir biji gandum (quinoa)
- 1 kaleng kacang koro (½ kg)
- ½ gelas remah-remah roti
- 1 butir telur, kocok sebentar
- 2 siung bawang putih, dicincang
- 2 sendok teh jintan bubuk
- ½ cangkir ketumbar
- Setengah perasan lemon
- ½ gelas kepingan kenari
- 1 sendok makan mentega
- 200 gram jamur pilihan Anda, iris
- 2 sendok makan minyak sayur
- 6 roti bun 100% gandum

Checkout

batam

Your Province
kepri

Cash On Delivery

2018-10-4 12:00:00

Phone Number
085264055552

1 Meat Burger 15000.0 USD,

Order: 15000.0 USD
Tax: 0.0%: 0.0 USD
Total: 15000.0 USD

Your Email
andhika.it09@gmail.com

Comment (Optional)
panas dan pedas ya

SEND

My Cart

Total Order : (Tax 0.0%) 15000.0 USD

Detail

Meat Burger 1 15000.0 USD

Figure 2.3.3.1: Illustrates the five steps in the frontend customer journey [19]

Figure 2.3.3.1 depicts the step-by-step procedure of using the Vege Application, taking users from the initial launch to the final purchase of vegetarian foods. The programme includes a product catalogue, full product information, real-time pricing and inventory availability, and a secure checkout process. The Vege Application is remarkable for its integration of the Cash on Delivery (COD) payment method. This payment method intends to boost customer confidence by letting purchasers to pay for their goods upon delivery, so reducing concerns about online fraud and non-delivery of purchased items.

The frontend customer journey shown in Figure 2.3.3.1 has the following steps:

Step 1 -Landing page to browse products.

Step 2- Product details page

Step 3- Price and availability

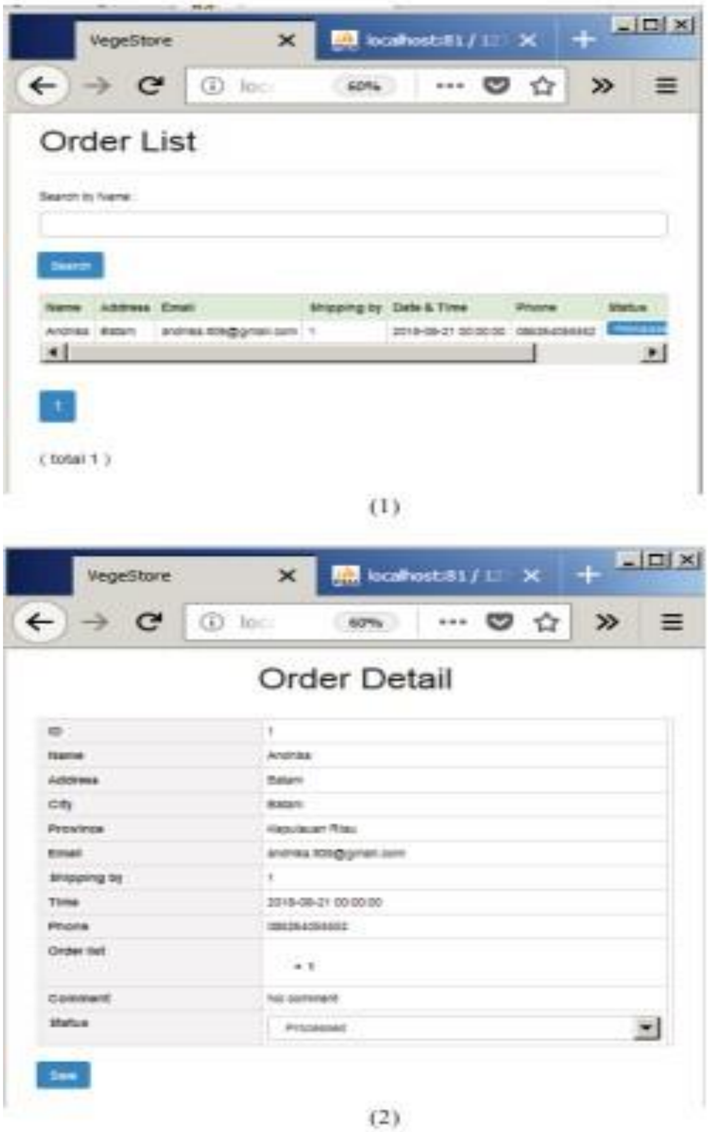
Step 4- Enter order and delivery details.

Step 5 -Confirmation and payment page

The authors emphasize the importance of a reliable back-end system for effective order processing and product upgrades. Figure 2.3.3.2 displays the back-end application interface, which allows sellers to track and handle orders, change product information, and maintain inventory levels. The backend system consists of:

1. Order monitoring dashboard.

2. Detailed order management and communications



Order List

Search by Name:

Name	Address	Email	Shipping by	Date & Time	Phone	Status
Andrika	Batam	[illegible]@gmail.com	1	2018-08-21 00:00:00	[illegible]	[illegible]

(total 1)

(1)

Order Detail

ID	1
Name	Andrika
Address	Batam
City	Batam
Province	Kepulauan Riau
Email	[illegible]@gmail.com
Shipping by	1
Time	2018-08-21 00:00:00
Phone	[illegible]
Order list	• 1
Comment	No comment
Status	Processed

(2)

Figure 2.3.3.2: Application back-end process [19]

The implementation section describes the hardware and software specs used in developing the Vege Application. The writers used Android Studio as the Integrated Development Environment (IDE), with Java and XML as the primary programming languages. The application was tested on a Lenovo G40-45 laptop and a Samsung A8 smartphone to ensure compatibility with several hardware platforms. The authors conclude by emphasising the Vege Application's potential benefits for encouraging a healthy lifestyle, enhancing customer happiness, and recruiting new consumers to vegetarian eateries. They see the Cash on Delivery function as a major aspect in increasing client trust and addressing concerns about online theft. Overall, the Vege Application offers an innovative solution to the difficulties that consumers encounter when finding and purchasing vegetarian cuisine. The project uses mobile technology to provide a convenient and trustworthy platform for promoting and supporting the adoption of healthy, plant-based diets.

2.3.4 Design of College Canteen Ordering System Based on Cortex-A9 and Android

In this reviewed paper [20], the authors describe a college canteen ordering system based on the Cortex-A9 processor and Android technologies. The proposed system is intended to overcome the disadvantages of existing manual ordering techniques, such as pricing inaccuracies, low efficiency, and a lack of data analytic capabilities. The system architecture consists of two primary components: the local terminal and the client. The local terminal comprises of a host powered by the Cortex-A9 CPU and a series of triggers located near each dish at the canteen counter. The ZigBee wireless network facilitates communication between the host and the triggers, allowing for seamless data flow.

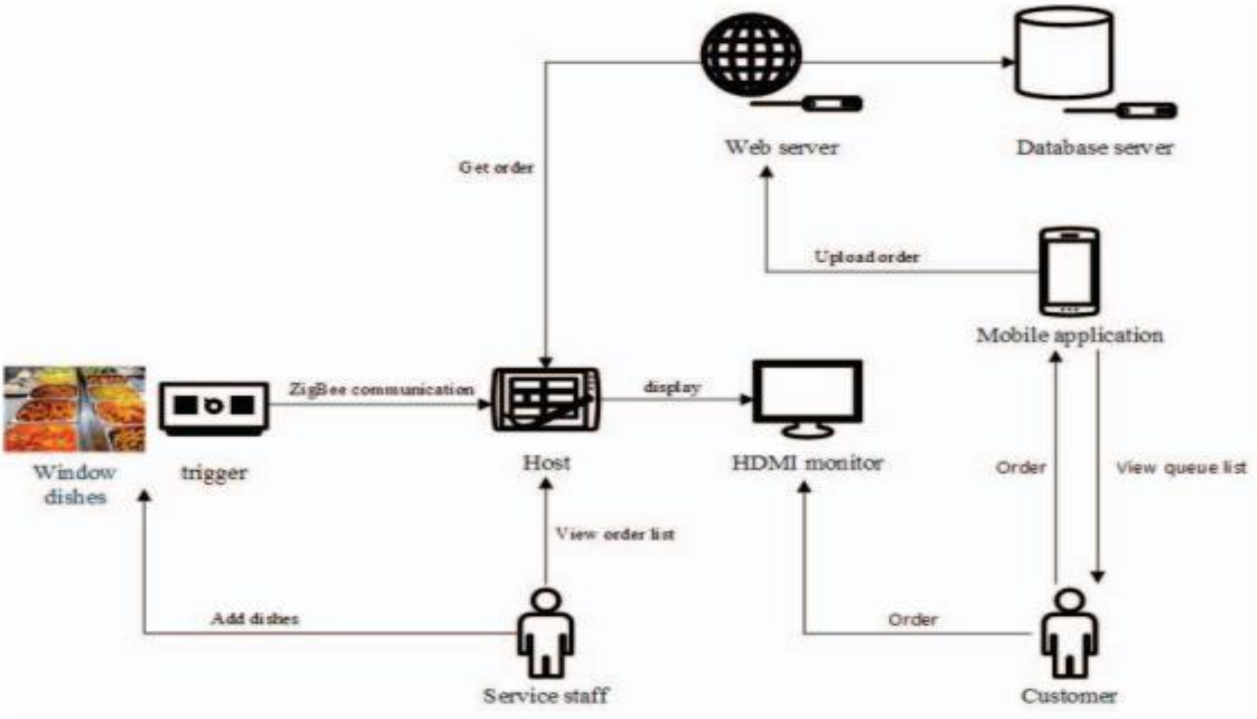


Figure 2.3.4.1: Illustration of the system structure diagram [20]

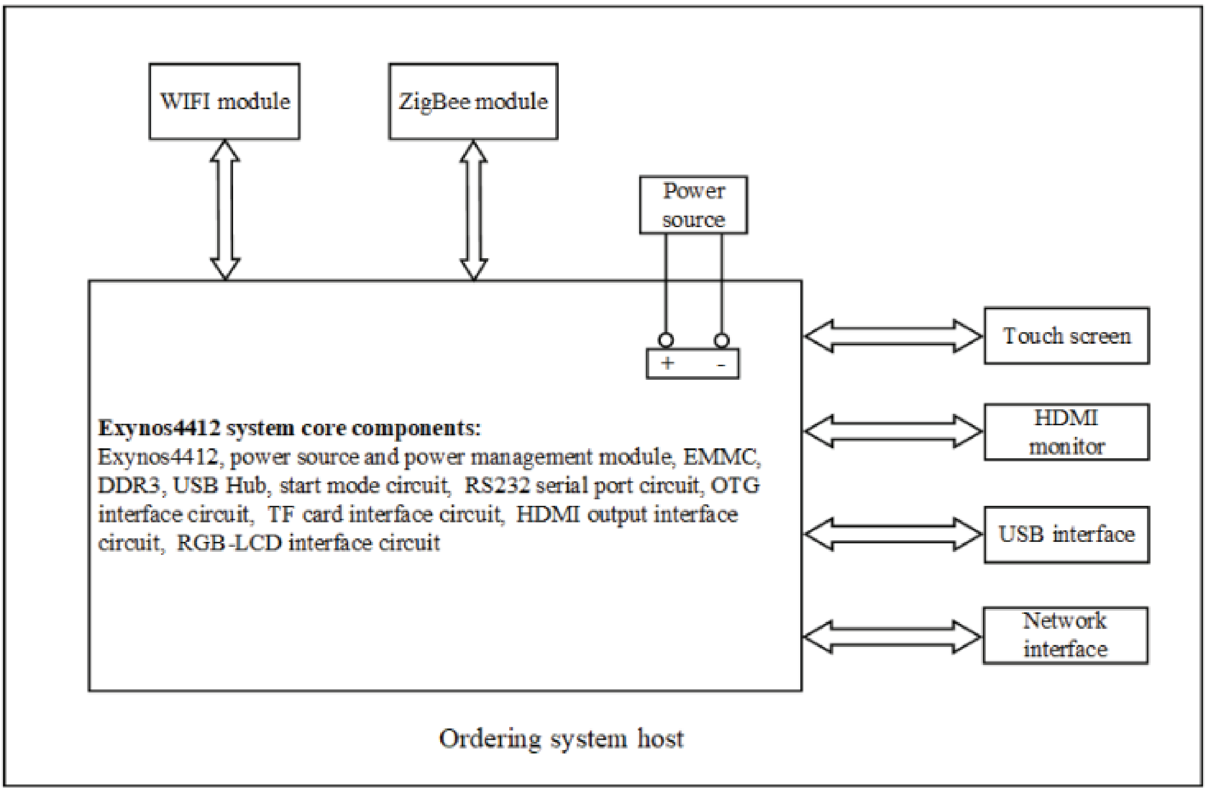

WIFI module
ZigBee module
Power source
+
-
Exynos4412 system core components:
Exynos4412, power source and power management module, EMMC, DDR3, USB Hub, start mode circuit, RS232 serial port circuit, OTG interface circuit, TF card interface circuit, HDMI output interface circuit, RGB-LCD interface circuit
Touch screen
HDMI monitor
USB interface
Network interface
Ordering system host

Figure 2.3.4.2: Host hardware architecture diagram [20]

The host hardware architecture, shown in Figure 2.3.4.2, includes the Samsung Exynos 4412 chip, DDR3 memory, eMMC flash storage, and a variety of interfaces such as WiFi, ZigBee, USB, and HDMI. The host runs Android 5.1, which was converted and configured specifically for the Cortex-A9 architecture. On the client side, the authors created a mobile application that allows customers to place orders via the Internet. The app includes functions including menu browsing, order placement, payment processing, order history tracking, service review, and personalized recommendations. To meet the diverse needs of its customers, the system offers two separate ordering modes. The first option involves local trigger-based ordering at the canteen counter, in which staff workers interact with triggers to choose the appropriate foods. The data is subsequently sent wirelessly to the host for processing and display. The second approach allows customers to place orders directly from the mobile application. Following successful payment, the order information are saved in a MySQL database on a web server. The host obtains the most recent order information from the database in real time, updating the order queue and notifying the service staff accordingly.

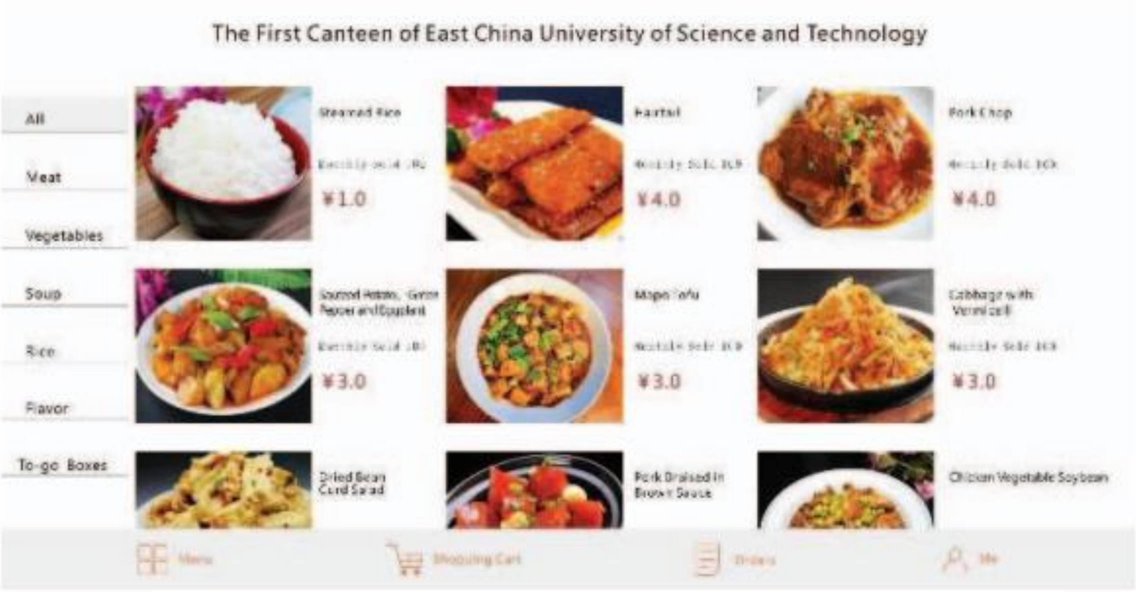


Figure 2.3.4.3: Host application software main [20]

Figure 2.3.4.3 shows the primary interface of the host application programmed, which includes modules for dish presentation, shopping cart management, mobile order handling, and user management. The authors go on to describe how they designed the trigger circuit board, which uses the STM32F103 microcontroller and incorporates components including the ZigBee module, rocker buttons, and LEDs. The communication between the host and the trigger group uses the ZigBee broadcast mode, which allows for efficient multi-point data exchange.

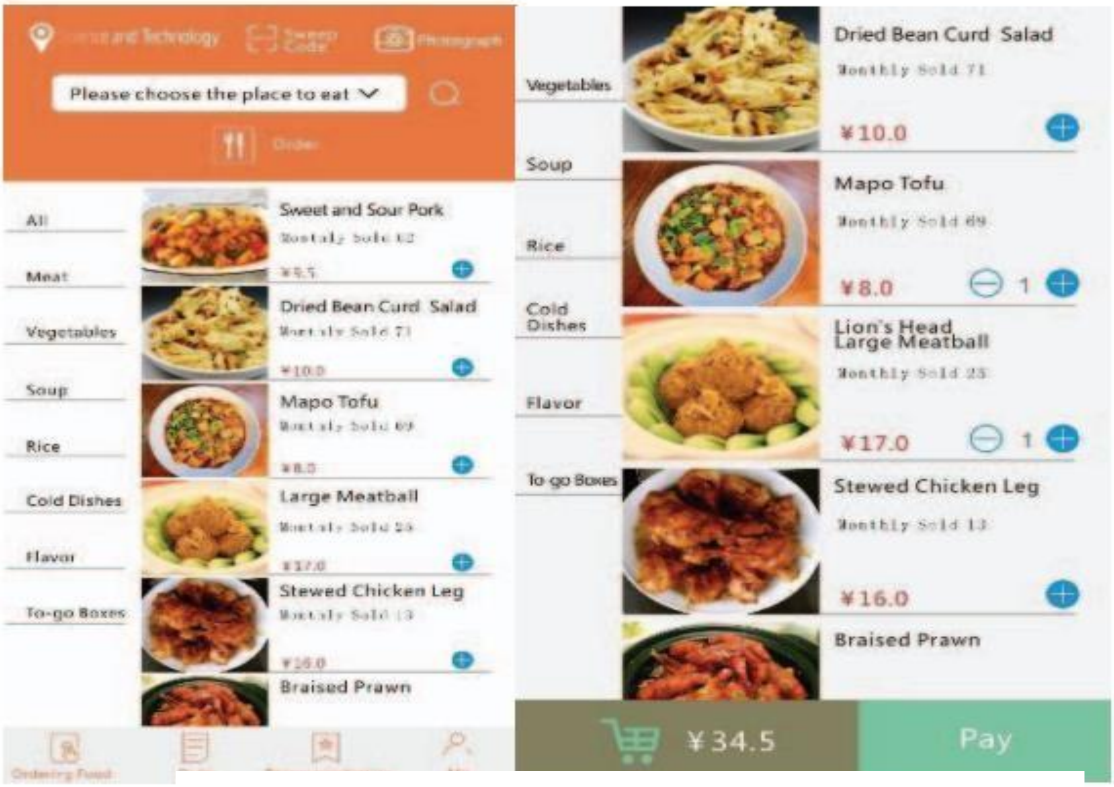


Figure 2.3.4.4: Client mobile app order interface [20]

On the client end, the application software includes modules for login, ordering, order history, statistical analysis, suggestions, and personal information management. Figure 2.3.4.4 depicts the main ordering interface of the client application. The authors used Socket network communication based on the TCP/IP protocol to exchange data between the client and web server. They also created a back-end framework using MyEclipse and set up a Tomcat server to handle client requests and database transactions.

Overall, this paper presents a full college canteen ordering system using the Cortex-A9 CPU, Android technology, and a variety of communication protocols and frameworks. The solution is intended to improve the ordering process, increase efficiency, and provide important data analysis capabilities to college canteens.

2.4 Comparison on The Existing Food ordering applications to "SmartCanteen"

In addition, a further analysis of the three similar food ordering applications above is listed in table 2.4.1.

Paper Title	Technique used	Advantage	Disadvantage
Wireless Food Ordering System Based on Web Services	• Integration of web services and wireless communication technology. • Employment of Web Service Description Language (WSDL) and Simple Object Access Protocol (SOAP) protocols. • Implementing the Secure Sockets Layer (SSL) protocol for end-to-end encryption.	• Seamless integration of wired and wireless connectivity. • Efficient and convenient food order operations • Real-time communication and status updates. • Personalised customisation and safe internet transactions • Increased efficiency and improved inventory management	• Potential network connectivity difficulties. • Initial hardware expenses for implementation. • Additional maintenance required. • Scalability issues with a high number of concurrent orders
Presto Canteen Management System with an Android Application	• Android app for remote ordering and payment	• Reduced queue waiting times for customers.	• Network connectivity issues can disrupt ordering.

	Bluetooth communication between app and ARM processor. • ARM processor for controlling printer, display, etc. • Thermal printer for generating bill/token. • LCD display for order details.	• Convenient ordering through mobile app. • Real-time order tracking and status updates. • Automated billing and payment processing. • Better inventory management for canteen.	• Upfront hardware costs for processor, printer, etc. • Additional maintenance required for hardware. • Scalability issues with large number of concurrent orders.
Design of College Canteen Ordering System Based on Cortex-A9 and Android	• Cortex-A9 processor - Provides computing power for running Android OS and applications. • Android operating system - Enables development of custom mobile app for ordering.	• Improved efficiency - Automated ordering is faster than manual processing. • Flexible ordering – Supports both counter triggers and mobile app. • User-friendly - Intuitive	• Training needed - Staff must learn to operate new automated system • Limited customization since standard mobile app has less flexibility

	- ZigBee wireless network - Allows communication between host and triggers. - MySQL database - Stores order data on back-end server. - Socket programming - Enables networking between client, server, and database.	mobile app interface	
Vege Application! Using Mobile Application to Buy Vegetarian Food	- Android mobile application development using Java and XML in Android Studio. - Modular front-end and back-end system design Cash-on-delivery	- Allows easy browsing and ordering of vegetarian products. - Cash-on-delivery reduces buyer anxiety about online payments. - Back-end system enables vendor	- Lack of advanced features such as reviews, recommendations, etc. - Dependence on courier system for timely fulfilment and delivery

	payment integration. • Use of couriers for order fulfilment and delivery	order management	
SmartCanteen Mobile Application	• Integration with Firebase backend for cloud data storage. • Developed Android app using Java with Android Studio's framework. • Created user-friendly ordering interface. • Provided real-time order progress and availability updates. • Integrated with electronic payment systems for	• Streamlined dining experience with shorter wait times. • Mobile ordering provides greater convenience for users. • Enhanced operational efficiency through automated processes. • Inventory and consumer preferences are managed via comprehensive data management. • Integration with different electronic	• User adaption to new technology and interfaces may necessitate training. • Scalability issues arise with an expanding user base and order volume. • Dependence on stable network connectivity for peak performance. • Using external APIs like Firebase for backend data storage

	convenient transactions	payment methods such as debit/ credit cards and e-wallets to improve accessibility.	

Table 2.4.1: Comparison on the existing Food ordering applications with SmartCanteen

2.5 Review on Commercial Mobile Food Ordering applications

2.5.1 GrabFood (Android & IOS)

Source link: https://food.grab.com/my/en/

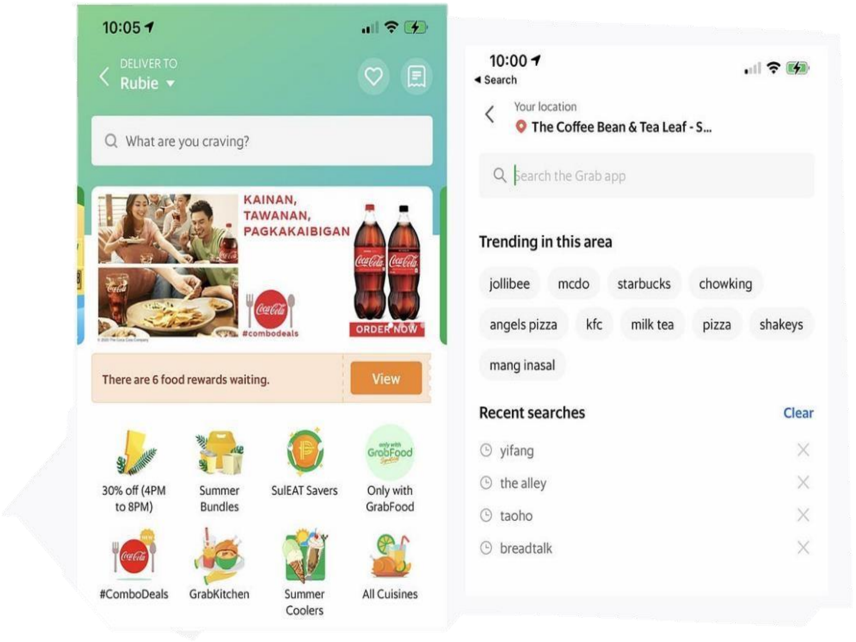


Figure 2.5.1.1 GrabFood user interface

GrabFood, a subsidiary of the popular Southeast Asian ride-hailing company Grab, has successfully entered the food delivery market. This development reflects Grab's strategy to diversify beyond transport services by leveraging its existing user base and technology infrastructure. GrabFood's platform connects customers with restaurants in many countries, delivering a diverse range of cuisines and menu items.

The application's user interface is designed for simplicity, with straightforward navigation and real-time order tracking. Transparency in the delivery process increases user trust and happiness. GrabFood's incorporation of secure payment alternatives simplifies the ordering process even more. GrabFood's service is unique for its seamless integration into the larger Grab ecosystem, allowing consumers to access a variety of services - from ride-hailing to food delivery - via a single platform.

Key Features:

- Predictive Demand Forecasting: GrabFood uses past data analysis to predict demand swings across various durations. This foresight allows restaurants to proactively alter their production capacity, potentially lowering food waste and increasing order fulfilment rates.
- Dynamic Pricing Model: The platform uses a dynamic pricing model that considers aspects such as food temperature, driver availability, and order volume. This strategy seeks to balance supply and demand, particularly during peak hours or under difficult delivery conditions.
- Driver Assignment Optimisation: GrabFood uses machine learning algorithms to efficiently match orders to drivers. To reduce delivery times and increase delivery success, the system considers criteria such as proximity, ideal routes, and vehicle capacity.
- Personalised Recommendation Systems: GrabFood provides personalised restaurant and dish recommendations based on user preferences, demographic information, location data, and current trends. This functionality improves the user experience while potentially increasing order frequency and client retention.

2.5.2 FoodPanda (Android & IOS)

Source link: https://www.foodpanda.my/

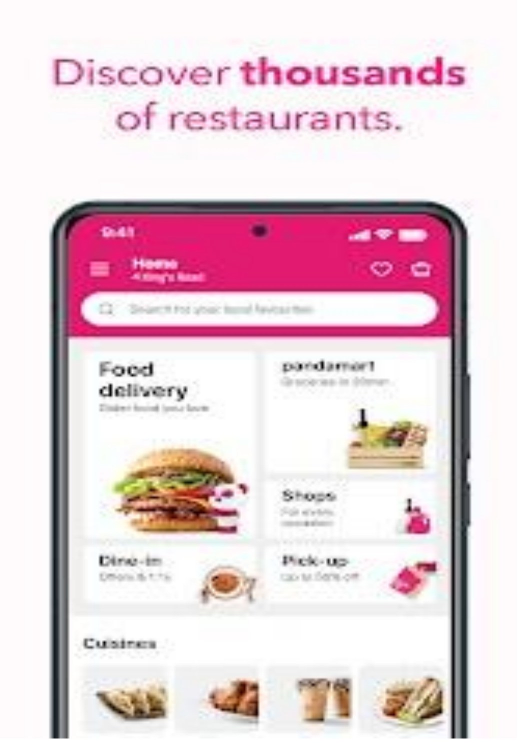


Figure 2.5.1.2 FoodPanda user interface

Foodpanda, a global food delivery platform owned by the Delivery Hero Group, has established itself as a major participant in the online food delivery business. Foodpanda, which operates in various countries in Southeast Asia, connects consumers with a comprehensive selection of local and international eateries, catering to a wide range of culinary preferences and pricing points.

The Foodpanda app was developed with the consumer in mind, emphasising simplicity and efficiency during the ordering process. This design strategy enables clients to place orders quickly, frequently with just a few taps. The platform's broad network of delivery partners is critical to its operational model, since it ensures timely delivery across all service regions.

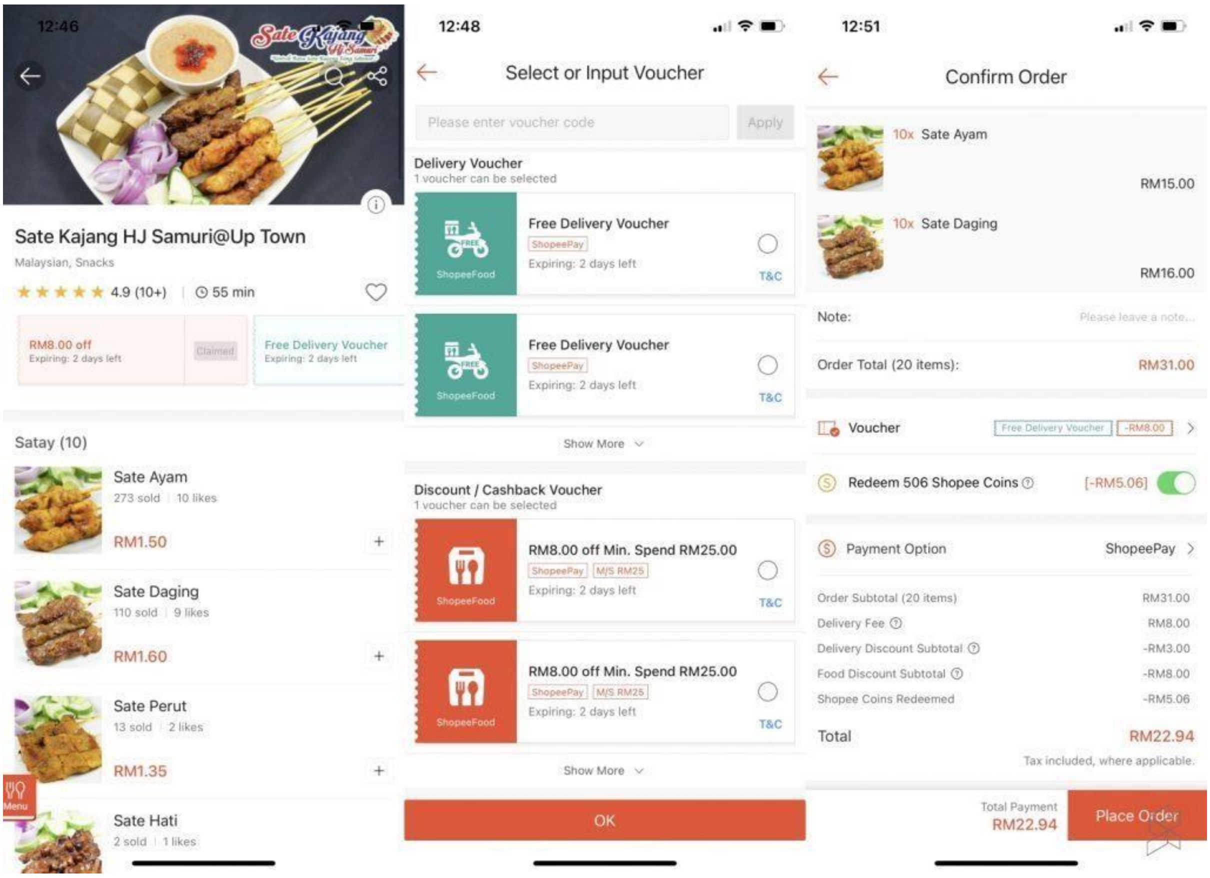
Key Features:

- Demand Prediction: Foodpanda uses advanced forecasting algorithms to predict order volume. By analysing past data, weather patterns, and local events, the platform can adjust its capacity to match changing demand.

- Load balancing: Foodpanda uses real-time order monitoring and predictive modelling to dynamically allocate orders. This technique helps to balance delivery capacity, ensuring that resources are used efficiently during peak and off-peak hours.
- Restaurant Ranking: The platform's restaurant rating system considers a variety of factors, including service time, cancellation rate, and food quality. This comprehensive technique strives to deliver more accurate and helpful rankings to users.
- Customer Segmentation: Foodpanda uses data mining techniques to divide customers into different segments based on order value, frequency, and demand trends. This segmentation allows for more targeted marketing and personalised user experiences.

2.5.3 ShopeeFood (Android & IOS)

Source link: https://www.shopeefood.com.my/



12:46

Sate Kajang HJ Samuri@Up Town

Malaysian, Snacks

★★★★★ 4.9 (10+) | 55 min

RM8.00 off
Expiring: 2 days left
Claimed

Free Delivery Voucher
Expiring: 2 days left

Satay (10)

Sate Ayam	273 sold | 10 likes	RM1.50
Sate Daging	110 sold | 9 likes	RM1.60
Sate Perut	13 sold | 2 likes	RM1.35
Sate Hati	2 sold | 1 likes	

12:48

Select or Input Voucher

Please enter voucher code
Apply

Delivery Voucher
1 voucher can be selected

Free Delivery Voucher
ShopeePay
Expiring: 2 days left
T&C

Free Delivery Voucher
ShopeePay
Expiring: 2 days left
T&C

Show More

Discount / Cashback Voucher
1 voucher can be selected

RM8.00 off Min. Spend RM25.00
ShopeePay M/S RM25
Expiring: 2 days left
T&C

RM8.00 off Min. Spend RM25.00
ShopeePay M/S RM25
Expiring: 2 days left
T&C

Show More

OK

12:51

Confirm Order

10x Sate Ayam RM15.00

10x Sate Daging RM16.00

Note: Please leave a note...

Order Total (20 items): RM31.00

Voucher: Free Delivery Voucher -RM8.00

Redeem 506 Shopee Coins [-RM5.06]

Payment Option: ShopeePay

Order Subtotal (20 items)	RM31.00
Delivery Fee	RM8.00
Delivery Discount Subtotal	-RM3.00
Food Discount Subtotal	-RM8.00
Shopee Coins Redeemed	-RM5.06
Total	RM22.94

Tax included, where applicable.

Total Payment RM22.94
Place Order

Figure 2.5.1.3 ShopeeFood user interface

Shopee Eats, a subsidiary of the renowned e-commerce platform Shopee, is a unique combination of online shopping and food delivery. This integration enables customers to seamlessly order meals alongside their regular purchasing activity, resulting in a more comprehensive and convenient digital marketplace experience.

Shopee Eats intends to provide an integrated experience that combines the appeal of e-commerce with the instant gratification of food delivery, leveraging its current user base and technological infrastructure. This strategy not only broadens Shopee's service offerings, but it also has the potential to boost user engagement and retention across all platforms.

Key Features:

- Cross-Platform User Data Utilisation: Shopee Eats takes advantage of the abundance of user data available through Shopee's e-commerce activities. The platform can analyse user demographics, surfing history, and purchasing trends to improve its meal delivery service.
- Targeted Marketing Campaigns: The platform uses its extensive user data to develop personalised marketing campaigns. These might include vendor-specific discounts, promotions for new food items, and personalised recommendations, which could boost conversion rates and customer happiness.
- Integrated Payment System: Shopee Eats takes advantage of the unified Shopee Pay wallet, which is available across all Shopee platforms. This integration simplifies the payment procedure for food orders, which may reduce friction in the user experience.
- Cross-Selling Opportunities: The platform employs cross-platform marketing tactics, informing consumers of one service about offers from other Shopee services. This strategy may involve in-app advertising and push notifications to enhance user engagement throughout Shopee's ecosystem.

2.6 Comparison on Currently Commercial Mobile Food Ordering Applications

In addition, a further analysis of the commercialised food ordering applications above is listed in table 2.6.1

Application Name	**Strengths**	**Weakness**
GrabFood	• Optimised driver routing methods.	• Limited to core SEA markets.

	- A wide range of delivery choices. - Integration with Grab payment system and strong focus on food delivery.	- Supply constraints during peak seasons.
ShopeeFood	- Integrated with Shopee ecosystem. - Large delivery fleet. - Access to customer data. - Cost reductions through pooled logistics	- Small market share as a new entrant. - Limited localization expertise. - Expensive fleet maintenance costs
FoodPanda	- Global presence in 40+ countries. - Hyperlocal focus with strong localization. - Advanced demand forecast algorithms. - Loyalty programmes	- Challenges with market-specific operational complexity and technology integration. - High customer acquisition costs.

Table 2.6.1 Comparison on Currently Commercial Mobile Food Ordering Applications

2.7 Conclusion

In summary, this literature study provides a thorough examination of existing systems and technical techniques related to mobile-based canteen ordering solutions. Examining numerous food ordering applications and technologies revealed important insights into mobile platforms, payment integration, and customer identification—all of which are critical components of the envisioned “SmartCanteen” system. This review intends to distil significant findings for the construction of an improved canteen ordering application by assessing the strengths and drawbacks of current offerings. It will enable a comparative analysis of technical alternatives, guiding the selection of relevant technologies that are consistent with the goals of “SmartCanteen” Finally, this literature survey seeks to provide a well-informed technical foundation for the creation of a new mobile-based canteen ordering system, aiming at speeding service and improving user satisfaction.

Chapter 3: System Methodology/Approach

3.1 Introduction.

This chapter will primarily explain the methodology used to align with the objectives of the “SmartCanteen” project. Each phase will be thoroughly examined, including how the project was analysed, designed, built, and implemented to create a mobile application that revolutionised canteen ordering. The emphasis will be on ensuring that the app achieves its objectives, such as providing an easy user experience, supporting seamless order placements, and determining user requirements using online survey questionnaires.

3.2 Design Specifications

3.2.1 Project Methodology

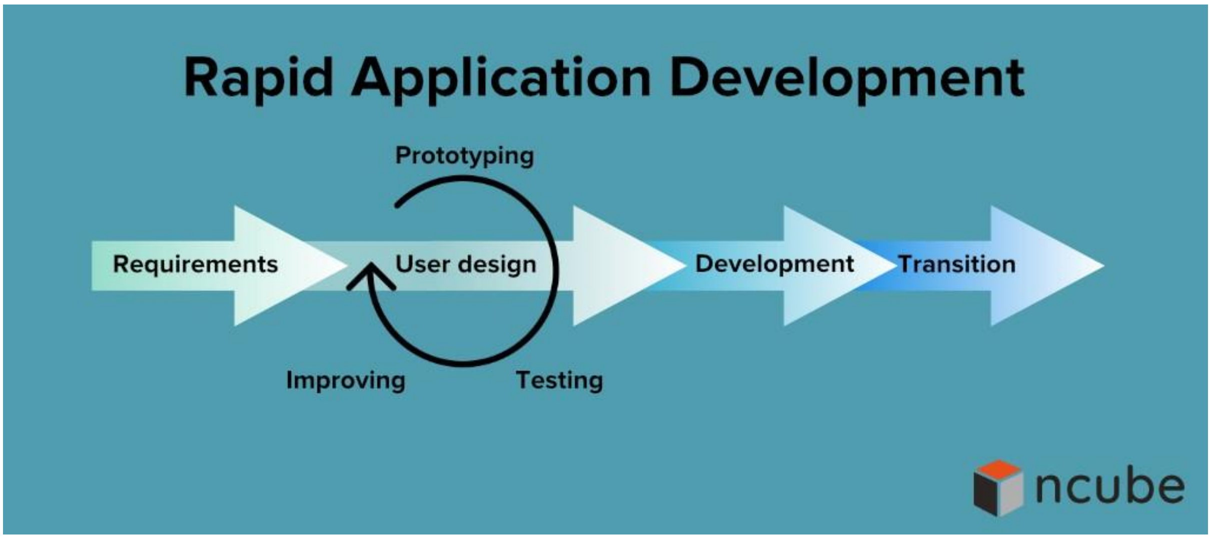


Figure 3.2.1: Rapid Development Methodology Explained [21]

RAD is a system development methodology that emphasises rapid development to deliver the system to the user faster. It's a method that prioritises rapid prototyping and user feedback over long development and testing cycles. The primary benefit of adopting RAD is that it allows developers to swiftly make several iterations and improvements to the product without having to start again. This ensures that the criteria meet user expectations. The process starts with identifying the requirements, then prototyping, receiving input, and finalising the program. The “SmartCanteen” system will be iteratively enhanced during the process to achieve the goals described in the preceding chapter.

3.2.2 Requirements Planning Phase

In the requirements planning phase, requirements-gathering techniques have been used to better understand ways to achieve the objectives. Users were heavily involved in this phase, which required them to provide and prioritise new system requirements and review the system's iterations. An online survey questionnaire served as the major data collection method, providing a low-cost means to reach a large community sample. This survey was designed for UTAR students and staff, to gather useful information to align user demands with project objectives. The questionnaire-based survey method was chosen because it is effective in gathering information from a broad population. The five main elements in the sample design process are: defining the target population, selecting the sampling frame, selecting a sampling technique, determining the sample size, and carrying out the sampling procedure. The convenience sampling method was chosen for its ease and cost-effectiveness in acquiring initial data [22]. A target sample size of 112 respondents was selected and data was gathered through an online questionnaire created using Google Forms. This platform was chosen for its ease of use and capacity to construct multiple question kinds, as well as providing summary reports on survey results. The target audience of the questionnaire will be conducted between the age of 18 and 60 with a focus on individuals who often visit the canteen.

An introduction will be written before conducting the online survey to let users have a summary of the “SmartCanteen” mobile application. The online questionnaire was broken into two sections:

- **Section A: Demographic information**
  - This section will ask general questions such as name, age etc.
- **Section B: Preferred features**
  - This section asks about the mobile application's chosen features and functional expectations.
  - Likert scale type questions will be used ranging from strongly disagree (1) to strongly agree (5).

3.2.3 User design phase

In the system design phase, detailed analysis will be carried out by analysing research articles and existing products to identify business operations linked with the proposed system area. Unified modelling language (UML) diagrams will be created to depict the flow of data.

The following diagrams will be used: use case diagram, activity diagrams for each module, sequence diagram, and class diagram. The diagrams will assist in gaining a better grasp of the different features and functions. For example, a use case diagram summarises the relationship between use cases, actors, and systems. Activity diagrams illustrate the workflows between use cases and provide a detailed understanding of their interactions. Next, a sequence diagram can depict the high-level interaction between the user and the system. Finally, class diagrams model systems from a business perspective, displaying the structure of classifiers in the system. Aside from that, many prototypes of the proposed project will be built and reviewed by the user. Throughout this phase, requirements and features will be re-examined and validated until users are pleased.

The mobile application's user interface will be created using Material design, which was developed by Google for Android developers. Aside from that, the data entered by the user will be checked using many specific functions, including regular expressions, and authentication will be performed using Google Firebase.

3.2.4 Development phase

In the development phase, the prototypes developed for the “SmartCanteen”: Transforming Canteen Ordering with a Mobile Application" project will be refined and adjusted in response to user feedback. The prototypes will be iteratively evaluated, with all comments carefully considered until the final product is given. The mobile application will be developed using Android Studio and Firebase, with user data being stored in the Firebase database. Initially, order data will be recorded within the app and sent to the Firebase database via API calls. This ensures that order information is safely kept in Firebase and easily accessible within the app, allowing users to follow their orders in real time. Following that, the application will undergo extensive functional testing to ensure that it meets all the criteria. To provide a consistent user experience, functional testing will include simulations of user activities such as menu navigation and order placement. The finished product will be checked for functionality and performance. Functional testing involves user-initiated tests to ensure application compliance with requirements. Furthermore, performance testing will be carried out to assess the application's performance under a variety of scenarios, including low memory, low battery, and poor network connectivity. In addition, detailed documentation will be created to show users how to properly navigate and use the system.

3.2.5 Transition

In the final transition phase of the “SmartCanteen” project, extensive testing and validation will be performed to verify that the app is ready for public release. This covers unit tests for specific components like UI screens, API endpoints, and database entities, as well as end-to-end integration testing to check full workflows like ordering, payment, and purchase history. Furthermore, exploratory testing will reveal edge cases and flaws, whilst usability testing with a limited number of canteen customers will provide input for future modification. Any discovered issues will be noted, prioritized, and resolved to ensure the app is reliable and user-friendly. After that, the final product of this proposed project which fulfilled the objectives will be fully complete in FYP2 in September 2024.

3.3 Justification of RAD

The “SmartCanteen” app will be developed using rapid iterative prototypes to get continuous user feedback and validation. This agile approach with rapid iterations allows quickly building app versions with minimal features to gather user feedback early. Users can then interact with these prototypes on their actual mobile devices which helps validate requirements and usability. The rapid cycles enable correcting course based on real user input rather than solely relying on initial requirements. Features and UI/UX can be incrementally improved to boost usability and adoption. Overall, this methodology will incorporate constant user feedback into the development loop leading to a user-centric design focused on addressing core ordering pain points. The rapid prototyping and iteration will maximize the app's utility and ease of use for the target canteen user base.

3.4 User Requirements

According to the survey conducted during the requirements planning phase, the proposed “SmartCanteen” application is required to meet user expectations to give a high-quality meal ordering experience. The requirements are divided into functional and non-functional requirements.

3.4.1 Functional Requirements

A. User Registration: Students and faculty should be able to create accounts with their email addresses and phone numbers.

B. Order Placement: Users should be able to navigate menus, select items, and place orders with ease.

C. Order Tracking: Users should get real-time updates on the status of their orders, from preparation to delivery.

D. Customization Options: Users should be able to customize their orders by selecting items or stating dietary preferences.

E. Payment Integration: For convenience, the application should accept a variety of payment methods, such as credit/debit cards and mobile wallets.

F. Feedback Mechanism: Users should be able to leave feedback and give ratings on their orders and overall experience.

3.4.2 Non-functional requirements

A. Reliability: The application must be always available and functional to meet users' ordering demands, especially during peak hours.

B. Performance: The application should be responsive and capable of supporting several concurrent users with minimal delays or outages.

C. Scalability: The application should be scalable enough to accommodate future expansion and increased user demand while maintaining performance.

3.5 System Design Diagram

In the system design phase, research papers and existing products will be reviewed to analyse the business activities related to the proposed system. Unified modelling language (UML) diagrams will be designed to depict the data flow. The diagrams will help to comprehend the many features and functions. Aside from that, various prototypes of the proposed project will be created for user review. During this phase, requirements and features will be re-examined and validated until all users are pleased. Material design, created by Google for Android developers, will be used to construct the mobile app's user interface. Additionally, the user's input data will be validated using regular expressions and authenticated using Google Firebase.

3.6 System Architecture

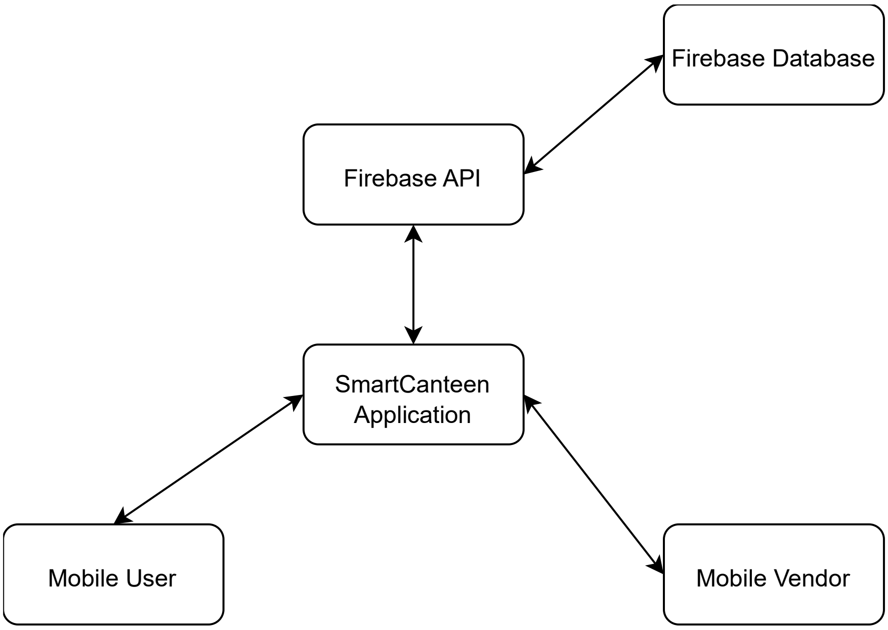


Figure 3.6.1: System Architecture of overall SmartCanteen Mobile Application

The “SmartCanteen” application is the frontend interface for both Mobile Users and Mobile Vendors. The Firebase API provides communication between the “SmartCanteen” App and the Firebase Database, allowing for real-time data synchronisation and interactivity. Firebase DB holds all data associated with user accounts, menu items, orders, and transaction records. Mobile users use the “SmartCanteen” app to explore menus, place orders, and manage their accounts.

Mobile vendors utilize the “SmartCanteen” App to accept and fulfil orders, alter menu items, and track order status. This architecture offers smooth communication between users and vendors while taking advantage of Firebase's real-time database capabilities for efficient data management and synchronization. The “SmartCanteen” App serves as a central hub, offering a simple interface for consumers and vendors to interact with the platform's features and functionalities.

3.7 UML diagram.

The business modelling, requirements, and analysis workflow is transformed into a unified modelling language (UML) diagram using information acquired during the requirements or planning phase. The diagrams will aid in understanding the unique features and functionalities of the proposed project.

3.7.1 Use Case Diagram

By using a use case diagram, it can summarize the user's possible interactions with the system. It is used to define and organize functional requirements in a system and represents the basic flow of events in a use case.

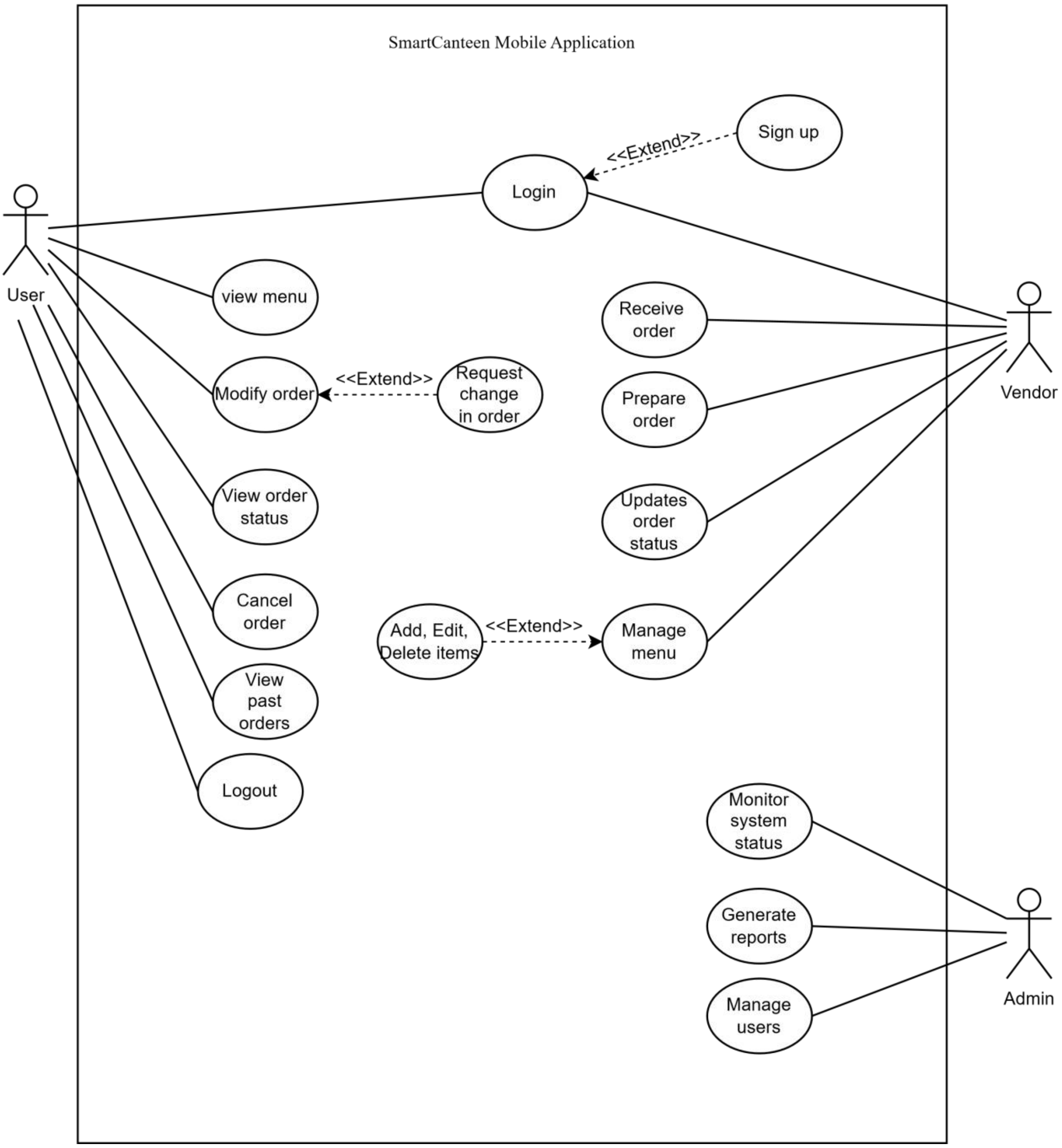


Figure 3.7.1.1: SmartCanteen Mobile Application use case diagram

In Figure 3.7.1.1, the use case diagram depicts the interactions of three major actors in a canteen ordering application: users, vendors, and administrators. Users can place, modify, and cancel orders, as well as examine order status, menus, and previous orders. Vendors receive and prepare orders, manage the menu by adding, updating, or deleting items, change order status, and see order history. Administrators are responsible for managing users, creating reports, and monitoring system status. These use cases collectively represent the functionalities required to allow for seamless order handling, menu administration, and system monitoring within the application.

3.7.2 Activity Diagram

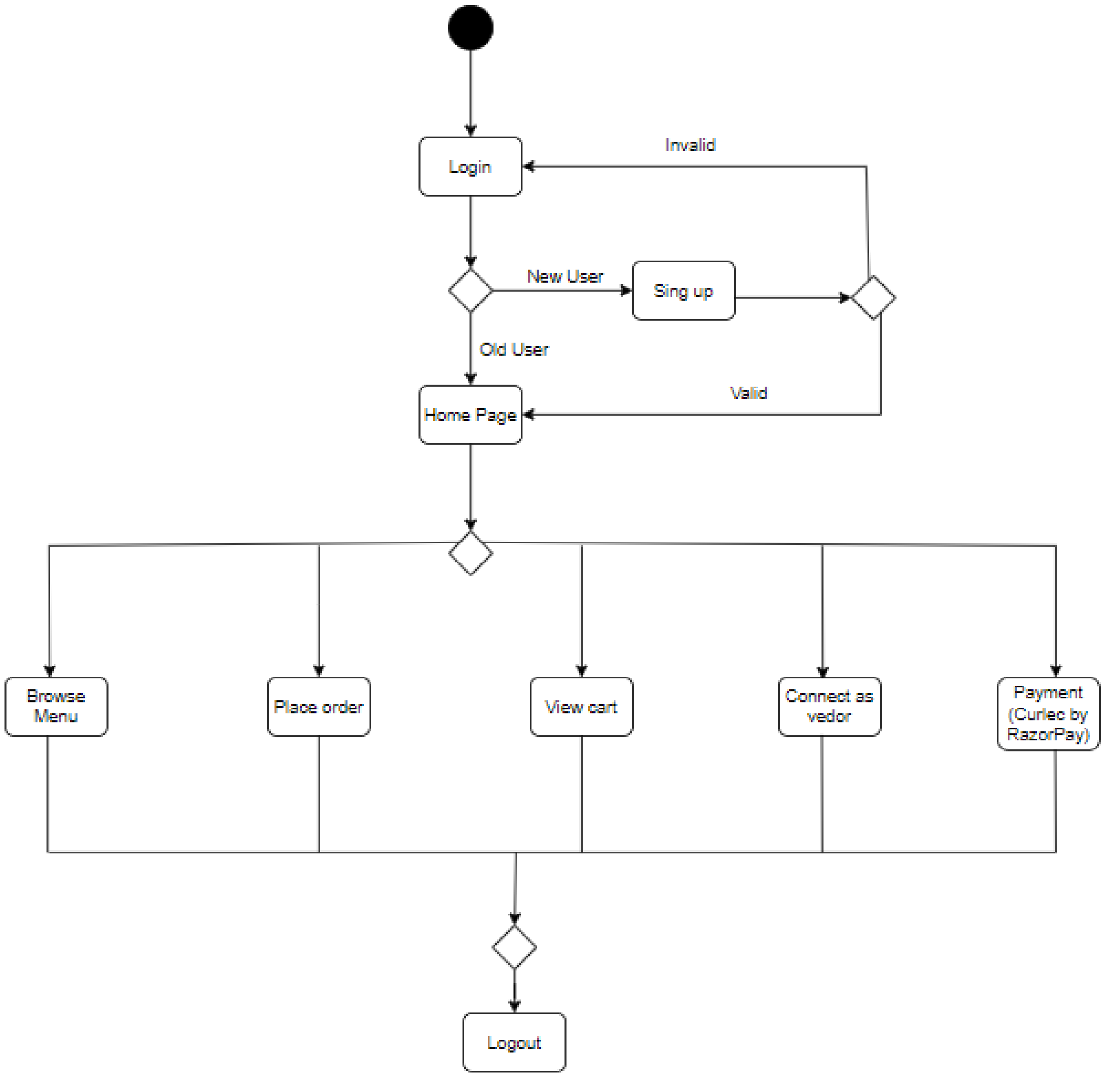


Figure 3.7.2.1: Activity diagram of the overall SmartCanteen Mobile Application

To simplify the system architecture, an activity diagram was created with the following activities: login, sign up, browsing menu, placing order, viewing cart, connecting as vendor, and integrating with payment gateway such as “Curlec” by RazorPay.

3.7.3 Browse Menu Activity Diagram

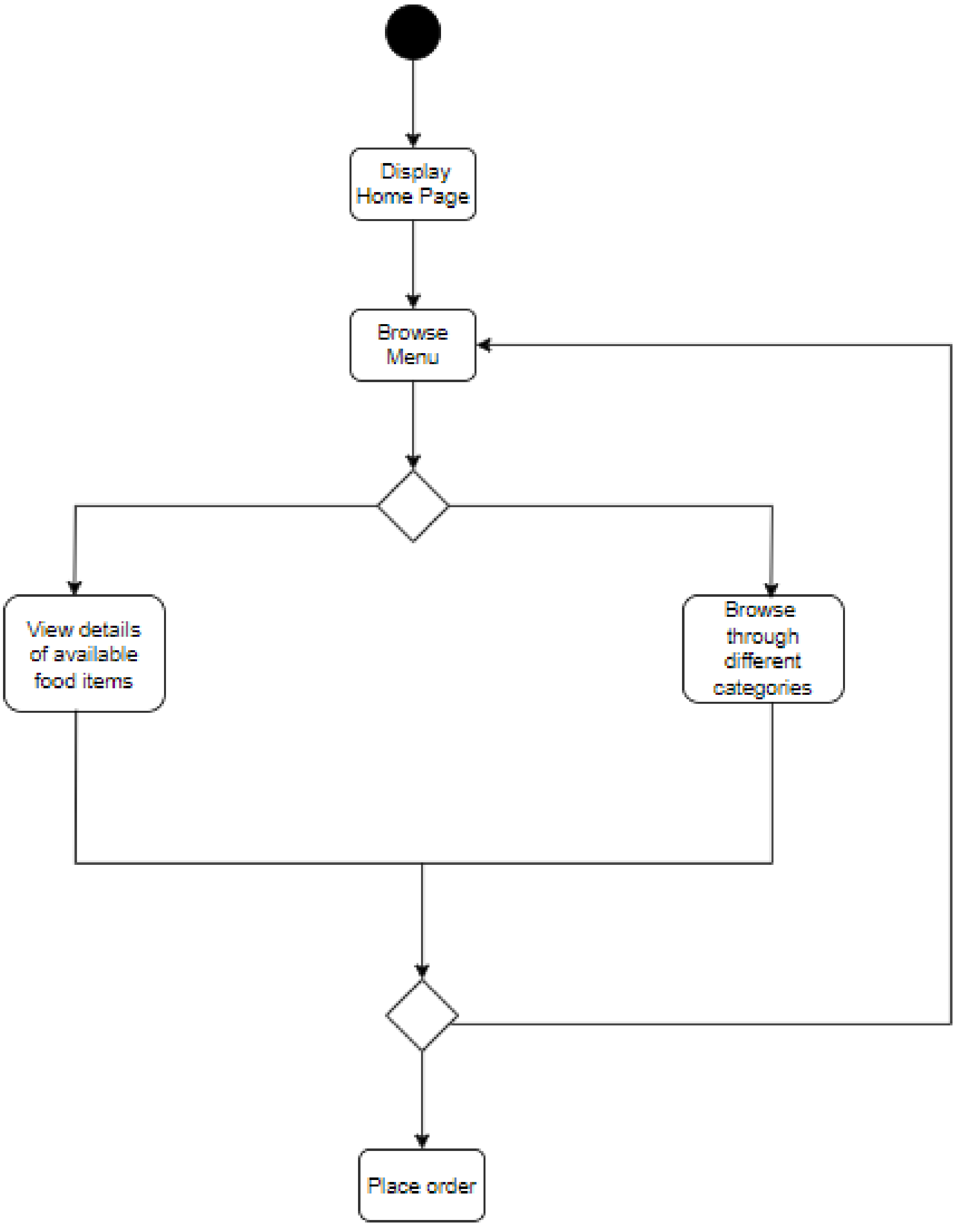

Display Home Page
Browse Menu
View details of available food items
Browse through different categories
Place order

Figure 3.7.3.1: Activity diagram of Browse Menu module.

Based on the figure above, when users navigate to the browse menu option, they are presented with a full list of available food products and categories provided by the canteen. Within this menu layout, consumers may easily navigate through categories such as main courses, desserts, and beverages to choose their favourite food items. Each food item is accompanied with extensive information such as name, description, and price. This enables people to make informed decisions when placing an order.

3.7.4 View Cart Activity Diagram

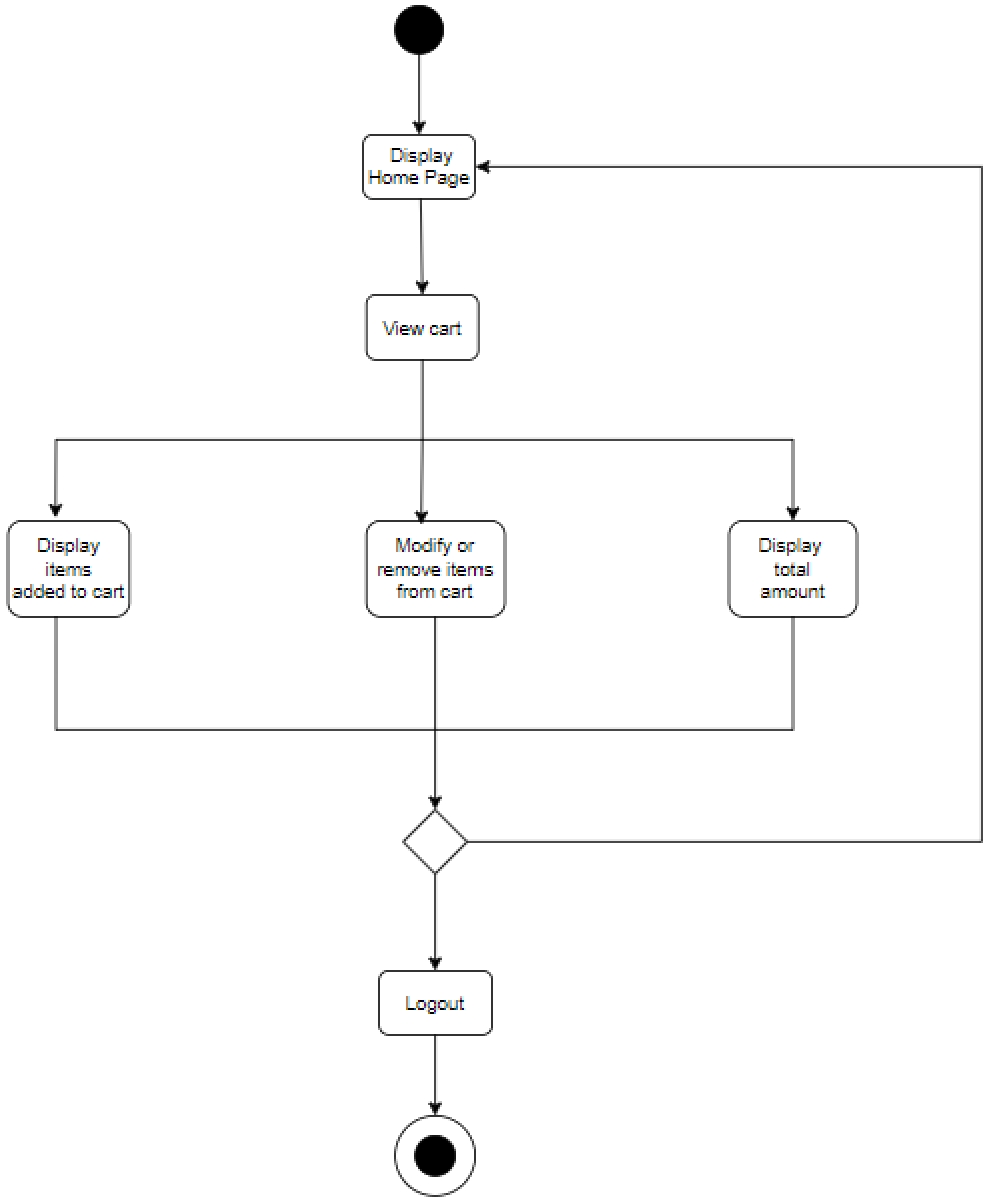


Figure 3.7.4.1: Activity diagram of View cart module.

Based on figure 3.7.4.1, the view cart module offers users a brief overview of all the items that they've put to their cart during their browsing session. Users can change the quantity of things in the cart interface or completely delete individual products, allowing them to tailor their order to their preferences. Furthermore, the system automatically calculates the overall cost of the order, allowing users to see the final amount they must pay at checkout.

3.7.5 Place Order Activity Diagram

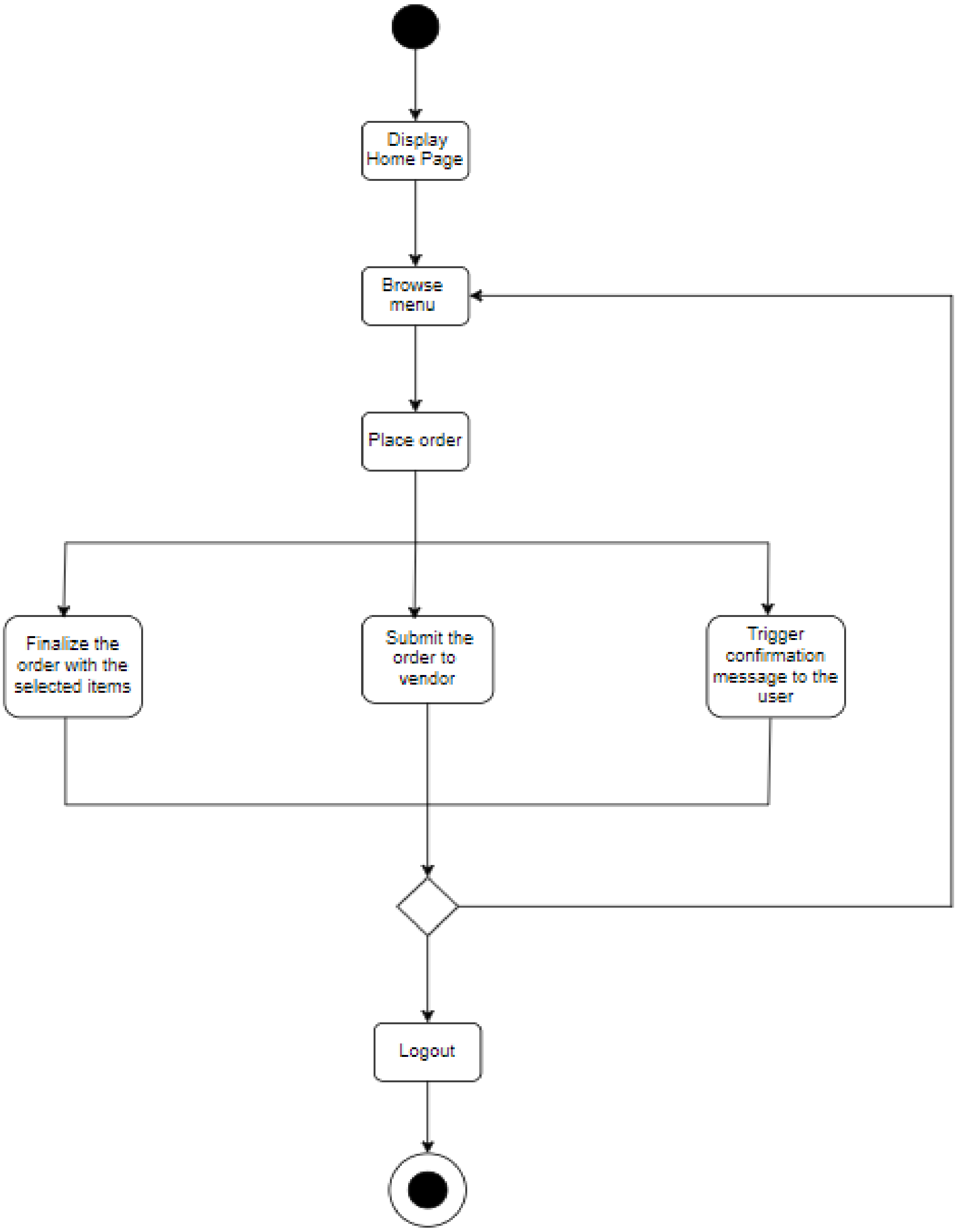

Display Home Page
Browse menu
Place order
Finalize the order with the selected items
Submit the order to vendor
Trigger confirmation message to the user
Logout

Figure 3.7.5.1: Activity diagram of Place Order module.

In figure 3.7.5.1, users can finalise their order by selecting the place order option after reviewing their cart and completing the payment process. Users commence the system's order processing cycle by confirming their selected products. The system then gets the order details, which include the list of items, payment information, and any special instructions provided by the customer. Users receive a confirmation message after successfully submitting their order, confirming that their order was successfully placed and offering an expected completion time, if applicable.

3.7.6 Payment Activity Diagram

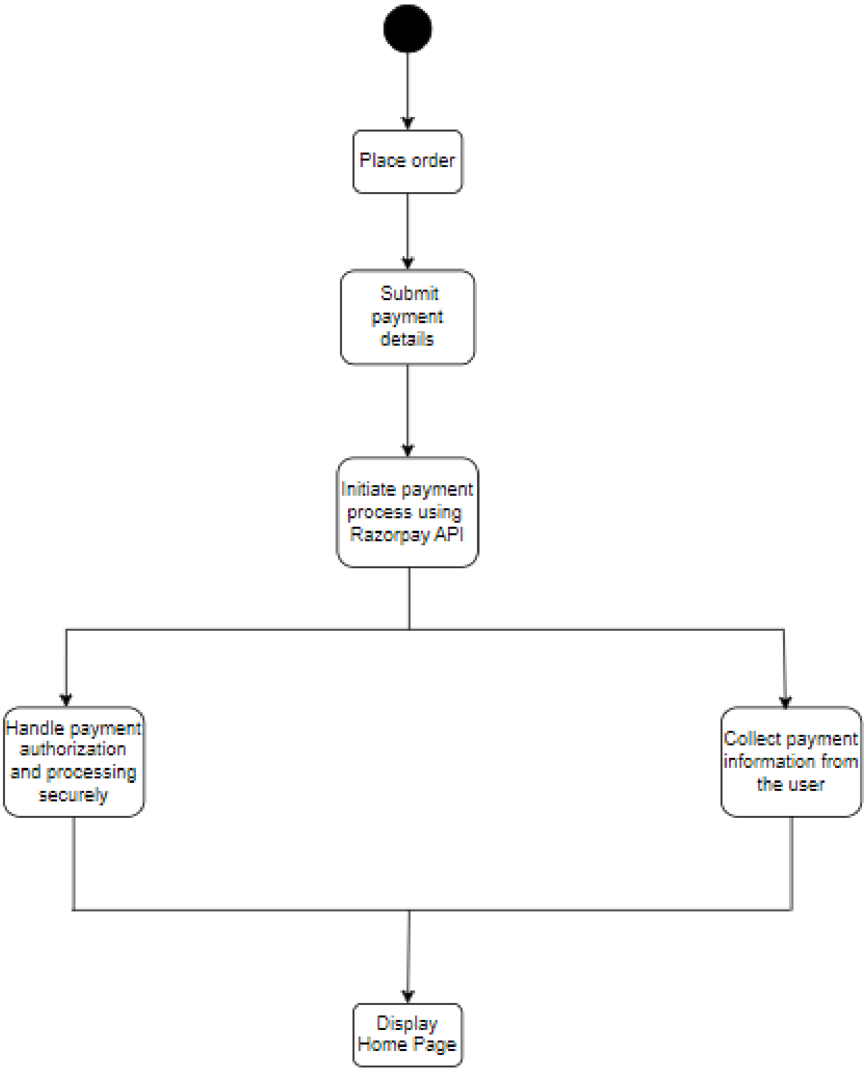

Place order
Submit payment details
Initiate payment process using Razorpay API
Handle payment authorization and processing securely
Collect payment information from the user
Display Home Page

Figure 3.7.6.1: Activity diagram of Payment module.

Based on figure 3.7.6.1, users may confidently and securely complete their transactions thanks to Curlec by Razorpay's payment features. When users go to check out, they are prompted to provide their payment information, which may include credit/debit card information or any other Razorpay-supported payment method. The system securely processes this data and authorises the transaction via the Curlec payment gateway, ensuring that sensitive data is safeguarded throughout the payment process.

3.7.7 Vendor Activity Diagram

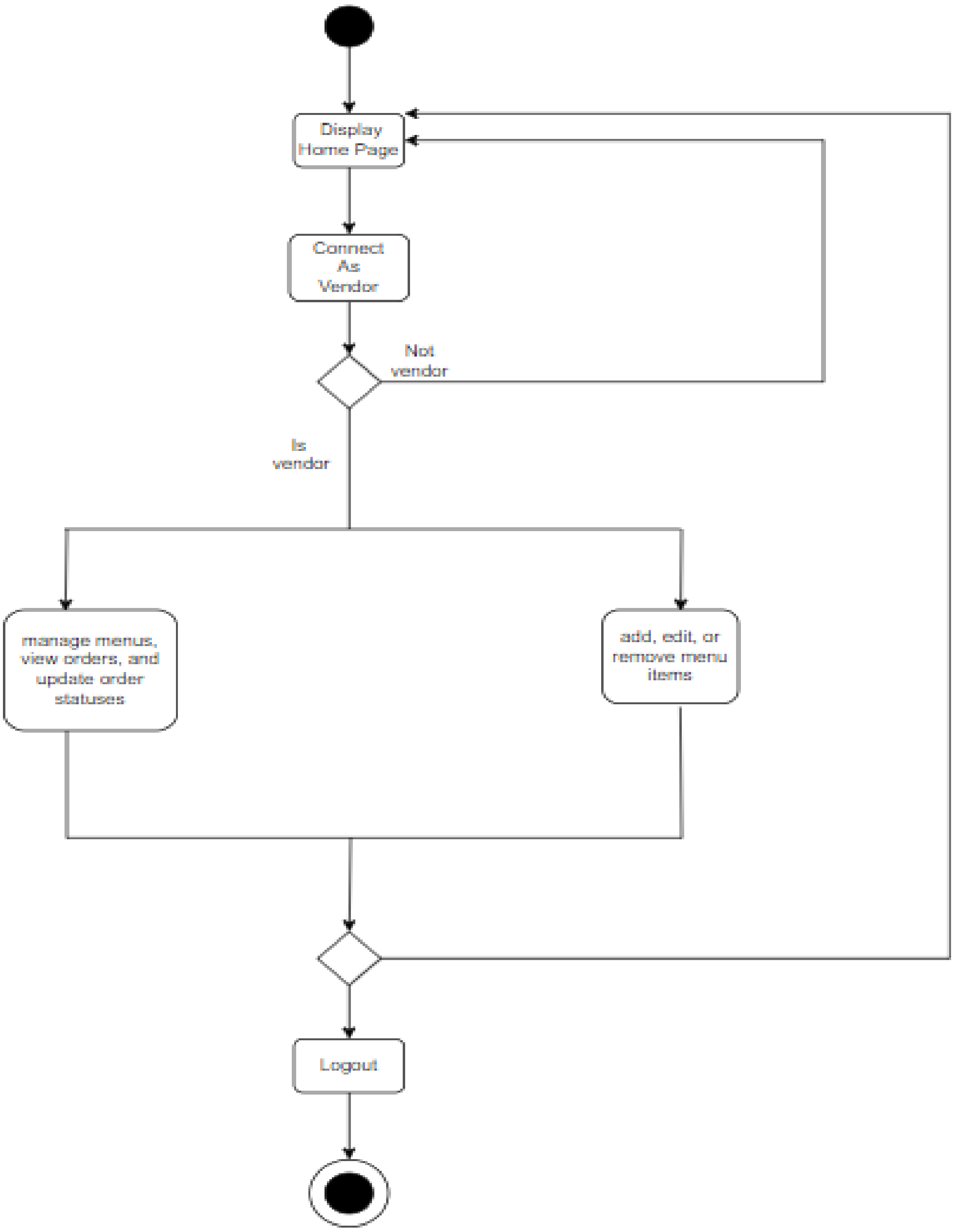

Display Home Page
Connect As Vendor
Not vendor
Is vendor
manage menus, view orders, and update order statuses
add, edit, or remove menu items
Logout

Figure 3.7.7.1: Activity diagram of Vendor module.

In figure 3.7.7.1, vendors can gain access to specific features by logging into their dedicated accounts using the connect as vendor option. Once logged in, vendors have access to a variety of vendor-specific features aimed at streamlining their operations. This includes the ability to customise their menus by adding new items, changing existing ones, and removing items that are no longer available. Additionally, merchants can efficiently handle incoming orders, see order information, and adjust order statuses to keep customers updated on the status of their orders.

3.7.8 Sequence Diagram

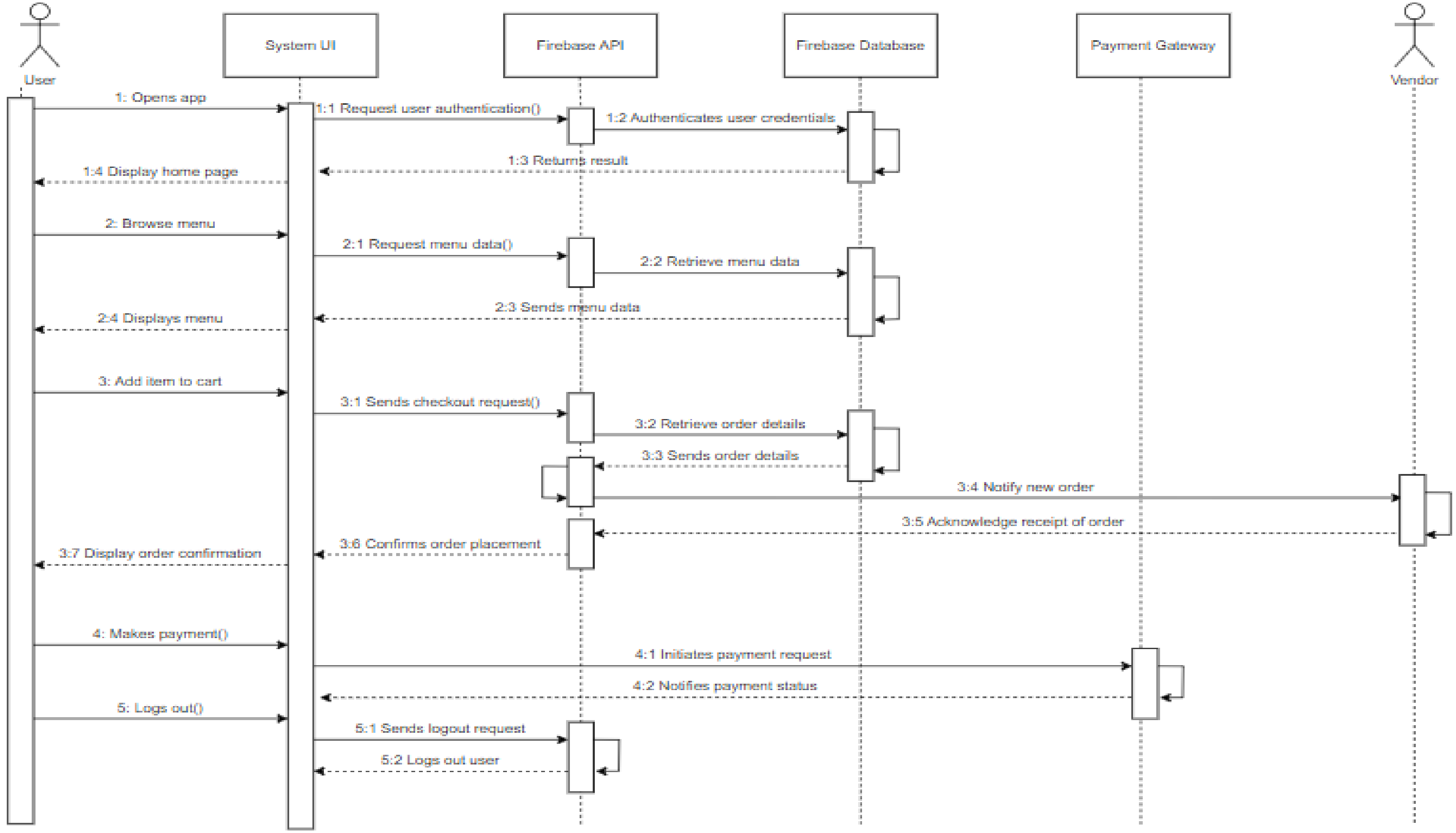


Figure 3.7.8.1: Sequence diagram of the overall SmartCanteen Mobile Application

The sequence diagram in figure 3.7.8.1 depicts the interaction of several components of the SmartCanteen mobile application ecosystem during a typical user transaction. Initially, the user interacts with the SmartCanteen app, which uses the Firebase API to authenticate the user's credentials. After successful authentication, the app displays the home screen, where the user can browse the menu and add items to their cart. When the user proceeds to checkout, the app sends the order details to the Firebase Database, which saves the data and tells the vendor of the new order. Meanwhile, the user initiates the payment process via the app, which then communicates with the Payment Gateway (Curlec by Razorpay) to complete the transaction. Once the payment is confirmed, the payment status is changed in the Firebase API, and the SmartCanteen app notifies the user that the payment has been completed. Finally, the user logs out, and the app communicates with the Firebase API to log user out of the system. This sequence ensures users of the SmartCanteen mobile app have a seamless and secure transaction experience.

3.7.9 Class Diagram

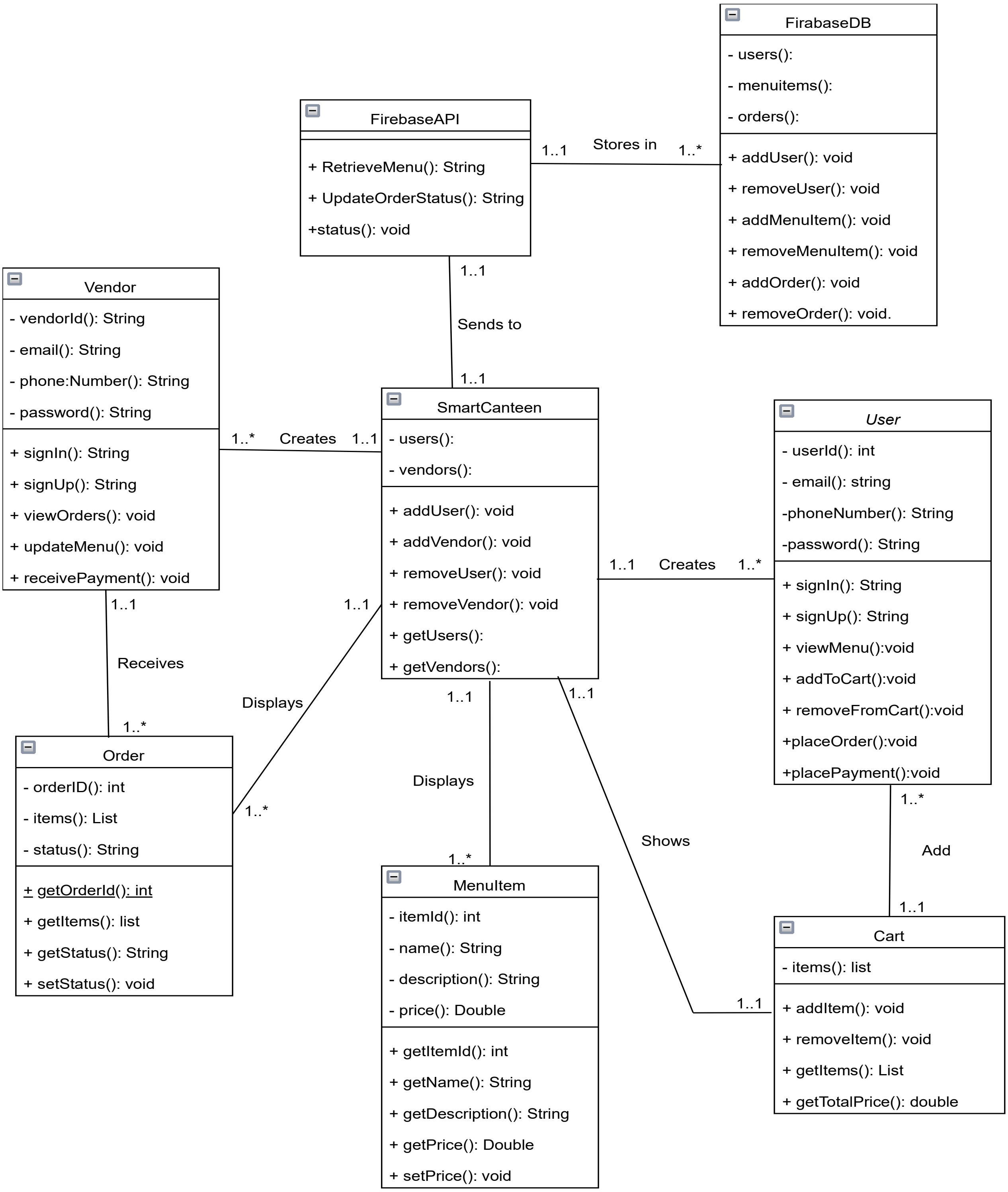


Figure 3.7.9.1: Class Diagram of the proposed system.

The "SmartCanteen" system's major components are outlined in the class diagram in figure 3.7.9.1, with a focus on architecture and functionality. The "SmartCanteen" class is important to the system, providing as the primary interface for managing users, vendors, menu items, orders, and carts. Users and vendors are represented as separate classes, with each including attributes such user IDs, email addresses, phone numbers, and login credentials. The MenuItem class represents individual food items in the system, and includes properties such as item IDs, names, pricing, and categories. Orders are tracked using the Order class, which includes information on the ordered items and amounts. The Cart class makes it easier for users to manage the products they want to buy by allowing them to add, remove, and clear items from their cart. The interaction with the Firebase API and Firebase database ensures that data is stored and retrieved seamlessly, increasing the "SmartCanteen" system's scalability and reliability.

3.8 Conclusion

In conclusion, the design specifications given in this project represent a thorough strategy to designing the "SmartCanteen" mobile application using known processes and iterative procedures. The use of the Rapid Application Development (RAD) approach helps agile development by allowing for quick prototype releases and iterative modifications to suit changing needs. User participation in the planning and design phases promotes alignment with organisational goals and user preferences, with a focus on active engagement and iterative feedback loops. The development phase is focused on iterative improvement in response to user feedback, with tools such as Android Studio and Firebase used for smooth integration and data management. Transitioning to the final step entails extensive testing and validation to ensure that the application is ready for public release. The justification for RAD emphasizes the project's dedication to user-centred design and continuous improvement. Overall, the initiative intends to modernize canteen ordering using innovative mobile technology, addressing crucial points, and improving user's dining experiences.

Chapter 4: System Design

4.1 Introduction.

This chapter will primarily cover the method chosen for the “SmartCanteen” project, including each aspect involved in developing the canteen ordering application. It will describe how the project was analysed, designed, and developed to achieve the objective of improving the dining ordering experience. Each phase will be described in depth to ensure that the proposed system meets user needs and provides a seamless and efficient solution.

4.2 System Setup and Configurations

This system setup act as a documentation for other programmers to have a reference if they want to build up this project on its own local development in the future.

4.2.1 “SmartCanteen” Application Set Up for Order Management

The initial stage of implementation included the visual layout of the "SmartCanteen" mobile application and its integration with the Firebase database.

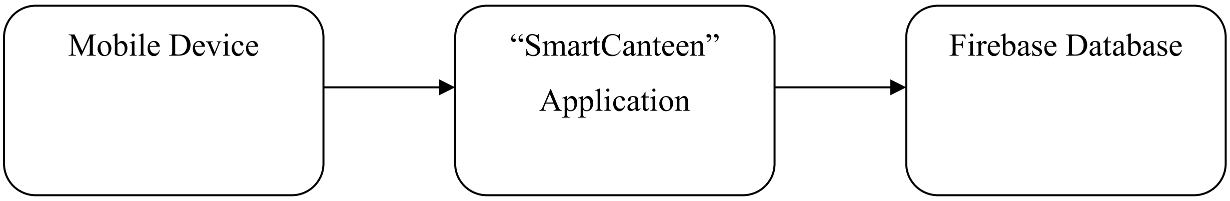


Figure 4.2.1.1 Process flow of the data from mobile device to the Firebase database

Figure 4.2.1.1 shows the logic and process flow of data from the mobile device to the Firebase database using the "SmartCanteen" application. The mobile device places food orders, which are then sent to Firebase for real-time data synchronisation. The Firebase database stores the order information, making it available to both users and vendors. Finally, the "SmartCanteen" application retrieves the updated data from Firebase, giving users a simple interface for managing orders, tracking their status, and sending order confirmations to vendors. The implementation starts with designing the user interface and developing the app's functionalities in Android Studio.

4.2.2 Application Setup

Listing 4.2.2.1 below shows code snippets of how the MainActivity.class of the application should look like. The SmartCanteen app's MainActivity serves as the central navigation hub, allowing users to access various features through a navigation drawer, including navigating to their cart, which is managed by the CartFragment, daily meals, favourite items, and more. The app also includes a toolbar for easy access to additional functionalities. Firebase is initialised in the activity to handle order management and store order data.

```
public class MainActivity extends AppCompatActivity {

    private AppBarConfiguration mAppBarConfiguration;
    private ActivityMainBinding binding;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        binding = ActivityMainBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        setSupportActionBar(binding.appBarMain.toolbar);

        // Firebase Initialization
        FirebaseDatabase database = FirebaseDatabase.getInstance();
        database.setPersistenceEnabled(true); // Enable offline persistence if necessary

        DrawerLayout drawer = binding.drawerLayout;
        NavigationView navigationView = binding.navView;

        // Passing each menu ID as a set of Ids because each menu should be considered as top
level destinations.
        mAppBarConfiguration = new AppBarConfiguration.Builder(
                R.id.nav_home, R.id.nav_daily_meal, R.id.nav_favourite, R.id.nav_my_cart)
```

```
            .setOpenableLayout(drawer)
            .build();
        NavController navController = Navigation.findNavController(this, R.id.nav_host_fragment_content_main);
        NavigationUI.setupActionBarWithNavController(this, navController, mAppBarConfiguration);
        NavigationUI.setupWithNavController(navigationView, navController);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }

    @Override
    public boolean onSupportNavigateUp() {
        NavController navController = Navigation.findNavController(this, R.id.nav_host_fragment_content_main);
        return NavigationUI.navigateUp(navController, mAppBarConfiguration)
                || super.onSupportNavigateUp();
    }

    public void welcome(View view) {
        startActivity(new Intent(MainActivity.this, WelcomeActivity.class));
    }

    public void vendor(View view) {
        startActivity(new Intent(MainActivity.this, VendorMainPage2.class));
    }

    public void profile(View view) {
        startActivity(new Intent(MainActivity.this, Profile.class));
    }
}
```

Listing 4.2.2.1: Code snippet of MainActivity.class of application

The CartFragment.class in the SmartCanteen app allows users to review and manage their chosen food items before proceeding to the checkout. For CartFragment.class to work properly, the CartAdapter.class and Order.class model must be implemented first. Listing 4.2.2.2 below shows the code snippet of how the CartAdapter.class should look like. The CartAdapter.class connects the list of food items in the cart to the RecyclerView, which displays them. It accepts a list of orders, assigns each item to a view, and manages user interactions such as quantity updates and cart removals.

```
public class CartAdapter extends RecyclerView.Adapter<CartAdapter.CartViewHolder> {

    private List<Order> cartItems;

    public CartAdapter(List<Order> cartItems) {
        this.cartItems = cartItems;
    }

    @NonNull
    @Override
    public CartViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int
viewType) {
        View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.item_cart, par-
ent, false);
        return new CartViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull CartViewHolder holder, int position) {
        Order item = cartItems.get(position);
        holder.foodNameTextView.setText(item.getFoodname());
        holder.foodPriceTextView.setText("RM " + item.getTotalPrice());
    }

    @Override
    public int getItemCount() {
        return cartItems.size();
    }

    public static class CartViewHolder extends RecyclerView.ViewHolder {

        TextView foodNameTextView, foodPriceTextView;

        public CartViewHolder(@NonNull View itemView) {
            super(itemView);
            foodNameTextView = itemView.findViewById(R.id.foodNameTextView);
            foodPriceTextView = itemView.findViewById(R.id.foodPriceTextView);
        }
    }
}
```

Listing 4.2.2.2: Code snippet of CartAdapter.class

Next, listing 4.2.2.3 below is the code snippet of Order.class model. The Order model describes the structure of an order, which typically includes fields like foodName, quantity, and totalPrice. This model is critical for storing and retrieving order information, particularly when integrating with Firebase. These components must be configured first to ensure that the RecyclerView in CartFragment.class displays the user's selected items correctly.

```
public class Order {
    private String foodname;
    private double totalPrice;

    public Order() {
    }

    public Order(String foodname, double totalPrice) {
        this.foodname = foodname;
        this.totalPrice = totalPrice;
    }

    public String getFoodname() {
        return foodname;
    }

    public void setFoodname(String foodname) {
        this.foodname = foodname;
    }

    public double getTotalPrice() {
        return totalPrice;
    }

    public void setTotalPrice(double totalPrice) {
        this.totalPrice = totalPrice;
    }
}
```

Listing 4.2.2.3: Code snippet of Order.class model

Listing 4.2.2.4 contains a code snippet from CartFragment.class. Users can modify their orders by changing quantities or removing items. A "Proceed to Checkout" button is provided, which takes users to the PaymentOptionActivity, where they can select cash or online payment. Once confirmed, the order details are sent to Firebase and stored in the Orders node, which includes the food name and total price. The CartFragment.class is accessible from the navigation drawer and is critical to completing the user's order.

```
public class CartFragment extends Fragment {


    private RecyclerView cartRecyclerView;
    private CartAdapter cartAdapter;
    private List<Order> cartItemList;
    private Button btnCheckout;
    private TextView totalPriceTextView;

    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater, @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.fragment_cart, container, false);

        cartRecyclerView = view.findViewById(R.id.cartRecyclerView);
        btnCheckout = view.findViewById(R.id.btnCheckout);
        totalPriceTextView = view.findViewById(R.id.totalPriceTextView);

        cartItemList = new ArrayList<>(); // List to hold cart items
        cartAdapter = new CartAdapter(cartItemList);

        cartRecyclerView.setLayoutManager(new LinearLayoutManager(getContext()));
        cartRecyclerView.setAdapter(cartAdapter);

        // Sample item added to the cart for demonstration
        cartItemList.add(new Order("Pizza", 10.00));
        cartItemList.add(new Order("Burger", 5.00));
        cartAdapter.notifyDataSetChanged();

        // Calculate total price
        calculateTotalPrice();

        // Proceed to checkout when "Checkout" button is clicked
        btnCheckout.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Redirect to PaymentOptionActivity
                Intent intent = new Intent(getActivity(), PaymentOptionActivity.class);
                startActivity(intent);
            }
        });

        return view;
    }

    private void calculateTotalPrice() {
        double totalPrice = 0;
        for (Order item : cartItemList) {
            totalPrice += item.getTotalPrice();
        }
        totalPriceTextView.setText("Total: RM " + totalPrice);
    }
}
```

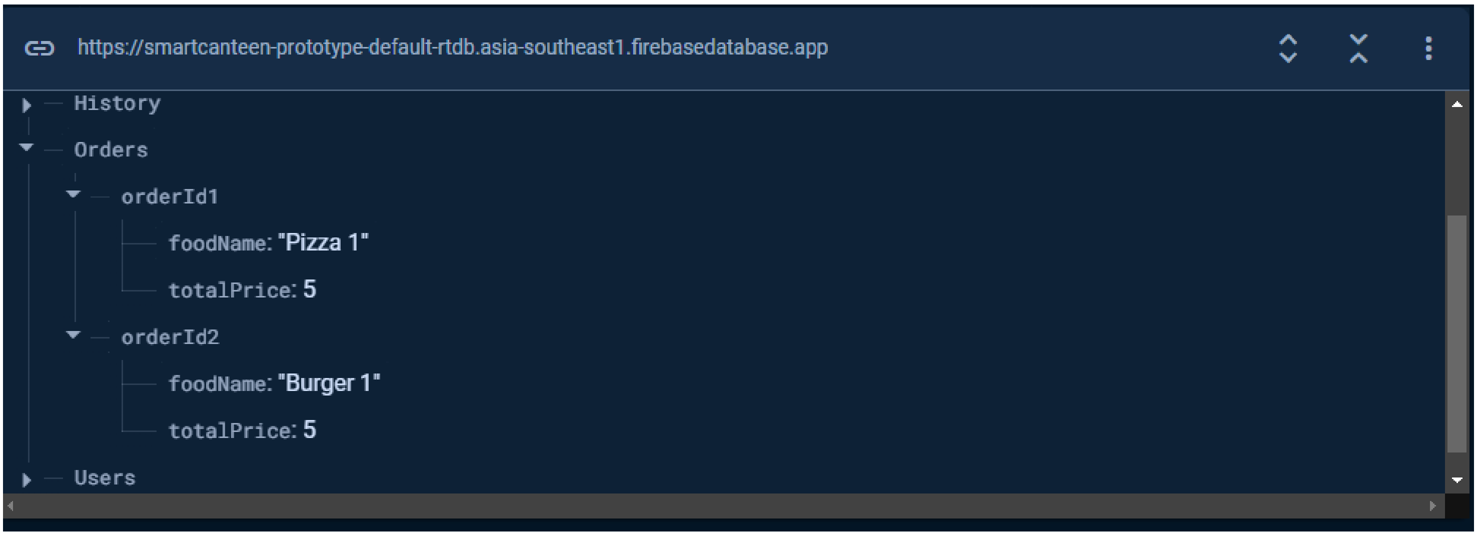
Listing 4.2.2.4: Code snippet of CartFragment.class

The PaymentOptionActivity configuration comes after the CartFragment activity, in which users review their selected items. Listing 4.2.2.5 shows a code snippet for PaymentOptionActivity. By enabling this, users are presented with two payment options: cash or online payment. When the user clicks the "Send Order" button, the order details, including the food name and total price, are sent to Firebase and stored in the Orders node. This node organises orders by unique order IDs, with each order containing fields such as foodname and totalPrice. A screenshot of the Orders node in the Firebase database is provided below in figure 4.2.2.1 to demonstrate how the data is organised and stored.

```
public class PaymentOptionActivity extends AppCompatActivity {


    private Button btnCash, btnOnlinePayment, btnSendOrder;
    private ProgressDialog loadingDialog;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_payment_option);

        btnCash = findViewById(R.id.btnPayByCash);
        btnOnlinePayment = findViewById(R.id.btnOnlinePayment);
        btnSendOrder = findViewById(R.id.btnSendOrder);

        // Show loading dialog
        loadingDialog = new ProgressDialog(this);
        loadingDialog.setMessage("Processing your order...");
        loadingDialog.setCancelable(false);

        // Cash Payment Option
        btnCash.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Toast.makeText(PaymentOptionActivity.this, "Paying by Cash",
Toast.LENGTH_SHORT).show();
            }
        });

        // Online Payment Option
        btnOnlinePayment.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(PaymentOptionActivity.this, PaymentActivity.class);
                startActivity(intent);
            }
        });

        // Send Order to Firebase
        btnSendOrder.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Show loading dialog
                loadingDialog.show();

                // Simulate sending order
                btnSendOrder.postDelayed(new Runnable() {
                    @Override
                    public void run() {
                        sendOrderToFirebase();

                        loadingDialog.dismiss();
                        Intent intent = new Intent(PaymentOptionActivity.this, CheckoutActivity.class);
                        startActivity(intent);
                        Toast.makeText(PaymentOptionActivity.this, "Order sent to Vendor", Toast.LENGTH_SHORT).show();
                    }
                }, 2000); // Simulate 2 seconds loading
            }
        });
    }

    private void sendOrderToFirebase() {
        FirebaseDatabase database = FirebaseDatabase.getInstance();
        DatabaseReference ordersRef = database.getReference("Orders");

        // Create unique order ID
        String orderId = ordersRef.push().getKey();

        // Add order details
        Order order = new Order("pizza1", 5.00); // Replace with actual item and price
        ordersRef.child(orderId).setValue(order);
    }
```

```
}
```

Listing 4.2.2.5: Code snippet of PaymentOptionActivity.class



https://smartcanteen-prototype-default-rtdb.asia-southeast1.firebasedatabase.app

```
History
Orders
  orderId1
    foodName: "Pizza 1"
    totalPrice: 5
  orderId2
    foodName: "Burger 1"
    totalPrice: 5
Users
```

Figure 4.2.2.1: Screenshot of Order node from Firebase Database

Lastly comes the OrderActivity.class is essential to the “SmartCanteen” app because it allows vendors to efficiently manage and track incoming customer orders, ensuring smooth communication and timely updates via features such as SMS notifications and real-time order management. Before setting up the OrderActivity.class, the OrderAdapter.class must first be implemented which handles order display in the RecyclerView. The OrderAdapter,class handles the binding of order data (such as food name and total price) to views, such as TextView elements, and includes the ability to mark an order as "Completed." This completion generates an SMS notification for the user. The adapter also includes swipe-to-remove functionality, which lets you remove orders from the list without deleting them from Firebase. Below is a listing which contains code snippets of OrderAdapter.class.

```
public class OrderAdapter extends RecyclerView.Adapter<OrderAdapter.OrderViewHolder> {

    private Context context;
    private ArrayList<Order> orderList;
    private DatabaseReference orderRef;

    public OrderAdapter(Context context, ArrayList<Order> orderList) {
        this.context = context;
        this.orderList = orderList;
        this.orderRef = FirebaseDatabase.getInstance().getReference("Orders");
    }

    @NonNull
    @Override
    public OrderViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(context).inflate(R.layout.order_item, parent, false);
        return new OrderViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull OrderViewHolder holder, int position) {
        Order order = orderList.get(position);
        holder.foodNameTextView.setText(order.getFoodName());
        holder.totalPriceTextView.setText("Price: RM" + order.getTotalPrice());

        // Handle Completed button click
        holder.completedButton.setOnClickListener(v -> {
            String foodName = order.getFoodName();
            String message = "Your order " + foodName + " is ready for pick up";

            // Send SMS to the user's phone number
            SmsManager smsManager = SmsManager.getDefault();
            smsManager.sendTextMessage("+60193837753", null, message, null, null);

            // Notify the user that the SMS has been sent
            Toast.makeText(context, "SMS sent: " + message, Toast.LENGTH_SHORT).show();
        });

        // Handle swipe to remove order from list (not from Firebase)
    }

    @Override
    public int getItemCount() {
        return orderList.size();
    }

    public static class OrderViewHolder extends RecyclerView.ViewHolder {
        TextView foodNameTextView, totalPriceTextView;
        Button completedButton;

        public OrderViewHolder(@NonNull View itemView) {
            super(itemView);
            foodNameTextView = itemView.findViewById(R.id.foodNameTextView);
            totalPriceTextView = itemView.findViewById(R.id.totalPriceTextView);
            completedButton = itemView.findViewById(R.id.completedButton);
        }
    }
}
```

Listing 4.2.2.6: Code snippet of OrderAdapter.class

Once the OrderAdapter has been configured, the OrderActivity.class is set up to display and manage incoming orders. It retrieves order data from Firebase via the Orders node and populates the RecyclerView with a list of orders using the previously defined OrderAdapter.class. The activity also sends SMS notifications to users when their orders are ready and offers swipe functionality to remove orders from the list. In addition, SMS permissions are required to send notifications. Below is a listing which has code snippets of OrderActivity.classes.

```
public class OrderActivity extends AppCompatActivity {


    private RecyclerView recyclerView;
    private OrderAdapter orderAdapter;
private ArrayList<Order> orderList;
private DatabaseReference orderRef;

private static final int SMS_PERMISSION_CODE = 100;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_order);

    // Check and request SMS permission
    checkAndRequestSmsPermission();

    recyclerView = findViewById(R.id.recyclerViewOrders);
    recyclerView.setLayoutManager(new LinearLayoutManager(this));

    orderList = new ArrayList<>();
    orderAdapter = new OrderAdapter(this, orderList);
    recyclerView.setAdapter(orderAdapter);

    // Get orders from Firebase
    orderRef = FirebaseDatabase.getInstance().getReference("Orders");
    orderRef.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            orderList.clear();
            for (DataSnapshot snapshot : dataSnapshot.getChildren()) {
                Order order = snapshot.getValue(Order.class);
                orderList.add(order);
            }
            orderAdapter.notifyDataSetChanged();
        }
```

```
            @Override
            public void onCancelled(@NonNull DatabaseError error) {
                // Handle database error
            }
        });

        // Attach ItemTouchHelper to RecyclerView
        new ItemTouchHelper(itemTouchHelperCallback).attachToRecyclerView(recyclerView);

    }

    private void checkAndRequestSmsPermission() {
        if (ContextCompat.checkSelfPermission(this, Manifest.permission.SEND_SMS)
                != PackageManager.PERMISSION_GRANTED) {
            // Request SMS permission
            ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.SEND_SMS}, SMS_PERMISSION_CODE);
        }
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        if (requestCode == SMS_PERMISSION_CODE) {
            if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
                // Permission granted
                Toast.makeText(this, "SMS Permission granted", Toast.LENGTH_SHORT).show();
            } else {
                // Permission denied
                Toast.makeText(this, "SMS Permission denied", Toast.LENGTH_SHORT).show();
            }
        }
    }

    // Callback for swipe functionality
    ItemTouchHelper.SimpleCallback itemTouchHelperCallback = new ItemTouchHelper.SimpleCallback(0, ItemTouchHelper.RIGHT) {
        @Override
        public boolean onMove(@NonNull RecyclerView recyclerView, @NonNull RecyclerView.ViewHolder viewHolder, @NonNull RecyclerView.ViewHolder target) {
            // We don't need move functionality, so return false
            return false;
        }

        @Override
        public void onSwiped(@NonNull RecyclerView.ViewHolder viewHolder, int direction) {
            // Remove item from the RecyclerView list, but not from Firebase
            int position = viewHolder.getAdapterPosition();
            orderList.remove(position);
            orderAdapter.notifyItemRemoved(position);
        }
    };

}
```

Listing 4.2.2.7: Code snippet of OrderActivity.class

4.2.3 Android Studio Setup

1. Download Android Studio from the official website and start a new project with an empty activity.

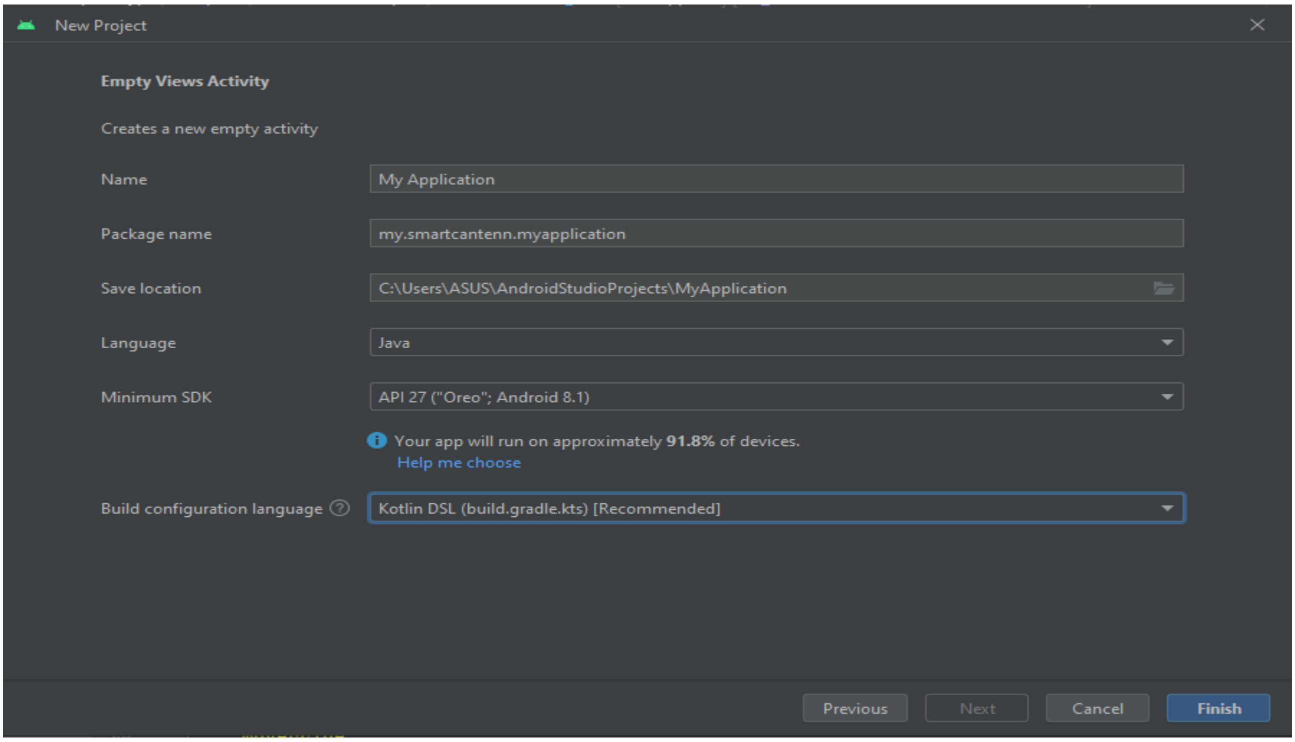
2. Enter your application and package names.

3. Save the project location.

4. Select a programming language, either Kotlin or Java.

5. Select the minimal SDK; the lower the API, the more devices you can support.

6. Click Finish and wait for the project to be created.



New Project

Empty Views Activity

Creates a new empty activity

Name: My Application

Package name: my.smartcantenn.myapplication

Save location: C:\Users\ASUS\AndroidStudioProjects\MyApplication

Language: Java

Minimum SDK: API 27 ("Oreo"; Android 8.1)

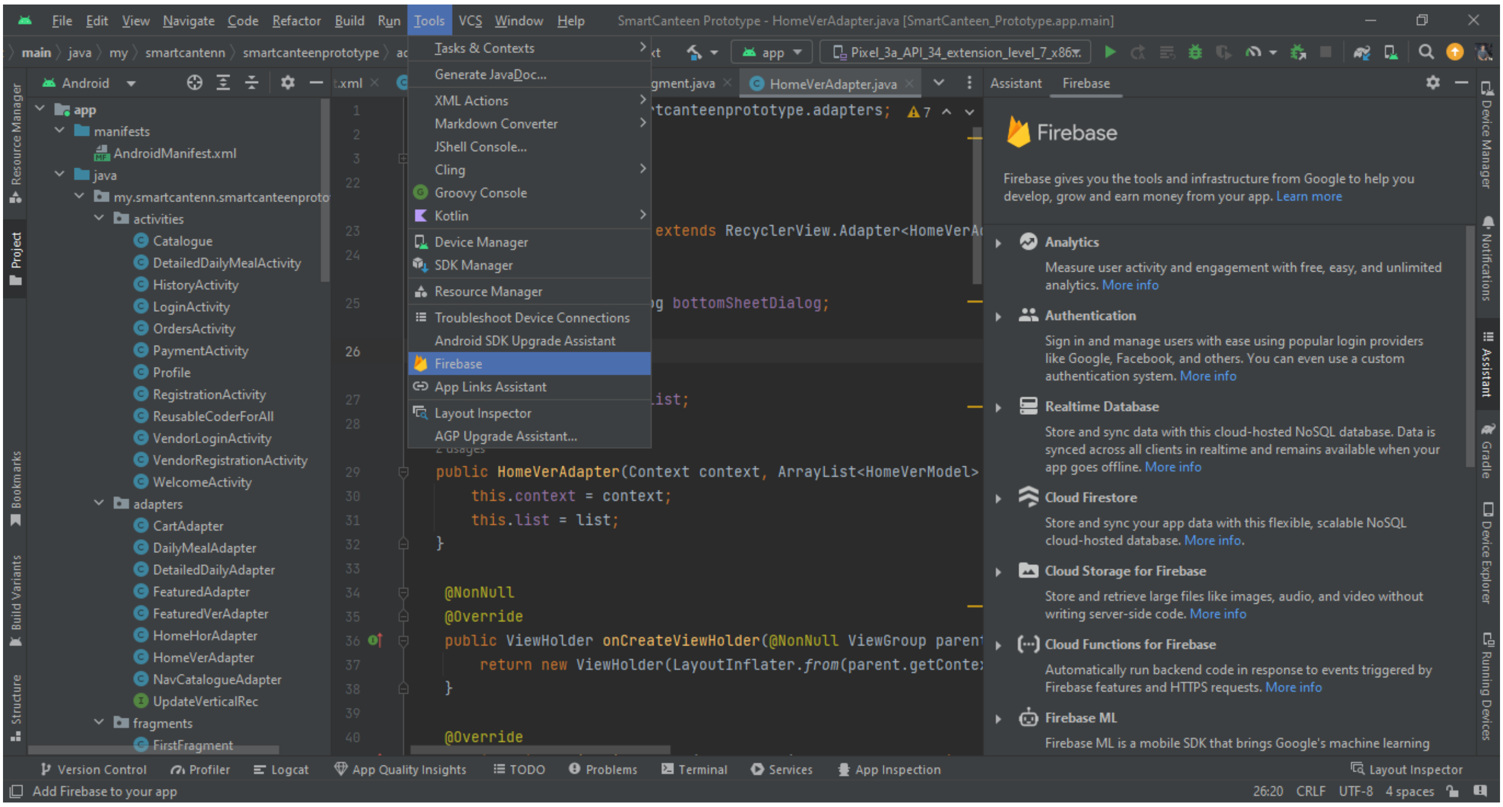
Your app will run on approximately **91.8%** of devices.
Help me choose

Build configuration language: Kotlin DSL (build.gradle.kts) [Recommended]

Figure 4.2.3.1: Creating new project in Android Studio

4.2.3 Firebase Setup in Android Studio

After installing Android Studio, users have two options for connecting their Android app to Firebase. The procedures outlined here use Android Studio Firebase Assistant to configure authentication using Google Sign-in.



```
...tcanteenprototype.adapters;

... extends RecyclerView.Adapter<HomeVerA...

...g bottomSheetDialog;

...ist;

    public HomeVerAdapter(Context context, ArrayList<HomeVerModel>
        this.context = context;
        this.list = list;
    }

    @NonNull
    @Override
    public ViewHolder onCreateViewHolder(@NonNull ViewGroup paren...
        return new ViewHolder(LayoutInflater.from(parent.getConte...
    }

    @Override
```

Firebase

Firebase gives you the tools and infrastructure from Google to help you develop, grow and earn money from your app. Learn more

Analytics

Measure user activity and engagement with free, easy, and unlimited analytics. More info

Authentication

Sign in and manage users with ease using popular login providers like Google, Facebook, and others. You can even use a custom authentication system. More info

Realtime Database

Store and sync data with this cloud-hosted NoSQL database. Data is synced across all clients in realtime and remains available when your app goes offline. More info

Cloud Firestore

Store and sync your app data with this flexible, scalable NoSQL cloud-hosted database. More info.

Cloud Storage for Firebase

Store and retrieve large files like images, audio, and video without writing server-side code. More info

Cloud Functions for Firebase

Automatically run backend code in response to events triggered by Firebase features and HTTPS requests. More info

Firebase ML

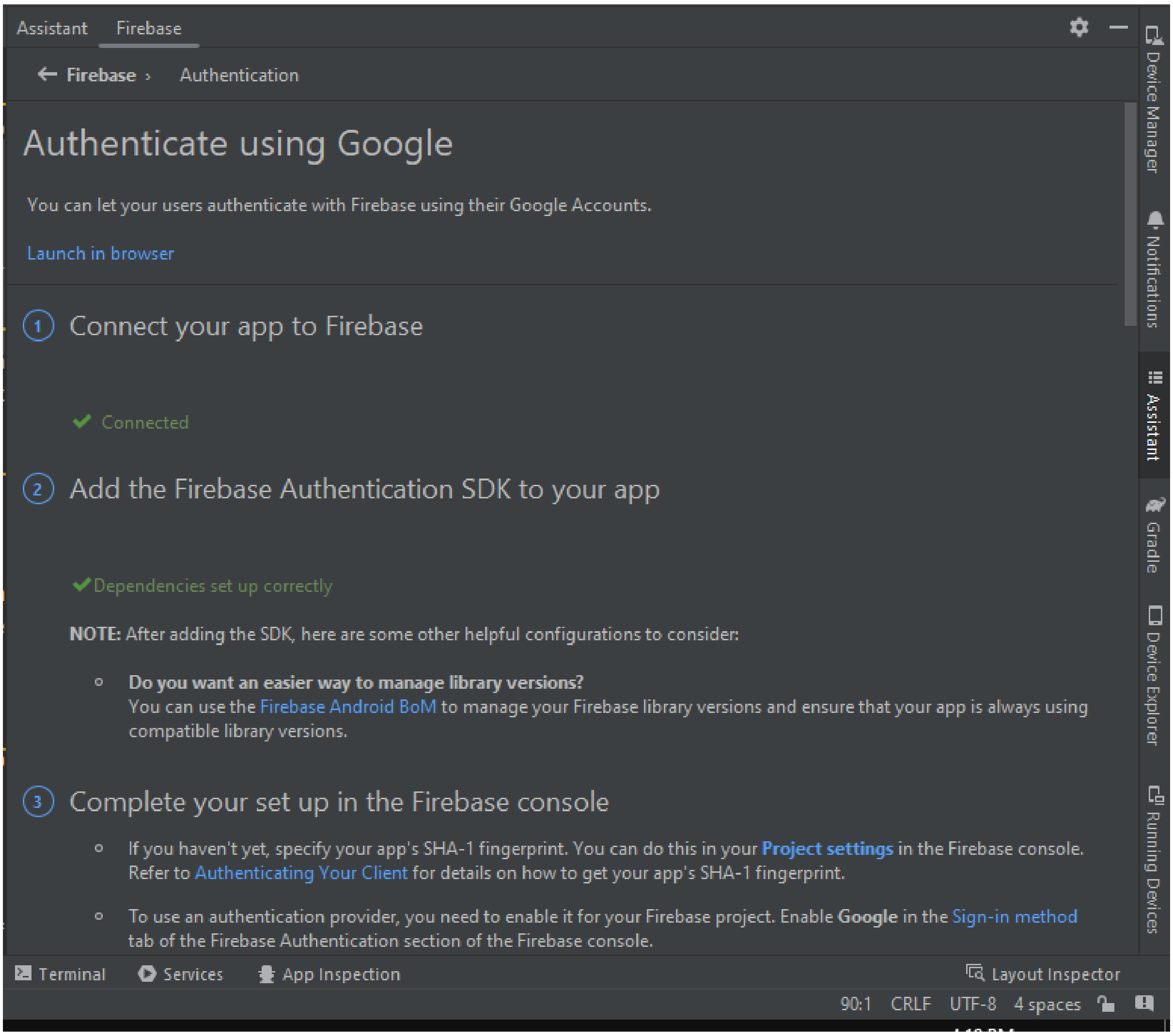
Firebase ML is a mobile SDK that brings Google's machine learning

Figure 4.2.3.2: Steps for connecting Firebase with Android Studio

1. Select Tools → Firebase to open an aid window on the left side.

2. Services selected for this project include authentication for login and sign up, analytics for measuring user activity and engagement, Cloud Firestore and Realtime Database for data storage and retrieval, and Cloud Storage for huge files.

3. Follow the procedures outlined in the assistance, and the application will now be able to gather the user's email address and password, as well as monitor application activities.



Authenticate using Google

You can let your users authenticate with Firebase using their Google Accounts.

Launch in browser

1 Connect your app to Firebase

✓ Connected

2 Add the Firebase Authentication SDK to your app

✓ Dependencies set up correctly

NOTE: After adding the SDK, here are some other helpful configurations to consider:

- **Do you want an easier way to manage library versions?** You can use the Firebase Android BoM to manage your Firebase library versions and ensure that your app is always using compatible library versions.

3 Complete your set up in the Firebase console

- If you haven't yet, specify your app's SHA-1 fingerprint. You can do this in your **Project settings** in the Firebase console. Refer to Authenticating Your Client for details on how to get your app's SHA-1 fingerprint.
- To use an authentication provider, you need to enable it for your Firebase project. Enable **Google** in the Sign-in method tab of the Firebase Authentication section of the Firebase console.

Figure 4.2.3.3: Steps provided by the assistant for using Google Sign-In authentication in the project.

4.3 Verification Plan (Testing)

Testing evaluates a system's performance against its stated objectives and capabilities. Black box testing was chosen for this project. This testing approach focusses on the system's input and output. Black box testing validates the system against predefined requirements.

There are several benefits to black box testing:

A. It works for broader coverage.

B. Test cases can be generated prior to development and immediately following specification.

C. Defects and inconsistencies can be identified during the initial stages of testing.

First, test cases and procedures for unit and integration will be developed based on requirements specifications and software component identification. The report will include the results of the test cases and procedures. Test cases can be divided into two types:

1. Testing the system's functionality with the correct input demonstrates that it works as expected.
2. Testing the system with incorrect input to ensure that it will function properly without failure or error.

Test cases are an example.

Test case module: Login using email module unit testing.

Test description: To test the input of email and password field.

Test steps	Step Description	Data Utilized	Expected Result	Actual Result	Pass/Fail /Haven't
1	Test a valid email and password email:kavirajmaniveel@gmail.com password: 12345678	Email and password are registered in Firebase	Login successful and go to the next page	Redirect new or existing user to the main menu	-
2	Test a valid email but invalid password Actual Data: email: rajmani@yahoo.com password: 12345678	Email is registered in the database.	Display error: Incorrect email or password entered	"Incorrect email or password entered"	-
3	Test an invalid email and password Actual Data: email:kavirajmaniveel@yahoo.com password: 12345678	Email is not registered in the database.	Display error: Invalid email format.	"Email is not registered yet"	-
…	…	…	…	…	

Table 4.3.1 Example of a Test Case

Chapter 5 System Implementation

5.1 Introduction

This chapter will discuss about tools used and system requirements used and the preliminary work which have been completed in this semester. Thus, each module or function will be described in details and label with screenshot picture.

5.2 Tool Involved:

Tools	Description
Software: Java programming in Android Studio.	The “SmartCanteen” app will be created in Android Studio with Java, a free and simple programming environment for Android apps.
Google Firebase	Google Firebase is an open-source framework for building mobile apps. Users' authentication and data will be saved to the Firebase Realtime database.
Google Analytics	Google Analytics is used to evaluate the application's performance and user engagement, allowing for data-driven decisions regarding future improvements and optimisations.
Curlec by RazorPay	Curlec by RazorPay provides a streamlined payment procedure, allowing users to make payments easily while also giving vendors with capabilities for managing payment transactions and seeing payment history.

Table 5.2.1: Tools that will be used to develop this project.

5.3 System Requirements

Tables 5.3.1 and 5.3.2 provide detailed software and hardware requirements for other researchers to understand the minimum needs for developing and running a pet location monitoring system.

Software Requirement	
Operating system	Windows 10
Android Studio	Version 4.4.1
Android Platform Version	API Level 29
Firebase	Authentication Cloud FireStore Realtime Database Storage

Table 5.3.1: Software Requirements

Computer Hardware Requirements	
Operating system	64 bits
RAM	8 GB
Hard disc storage	256 GB

Table 5.3.2: Computer Hardware Requirements

5.4 User Interface of Application

5.4.1 Splash Screen

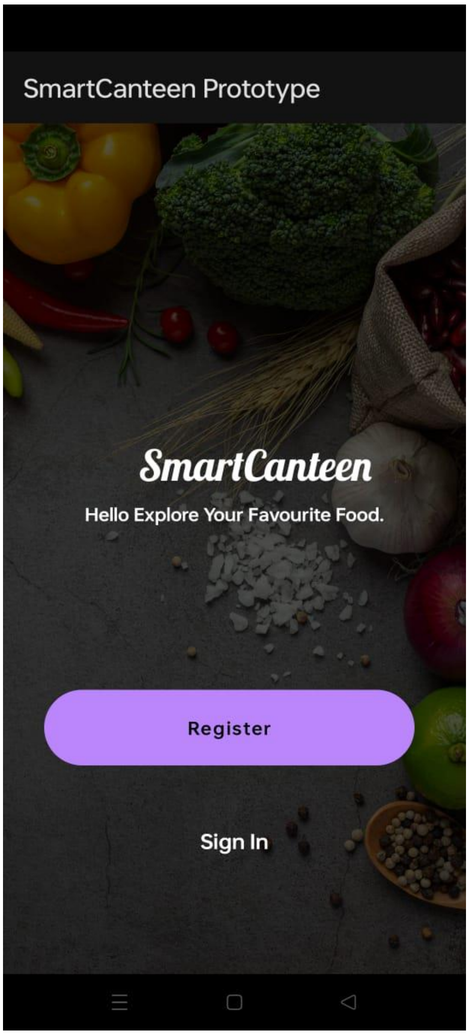


Figure 5.4.1.1: Splash Screen.

Figure 5.4.1.1 shows the proposed system's splash screen. When users open the application, they will see a splash screen with the name of the application. The next step is for users to choose between signing in with an existing account or creating a new account.

5.4.2 Login and Sign-Up Screen Module

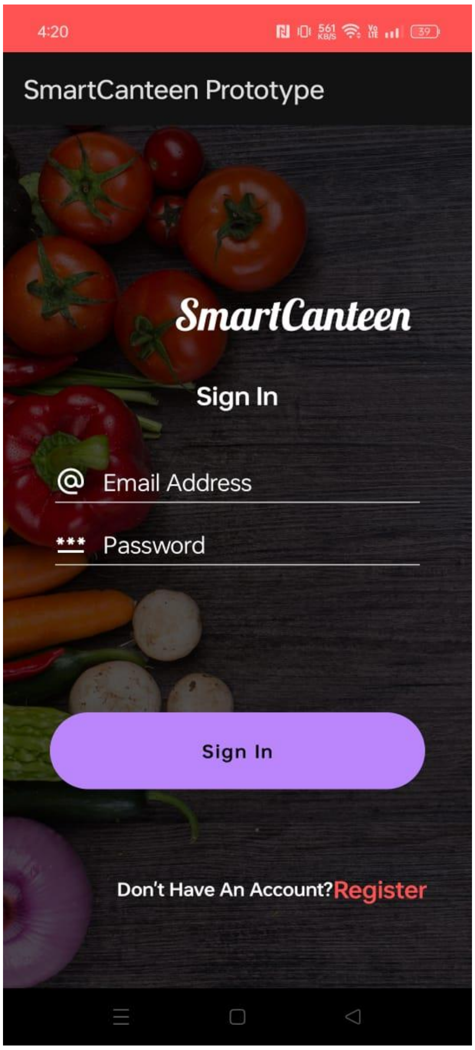


Figure 5.4.2.1: Login Page.

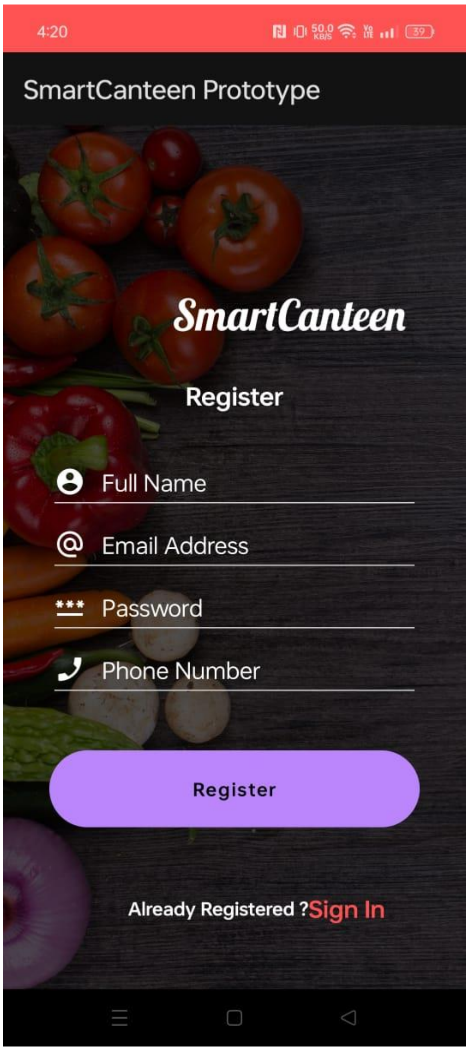


Figure .4.2.2: Register Page.

Following that, users can sign in by email. New users can click the register button, which will take them to Figure 5.4.2.2. This module uses Firebase Authentication. Users can establish a new account by entering the email address and password shown in Figure 5.4.2.1.

5.4.3 Home page Module

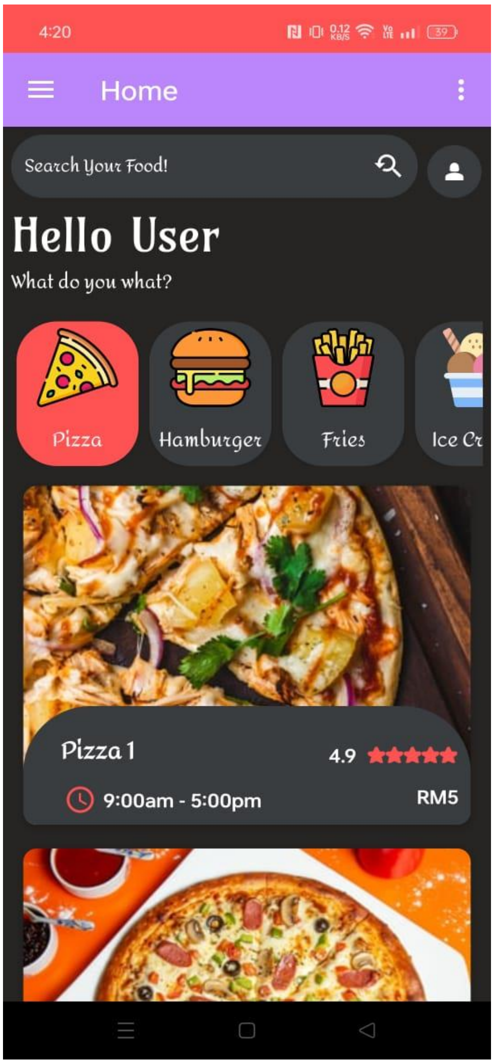


Figure 5.4.3.1: Home Page.

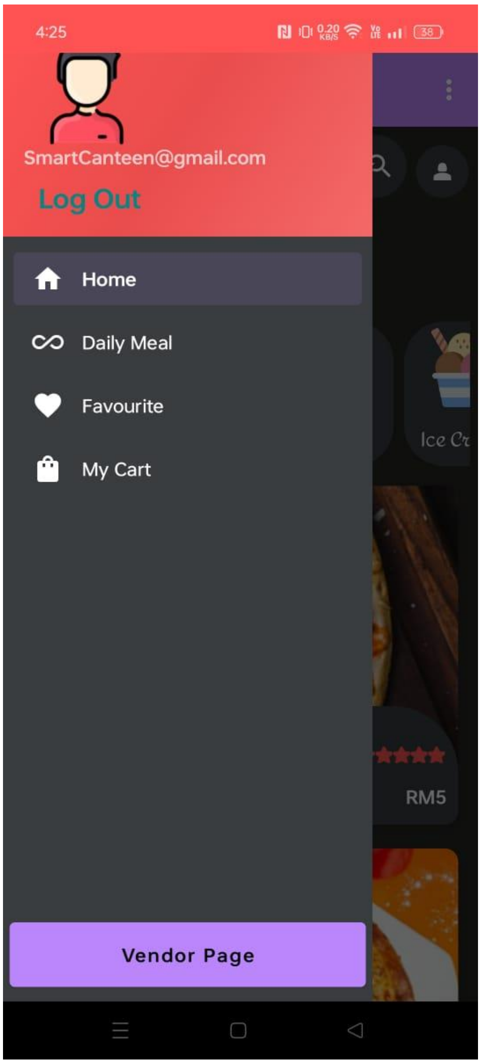


Figure 5.4.3.2: Navigation panel.

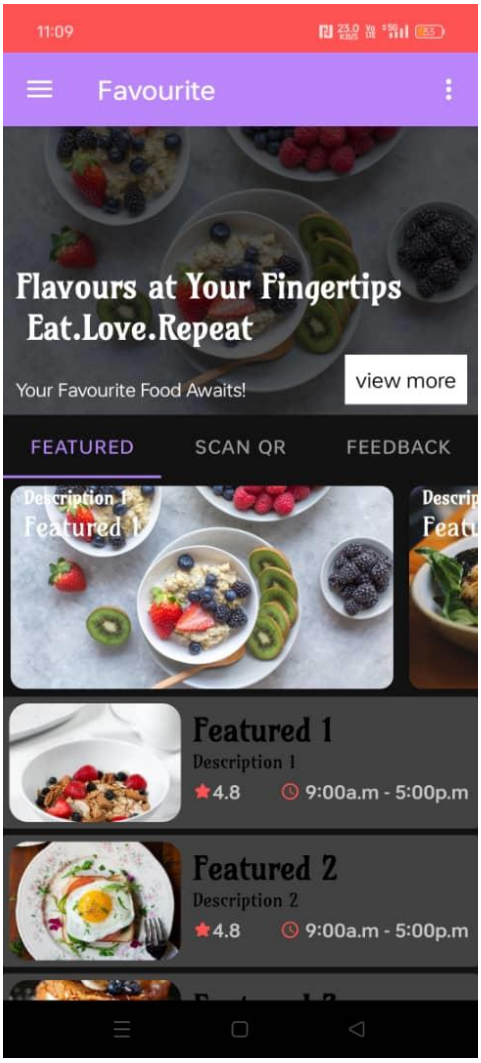


Figure 5.4.3.3: Favourite Page

In figure 5.4.3.1, users can browse the menu and choose their preferred food on the home page. The side navigation bar allows users to view daily meals, favourites with featured dishes, and the main page. In figure 5.4.3.3, users not only can view featured dishes but also provide feedback and scan QR code upon collecting their meal. Users can also connect as vendors via the vendor page button.

5.4.4 View Cart Module

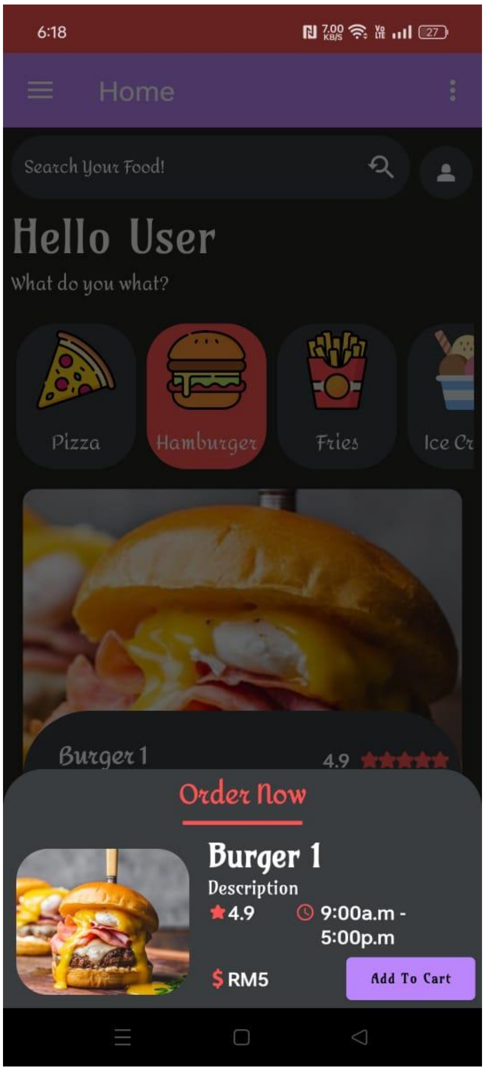


Figure 5.4.4.1: Add to cart feature.

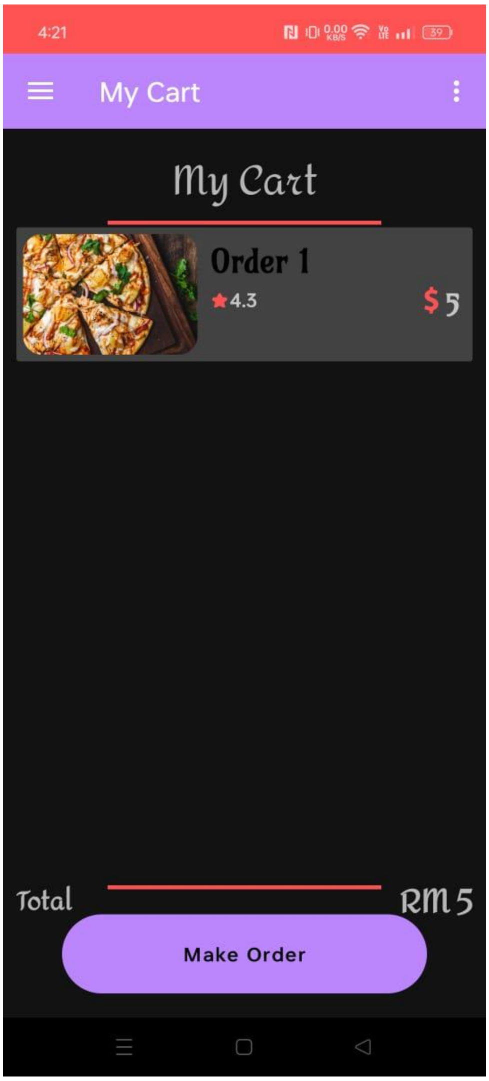


Figure 5.4.4.2: View of cart.

The "Add to cart" function in figure 5.4.4.1 allows users to easily check and alter their selected items before placing their order. With a rapid overview of their chosen goods, including name, description, and price, users can easily adjust or remove orders as needed, ensuring that their order truly reflects their preferences.

5.4.5 Cart Module

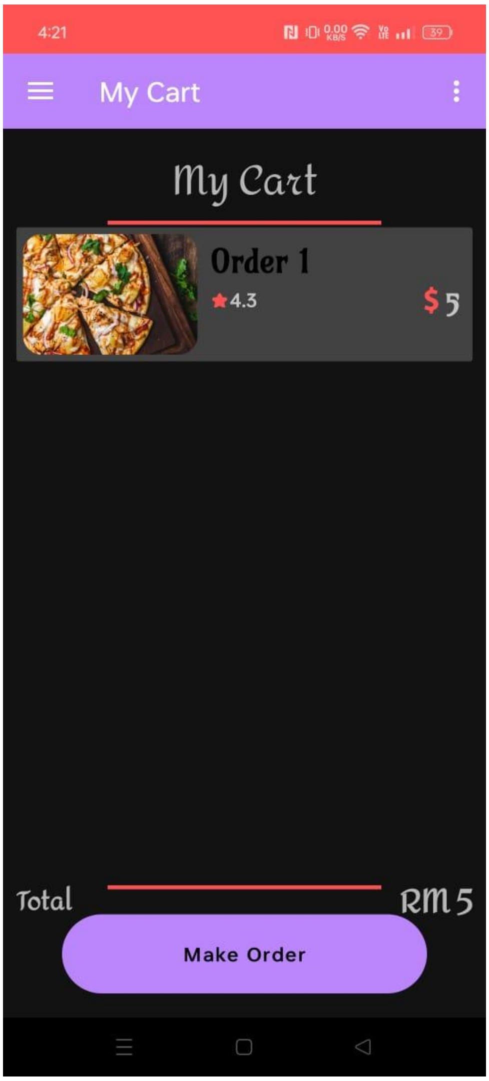


Figure 5.4.5.1: View of cart.

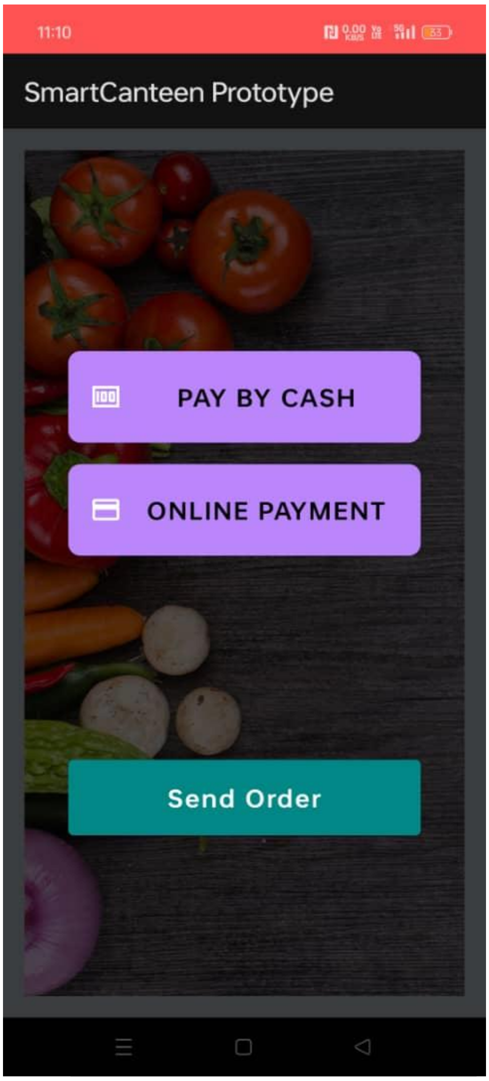
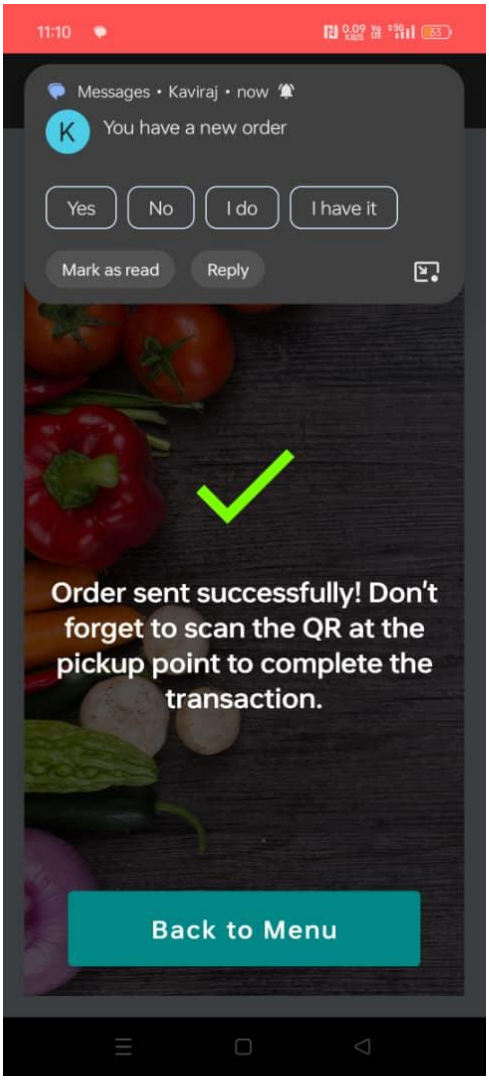


Figure 5.4.5.2: Checkout page.


Messages • Kaviraj • now

You have a new order

Order sent successfully! Don't forget to scan the QR at the pickup point to complete the transaction.

Figure 5.4.5.3: Order Conformation page.

Users can view their cart in Figure 5.4.5.1, which shows a list of items along with their order's picture, name, rating, and price. There is also a "Make Order" button, which, when clicked, takes users to the checkout page. Figure 5.4.5.2 shows how users can proceed to the checkout page and select their preferred payment method, either cash or online payment. The page also includes a "Send order" button, which, when clicked, confirms the order and advances to the final step. Figure 5.4.5.3 shows the Order Confirmation page, which informs users that their order has been successfully sent to the vendor. Additionally, the vendor receives an SMS notification stating, "You have a new order."

5.4.6 Connect as Vendor Module

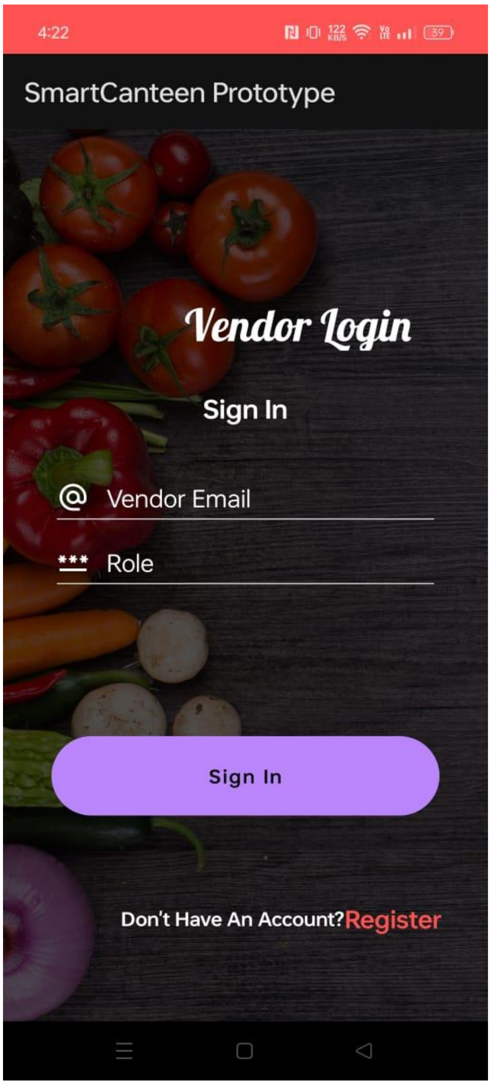


Figure 5.4.6.1: Vendor Login Page

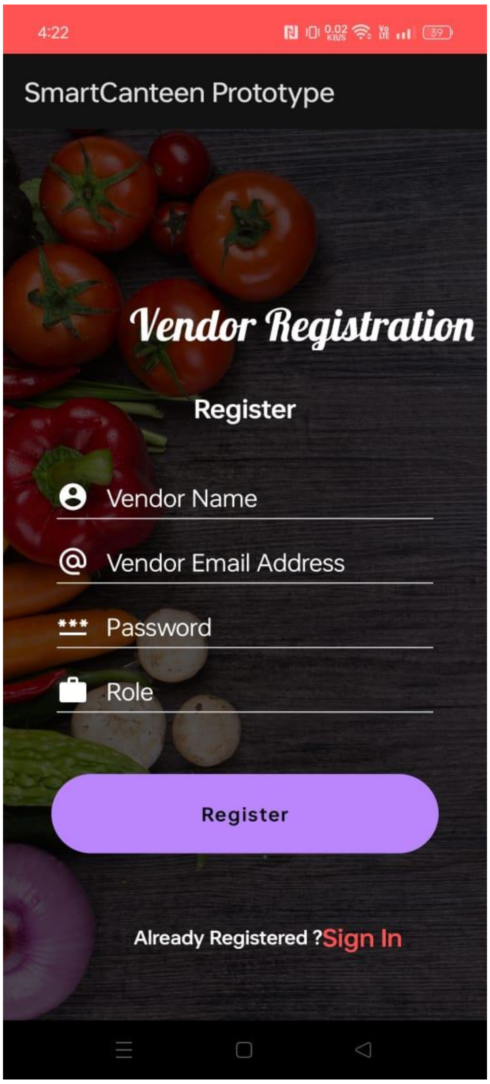


Figure 5.4.6.2: Vendor Registration Page

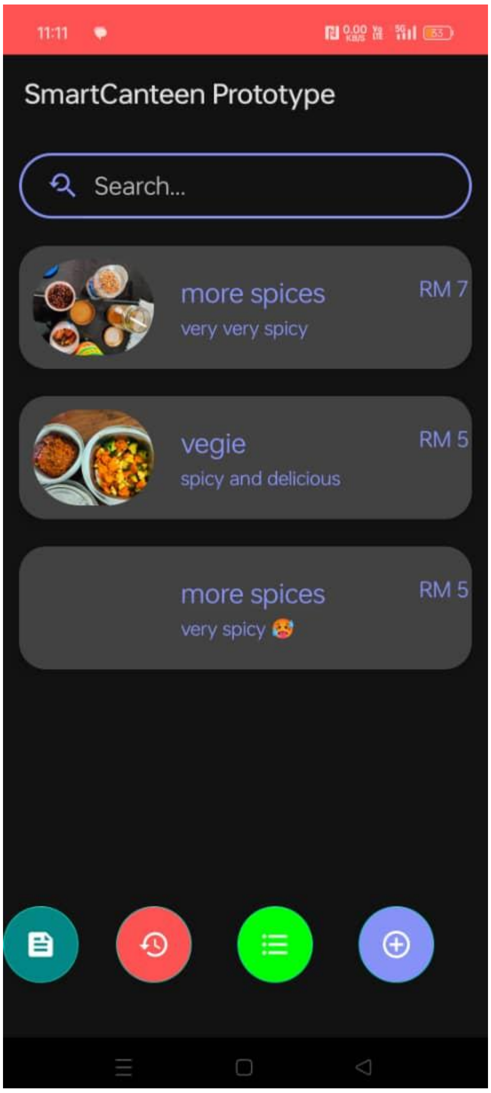
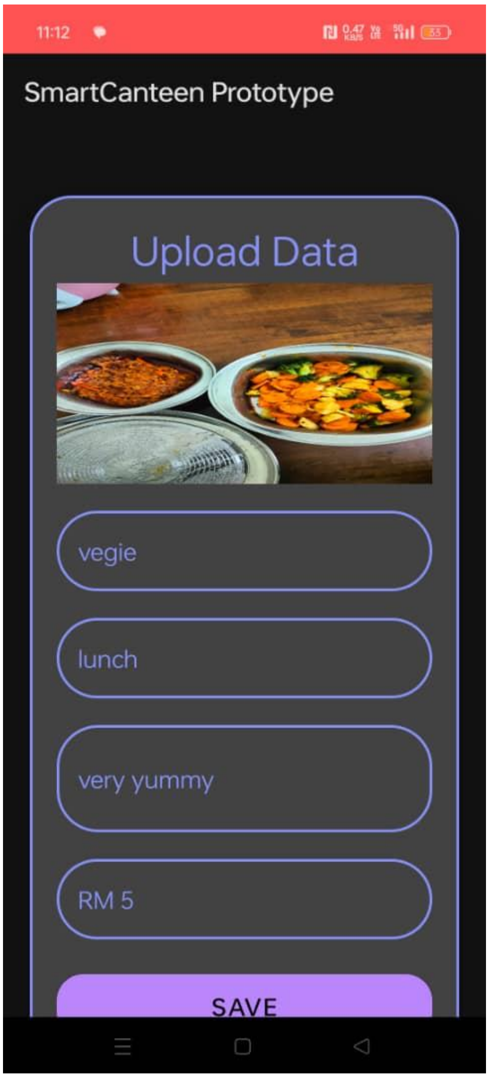


Figure 5.4.6.3: Vendor Home Page

The vendor login page in figure 5.4.6.1 serves as the portal's initial entry point, needs vendors to enter their registered email and role credentials to gain secure access. The vendor registration page in figure 5.4.6.2, on the other hand, quickens process of gathering necessary information such as the vendor's name, email address, password, and system role. When vendors log in, they are directed to the vendor home page in figure 5.4.6.3, it shows a list of food items uploaded by the vendor. There are four buttons below that provide key functionality. The dark green button opens the feedback section, which allows vendors to view user feedback. The red-green button allows vendors to view their order history and download it as an.xlsx file, and the generate QR code function is also available here. The light green button displays incoming orders, while the violet button allows vendors to add a new dish to the menu.

5.4.7 Add, Edit and Remove Dish Modules



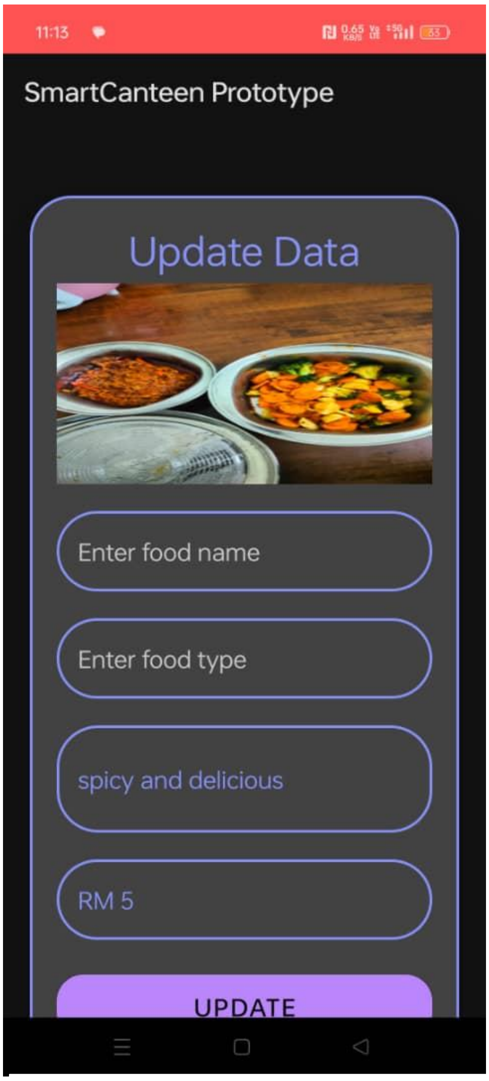
vegie

lunch

very yummy

RM 5

Figure 5.4.7.1: Add Dish Page



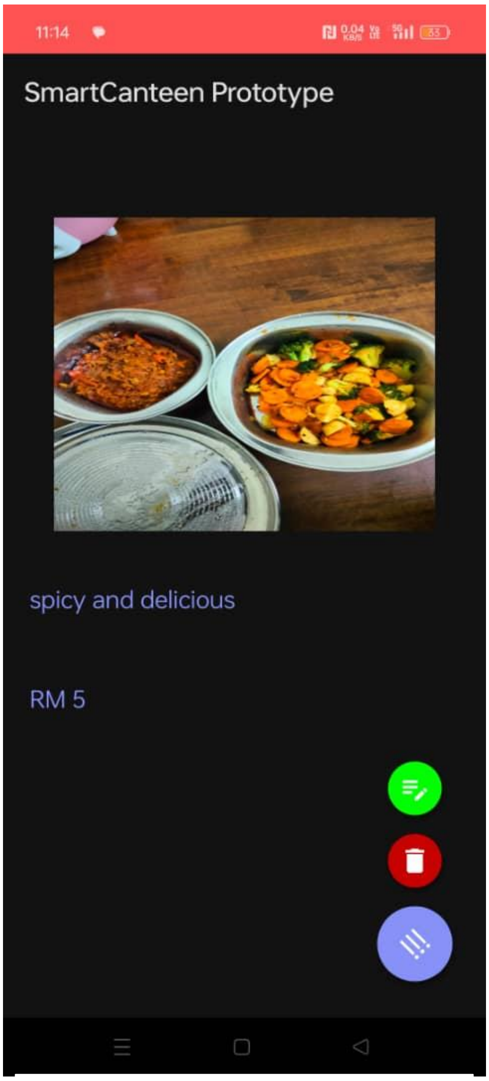
Enter food name

Enter food type

spicy and delicious

RM 5

Figure 5.4.7.2: Edit Dish Page



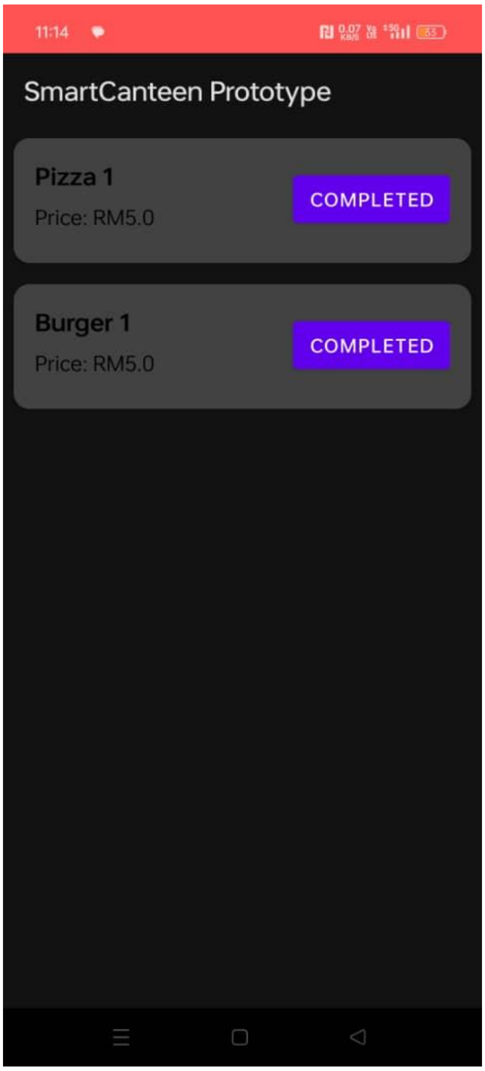
spicy and delicious

RM 5

Figure 5.4.7.3: Remove / Edit Dish Functionality

Figure 5.4.7.1 shows the Add Dish page, which allows vendors to upload a dish by entering details such as the dish name, type (breakfast or lunch), description, and price, as well as uploading an image of the dish. When the "Save" button is pressed, the dish is uploaded to the Firebase database and made available for users to view within the app. Figure 5.4.7.2 depicts the Edit Dish page, which allows vendors to update any dish-related information, including the picture, description, and price, with changes reflected in Firebase. Finally, in Figure 5.4.7.3, the Remove/Edit Dish functionality offers two options. The green button allows vendors to edit the dish, whereas the red button, when clicked, removes the dish entirely from the app and Firebase.

5.4.8 Orders Module


SmartCanteen Prototype

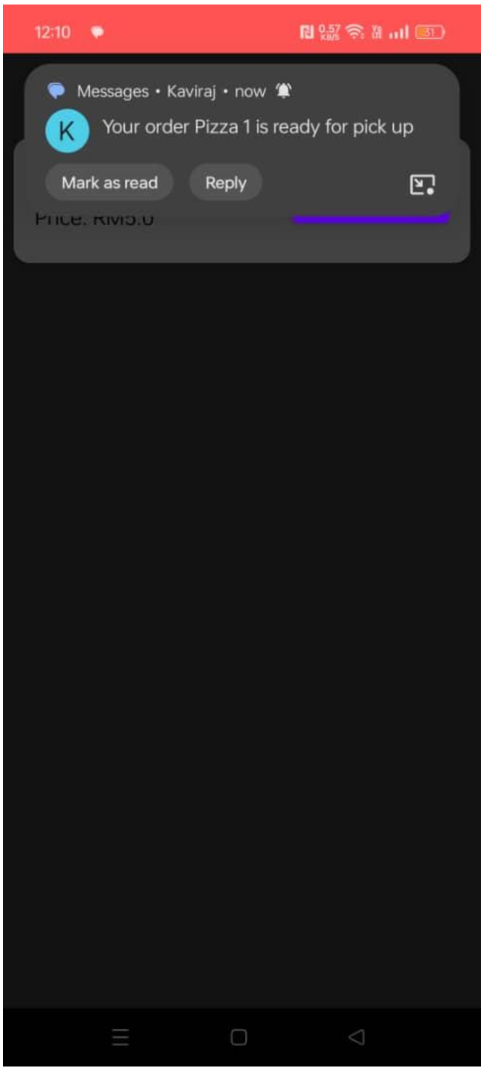
Pizza 1

Price: RM5.0

Burger 1

Price: RM5.0

Figure 5.4.8.1: Incoming Orders Page

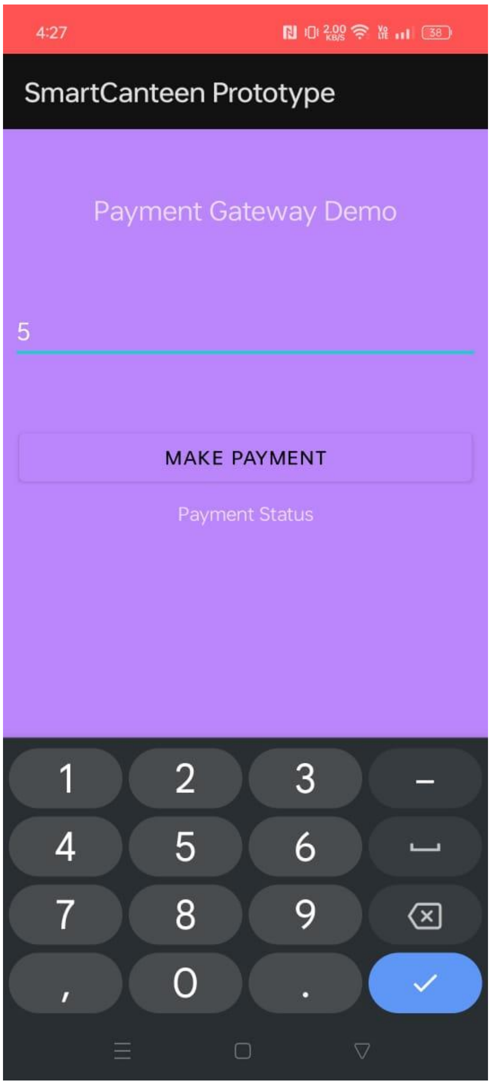

Messages • Kaviraj • now

Your order Pizza 1 is ready for pick up

Figure 5.4.8.2: Order Completion Functionality

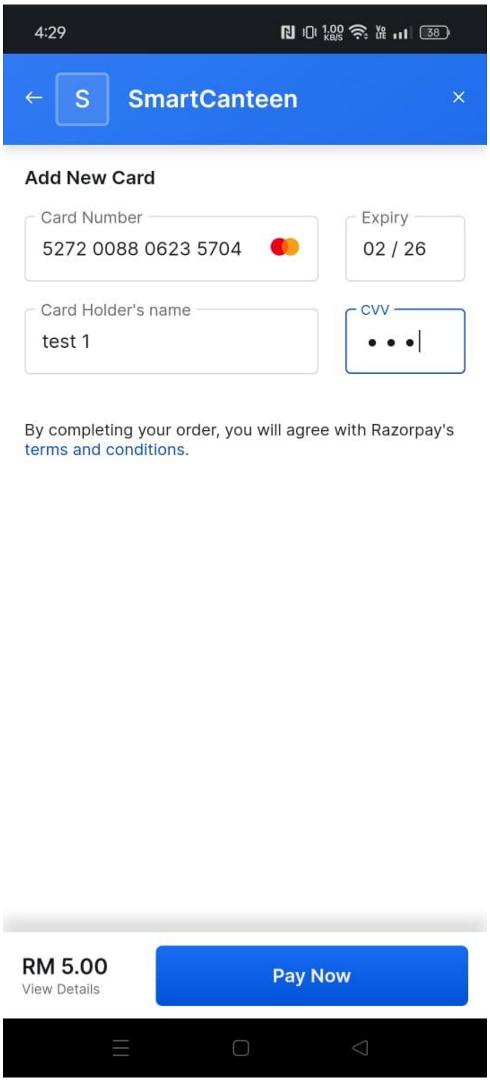
Figure 5.4.8.1 depicts the Incoming Orders page, which displays a list of orders that are sent to the Firebase database and then sent to vendors. Every order has a "Completed" button next to it. Figure 5.4.8.2 shows how the Order Completion functionality is activated when the vendor clicks the "Completed" button. The order is removed from both the list and Firebase, and the user receives an SMS notification that reads, "Your order is ready for pickup."

5.4.9 Payment Gateway Module



5

Figure 5.4.9.1: Payment Page



Card Number: 5272 0088 0623 5704

Expiry: 02 / 26

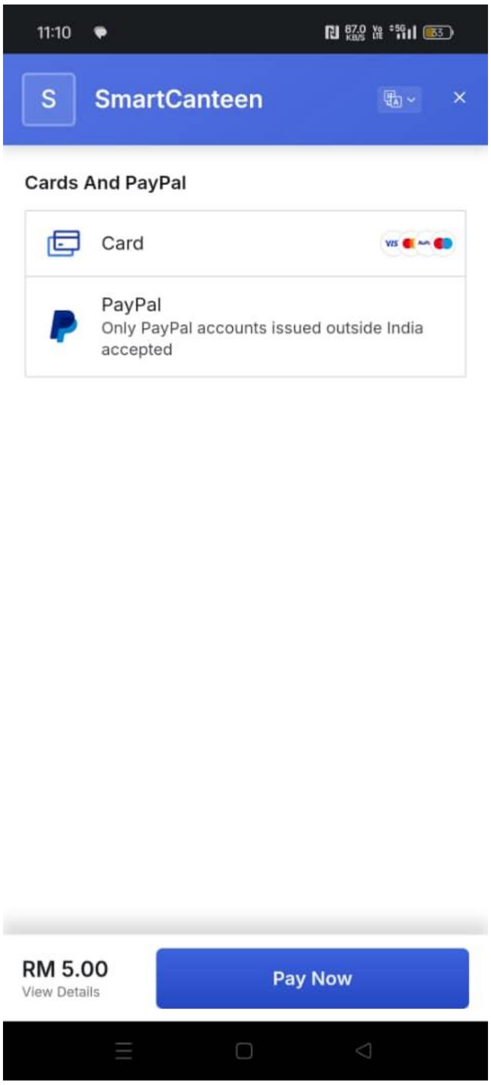
Card Holder's name: test 1

CVV: •••

By completing your order, you will agree with Razorpay's terms and conditions.

RM 5.00

Figure 5.4.9.2: Payment Option Page



Only PayPal accounts issued outside India accepted

RM 5.00

Figure 5.4.9.3: Payment Detail

Figure 5.4.9.1 shows the payment screen, where users can enter the amount, they want to pay, giving them freedom and control over their purchase. Figure 5.4.9.2 shows the succeeding payment option page, which provides a variety of payment methods, including debit and credit card payments and Paypal, to accommodate a wide range of user preferences. Finally, the payment details page in figure 5.4.9.3 allows users to enter their bank card information safely while leveraging Curlec by Razorpay's rigorous security standards. This payment gateway solution offers a smooth and dependable payment process, improving the overall user experience and allowing for hassle-free transactions inside the “SmartCanteen” ecosystem.

5.5 Conclusion

This chapter had discussed about the tools used and software used, system requirements and the user interface and explanations and descriptions of each function are documented and label with screenshot pictures.

Chapter 6: System Evaluation and Discussion

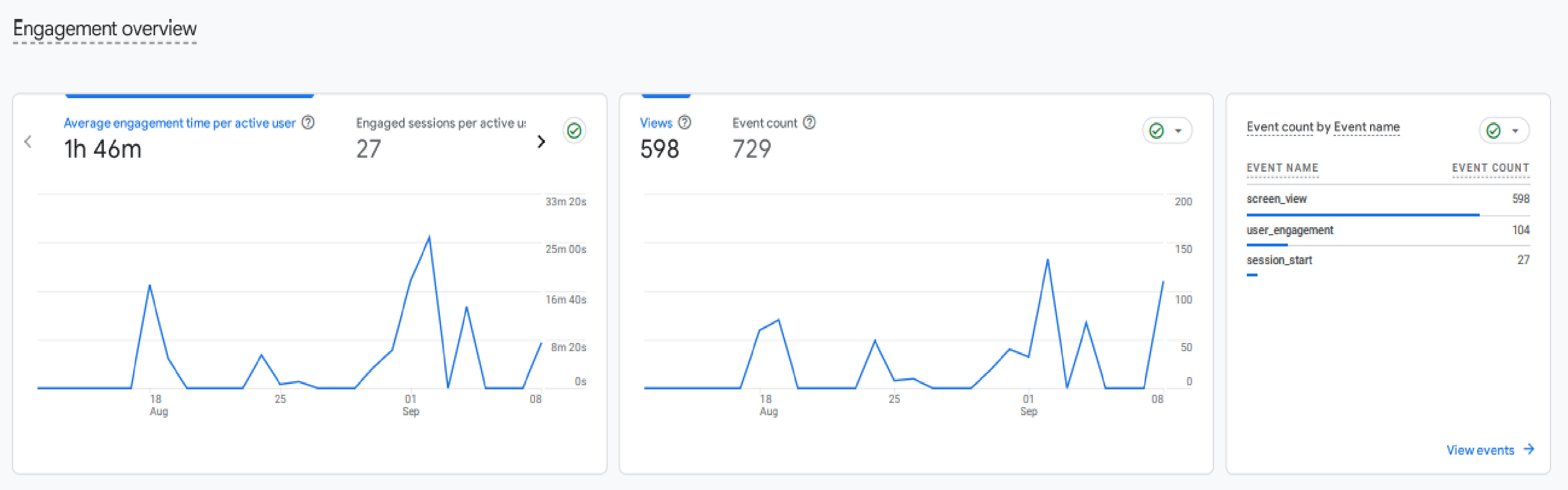
6.1 Introduction

This chapter will discuss the results of system testing and the performance metrics for the “SmartCanteen” mobile application. Next, each module's test cases will be documented. The results and analysis from the user feedback survey will also be discussed. In addition, the project timelines for FYP 1 and FYP 2 will be examined, and the current semester's planning will be detailed.

6.2 System Testing and Performance Metrics

In this section, the performance metrics are gathered during system testing, with a focus on user engagement and the overall application activity. The data is obtained from Google Analytics, which provides insights into user behaviour and the effectiveness of various features.

6.2.1 Engagement Overview



Engagement overview

Average engagement time per active user 1h 46m

Engaged sessions per active u: 27

Views 598

Event count 729

Event count by Event name

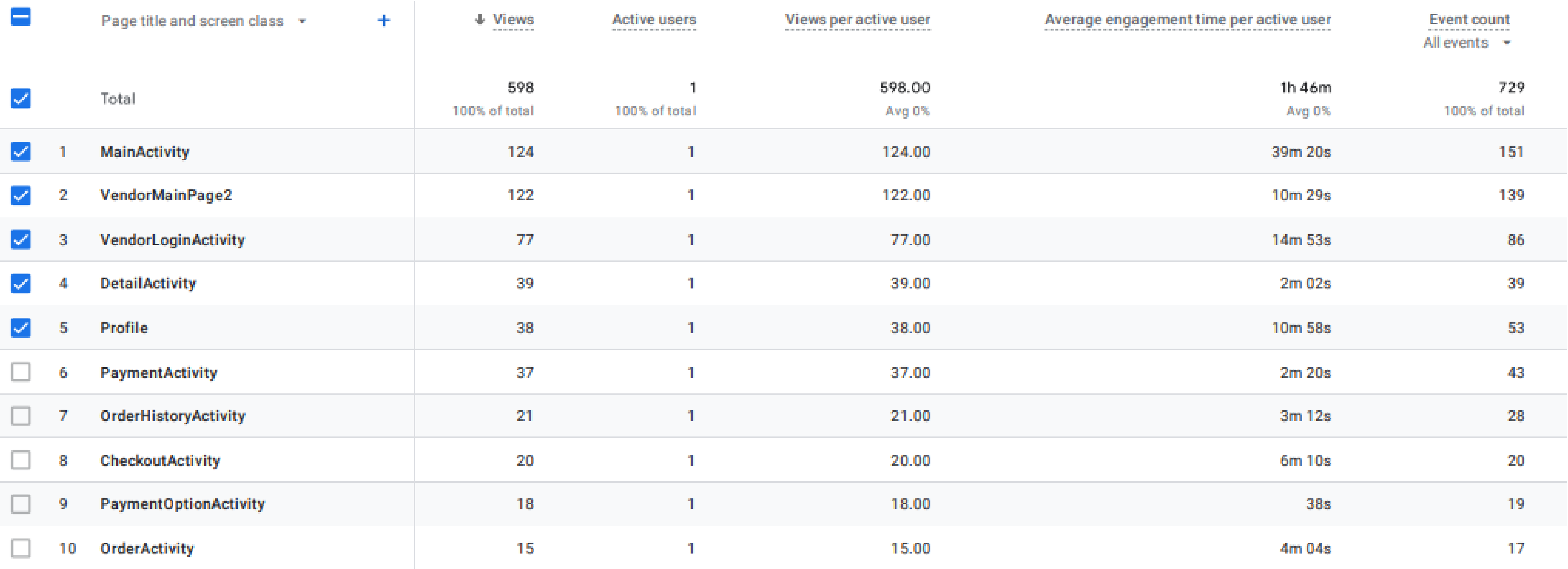
EVENT NAME	EVENT COUNT
screen_view	598
user_engagement	104
session_start	27

View events

Figure 6.2.1.1: Engagement Overview of "SmartCanteen" Application

Figure 6.2.1.1 depicts user engagement rates, with an average engagement duration per active user of 1 hour and 46 minutes. This measure indicates that users are spending a significant amount of time within the application, indicating that the application successfully maintains user interest throughout several sessions. With 27 engaged sessions per active user, it is obvious that users are returning to the app, most likely because they value the content and features enough to interact several times. The 729 total event count across 598 views demonstrates active participation. This implies users are interacting with the app's many features, whether through button pushes, form submissions, or other in-app actions.

6.2.2 Page Views and Event Counts



⊟		Page title and screen class ▾ +	↓ Views	Active users	Views per active user	Average engagement time per active user	Event count All events ▾
☑		Total	598 100% of total	1 100% of total	598.00 Avg 0%	1h 46m Avg 0%	729 100% of total
☑	1	MainActivity	124	1	124.00	39m 20s	151
☑	2	VendorMainPage2	122	1	122.00	10m 29s	139
☑	3	VendorLoginActivity	77	1	77.00	14m 53s	86
☑	4	DetailActivity	39	1	39.00	2m 02s	39
☑	5	Profile	38	1	38.00	10m 58s	53
□	6	PaymentActivity	37	1	37.00	2m 20s	43
□	7	OrderHistoryActivity	21	1	21.00	3m 12s	28
□	8	CheckoutActivity	20	1	20.00	6m 10s	20
□	9	PaymentOptionActivity	18	1	18.00	38s	19
□	10	OrderActivity	15	1	15.00	4m 04s	17

Figure 6.2.2.1: Page Views and Event Counts of "SmartCanteen" Application

Based on figure 6.2.2.1, the Page Views and Event Counts section shows an overview of user activity across several screens. This detailed analysis shows which sections of the app are most frequently visited by users, how long they spend on each screen and the level of interaction taking place within those pages.

The data shows the following trends:

- MainActivity and VendorMainPage2 are the most viewed screens, with 124 and 122 views, respectively. These activities have the highest engagement time, with MainActivity lasting 39 minutes and 20 seconds and VendorMainPage2 lasting 10 minutes and 29 seconds. This indicates that these pages are essential to the user experience, most likely serving as entry points or crucial functioning screens.
- VendorLoginActivity has 77 views and an engagement duration of 14 minutes and 53 seconds, indicating that people are effectively accessing the system using the vendor login option. The high engagement time suggests that consumers spend a large amount of time on this screen, possibly owing to the login or authentication processes.
- Profile activity had 38 views, with an average engagement time of 10 minutes and 58 seconds, indicating that users devote significant effort to managing their personal information. This may highlight the significance of profile customisation or account management features in the user journey.
- DetailActivity and PaymentActivity had much lower engagement times of 2 minutes 2 seconds and 2 minutes 20 seconds, respectively despite having high view counts. This suggests that users are moving swiftly across these screens, indicating efficient functionality or locations where more engagement could be improved.
- OrderHistoryActivity, CheckoutActivity, and OrderActivity all received fewer views and had shorter interaction periods. Despite their modest utilisation, these features are still important to the user experience, notably for tracking orders and completing transactions. Optimising these actions for increased user retention and efficient processes may improve the entire application experience.

In conclusion, the system testing data show that the application performs effectively in terms of user engagement, with key activities creating significant interaction. The data contributes to continuous optimisation efforts to improve lesser-used features while preserving the core features' strengths.

6.3 Test Cases for Completed Module

The test cases for each module will be documented as follows. Each module will test inputs, buttons, functions, pages, and expected results from database operations. The testing was done in a controlled environment, with both actual devices and emulators. Functional testing ensured that all functionality, including menu browsing and order tracking, was performed properly. The usability testing was done with a group of students and faculty provided input on the user interface and overall experience. The results showed that the system maintained acceptable reaction times and reliability in normal usage scenarios such as ordering multiple orders simultaneously.

6.3.1 Login module

Test case module: Login Module unit testing

Test Description: To test Login by using email and password

Test steps	Step Description	Data Utilized	Expected Result	Actual Result	Pass/Fail /Haven't
1	Test a valid email and password email:kavirajmaniveel@gmail.com password: 12345678	Email and password are registered in Firebase	Login successful and go to the next page	Redirect new or existing user to the main menu	Pass
2	Test a valid email but invalid password	Email and password are registered in Firebase	Display error: Sorry invalid email and password.	"Incorrect email or password entered"	Pass
3	Test an invalid email	-	Display error: Invalid email format.	"Email is not registered yet"	Pass
4	Email text and fields empty and click "Login" button	-	Display error: Please fill in all the empty values.	Display error: Please fill in all the empty values.	Pass

Table 6.3.1.1 Login Module unit testing

6.3.2 MainActivity Module

Test Case Module: Unit Testing for the MainActivity Module.

Test Description: To test main user operations such as searching for dishes, viewing unusual orders, and placing orders.

Test steps	Step Description	Data Utilized	Expected Result	Actual Result	Pass/Fail /Haven't
1	Search dish by name	Dish: Pizza	The search results include foods with the name "Pizza."	Search results show the correct dish with details.	Pass
2	Select a dish and place an order.	Dish: Pizza. Price: RM 5.00	The dish is added to the cart, and the transaction is completed successfully	Dish was successfully added to the cart, and the order was placed.	Pass
3	Select the "Pay by Cash" option and complete the order.	Payment Method: Cash	The order is confirmed, the payment method is recorded, and confirmation is displayed.	Order placed with cash payment confirmation.	Pass
4	Select the "Online Payment" option and complete the order.	Payment Method: Online payment	The order is confirmed, and the payment process is initiated and completed.	Online payment was successfully processed, and the order was placed.	Pass
5	Select no payment option	-	The order is not confirmed and choose payment type is required	"Choose a payment type before placing an order"	Pass

Table 6.3.2: Unit Testing for the MainActivity Module.

6.3.3 Vendor Module

Test Case Module: Unit Testing for Vendor Module.

Test Objective: To evaluate vendor functionalities such as order management, adding, editing, deleting, and searching for dishes.

Test steps	Step Description	Data Utilized	Expected Result	Actual Result	Pass/Fail /Haven't
1	View all the incoming orders.	Order ID: orderID1	Display a list of all current orders, including item names and prices.	All the current orders are listed.	Pass
2	Mark an order as "completed".	Order ID: orderID1	The order's status was changed to "Completed" and it was moved into history.	Order is marked as completed and archived in history.	Pass
3	Include a new dish on the menu.	Dish: burger, Price: RM 5.00	A new dish has been added to the menu and is visible to customers.	Dish was successfully added to the menu.	Pass
4	Edit an existing dish's details.	Dish: Burger, New Price: RM5.00.	Dish information is updated and reflected on the menu.	Dish details were successfully updated.	Pass
5	Delete a dish from the menu.	Dish: Sandwich	Dish has been removed from the menu and is no longer visible to customers.	Dish was successfully removed from the menu.	Pass
6	Search for a dish by name.	Dish: Ice cream.	Search results show the dish's name and details.	Search results show the correct dish.	Pass

Table 6.3.3.1: Unit Testing for Vendor Module.

6.3.4 History module.

Test Case Module: Unit testing for the History Module.

Test Description: This test will show previous orders in history.

Test steps	Step Description	Data Utilized	Expected Result	Actual Result	Pass/Fail /Haven't
1	View the previous order details.	Order ID: historyID2.	Display all previous order information, including food items and prices.	Past order details were displayed correctly.	Pass
2	Order again from history.	Order ID: historyID2.	Add the items from the previous order to your current cart for checkout.	Order was successfully added to the cart.	Pass
3	Download an excel file of previous order history	-	Allows vendors to download a record of previous orders received.	"History.xlsx downloaded successfully"	Pass

Table 6.3.4.1: Unit testing for the History Module.

6.3.5 Notification module.

Test Case Module: Unit testing of the Notification Module.

Test Description: This test will send notifications to users when an order is ready or completed.

Test steps	Step Description	Data Utilized	Expected Result	Actual Result	Pass/Fail /Haven't
1	Send an SMS when a new order is placed.	Telephone number: +60193837753.	SMS was sent to the vendor with the message, "You have a new order."	SMS was successfully sent to the vendor.	Pass
2	Send an SMS when an order is completed.	Telephone number: +60162345592.	The user receives a notification stating, "Your order has been completed."	SMS was successfully sent.	Pass
3	Check to see if notifications continue to appear in the notification centre.	-	Notifications will remain in the user's notification centre until they are cleared.	Notifications remain in the notification centre.	Pass

Table 6.3.5.1: Unit testing of the Notification Module.

6.4 Analysis of Survey

An online questionnaire survey was carried out to determine the user requirements for the “SmartCanteen” mobile application. The survey obtained 52 valid responses from students and staff, with data collection and analysis conducted using Google Forms. The analysis is presented in tables and histograms to better understand the user preferences.

6.4.1 Demographic data

The demographic data revealed that most respondents were between the ages of 18 and 24, accounting for 78.6% of all participants, with both male and female respondents coming from student and faculty backgrounds. About 98.2% of respondents had prior experience using mobile applications to order food, demonstrating the study's relevance in meeting the needs of the target audience.

Demographics		Sample	Percentage
Gender	Male	61	54.5%
	Female	51	45.5%
Age	< 18	3	2.7%
	18-24	88	78.6%
	25-34	18	16.1%
	35-44	2	1.8%
	45+	1	0.9%
Occupation	Student	93	83%
	Faculty/Staff	18	16.1%
	Alumni	1	0.9%
Have you ever used a mobile application or system to order food from a canteen or restaurant before?	Yes	110	98.2%
	No	2	1.8%

Table 6.4.1.1 Demographic data of the respondents

6.4.2 Survey results

The results in Table A and Table B in Appendix showed that 82 (73.2%) agreed on an intuitive interface, 91 (81.2%) valued detailed menu information, and 87 (77.7%) valued order history. 91 (81.3%) people prioritised real-time menu availability, while 89 (79.5%) valued order tracking. Nearly all respondents, 110 (98.2%) and 109 (97.3%) valued login functionality and multiple payment options, respectively. These findings show broad satisfaction with the app's core functionalities, implying that it effectively meets user needs in interface design, payment flexibility, and order management.

Other than that, the survey results also provide useful information about user preferences for key features of the "SmartCanteen" mobile application. Below focuses on the four specific features that showed varied responses from participants:

1. Previous Order Memory.

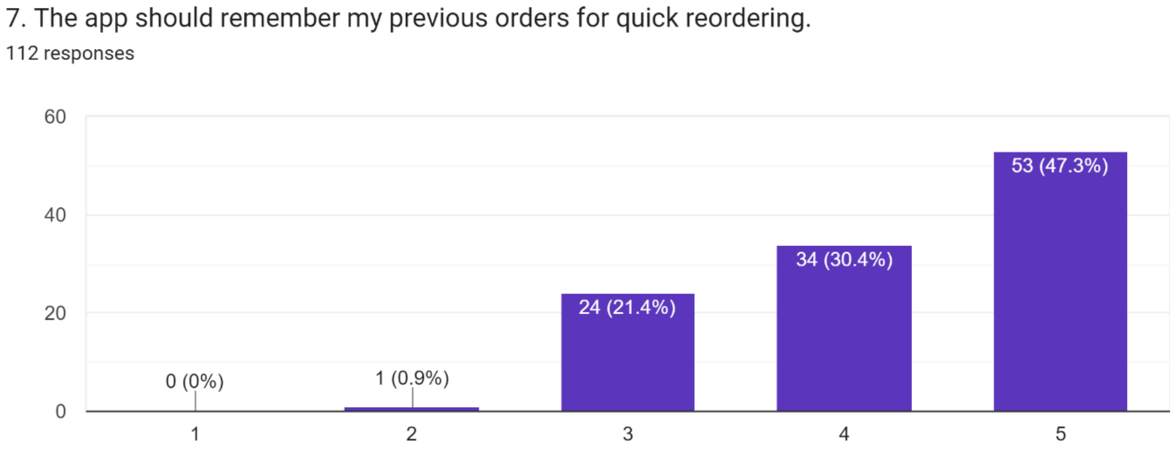


Figure 6.4.2.1: Bar Chart of respondents for previous order memory feature.

Based on the figure 6.4.2.1, the statement "The app should remember my previous orders for quick reordering" received generally positive feedback.

- Agree (strongly agree + agree): 77.7%.
- Neutral: 21.4%.
- 0.9% of respondents disagreed (strongly disagreed plus disagreed).

This feature is undoubtedly important to most users. The ability to quickly reorder previous meals saves time and improves user convenience. However, the relatively high neutral percentage (21.4%) indicates that, while useful, this feature may not be required by all users. The low percentage of disagreement suggests that few users are concerned about the app storing their order history.

2. Order tracking and notification.

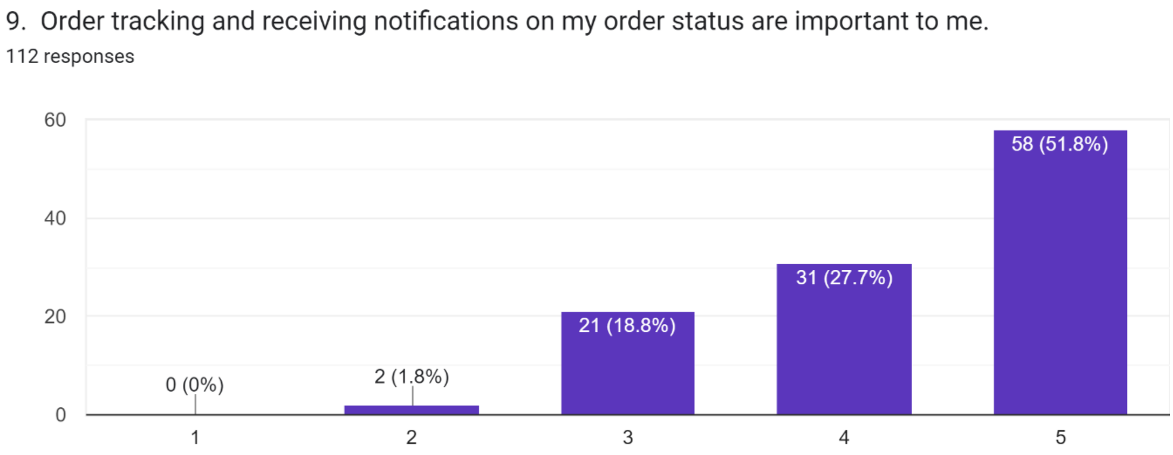


Figure 6.4.2.2: Bar Chart of respondents for order tracking and notification feature.

In figure 6.4.2.2, responses collected for the statement "Order tracking and receiving notifications on my order status are important to me" were:

- Agree (strongly agree + agree): 79.5 %.
- Neutral: 18.8%.
- 1.8% of respondents disagreed (strongly disagreed plus disagreed).

This feature received strong support from users, with more than 80% agreeing on its importance. This high agreement rate demonstrates how important it is for users to know the status of their orders. The ability to track orders and receive timely notifications can reduce uncertainty and enhance the overall user experience. The low disagreement rate indicates that very few users think this feature is unnecessary or potentially annoying.

3. Pre-ordering capability.

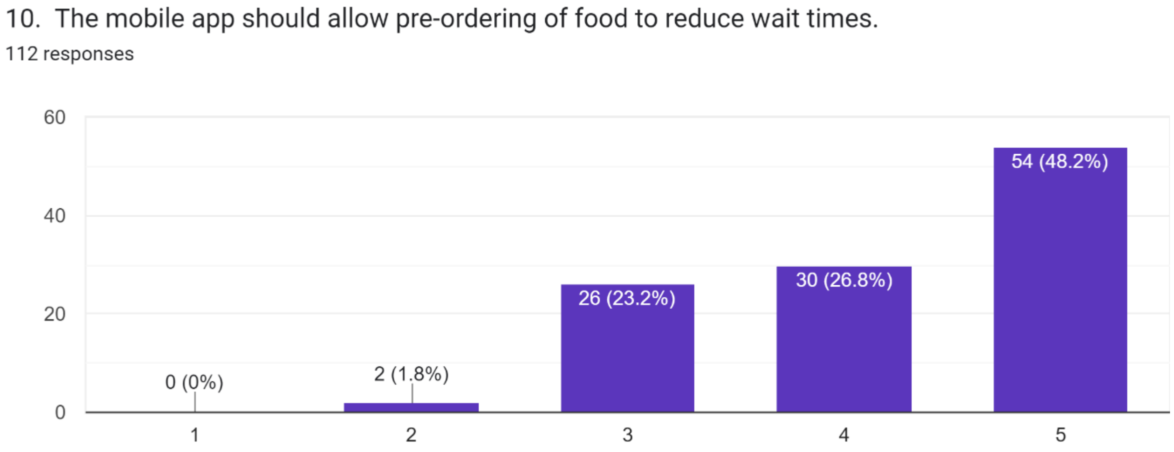


Figure 6.4.2.3: Bar Chart of respondents for pre-ordering capability feature.

Based on the bar chart in figure 6.4.2.3, for the statement "The mobile app should allow pre-ordering of food to reduce wait times" received mixed feedback.

- Agree (strongly agree + agree): 75 %.
- Neutral: 23.2%.
- 1.8% of respondents disagreed (strongly disagreed plus disagreed).

Despite receiving most positive responses, this feature had the lowest agreement rate of the four. The higher neutral percentage (23.2%) indicates that a quarter of respondents are unconcerned about this feature or are unsure of its benefits. This could be due to differences in personal schedules or preferences for food freshness. Despite this, the overall positive response suggests that pre-ordering is a useful feature for many users, potentially assisting in managing busy schedules and reducing wait times during peak hours.

4. Feedback option.

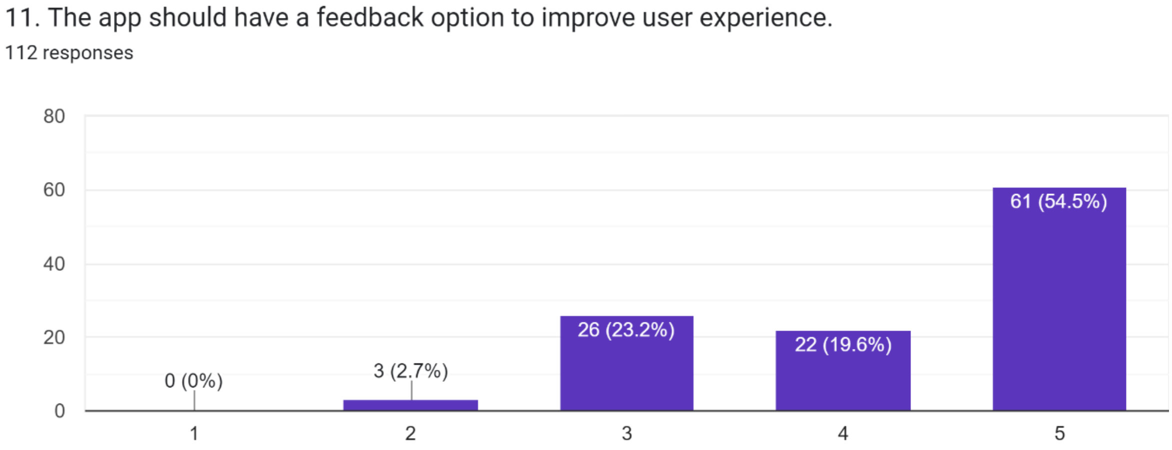


Figure 6.4.2.4: Bar Chart of respondents for feedback option feature.

Figure 6.4.2.4 shows the responses to the statement "The app should have a feedback option to improve user experience".

- Agree (strongly agree + agree): 74.1%.
- Neutral: 23.2%.
- 2.7% of respondents disagreed (strongly disagreed + disagreed).

The feedback option was widely supported, with more than three-quarters of respondents agreeing on its importance. This suggests that users value the opportunity to help improve the app and have their voices heard. The relatively high neutral percentage (23.2%) may indicate that, while users value the option, they do not see themselves using it frequently. The very low disagreement rate (2.7%) indicates that almost no users believe a feedback option is detrimental to their experience.

In conclusion, the four features received most of the support from respondents, demonstrating their importance in the “SmartCanteen” application. Order tracking and notifications were the most popular of these four features, followed by previous order memory and feedback. While still important, the pre-ordering feature received slightly less enthusiasm, implying that it may be considered a secondary feature during the initial development phase. The functional and non-functional requirements are constructed in Chapter 3 are results from the online survey questionnaire from 52 valid respondence.

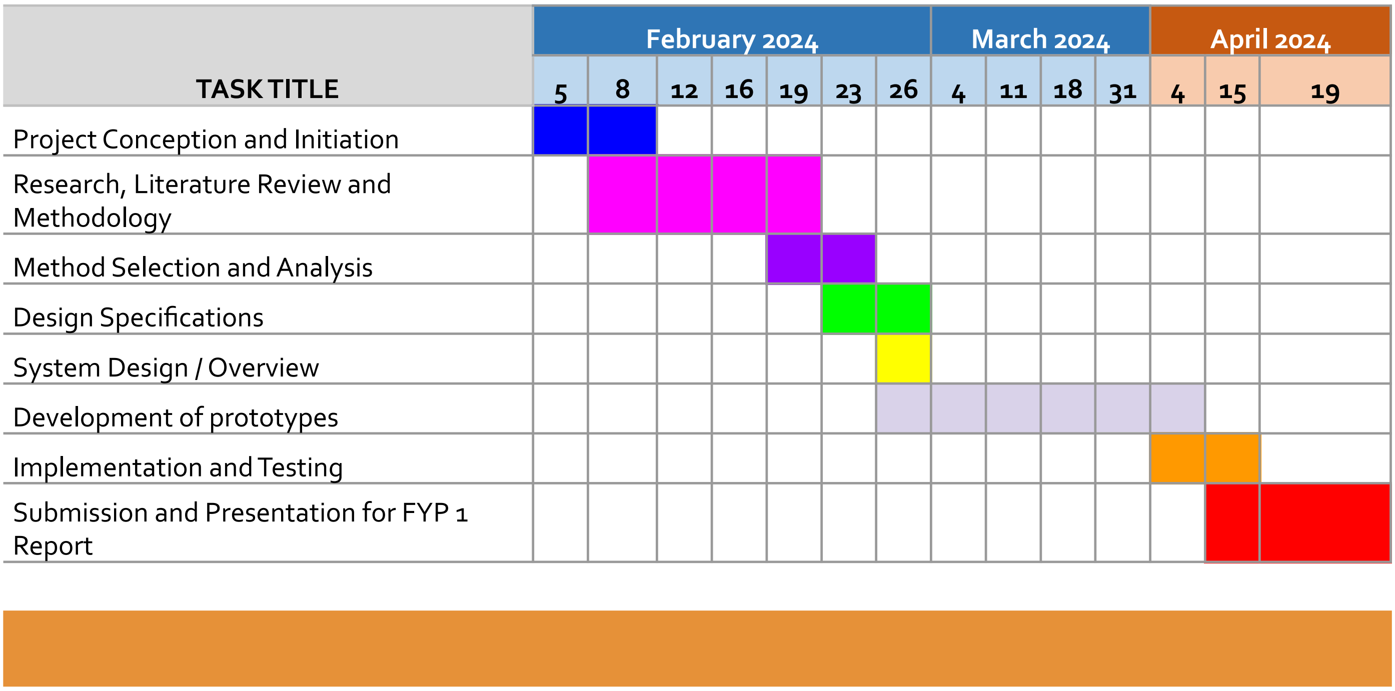
6.5 Project Challenges

The development of the “SmartCanteen” application faced several significant challenges, each of which required careful solutions. One of the main issues was ensuring a smooth integration with Firebase, particularly regarding real-time data synchronisation for orders. Given the importance of real-time data exchange between users and vendors, delays or errors during peak usage could result in significant disruptions. This required consistent troubleshooting and optimisation of Firebase's real-time database features to ensure data was always up to date and synchronised across devices. Another significant challenge was designing an intuitive user interface that catered to a wide range of users, including tech-savvy students and less experienced elderly staff. Finding a balance between simplicity and functionality was critical. Early feedback indicated that the design was too complex for some users, prompting iterative changes aimed at simplifying navigation while retaining the app's key features.

The final challenge was incorporating various payment methods. The goal was to provide users with a variety of options, including e-wallets and internet banking. However, due to technical limitations and integration issues, the final app only accepted cash, credit/debit card, and PayPal payments. Ensuring the security of the payment process while attempting to implement additional options proved too difficult, requiring a simplified approach to maintain functionality and data security standards.

6.6 Timeline

6.6.1 Gantt chart of FYP1



TASK TITLE	February 2024							March 2024				April 2024		
	5	8	12	16	19	23	26	4	11	18	31	4	15	19
Project Conception and Initiation	■	■												
Research, Literature Review and Methodology		■	■	■	■									
Method Selection and Analysis					■	■								
Design Specifications						■	■							
System Design / Overview							■							
Development of prototypes							■	■	■	■	■	■		
Implementation and Testing												■	■	
Submission and Presentation for FYP 1 Report													■	■

Figure 6.6.1.1: Gantt chart of FYP1 from February 16, 2024, to April 19, 2024

The Gantt chart provided explains a detailed timeline for the Final Year Project (FYP), with tasks covering project conception and initiation to implementation and testing. The chart, which spans from February 16, 2024, to April 19, 2024, divides the project into weekly segments, making it easier to track progress and assign tasks. Each element, including research, literature review, methodology, and design specifications, is assigned a set period to ensure concentrated efforts and prompt completion. Notably, the addition of the FYP 1 Report Submission and Presentation column signifies the project's completion, which is set for the week of April 15, 2024. This systematic method allows for effective project activity management while also ensuring that important deadlines are met.

6.6.2 Planning for current semester (FYP2)

TASK TITLE	July 2024							August 2024				September 2024		
	5	8	12	16	19	23	26	4	11	18	31	2	4	6
Introduction	🟦	🟦												
Research, Literature Review and Methodology		🟪	🟪	🟪	🟪									
Design Specifications						🟩	🟩							
System Design / Overview							🟨							
Development							▨	▨	▨	▨	▨	▨		
Implementation and Testing												🟧	🟧	
Submission and Presentation for FYP 2Report													🟥	🟥

Figure 6.6.2.1: Gantt chart of FYP2 from July 5, 2024, to September 6, 2024

The Gantt chart shows the estimated timeline for the second phase of the Final Year Project (FYP 2), which runs from July 5, 2024, to September 6, 2024. It includes the following project phases: introduction, research, literature review, and methodology, followed by design specifications and system design/overview. Notably, the Development phase takes up most of the weeks, highlighting the significant effort necessary for this stage. This phase is critical for improving the prototypes and fine-tuning the project's technical components to complete the project itself. Implementation and testing are then scheduled to ensure that the project is functional and reliable. The submission and presentation of the FYP 2 report marks the project's conclusion and documentation for assessment.

6.7 Conclusion

In conclusion, system testing data from Google Analytics reveals high user engagement with “SmartCanteen” particularly with key screens such as MainActivity and VendorMainPage2, while also highlighting areas for improvement in features with shorter interactions. Other than that, the unit tests of each completed module have been discussed in detail. All the result of the modules meets the expected result. The results and analysis of the survey and project timeline for FYP 1 and FYP2 and lastly planning for the current semester was also documented.

Chapter 7: Conclusion and Recommendation

7.1 Introduction

In conclusion, the proposed project aims to modernise the existing canteen ordering system by implementing a mobile-based application, thereby improving efficiency and user experience. The "SmartCanteen" application allows users to browse menus, place orders, and make payments using their smartphones, eliminating the need for physical queues. This system also includes real-time order tracking and a variety of payment options, making it more convenient for both students and staff.

Several objectives and modules were successfully implemented by the end of this project, as highlighted in the sections that follow. The project addressed key inefficiencies in traditional canteen operations, resulting in a more streamlined and user-friendly ordering system. Customers can now avoid long lines, receive order status updates, and manage their purchases more efficiently using the mobile application.

This project report documents the goals and objectives achieved, as well as recommendations for future improvements to address the limitations encountered. These proposed solutions will serve as a foundation for future development aimed at improving the overall user experience and system performance.

7.2 Findings

The project report concludes with a summary of achieved goals and objectives for the next session. Future work to improve the problem will be documented based on the identified limitations and solutions.

The following objectives and sub-objectives have been met:

1st Objective: To develop an intuitive, user-friendly mobile application for canteen ordering.

a) The program has a basic, clean layout with large buttons/icons for easy navigation and high contrast colours for visibility. (Done)

b) Text size and colour schemes can be customised for visually impaired users. (Done)

2nd Objective: To facilitate seamless lunch order placements and efficient food preparation timelines.

a) Category-based menu navigation allows users to quickly browse menus and place orders. (Done)

b) Vendors receives real-time order information, including prep time indicators. (Done)

3rd Objective: To integrate various digital payment options for seamless checkout.

a) Payment processing via credit/debit cards, PayPal, and cash has been implemented, though mobile wallets and campus dining accounts were not integrated. (Partially achieved)

b) Customers can securely store their payment credentials within the app for faster checkout on future orders. (Done) c) Digital receipts and order histories are provided to users. (Done)

4th Objective: To determine user requirements through online survey questionnaires.

a) Comprehensive surveys were created and distributed to gather user requirements for the canteen ordering system. (Done).

b) The survey results were analysed to help shape the system design. (Done)

The modules that have been achieved:

a. Sign-up and login functionality. (Fully)

b. Browse the digital menu and place an order. (Fully)

c. Live order tracking and status updates. (Fully)

d. Centralised order and menu data management. (Fully)

e. Integration of various digital payment methods. (Partially)

f. A vendor management interface for order processing. (Fully)

7.3 Limitation

Several limitations and issues were identified during the development process of the "SmartCanteen" mobile application, which must be considered:

1. The application is currently only available to Android users, limiting its availability to those using iOS or other platforms.
2. Due to resource constraints, mobile wallets and campus dining accounts were not integrated, leaving only credit/debit cards, PayPal, and cash as payment options.

3. Network dependency is an important consideration; users require a consistent internet connection to access the menu, place orders, and track them in real time, which may impede the user experience in areas with limited connectivity.
4. The application does not currently support multilingual options, which may limit its usefulness for non-English speakers.
5. The vendor interface is functional, but it could be improved for a better user experience, especially for vendors that handle high order volumes during peak hours.
6. Due to time and financial constraints, advanced data analytics for tracking customer preferences and improving inventory management were not fully implemented.

7.4 Future Recommendation

1. This section provides recommendations to address the limitations mentioned earlier:
2. Expand the application to include iOS support, making the system cross-platform and accessible to a larger audience.
3. Integrate additional payment methods, such as mobile wallets and campus dining accounts, to give users more flexibility and convenience.
4. Improve the application's offline functionality, allowing users to browse menus and place orders in locations with limited or no internet access. Once connectivity is restored, orders can be synced.
5. Add multi-language support to accommodate users who do not speak English, making the application more accessible and inclusive.
6. Enhance the vendor interface to increase efficiency, especially during peak hours, by including features such as order prioritisation and inventory management.
7. Implement advanced data analytics to help vendors understand customer preferences, resulting in better inventory management and personalised recommendations.

7.5 Conclusion

In conclusion, this chapter summarised the issues with existing canteen ordering systems and described the methods proposed to address them. This chapter documents the successfully completed objectives and modules, as well as the project's limitations. In addition, recommendations were made to improve the functionality, accessibility, and scalability of the "SmartCanteen" application. Overall, the proposed project produced a prototype product that met most of the objectives outlined in Section 1.5. While the primary goal was to streamline the canteen ordering process, the features and concepts developed in this project could be applied to other service-based environments such as cafeterias, small restaurants, or even retail stores.

Moving forward, with the addition of new features such as improved payment options, cross-platform availability, and enhanced vendor tools, the system has the potential to evolve into a comprehensive solution that meets the needs of a larger user base while addressing operational challenges in a variety of industries.

REFERENCES

[1] S. Joshi, B. Kasaju, P. Karki, S. K. Aryal, and S. B. Chhetri, “Smart Canteen Management System,” *2022 IEEE 2nd Mysore Sub Section International Conference (MysuruCon)*, Oct. 2022, doi: 10.1109/mysurucon55714.2022.9971803.

[2] H. Auto, “Long queues at hypermarts, restaurants in Malaysia ahead of expected Covid19 curbs,” *The Straits Times*, Jan. 11, 2021. [Online]. Available: https://www.straitstimes.com/asia/se-asia/long-queues-at-hypermarts-restaurants-in-malaysia-amid-talk-of-fresh-covid-19-curbs

[3] N. Pandey, S. Sharma, V. Sharma, and T. Garg, “Automated Canteen Ordering system,” *International Journal for Research in Applied Science and Engineering Technology*, vol.
10, no. 5, pp. 3617–3620, May 2022, doi: 10.22214/ijraset.2022.43173.

[4] C. Antony, “Canteen food Ordering and Managing system,” *International Journal of Current Science Research and Review*, vol. 05, no. 06, Jun. 2022, doi: 10.47191/ijcsrr/v5i6-51.

[5] T. Dirsehan and E. Cankat, “Role of mobile food-ordering applications in developing restaurants’ brand satisfaction and loyalty in the pandemic period,” *Journal of Retailing and Consumer Services*, vol. 62, p. 102608, Sep. 2021, doi:
10.1016/j.jretconser.2021.102608.

[6] M. Y.-C. Yim and C. Y. Yoo, “Are Digital Menus Really Better than Traditional Menus? The Mediating Role of Consumption Visions and Menu Enjoyment,” *Journal of Interactive Marketing*, vol. 50, no. 1, pp. 65–80, May 2020, doi:
10.1016/j.intmar.2020.01.001.

[7] N. Abdullah, F. Redzuan, and N. A. Daud, “E-wallet: factors influencing user acceptance towards cashless society in Malaysia among public universities,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 20, no. 1, p. 67, Oct. 2020, doi:
10.11591/ijeecs.v20.i1.pp67-74.

[8] A. Bonfanti, C. Rossato, V. Vigolo, and A. V. Sánchez, “Improving online food ordering and delivery service quality by managing customer expectations: evidence from Italy,” *British Food Journal*, vol. 125, no. 13, pp. 164–182, Feb. 2023, doi: 10.1108/bfj-08-20220694.

[9] A. R, A. Singh, S. Pathan, and V. Kanade, “Online food ordering system,” *International Journal of Computer Applications*, vol. 180, no. 6, pp. 22–24, Dec. 2017, doi: 10.5120/ijca2017916046.

[10] N. Z. Nizam, Y. Arshad, and S. H. Supaat, “The understanding of sales promotions’ influence on food and beverages products in Malaysia,” *International Journal of Academic Research in Business & Social Sciences*, vol. 8, no. 10, Nov. 2018, doi: 10.6007/ijarbss/v8-i10/5313.

[11] P. Brewer and A. G. Sebby, “The effect of online restaurant menus on consumers’ purchase intentions during the COVID-19 pandemic,” *International Journal of Hospitality Management*, vol. 94, p. 102777, Apr. 2021, doi: 10.1016/j.ijhm.2020.102777.

[12] Y. Sakai, Y. Y. S. Rahayu, and T. Araki, “Nutritional value of canteen menus and dietary habits and intakes of university students in Indonesia,” *Nutrients*, vol. 14, no. 9, p. 1911, May 2022, doi: 10.3390/nu14091911.

[13] C.-C. B. Chen, H. Chen, and Y.-C. Wang, “Cash, credit card, or mobile? Examining customer payment preferences at chain restaurants in Taiwan,” Journal of Foodservice Business Research, pp. 1–20, Jun. 2021, doi:

[14] A. Sri Jeevagowry, “SJKT Azad is first primary school in Penang to adapt cashless canteen culture,” *The Malaysian Reserve*, p. 1, Jan. 11, 2023. Accessed: Feb. 23, 2024. [Online]. Available:

[15] X. Hongzhen, T. Bin, and S. Wenlin, “Wireless Food Ordering System Based on Web Services,” IEEE Xplore, Oct. 01, 2009. https://ieeexplore.ieee.org/abstract/document/5288330/ (accessed Apr. 16, 2022).

[16] G. M. Aditya et al., “Machine Learning Based Platform and Recommendation System for Food Ordering Services within Premises,” IEEE Xplore, Oct. 01, 2021.

[17] Tejas Raibagi, Ashwin Vishwakarma, J. Naik, R. Chaudhari, and Guntis Kalme, “Orderista - AI-based Food Ordering Application,” International Conference on Artificial Intelligence, Mar. 2021, doi: 10.1109/icais50930.2021.9396040.

[18] P. Manikandan, K. K. Babu, G. M. Reddy, G. Kalayan, and V. Muneeswaran, “KARE - Presto Canteen Management System with an Android Application,” IEEE Xplore, Apr. 01, 2023.

[19] Andhika, Y. Pernando, I. Verdian, Yodi, and M. R. Pradana, “Vege Application! Using Mobile Application to Buy Vegetarian Food,” IEEE Xplore, Oct. 01, 2019.

[20] A. Hongyu, L. Lizong, Q. Dixin, and O. Baibing, “Design of College Canteen Ordering System Based on Cortex-A9 and Android,” IEEE Xplore, Jun. 01, 2020.

[21] “What is Rapid Application Development?” NCube. https://ncube.com/blog/what-is-rapid-application-development

[22] M. N. K. Saunders, P. Lewis, A. Thornhill, and A. Bristow, “Understanding research philosophy and approaches to theory development,” oro.open.ac.uk, Jul. 23, 2019.

APPENDIX

TABLE A. SUMMARY OF RESPONSES TO 52 SURVEY QUESTIONS USING 5 POINT LIKERT SCALE

PF/N		Survey Questions	% Strongly agree	% Agree	% Neutral	% Disagree	% Strongly disagree
TD	1	The app should have a simple and intuitive user interface.	44.6	28.6	26.8	0	0
	2	The app should display detailed information about each menu item (ingredients, nutritional info).	45.5	35.7	17.9	0.9	0
	3	The app should remember my previous orders for quick reordering.	47.3	30.4	21.4	0.9	0
	4	Viewing the menu with real-time availability is essential for making decisions.	52.7	28.6	18.8	0	0
	5	Order tracking and receiving notifications on my order status are important to me.	51.8	27.7	18.8	1.8	0
	6	The mobile app should allow pre-ordering of food to reduce wait times.	48.2	26.8	23.2	1.8	0
	7	The app should have a feedback option to improve user experience.	54.5	19.6	23.2	2.7	0
F	8	I can register and log in to the mobile application.	83	15.2	0.9	0.9	0
	9	I can use multiple payment options, PayPal and cards.	85.7	11.6	2.7	0	0
	10	I can add, edit, delete, and view my orders in the application	87.5	10.7	1.8	0	0
	11	I can try new menu items when I order through an app.	77.7	14.3	8	0	0
	12	I can order from multiple campus food outlets in one transaction	79.5	11.6	6.3	1.8	0.9

TABLE B. COMBINED RESPONSES OF RESPONDENTS

PF/N		Survey Questions	Agree	Disagree
TD	1	The app should have a simple and intuitive user interface.	82 (73.2%)	0 (0%)
	2	The app should display detailed information about each menu item (ingredients, nutritional info).	91 (81.2%)	1 (0.9%)
	3	The app should remember my previous orders for quick reordering.	87 (77.7%)	1 (0.9%)
	4	Viewing the menu with real-time availability is essential for making decisions.	91 (81.3%)	0 (0%)
	5	Order tracking and receiving notifications on my order status are important to me.	89 (79.5%)	2 (1.8%)
	6	The mobile app should allow pre-ordering of food to reduce wait times.	84 (75%)	2 (1.8%)
	7	The app should have a feedback option to improve user experience.	83 (74.1%)	3 (2.7%)
F	8	I can register and log in to the mobile application.	110 (98.2%)	1 (0.9%)
	9	I can use multiple payment options, PayPal and cards.	109 (97.3%)	0 (0%)
	10	I can add, edit, delete, and view my orders in the application	110 (98.2%)	0 (0%)
	11	I can try new menu items when I order through an app.	103 (92%)	0 (0%)
	12	I can order from multiple campus food outlets in one transaction	102 (91.1%)	3 (2.7%)

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Year 3 Semester 3	**Study week no.: Week 1**
Student Name & ID: Kaviraj Maniveel, 21ACB06054	
Supervisor: Ts Dr Phan Koo Yuen	
Project Title: "SmartCanteen:" Transforming Canteen Ordering with a Mobile Application	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Rewrite chapter 1 and 2
- Review IIPSW report and guideline

2. WORK TO BE DONE

- Complete Chapter 1 and 2
- Progressing on Order Module

3. PROBLEMS ENCOUNTERED

N/A

4. SELF EVALUATION OF THE PROGRESS

Keep up on the good work

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Year 3 Semester 3	**Study week no.: Week 3**
Student Name & ID: Kaviraj Maniveel, 21ACB06054	
Supervisor: Ts Dr Phan Koo Yuen	
Project Title: "SmartCanteen:" Transforming Canteen Ordering with a Mobile Application	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Completed Chapter 1 and 2
- Completed on Order Module

2. WORK TO BE DONE

- Complete chapter 3
- Complete Vendor module

3. PROBLEMS ENCOUNTERED

N/A

4. SELF EVALUATION OF THE PROGRESS

Don't give it up. You can do this.

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Year 3 Semester 3	**Study week no.: Week 5**
Student Name & ID: Kaviraj Maniveel, 21ACB06054	
Supervisor: Ts Dr Phan Koo Yuen	
Project Title: “SmartCanteen:” Transforming Canteen Ordering with a Mobile Application	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Completed chapter 3
- Completed Vendor module

2. WORK TO BE DONE

- Work on chapter 4
- Test each module

3. PROBLEMS ENCOUNTERED

problems occur when fetching order history from Firebase Database

4. SELF EVALUATION OF THE PROGRESS

Don’t give up

Supervisor’s signature

Student’s signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Year 3 Semester 3	**Study week no.: Week 7**
Student Name & ID: Kaviraj Maniveel, 21ACB06054	
Supervisor: Ts Dr Phan Koo Yuen	
Project Title: “SmartCanteen:” Transforming Canteen Ordering with a Mobile Application	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Completed chapter 4

2. WORK TO BE DONE

- Fix bugs on the certain modules

3. PROBLEMS ENCOUNTERED

N/A

4. SELF EVALUATION OF THE PROGRESS

N/A

Supervisor’s signature

Student’s signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Year 3 Semester 3	**Study week no.: Week 9**
Student Name & ID: Kaviraj Maniveel, 21ACB06054	
Supervisor: Ts Dr Phan Koo Yuen	
Project Title: "SmartCanteen:" Transforming Canteen Ordering with a Mobile Application	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Fixed the bugs on some of the module
- Finish the testing on each module

2. WORK TO BE DONE

- Complete chapter 5

3. PROBLEMS ENCOUNTERED

N/A

4. SELF EVALUATION OF THE PROGRESS

Just do it

Supervisor's signature

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Year 3 Semester 3	**Study week no.: Week 11**
Student Name & ID: Kaviraj Maniveel, 21ACB06054	
Supervisor: Ts Dr Phan Koo Yuen	
Project Title: “SmartCanteen:” Transforming Canteen Ordering with a Mobile Application	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

• Complete chapter 5

2. WORK TO BE DONE

• Complete chapter 6

3. PROBLEMS ENCOUNTERED

N/A

4. SELF EVALUATION OF THE PROGRESS

Just do it

Supervisor’s signature

Student’s signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Year 3 Semester 3	**Study week no.: Week 13**
Student Name & ID: Kaviraj Maniveel, 21ACB06054	
Supervisor: Ts Dr Phan Koo Yuen	
Project Title: "SmartCanteen:" Transforming Canteen Ordering with a Mobile Application	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- Completed chapter 6

2. WORK TO BE DONE

- Complete everything

3. PROBLEMS ENCOUNTERED

N/A

4. SELF EVALUATION OF THE PROGRESS

I can do it

Supervisor's signature

Student's signature

POSTER

SmartCanteen

MOBILE APPLICATION

INTRODUCTION

Traditional canteen ordering methods are inefficient due to long waits, a lack of menu visibility, order problems, and limited payment alternatives. Therefore, a creative method to transform the dining experience using an innovative smartphone application: "SmartCanteen."

PROJECT SCOPE

- Develop a fully functional Android app for convenient canteen ordering.
- Integrate with the Firebase API to enable real-time order processing, tracking, and communication.
- Develop an administrative dashboard for vendor order management and analysis.

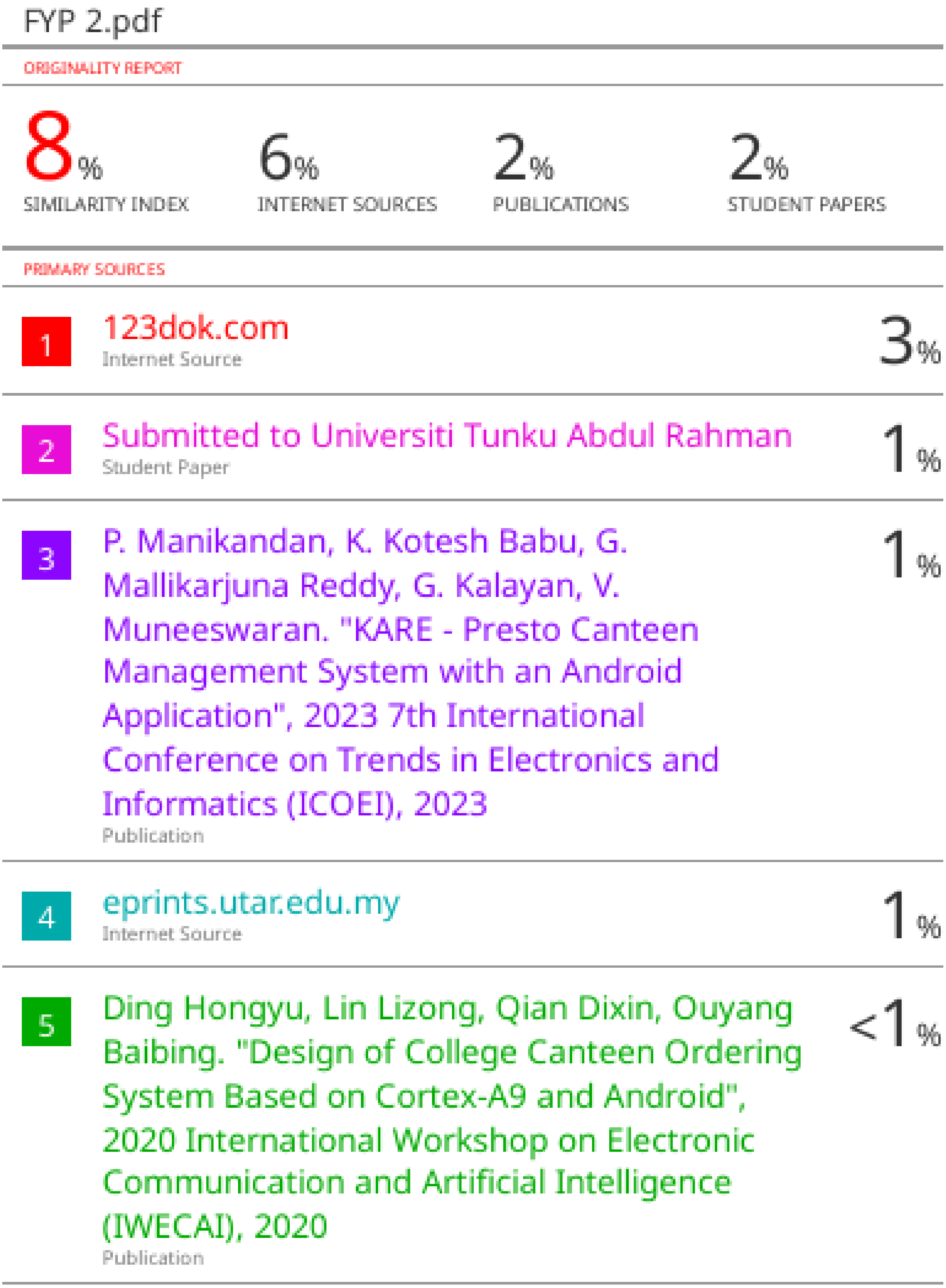
LIST OF MODULES

1. User Registration and Authentication
2. View and Browse Digital Menu
3. View cart and Order Placement
4. Integrated Electronic Payment Gateway
5. Vendor Order Management and Analytics

PROJECT OBJECTIVES

- Develop a user-friendly mobile app for efficient canteen ordering.
- Facilitate seamless order placements and streamline food preparation.
- Integrate diverse digital payment options for convenient transactions.
- Enhance overall operational efficiency and customer satisfaction.

PLAGIARISM CHECK RESULT



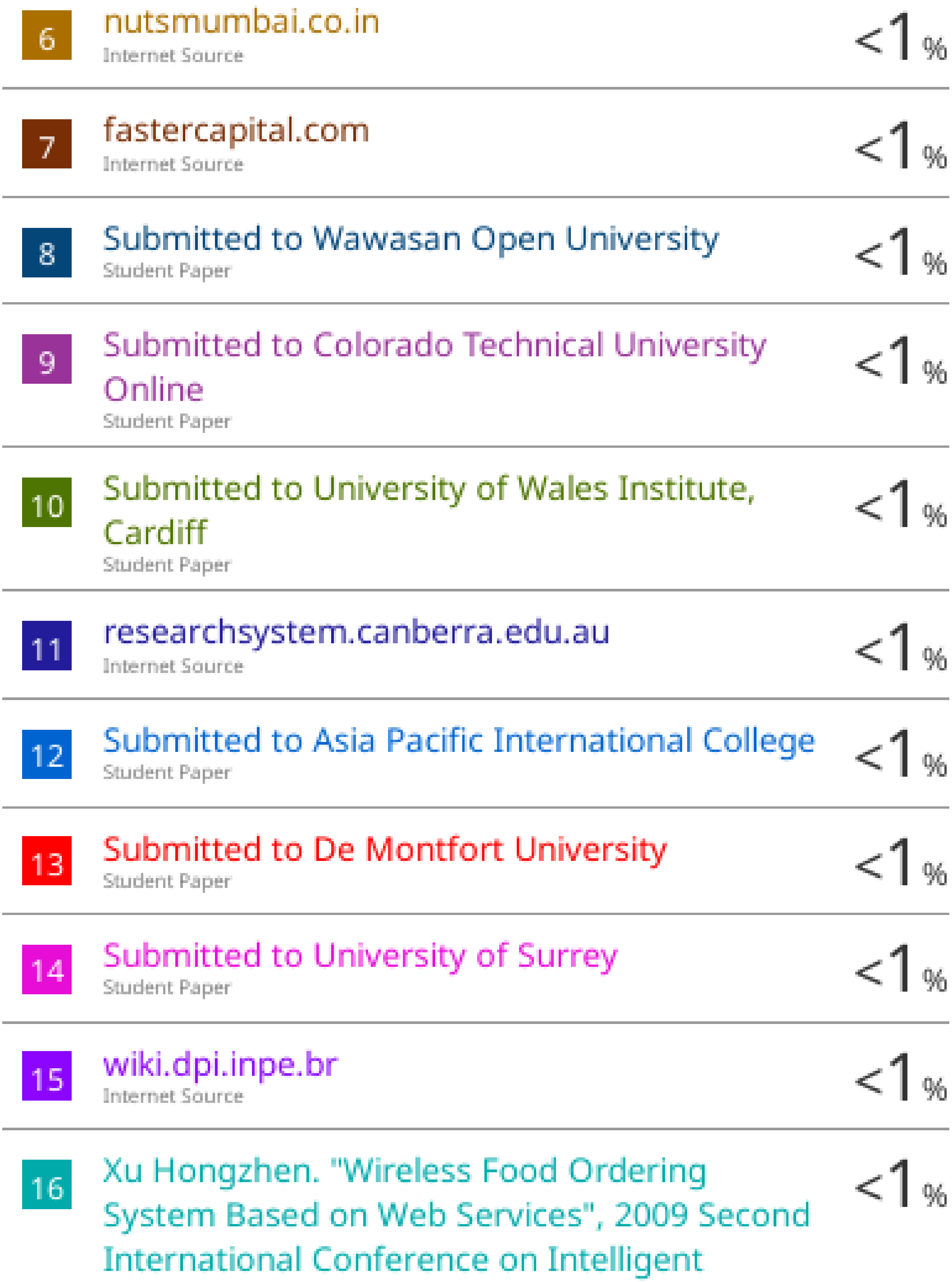
FYP 2.pdf

ORIGINALITY REPORT

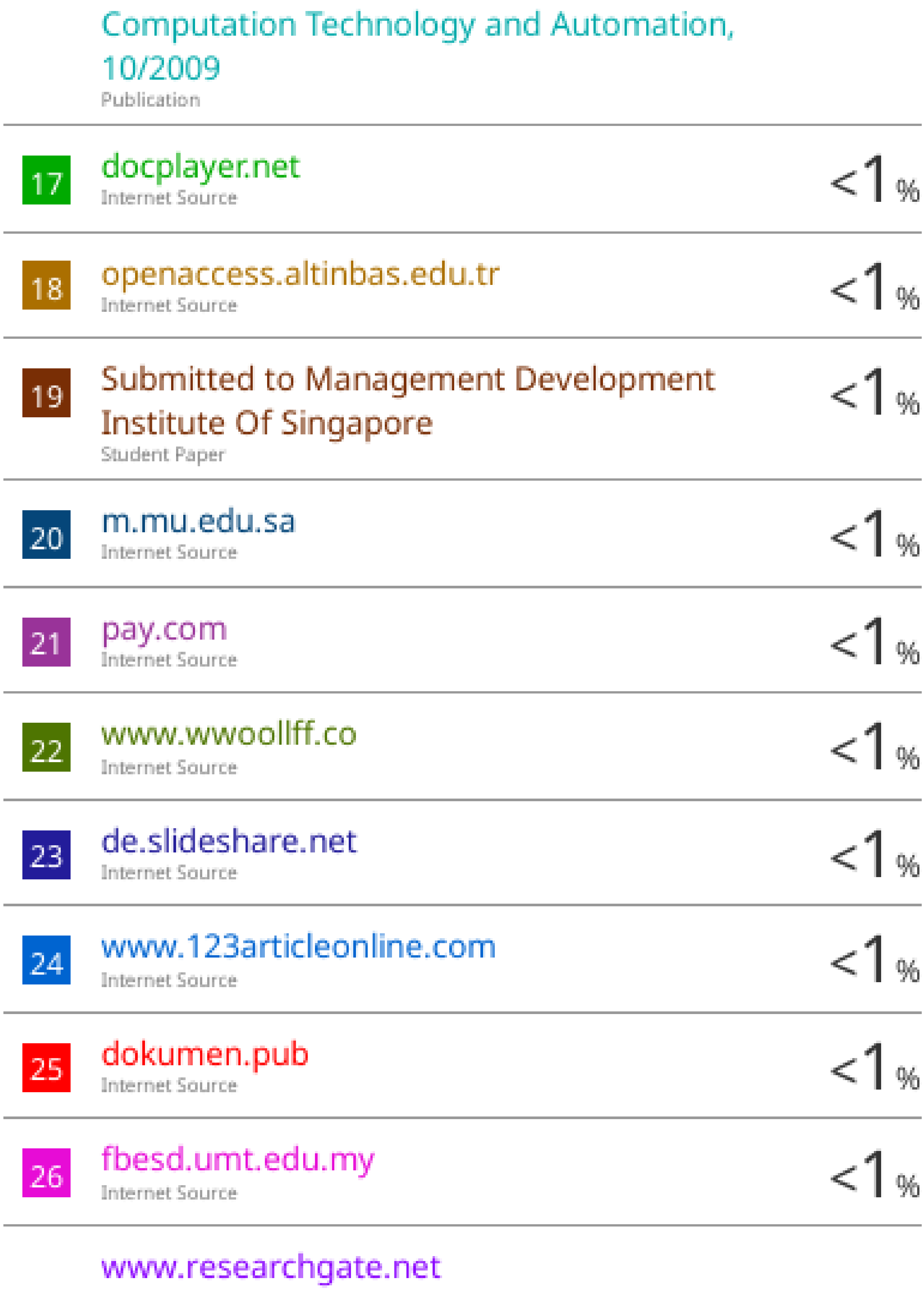
8%	6%	2%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	123dok.com Internet Source	3%
2	Submitted to Universiti Tunku Abdul Rahman Student Paper	1%
3	P. Manikandan, K. Kotesh Babu, G. Mallikarjuna Reddy, G. Kalayan, V. Muneeswaran. "KARE - Presto Canteen Management System with an Android Application", 2023 7th International Conference on Trends in Electronics and Informatics (ICOEI), 2023 Publication	1%
4	eprints.utar.edu.my Internet Source	1%
5	Ding Hongyu, Lin Lizong, Qian Dixin, Ouyang Baibing. "Design of College Canteen Ordering System Based on Cortex-A9 and Android", 2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI), 2020 Publication	<1%



6	nutsmumbai.co.in Internet Source	<1 %
7	fastercapital.com Internet Source	<1 %
8	Submitted to Wawasan Open University Student Paper	<1 %
9	Submitted to Colorado Technical University Online Student Paper	<1 %
10	Submitted to University of Wales Institute, Cardiff Student Paper	<1 %
11	researchsystem.canberra.edu.au Internet Source	<1 %
12	Submitted to Asia Pacific International College Student Paper	<1 %
13	Submitted to De Montfort University Student Paper	<1 %
14	Submitted to University of Surrey Student Paper	<1 %
15	wiki.dpi.inpe.br Internet Source	<1 %
16	Xu Hongzhen. "Wireless Food Ordering System Based on Web Services", 2009 Second International Conference on Intelligent	<1 %



	Computation Technology and Automation, 10/2009 Publication	
17	docplayer.net Internet Source	<1%
18	openaccess.altinbas.edu.tr Internet Source	<1%
19	Submitted to Management Development Institute Of Singapore Student Paper	<1%
20	m.mu.edu.sa Internet Source	<1%
21	pay.com Internet Source	<1%
22	www.wwoollff.co Internet Source	<1%
23	de.slideshare.net Internet Source	<1%
24	www.123articleonline.com Internet Source	<1%
25	dokumen.pub Internet Source	<1%
26	fbesd.umt.edu.my Internet Source	<1%
	www.researchgate.net	

27	Internet Source	<1 %
28	Mohamed E. Fayad, Huascar A. Sanchez, Srikanth G.K. Hegde, Anshu Basia, Ashka Vakil. "Software Patterns, Knowledge Maps, and Domain Analysis", Auerbach Publications, 2019 Publication	<1 %

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	Kaviraj Maniveel
ID Number(s)	21ACB06054
Programme / Course	Bachelor of Computer Science (Honours)
Title of Final Year Project	SMARTCANTEEN: TRANSFORMING CANTEEN ORDERING WITH A MOBILE APPLICATION

Similarity	**Supervisor's Comments** **(Compulsory if parameters of originality exceed the limits approved by UTAR)**
Overall similarity index: 8 % **Similarity by source** Internet Sources: 6 % Publications: 2 % Student Papers: 2 %	
Number of individual sources listed of more than 3% similarity: 0	
Parameters of originality required, and limits approved by UTAR are as Follows: **(i) Overall similarity index is 20% and below, and** **(ii) Matching of individual sources listed must be less than 3% each, and** **(iii) Matching texts in continuous block must not exceed 8 words** ***Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.***	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

Signature of Supervisor

Name: Phan Koo Yuen

Date: 13/9/2024

Signature of Co-Supervisor

Name:

Date:

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY

(KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	21ACB06054
Student Name	KAVIRAJ MANIVEEL
Supervisor Name	TS. DR PHAN KOO YUEN

TICK (√)	DOCUMENT ITEMS Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
✓	Title Page
✓	Signed Report Status Declaration Form
✓	Signed FYP Thesis Submission Form
✓	Signed form of the Declaration of Originality
✓	Acknowledgement
✓	Abstract
✓	Table of Contents
✓	List of Figures (if applicable)
✓	List of Tables (if applicable)
	List of Symbols (if applicable)
✓	List of Abbreviations (if applicable)
✓	Chapters / Content
✓	Bibliography (or References)
✓	All references in bibliography are cited in the thesis, especially in the chapter of literature review
✓	Appendices (if applicable)
✓	Weekly Log
✓	Poster
✓	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
✓	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.

____

(Signature of Student)

Date: 13/9/2024