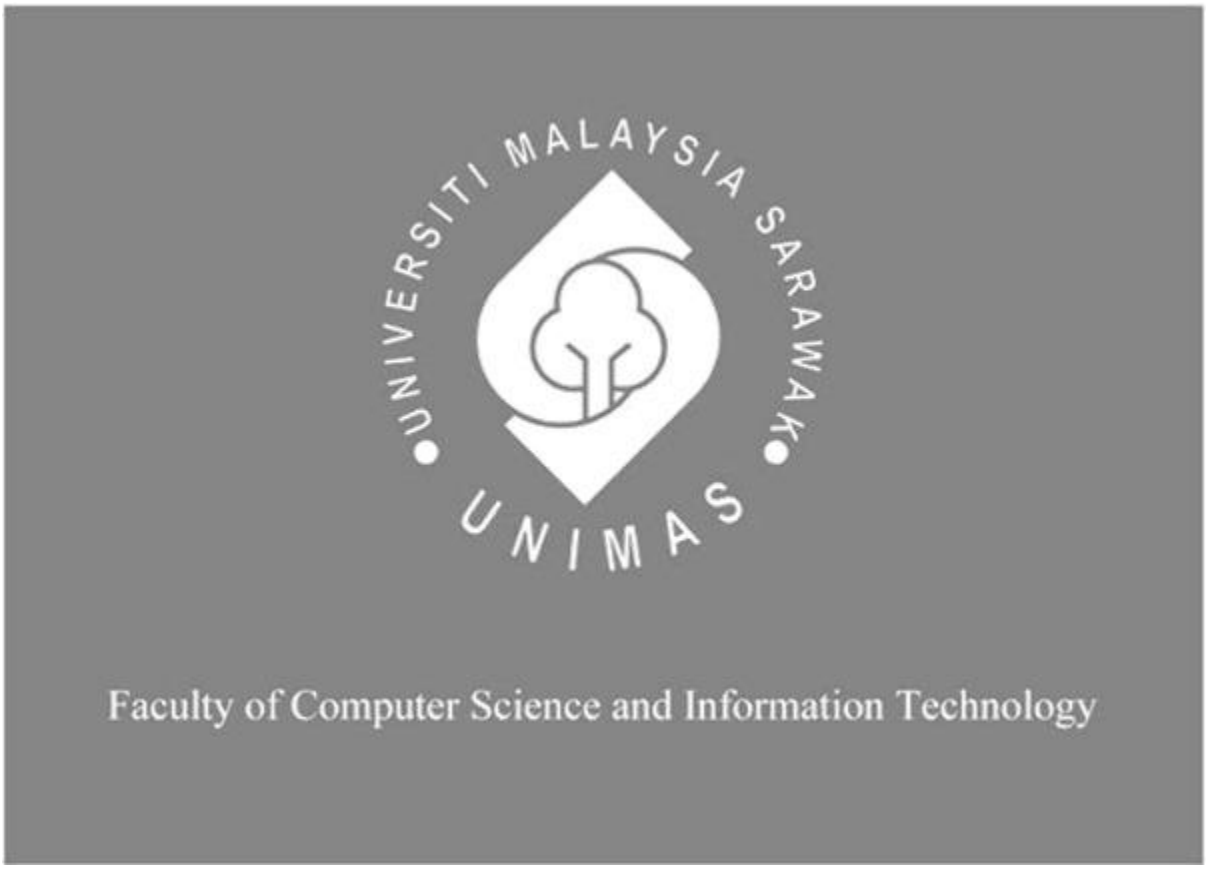


Smart Budget Personal Finance Tracking Mobile Application

ERIN WONG SING YING

Bachelor of Software Engineering with Honors

2025

Smart Budget Personal Finance Tracking Mobile Application

ERIN WONG SING YING

This project is submitted in partial fulfilment of

the requirements for the degree of Bachelor of Software Engineering with Honors

Faculty of Computer Science and Information Technology

UNIVERSITI MALAYSIA SARAWAK

2025

Aplikasi Mudah Alih Penjejakan Kewangan Peribadi Bajet Pintar

ERIN WONG SING YING

Projek ini merupakan salah satu keperluan untuk Ijazah Sarjana Muda Kejuruteraan Perisian dengan Kepujian

Fakulti Sains Komputer dan Teknologi Maklumat

UNIVERSITI MALAYSIA SARAWAK

2025

UNIVERSITI MALAYSIA SARAWAK

THESIS STATUS ENDORSEMENT FORM

TITLE **SMART BUDGET PERSONAL FINANCE TRACKING MOBILE APPLICATION**

ACADEMIC SESSION: 2024/2025
ERIN WONG SING YING
(CAPITAL LETTERS)

hereby agree that this Thesis* shall be kept at the Centre for Academic Information Services, Universiti Malaysia Sarawak, subject to the following terms and conditions:

1. The Thesis is solely owned by Universiti Malaysia Sarawak
2. The Centre for Academic Information Services is given full rights to produce copies for educational purposes only
3. The Centre for Academic Information Services is given full rights to do digitization in order to develop local content database
4. The Centre for Academic Information Services is given full rights to produce copies of this Thesis as part of its exchange item program between Higher Learning Institutions [or for the purpose of interlibrary loan between HLI]
5. ** Please tick (√)

☐	CONFIDENTIAL	(Contains classified information bounded by the OFFICIAL SECRETS ACT 1972)
☐	RESTRICTED	(Contains restricted information as dictated by the body or organization where the research was conducted)
√	UNRESTRICTED	

Validated by

(AUTHOR'S SIGNATURE)

Dr Liew Siaw Hong
Lecturer
Faculty of Computer Science and Information Technology
Universiti Malaysia Sarawak

(SUPERVISOR'S SIGNATURE)

Permanent Address

31-I, Jalan Huo Ping, 96000, Sibu, Sarawak

Date: 19/7/2025

Date: 23/07/2025

Note * Thesis refers to PhD, Master, and Bachelor Degree
** For Confidential or Restricted materials, please attach relevant documents from relevant organizations / authorities

DECLARATION

I hereby declare that the work contained in this project entitled "Smart Budget Personal Finance Tracking Mobile Application" is my original work except where explicitly stated otherwise. This project has been carried out as part of the requirements for the degree of Bachelor of Software Engineering (Honours) at Universiti Malaysia Sarawak (UNIMAS). I confirm that this work has not been submitted for any other academic qualification at this or any other institution of learning. All sources of information and literature used in this project have been acknowledged appropriately. I understand that the university has a policy regarding plagiarism and academic integrity, and I affirm that this work complies with all such policies.

Signed by,

ERIN WONG SING YING,

Faculty of Computer Science and Information Technology

Universiti Malaysia Sarawak.

ACKNOWLEDGEMENT

I would like to thanks to Universiti Malaysia Sarawak (UNIMAS) for providing the resources and facilities needed to complete this project successfully. The availability of tools, software, and a supportive environment for research made it easier for me to work efficiently and achieve my project goals.

Besides that, I would like to sincerely thank my supervisor, Dr. Liew Siaw Hong, for her guidance and support throughout my final year project. Her valuable advice, helpful feedback, and constant encouragement have been a great help in overcoming the challenges that I had faced during this project. I am truly grateful for her patience and dedication to ensuring the quality of my work.

My heartfelt thanks go to all the respondents who completed my survey forms. Their input and honest feedback were very important and played an important role in the success of this research. This project would not have been possible without their cooperation.

I am also deeply grateful to my family for their constant love, support, and belief in me throughout this journey. Their encouragement gave me strength during difficult times. I also want to thank my friends for their emotional and intellectual support, as well as their advice and motivation whenever I needed it.

Lastly, I would like to thank everyone who has helped me in any way during this project. Your support and kindness have made this success possible, and I will always remember and appreciate your contributions.

ABSTRACT

The Smart Budget Personal Finance Tracking Mobile Application is designed to help users to manage their personal finances easily and effectively. This application simplifies expense tracking, budgeting, and financial planning by integrating key features such as automated transaction logging through receipt scanning using Azure Form Recognizer, monthly budget assignment per category, spending summary reports, push notification reminders, and an AI chatbot powered by the Gemini API for financial guidance. It is developed using the Flutter framework and Dart programming language to ensure seamless performance on Android device. The system uses Firebase for secure data storage and hosting, allowing real-time access to user information. To evaluate the system's effectiveness, functional and non-functional testing were conducted, along with usability testing involving 25 participants. Results showed that 100% of users agreed the app was easy to use, while 92% found the receipt scanning feature helpful in simplifying financial tracking. Additionally, 80% of users reported that the AI chatbot was beneficial for providing real-time financial suggestions. Overall, this project aims to address the limitations of traditional budgeting methods and existing financial apps which help users to develop better financial habits and achieve their financial goals effectively.

ABSTRAK

Aplikasi Mudah Alih Penjejakan Kewangan Peribadi Smart Budget direka untuk membantu pengguna mengurus kewangan peribadi mereka dengan mudah dan berkesan. Aplikasi ini memudahkan proses penjejakan perbelanjaan, perancangan bajet dan kewangan melalui integrasi ciri-ciri utama seperti log transaksi automatik melalui imbasan resit menggunakan Azure Form Recognizer, penetapan bajet bulanan mengikut kategori, laporan ringkasan perbelanjaan, peringatan melalui notifikasi, dan chatbot AI yang dikuasakan oleh Gemini API bagi memberikan panduan kewangan secara masa nyata. Aplikasi ini dibangunkan menggunakan rangka kerja Flutter dan bahasa pengaturcaraan Dart bagi memastikan prestasi yang lancar pada peranti Android. Firebase digunakan sebagai platform storan data yang selamat dan hos aplikasi, membolehkan akses maklumat pengguna secara masa nyata. Bagi menilai keberkesanan sistem, ujian fungsi dan bukan fungsi telah dijalankan, termasuk ujian kebolehgunaan yang melibatkan seramai 25 orang peserta. Hasil ujian menunjukkan bahawa 100% pengguna bersetuju aplikasi ini mudah digunakan, manakala 92% mendapati ciri imbasan resit membantu dalam memudahkan penjejakan kewangan. Selain itu, 80% pengguna melaporkan bahawa chatbot AI memberikan panduan kewangan masa nyata yang berguna. Secara keseluruhan, projek ini bertujuan untuk mengatasi kelemahan kaedah bajet tradisional dan aplikasi kewangan sedia ada, serta membantu pengguna membentuk tabiat kewangan yang lebih baik dan mencapai matlamat kewangan mereka dengan lebih berkesan.

Table of Contents

DECLARATION i

ACKNOWLEDGEMENT ii

ABSTRACT iii

CHAPTER 1: INTRODUCTION 1

1.1 Introduction 1

1.2 Problem Statement 3

1.3 Objectives 3

1.4 Methodology 4

1.5 Project Scope 5

1.6 Significance of Project 6

1.7 Project Schedule 7

1.8 Expected Outcome 8

1.9 Project Outlines 8

CHAPTER 2: BACKGROUND STUDY OR LITERATURE REVIEW 11

2.1 Introduction 11

2.2 Review of Similar Existing Systems/ Application 11

2.2.1 Goodbudget 11

2.2.2 You Need A Budget (YNAB) 15

2.2.3 Bluecoins 18

2.3 Comparison of Features among Different Systems 22

2.4 Receipt scanning through Azure Form Recognizer 23

2.5 AI Chatbot Support 24

2.6 Summary 25

CHAPTER 3: REQUIREMENTS ANALYSIS AND DESIGN 26

3.1 Introduction 26
3.2 Project Development Methodology 26
3.2.1 Prototype Model 26
3.3 Requirement Analysis of the Proposed System 27
3.3.1 Analysis of the Current Personal Finance Tracking Situation 27
3.3.2 Analysis of the Proposed Mobile Application 27
3.4 System Requirements 28
3.4.1 Functional Requirements 28
3.4.2 Requirement Elicitation 31
3.4.3 Non-Functional Requirements 52
3.5 System Design 52
3.5.1 Use Case Diagram 53
3.5.2 Activity Diagram 63
3.5.3 Sequence Diagram 70
3.6 User Interface Design 79
3.6.1 Login & Register Page 79
3.6.2 Home Page 80
3.6.3 Budget Page 81
3.6.4 Manual Entry for Expenses or Income Page 82
3.6.5 Automated Transaction Tracking Via Receipt Scanning 83
3.6.6 Transaction History Page 84
3.6.7 Report Page 85
3.6.8 Chatbot Page 86
3.6.9 Profile Page 87
3.7 Summary 87

CHAPTER 4: SYSTEM IMPLEMENTATION 88
4.1 Introduction 88
4.2 Hardware and Software Development Tools Requirements 88
4.2.1 Hardware Development Tools 88
4.2.2 Software Development Tools 90
4.3 System Architecture 95
4.3.1 Presentation Layer 96
4.3.2 Business logic Layer 97
4.3.3 Data Storage Layer 98
4.4 Future Saver User's Description 98
4.5 Future Saver Module Implementation 99
4.6 Interface of Future Saver 101
4.6.1 Login Page 101
4.6.2 Register Page 102
4.6.3 Reset Password Page 103
4.6.4 Home Page 104
4.6.5 Budget Page 105
4.6.6 Manually Entry Transaction Page 106
4.6.7 Receipt Scanning Transaction Page 107
4.6.8 Spending by Category Report Page 108
4.6.9 Income vs Spending Report Page 109
4.6.10 Chatbot Page 110
4.6.11 Profile Page 111
4.6.12 Notifications Page 112
4.7 Summary 113

CHAPTER 5: SYSTEM TESTING 114
5.1 Introduction 114
5.2 Functional Testing 114
5.2.1 Test Case for Register Functionality 115
5.2.2 Test Case for Register Functionality 118
5.2.3 Test Case for Login Functionality 123
5.2.4 Test Case for Login Functionality 125
5.2.5 Test Case for Reset Password Functionality 128
5.2.6 Test Case for Budget Functionality 131
5.2.7 Test Case for Budget Functionality 135
5.2.8 Test Case for Manually Entry Functionality 138
5.2.9 Test Case for Manually Entry Functionality 144
5.2.10 Test Case for Receipt Scanning Functionality 153
5.2.11 Test Case for Receipt Scanning Functionality 156
5.2.12 Test Case for Spending by Category Report Functionality 158
5.2.13 Test Case for Income vs Spending Report Functionality 162
5.2.14 Test Case for Chatbot Functionality 166
5.2.15 Test Case for Profile Functionality 169
5.2.16 Test Case for Profile Functionality 173
5.3 Non-Functional Testing 177
5.3.1 Performance Testing 177
5.3.2 Usability Testing 182
5.3.3 Reliability Testing 185
5.4 User Acceptance Testing (UAT) 188
5.5 Summary 200

CHAPTER 6: CONCLUSION AND FUTURE WORK 201

6.1 Conclusion 201

6.2 Achievement of Objectives 201

6.3 Limitations 203

6.4 Recommendations and Future Work 204

References 206

Appendix A: Questionnaire 209

Appendix B: User Acceptance Testing (UAT) Questionnaire for Future Saver 218

LIST OF TABLES

Table 2.1: Comparison Between Three Similar Existing System 22

Table 3.1: Use Case Specification for Register New User 54

Table 3.2: Use Case Specification for Login Existing User 55

Table 3.3: Use Case Specification for Adding New Budget 56

Table 3.4: Use Case Specification for Adding New Transaction 57

Table 3.5: Use Case Specification for Viewing Transaction History 58

Table 3.6: Use Case Specification for Generating Report 59

Table 3.7: Use Case Specification for Interacting with AI Chatbot 60

Table 3.8: Use Case Specification for Viewing Profile 61

Table 3.9: Use Case Specification for Viewing Notification 62

Table 4.1: Hardware Requirements of Laptop 89

Table 4.2: Hardware Requirements of Mobile Device 89

Table 4.3: Software Requirements 90

Table 4.4: File Structure of Future Saver App 97

Table 4.5: Future Saver App's Module 99

Table 5.1: Test Case for Register Functionality 115

Table 5.2: Test Case for Register Functionality 118

Table 5.3: Test Case for Login Functionality 123

Table 5.4: Test Case for Login Functionality 125

Table 5.5: Test Case for Reset Password Functionality 128

Table 5.6: Test Case for Budget Functionality 131

Table 5.7: Test Case for Budget Functionality 135

Table 5.8: Test Case for Manually Entry Functionality 138

Table 5.9: Test Case for Manually Entry Functionality 144

Table 5.10: Test Case for Receipt Scanning Functionality 153

Table 5.11: Test Case for Receipt Scanning Functionality 156

Table 5.12: Test Case for Spending by Category Report Functionality 158

Table 5.13: Test Case for Income vs Spending Report Functionality 162

Table 5.14: Test Case for Chatbot Functionality 166

Table 5.15: Test Case for Profile Functionality 169

Table 5.16: Test Case for Profile Functionality 173

Table 5.17: Test Case of Performance Testing 178

Table 5.18: Test Case for Usability Testing 182

Table 5.19: Test Case of Reliability Testing 185

LIST OF FIGURES

Figure 1.1: Number of mobile users worldwide 1

Figure 1.2: Prototyping Methodology 4

Figure 1.3: FYP Project Schedule 7

Figure 2.1: Goodbudget Mobile Application 11

Figure 2.2: Screenshot of Envelopes Page 12

Figure 2.3: Screenshot of Transaction Page 13

Figure 2.4: Screenshot of Report Page 13

Figure 2.5: YNAB Mobile Application 15

Figure 2.6: Screenshot of Budget Page 15

Figure 2.7: Screenshot of Transaction Page 16

Figure 2.8: Screenshot of Report Page 17

Figure 2.9: Screenshot of YNAB chatbot 18

Figure 2.10: Bluecoins Mobile Application 18

Figure 2.11: Screenshot of Budget Page 19

Figure 2.12: Screenshot of Transaction Page 20

Figure 2.13: Screenshot of Reminder Page 20

Figure 2.14: Screenshot of Report Page 21

Figure 3.1: The Questionnaires: What is your primary goal of using a finance tracking tool? 32

Figure 3.2: The Questionnaires - Which tools do you currently use to track your expenses and income? 33

Figure 3.3: The Questionnaires - How often do you track your expense? 34

Figure 3.4: The Questionnaires - Do you currently have a budget for your monthly expenses? 35

Figure 3.5: The Questionnaires - Have you ever faced any challenges in manging your finances? 36
Figure 3.6: The Questionnaires - What are the main challenges you face in managing finances? 37
Figure 3.7: The Questionnaires: How often do you forgot to log your expenses manually? ..38
Figure 3.8: The Questionnaires – Have you spent over your planned budget? 39
Figure 3.9: The Questionnaires - How Comfortable are you using technology for personal finance management? 39
Figure 3.10: The Questionnaires - How important for the app to help you to manage personal finance? 40
Figure 3.11: The Questionnaires - What device would you prefer to use the finance tracking app? 41
Figure 3.12: The Questionnaires - Would you prefer a free app or a paid subscription-based app? 42
Figure 3.13: The Questionnaires - How important is a visually appealing design in a finance tracking app for you? 43
Figure 3.14: The Questionnaires - What type of expenses category would you like to have? 44
Figure 3.15: The Questionnaires - What income category would you like to have? 45
Figure 3.16: The Questionnaires - What features would you like to see in a personal finance tracking app? 46
Figure 3.17: The Questionnaires - Would you like to receive the reminders for expenses tracking and budget limit warnings? 47
Figure 3.18: The Questionnaires - How often would you like to receive notifications or reminders from the app? 48

Figure 3.19: The Questionnaires - How important is it for the app to provide automated receipt scanning through Azure Form Recognizer API for logging expenses? 49
Figure 3.20: The Questionnaires - Do you prefer visual insights (e.g., charts, graphs) for financial analysis? 50
Figure 3.21: The Questionnaires - How important is it for the app to provide AI chatbot for real-time financial guidance? 51
Figure 3.22: Use Case Diagram 53
Figure 3.23: Activity Diagram of Register and Login Page 63
Figure 3.24: Activity Diagram of Budget Page 64
Figure 3.25: Activity Diagram of Add Transaction Page 65
Figure 3.26: Activity Diagram of View Transaction History Page 66
Figure 3.27: Activity Diagram of Report Page 67
Figure 3.28: Activity Diagram of Chatbot Page 68
Figure 3.29: Activity Diagram of Profile Page 69
Figure 3.30: Sequence Diagram of Register Page 70
Figure 3.31: Sequence Diagram of Login Page 71
Figure 3.32: Sequence Diagram of Budget Page 72
Figure 3.33: Sequence Diagram of Add Transaction Page 73
Figure 3.34: Sequence Diagram of View Transaction History Page 74
Figure 3.35: Sequence Diagram of Report Page 75
Figure 3.36: Sequence Diagram of Chatbot Page 76
Figure 3.37: Sequence Diagram of Profile Page 77
Figure 3.38: Sequence Diagram of Notification Page 78
Figure 3.39: Login & Register Page 79
Figure 3.40: Home Page 80

Figure 3.41: Budget Page 81
Figure 3.42: Manual Entry for Expenses or Income Page 82
Figure 3.43: Automated Transaction Tracking Via Receipt Scanning 83
Figure 3.44: Transaction History Page 84
Figure 3.45: Reports Summary Page 85
Figure 3.46: Chatbot Page 86
Figure 3.47: User Profile Page 87
Figure 4.1: Android Studio (Google Developers, 2024) 91
Figure 4.2: Flutter (Flutter, 2024) 92
Figure 4.3: Firebase (Google, 2024) 93
Figure 4.4: Architecture Diagram of Future Saver App 95
Figure 4.5: Home Page of Future Saver App 96
Figure 4.6: Firebase Firestore Database of Future Saver 98
Figure 4.7: Login Page of Future Saver 101
Figure 4.8: Register Page for Future Saver 102
Figure 4.9: Reset Password Page for Future Saver 103
Figure 4.10: Home page of Future Saver 104
Figure 4.11: Budget Page for Future Saver 105
Figure 4.12: Manually Entry Transaction for Future Saver 106
Figure 4.13: Receipt Scanning Transaction for Future Saver 107
Figure 4.14: Spending by Category Report for Future Saver 108
Figure 4.15: Income vs Spending Report for Future Saver 109
Figure 4.16: Chatbot Page for Future Saver 110
Figure 4.17: Profile Page for Future Page 111
Figure 4.18: Notifications Page of Future Saver 112

Figure 5.1: The Questionnaires - Did you experience any issues when adding, editing, or deleting budget categories?.....188
Figure 5.2: The Questionnaires - How easy was it to add, edit, or delete transactions? 189
Figure 5.3: The Questionnaires – Does the receipt scanning feature working as expected? Did it capture all the necessary details?.....190
Figure 5.4: The Questionnaires – Does the app accurately track and update your available budget amounts after adding transactions?.....191
Figure 5.5： The Questionnaires – How accurately do the total income, total expenses, and account balance update on the home page after adding a new transaction?.....192
Figure 5.6: The Questionnaires - Is the report of financial summaries easy to view and understand? 193
Figure 5.7: The Questionnaires - How helpful and relevant do you find the recommendations provided by the Future Saver AI Chatbot? 194
Figure 5.8: The Questionnaires - How clear and helpful were the error messages when erros occurred?.....195
Figure 5.9: The Questionnaires - How would you rate the app's speed when loading data or switching between screens? 196
Figure 5.10: The Questionnaires - Is the overall flow of using the app logical and easy to follow? 197
Figure 5.11: The Questionnaires - How easy is it to navigate through the modules in the Future Saver App?.....198
Figure 5.12: The Questionnaires – Do you feel that the app’s design (both layout and visual style, including colours, icons, and text size) is intuitive and easy to use? 199
Figure 5.13: The Questionnaires - Please rate your overall satisfaction with the Future Saver App.....200

CHAPTER 1: INTRODUCTION

1.1 Introduction

In the modern technology era, mobile application has become essential tools for managing personal finance which can help users to maintain the financial stability and independence. Effective personal finance management includes budgeting, saving, investing, and tracking expenses, which are crucial for achieving the long-term financial goals such as buying a home, traveling, or planning for retirement. According to a report by Statista, over 7.41 billion people worldwide now own mobile phones, with this number expected to reach to 7.49 million by 2025 (Taylor, 2023). The diagram below illustrates the number of users worldwide (Taylor, 2023).

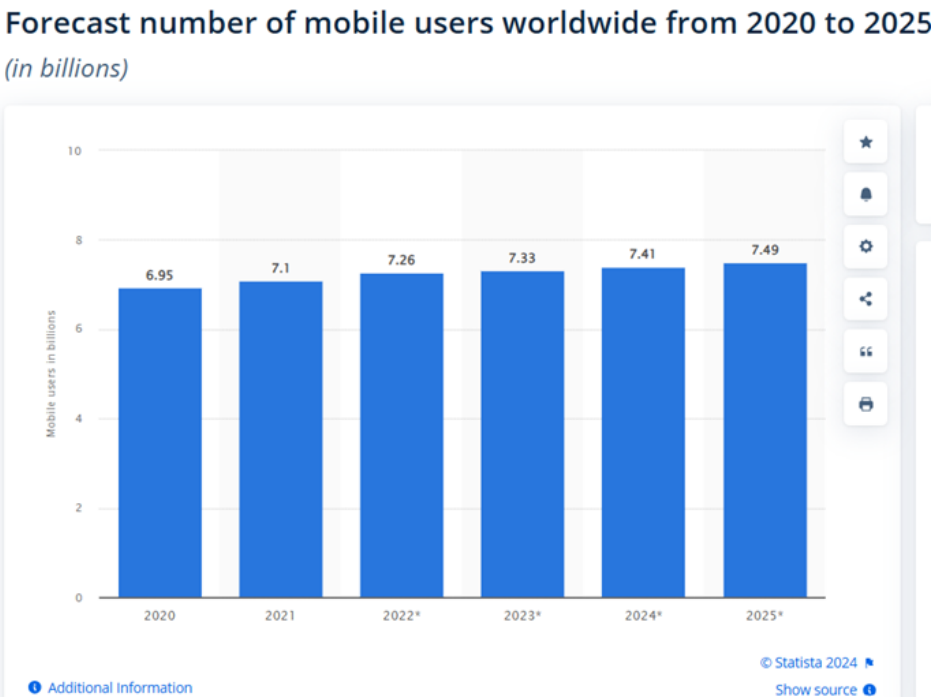


Figure 1.1: Number of mobile users worldwide

As smartphone usage increases, mobile apps have become a popular medium for personal finance management due to the accessibility and convenience. It provides users with instant, convenient access to track expenses, set budgets, and manage finances at anytime and anywhere. This accessibility allows for continuous, real time financial monitoring, helping

users avoid overspending and stick to their financial goals without the limitations of traditional manual tracking methods.

Adepegba et al. (2019) further explain that the increasing use of smartphones has encouraged many users to depend on these devices for managing financial transactions, providing a convenient platform to record and monitor daily expenses. This is further supported by Bhatt et al. (2024) which has shown that the digital budgeting tools significantly improve financial well-being by helping users to manage their finances easily, set budgets, avoid overspending, and develop a better saving habit. By using the mobile apps, users can easily track their income and expenses, provide a clearer view of their spending habits and enable more disciplined financial decisions as compared to the traditional methods such as using a notebook or Excel sheet to track the expenses manually. It can be time-consuming and lead to human mistakes.

The Smart Budget Personal Finance Tracking Mobile Application seeks to provide a user-friendly, efficient, and accessible solution for managing monthly expense. It allows users to track expenses effortlessly, set financial goals, and view spending patterns through insightful analytics. By establishing and working towards these goals, users can stay motivated and direct their finances toward meaningful achievements. This project intends to showcase how mobile apps like Smart Budget can help users to achieve a better financial control in order to stay within the budgets and improve the financial habits.

1.2 Problem Statement

Nowadays, most of the people still use the traditional methods like sticky notes, notebooks, and Excel spreadsheets to record and track their expenses (Wong & Tan, 2024). These methods can lead to mistakes, such as recording incorrect amounts, forgetting transactions and making calculation errors. It is also becoming difficult to track back past expenses when relying on paper records as it can be lost or damaged (Adepegba et al, 2019). Although Excel is a digital tool, it can be challenging for some users, especially elders, who may find it hard to understand and use the formulas for calculations. This makes Excel less accessible for those who want a simple way to manage their money. Besides that, most of the existing mobile budgeting apps requires users to enter every detail of their transactions manually, including the amount, name, and category. This manual entry process is time consuming, boring and increase the risk of errors (Wong & Tan, 2024). Moreover, current finance management applications do not have a virtual assistant that can provide real-time guidance and personalized financial insights to user. Without this support, user miss out on advice that could help them to make informed financial decisions and stay on track with their goals (Agarwal et al, 2024).

1.3 Objectives

a. To develop a user-friendly mobile application that allows users to easily track their daily expenses, while providing reminders through push notifications to help maintain their budgeting habits.
b. To implement automated data extraction from receipts to simplify the process of adding transactions.
c. To build an AI chatbot for user interaction that provides real-time financial suggestions and insights.

1.4 Methodology

In this project, a prototyping model is selected for developing this mobile application. The prototyping methodology is an approach in software development that focuses on creating an early model or prototype of a system to understand its functionality and gather user feedback before moving to full development (GeeksforGeeks, 2023). This methodology contains six stages which are requirements gathering, quick design, building a prototype, initial user evaluation, refining prototype, and system implementation.

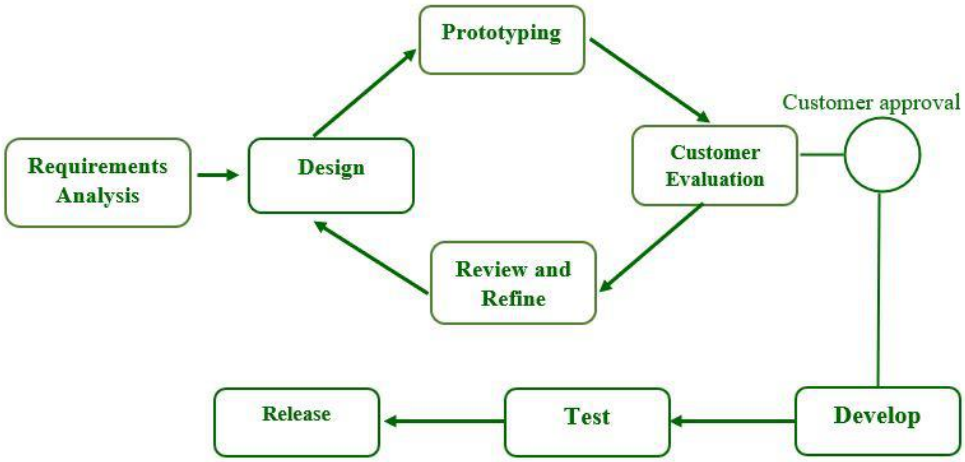


Figure 1.2: Prototyping Methodology

1. Requirement gathering is the first step in the prototyping model which involves defining the system's requirements. During this phase, user expectations and needs are identified through questionnaires, helping to create a clear foundation for development.
2. The second phase consist of a preliminary design or a quick design which provides a basic layout of the system. Though it is not a complete design, this stage gives users an initial overview of the system's structure and functionality.
3. Based on a quick design, an actual prototype is developed as a simple, functional model of the intended system.
4. The proposed system is then presented to the client for initial testing and evaluation. At this stage, users assess the prototype's functionality and provide feedback on its

strengths and areas for improvement. Client feedback and suggestions are gathered and forwarded to the developer.

5. If users identify areas that need improvement, the prototype is refined to address these insights. This cycle of refinement continues until users are satisfied with the model and approve it for further development.
6. Once the final version of the prototype is approved, it is developed into the complete system. The system undergoes rigorous testing before deployment to ensure stability and minimize the potential failures.

This methodology allows for continuous refinement based on user input, ensuring that the final application aligns closely with user expectations and needs.

1.5 Project Scope

The scope of this project is to develop an Android mobile application for personal finance management by using flutter and dart within the android studio. This app is designed for users who want to easily record the transactions and track their cash flow daily in order to maintain a clear and organized view of their finances. Key features of this mobile apps include tools like automated data extraction from receipt which allow users to capture the transaction details for accurate daily transaction logging and cash flow tracking. To enhance the user experience, personalized financial reminders will be delivered through the push notifications to keep the users informed and on track with their budget's goals. Additionally, it will include a built-in AI chatbot that provides real-time guidance, answer user questions, and provide personalized budgeting tips to help users in making the informed financial decisions. By focusing on accessibility, automation, and usability, this project aims to enable users to build a better financial habit, improve budgeting, and achieve long-term financial stability. By addressing the limitations of traditional budgeting and existing finance apps, this project seeks to deliver a more comprehensive and user-friendly solution for managing personal finances.

1.6 Significance of Project

A key significance of this project is its ability to replace the traditional and manual methods of tracking daily expenses with a modern mobile application which make the financial management simple and more efficient. By digitizing the expense tracking, the app reduces the time-consuming work and error of manual entry which can provide users with an accessible and efficient way to monitor their finances daily. Additionally, this mobile app contributes to the field of computer science, particularly in the areas of mobile application development and human-computer interaction (HCI) by examining how users interact with mobile financial tools. By prioritizing ease of use and accessibility, it aims to enable users to manage their finances more effectively.

Besides that, the Smart Budget Personal Finance Tracking Mobile Application makes a valuable contribution to financial technology (fintech). This project uses modern technologies like Azure Form Recognizer to enable the automatic extraction of text from images, which can be particularly useful for expense tracking. By integrating this technology, Smart Budget will allow users to capture images of receipts and convert them into text entries which can streamline the tracking process. Additionally, the app includes an AI chatbot that enhance user interaction and support. This innovative use the artificial intelligence (AI) in fintech domain which offer new insights into the usability and accuracy of automated tools in the context of personal finance.

In conclusion, the Smart Budget Personal Finance Tracking Mobile Application not only advances the practical applications of computer science in everyday life but also contributes to fintech field by enhancing financial management and improving user experience through innovative technology.

1.7 Project Schedule

FYP Project Schedule

Project start: Wed, 10/9/2024

Display week: 1

TASK	PROGRESS	START	END
Phase 1: Proposal			
Submission of Approved Brief Proposal	100%	10/10/24	20/10/24
Feedback and Comment from Supervisor	100%	21/10/24	5/11/24
Submission of Full Proposal	100%	6/11/24	14/11/24
Phase 2: Chapter 1			
Introduction	100%	15/11/24	21/11/24
Phase 3: Chapter 2			
Background Study / Literature Review	100%	22/11/24	13/12/24
Phase 4: Chapter 3			
Requirement Analysis & Design	100%	14/12/24	5/1/25
Submission of FYP 1 Final report	100%	6/1/25	17/1/25
Phase 5: Chapter 4			
System Development	100%	23/2/25	15/5/25
Phase 6: Chapter 5			
System Testing	100%	16/5/25	31/5/25
Phase 7: Chapter 6			
Conclusion & Future Work	100%	16/5/25	31/5/25
Submission of draft FYP 2 report	100%	1/6/25	10/6/25
Submission of FYP2 report	100%	11/6/25	23/6/25
Preparation of Presentation	100%	24/6/25	29/6/25
Submission of amendment FYP 2 report	100%	30/6/25	28/7/25

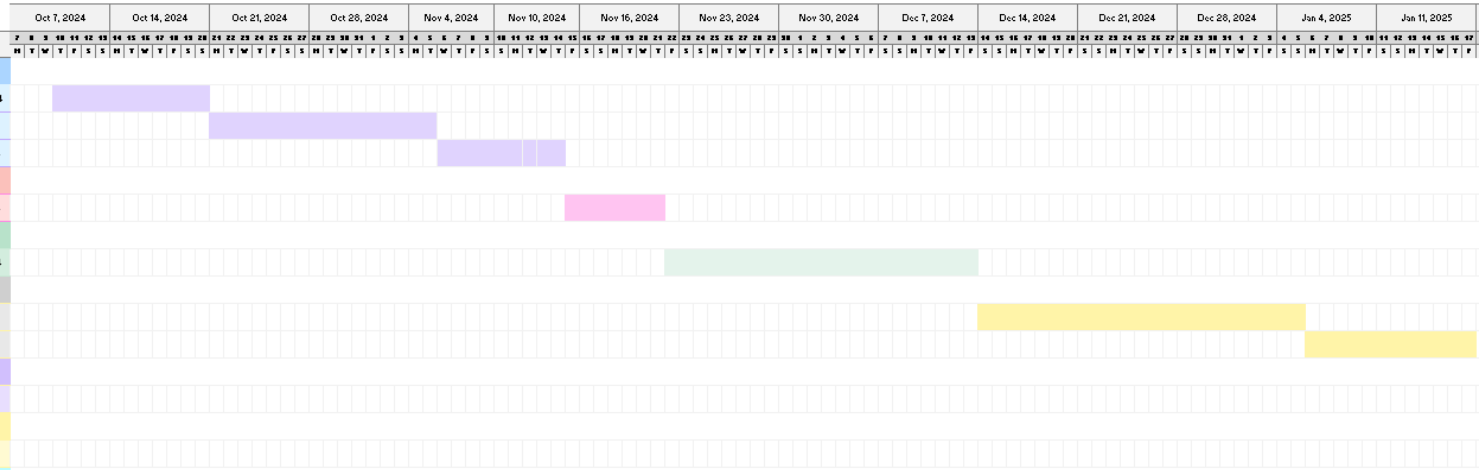


Figure 1.3: FYP Project Schedule

1.8 Expected Outcome

The finance mobile app is designed to be user-friendly which will make it easy for user to track their daily expenses, cash flow, and savings. It uses Azure Form Recognizer to automatically scan receipts and add transaction quickly which can reduce the need for manual data entry. Besides that, the app also features an AI chatbot to assist users with any questions whether related to specific transactions, budgeting advice or general app guidance. The chatbot provide support whenever it is need and help enhance the user experience. To help users on top of their finances, the app will also send reminders through push notifications. These notifications help user remember to record their expenses and stay on track of their budgeting goals, ensuring no transactions are missed.

1.9 Project Outlines

This project is structured into six chapters as outlined below.

Chapter 1: Introduction

This chapter provides a comprehensive overview of the Smart Budget Personal Finance Tracking Mobile Application and highlights its importance in helping users to manage their finances effectively. It includes the problem statement, identifying challenges with traditional methods and existing mobile application of recording daily expenses. The chapter outlines the objectives of the project, such as simplifying expense tracking, improving financial awareness, and supporting better decision-making. Additionally, the scope of the project is defined, including its target users and core feature of the application. The significance of the project is also discussed, highlighting its contribution to the field of mobile development and financial technology (fintech). Finally, the chapter includes the methodology, project schedule and the expected outcome.

Chapter 2: Background Study or Literature Review

This chapter provides a review of existing research and system which related to the personal finance management by analysing their strengths and weakness. It examines the related studies to identify the challenges faced by the users in financial tracking and highlights the limitations of the current solutions. It justifies the need for a smart, user-friendly app designed for efficient budgeting and introduce features like image processing for receipt scanning and AI chatbot.

Chapter 3: Requirement Analysis and Design

This chapter outlines the requirements for the application, including both functional and non-functional aspects. It highlights the key considerations for the user interface design such as simplicity and accessibility. To provide a clear understanding of the application's functionality, this chapter includes the use case diagram, activity diagram and sequence diagram. It emphasizes the importance of user feedback in refining the application to ensure it effectively meets the user needs.

Chapter 4: System Implementation

This chapter focuses on developing the mobile application prototype during the implementation phase which based on the defined design specifications. It describes the tools, technologies, and programming frameworks used, including flutter, dart languages within the android studio. The chapter details the implementation of core feature of the mobile app such as budget setting, daily transaction logging, automated receipt scanning, monthly summaries report and AI chatbot.

Chapter 5: System Testing

This chapter focuses on the testing phase, which includes both functional and non-functional testing. Functional testing focuses on verifying that the application's features work as expected. Non-functional testing assesses performance, reliability, and user experience to

ensure efficient application operation. Additionally, user feedback is gathered to ensure that the application meets functionality, performance, and usability standards.

Chapter 6: Conclusion and Future Work

This chapter summarizes the project's outcomes, highlighting how the Smart Budget Personal Finance Tracking Mobile Application achieves its goals of simplifying expense management and improving budgeting. It also discusses the limitations encountered during development. Finally, the chapter concludes with recommendations for future enhancements to further improve the application’s functionality and user experience.

CHAPTER 2: BACKGROUND STUDY OR LITERATURE REVIEW

2.1 Introduction

This chapter provides a literature review on the existing mobile application systems which relevant to the development of Future Saver, a Smart Budget Personal Finance Tracking Mobile Application. The systems are evaluated and analysed based on their features, usability, strengths and weaknesses in order to develop a new and improved personal finance tracking mobile application.

2.2 Review of Similar Existing Systems/ Application

This section examines three similar personal finance tracking mobile applications systems on Google Play Store: Good budget, YNAB, and Bluecoins.

2.2.1 Goodbudget

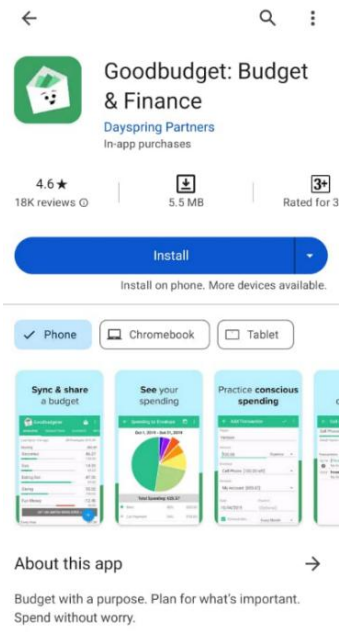


Figure 2.1: Goodbudget Mobile Application

Goodbudget, previously known as the Easy Envelope Budget Aid (EEBA). It is a financial mobile application which developed by Dayspring (Dayspring Partners, 2021).

Goodbudget is designed for household budget planning, and it is a digital adaptation of the envelope budgeting method (Dayspring Partners, 2021).

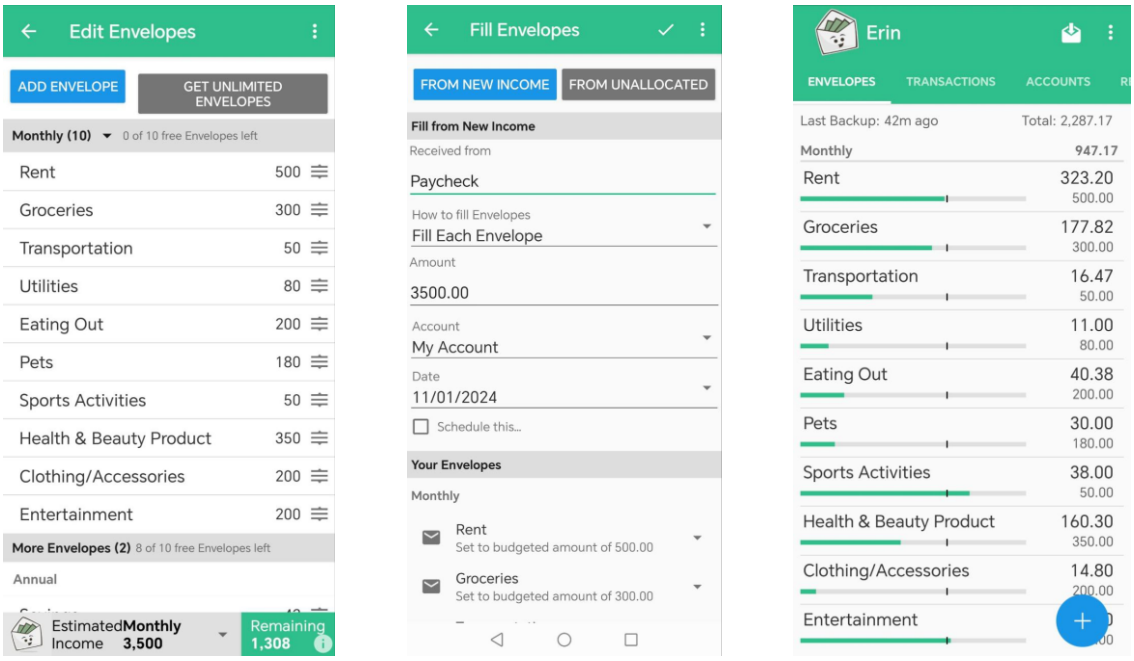


Figure 2.2: Screenshot of Envelopes Page

The envelope budgeting method simplifies budgeting by dividing the expenses into specific categories, referred to as envelope, such as groceries, eating out, and rent. Then, users can plan budgets for each envelope category. When users receive a paycheque, they can allocate funds to these virtual envelopes for different expense categories. This can help them to manage spending according to their predefined budgets. Once an envelope is empty, users are encouraged to stop spending in that category or transfer funds from another envelope (Luthi, 2022).

Figure 2.3: Screenshot of Transaction Page

When users make or receive a payment, they must enter the transaction manually into the app. After that, the transaction is recorded and categorized into its respective category, as shown in the diagram above. This ensures that all financial activities are organized for easy tracking and management. Additionally, the app includes a search function that allows users to find specific transactions using a filter feature.

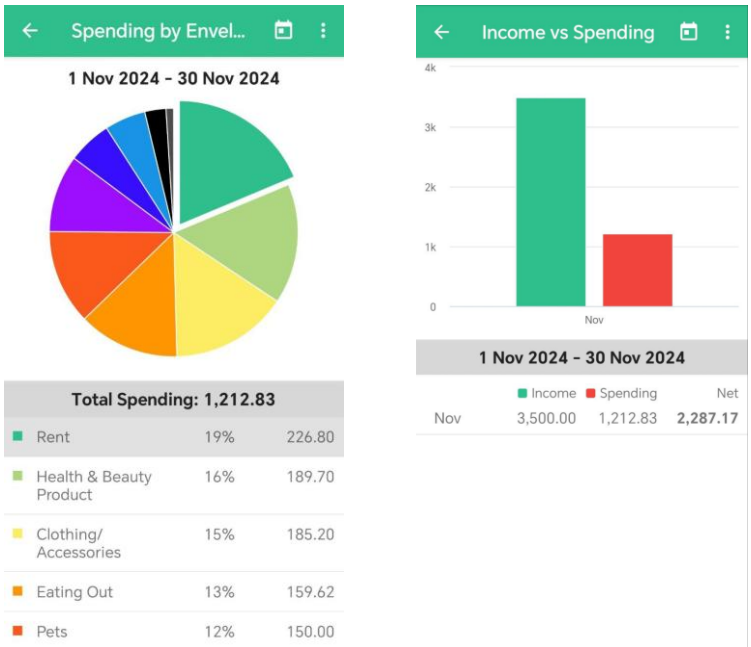


Figure 2.4: Screenshot of Report Page

One of the Goodbudget's key strength is its ability to provide a detailed charts and reports, including spending by envelope and Income vs Spending. These tools provide a clear insight into financial habits, highlighting the category where users spend the most and helping them to make an informed decision (Luthi, 2022). Another advantage of the app is its simplicity and ease of use which make it accessible for users without any financial knowledge. Besides that, Goodbudget helps users to determine how much money is needed to allocate for each envelope category which can help them to stay within their planned budgets and meet their financial goals. Furthermore, it includes a data back-up feature to prevent data loss and ensure the security of financial records.

One limitation of Goodbudget is it requires users to manually input and update the transaction daily, which can be inconvenient and time-consuming. It does not integrate with bank accounts to automatically import the transactions (Luthi, 2022). Additionally, the free version limits users to a maximum number of 10 envelopes only. If user want to access the unlimited envelopes, it requires a premium subscription which costs RM34.99 per month or RM309.99 per year. This cost may be considered expensive by some users. Another limitation is the app's user interface is less visually appealing and lacks icons alongside text which make the navigation less intuitive.

2.2.2 You Need A Budget (YNAB)

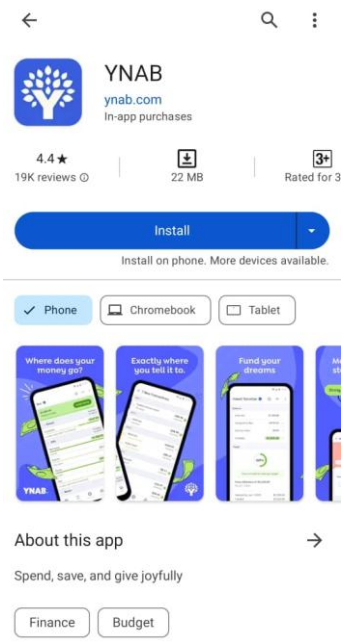


Figure 2.5: YNAB Mobile Application

YNAB which stands for "You Need A Budget". It is a personal budgeting software which was founded in 2004 by husband-and-wife team, Julie and Jesse to help users to take control of their finances (YNAB, 2024).

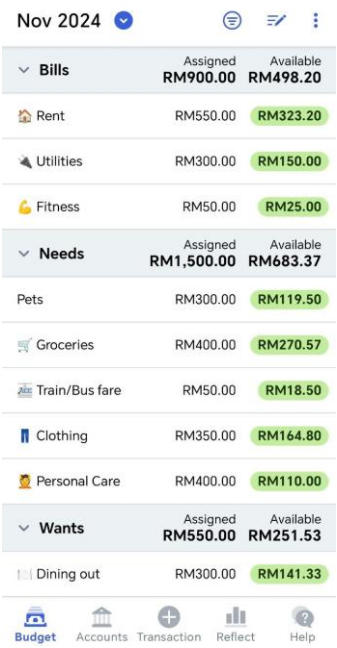


Figure 2.6: Screenshot of Budget Page

YNAB uses a zero-based budgeting system, a method where expenses are equal to income (Tindall & Neidel, 2024). It encourages users to allocate every dollar of income to specific expenses. It organizes users' finances into three categories — bills, wants, and needs. Within these categories, users can assign specific budgets for individual line items, allowing for a tailored money management approach. User can then track and adjust their spending throughout the month to stay on budget.

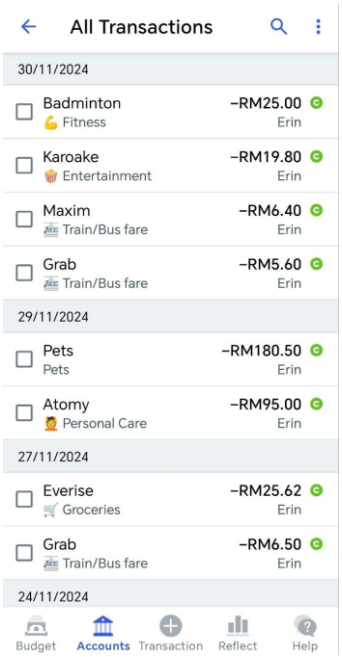


Figure 2.7: Screenshot of Transaction Page

All transactions manually entered by users are automatically recorded and categorized into their respective categories. This feature ensures that each transaction is accurately assigned to the appropriate budget category which make it easier for users to track their expenses and maintain an organized financial overview. Additionally, the app includes a search function that allows users to find specific transactions easily.

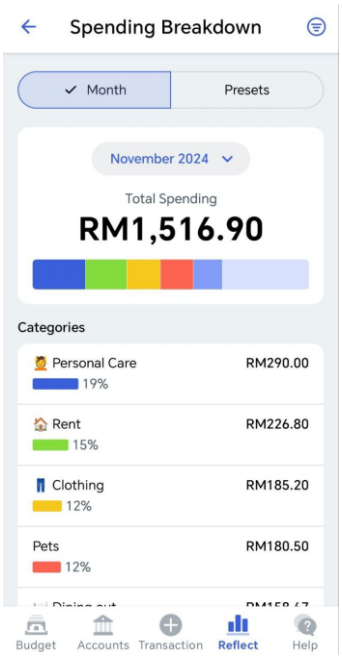


Figure 2.8: Screenshot of Report Page

One of the key strengths of YNAB is it can generate a detailed progress bar which can help users to analyse their spending habits effectively. The app offers users the flexibility to manage their finances either by manually entering each transaction or linking supported bank and credit card accounts for automatic transaction imports. YNAB provides intuitive visual tools that allow users to track their spending easily and monitor how much money remains in each spending category. Apart from that, the app includes a chatbot feature that allow users to interact with an AI assistant for spending-related questions. If the AI assistant cannot provide with the desired answer, users can reach out to the support team through email for further assistance. The diagram below illustrates YNAB's chatbot interface.

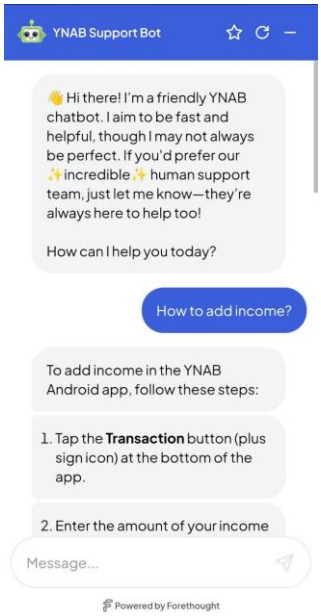


Figure 2.9: Screenshot of YNAB chatbot

One limitation of YNAB is that it is a paid budgeting app with no free version. New users can start with a free trial for 34 days to explore its feature. After the trial period, it costs $14.99 per month or $109 per year which may be expensive for some users (Tindall & Neidel, 2024). Its unique zero-based budgeting approach can be challenging for some new users to understand, especially those who are unfamiliar with this method of budgeting.

2.2.3 Bluecoins

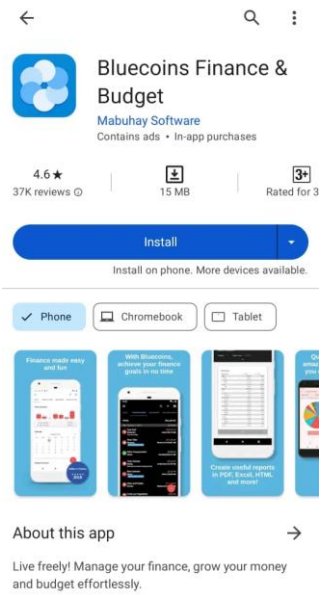


Figure 2.10: Bluecoins Mobile Application

Bluecoins is a simple personal finance management mobile application which is developed by Mabuhay Software (Appedus, 2021). It serves as an effective tool for user to set budgets, monitor their income and track expenses.

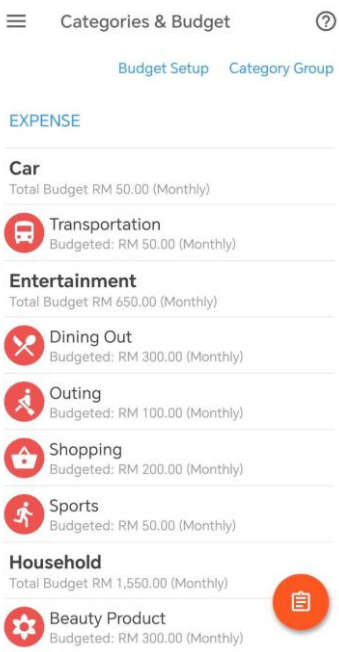


Figure 2.11: Screenshot of Budget Page

Bluecoins enables users to set budgets either at the category group level or the individual category level. For example, users can allocate a budget for a broader category group like 'Entertainment' or specify budgets for individual subcategories such as 'Dining Out' and 'Shopping.' This flexibility allows users to customize their budgeting strategy to align with their financial needs and goals effectively.

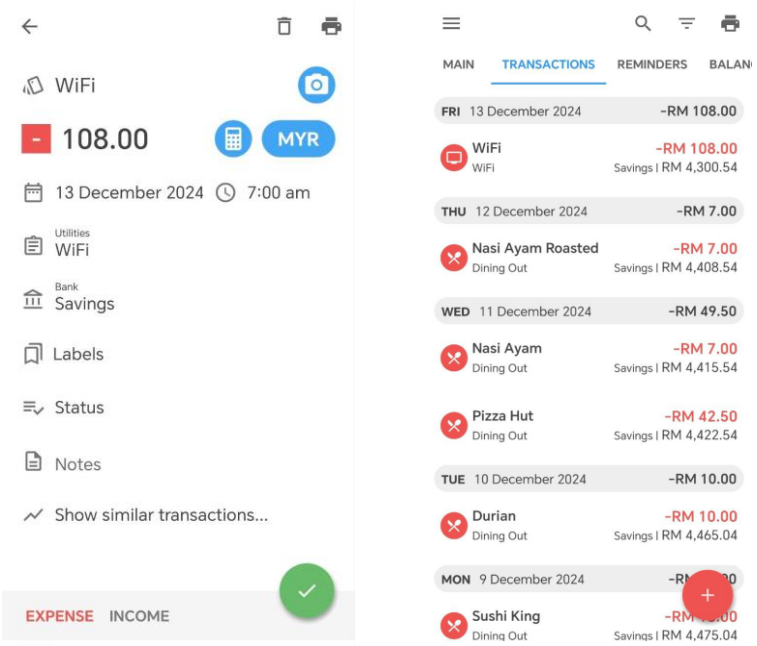


Figure 2.12: Screenshot of Transaction Page

When adding a new transaction for expense or income, Bluecoins allow users to attach receipt in the form of document, PDF or photos. This feature is a strength of the app which enable users to keep the digital record of their transaction. Additionally, the app can function offline which enable users to manage their finance anytime and anywhere without needing an internet connection.

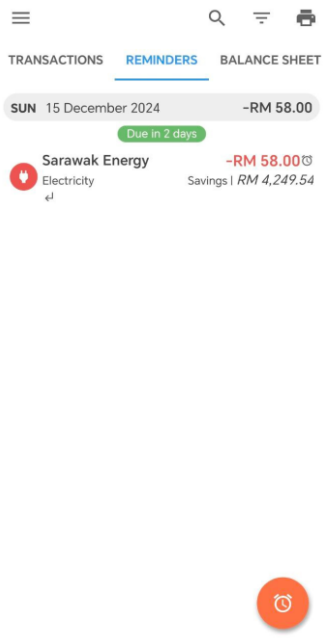


Figure 2.13: Screenshot of Reminder Page

Besides that, Bluecoins allows users to set daily reminders for expense logging. It will notify users through push notifications if no transactions have been recorded within the last 24 hours. The app also supports scheduled transactions, automatically logging them on the due date for convenience. To ensure consistent tracking, Bluecoins provides advanced filtering options which make it easier for users to categorize and review their financial data.

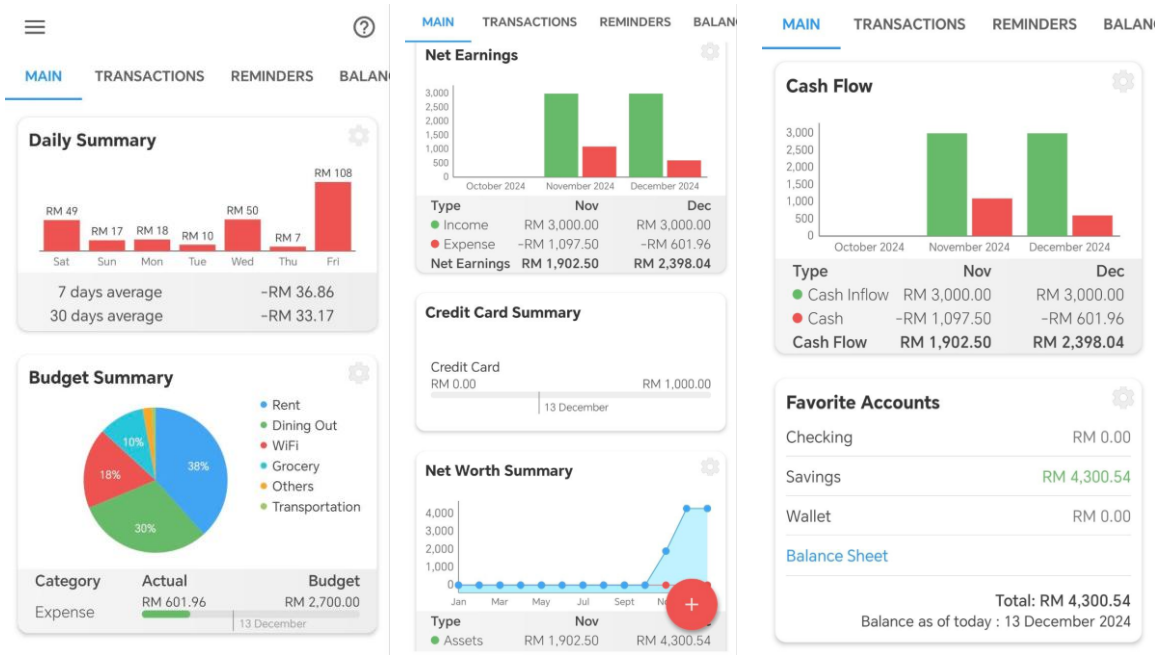


Figure 2.14: Screenshot of Report Page

Bluecoins generates a detailed report such as daily expense summary, budget summary, cash flow analyses, and net worth calculations. These insights make it easier for users to track their financial activities effectively, identify spending patterns, and understand their overall financial health. For enhanced security, users can set passwords or enable fingerprint authentication. One of the main advantages of Bluecoins is it offers a free version with essential features, and a premium option is available through a one-time payment of RM40. This makes it cost-effective as compared to the subscription-based apps.

One limitation of Bluecoins is it does not integrate directly with bank account which require user to manually input the transactions. Bluecoins provides the option to back up and sync data to Google Drive or Dropbox, but this feature is only available in the premium version.

Additionally, feature like exporting reports in various formats, such as PDF, Excel, and HTML, or sending reports directly to a printer for users who need the hard copies of their financial records is also exclusive to the premium option. The free version includes advertisements, which may affect the user experience.

2.3 Comparison of Features among Different Systems

When comparing the features of Goodbudget, YNAB and Bluecoins with Future Saver, it is important to consider the specific needs and requirements of users. Table 1 provides a detailed comparison of these three existing mobile application systems with Future Saver.

Table 2.1: Comparison Between Three Similar Existing System

Features	Goodbudget	YNAB	Bluecoins	Proposed System: Future Saver
Able to set budget	✓	✓	✓	✓
Budget editing	✓	✓	✓	✓
Budget deleting	✓	✓	✓	✓
Able to record income and expense	✓	✓	✓	✓
Transaction editing	✓	✓	✓	✓
Transaction deleting	✓	✓	✓	✓
Transaction filtering	✓	✓	✓	✓
Transaction history tracking	✓	✓	✓	✓
Receipt scanning through Azure Form Recognizer	✗	✗	✗	✓
Push notification reminder	✗	✓	✓	✓
Able to view summary report	✓	✓	✓	✓
AI chatbot support	✗	✓	✗	✓
Free of charges	✗	✗	✗	✓

2.4 Receipt scanning through Azure Form Recognizer

The Future Saver application integrates Azure Form Recognizer to automate receipt scanning which offer clear advantages over traditional Optical Character Recognition (OCR) tools. While conventional OCR systems like Tesseract can extract raw text from images, they often struggle with layout inconsistencies and require extensive post-processing to identify relevant fields such as transaction date, merchant name, or total amount. In contrast, Azure Form Recognizer uses deep learning and pre-trained models specifically designed for structured documents which allow it to extract key-value pairs directly from receipts and return the output in a structured JSON format. A recent industry analysis by IntuitionLabs (2025) confirms that Azure's AI-powered OCR models consistently outperform open-source OCR in extracting structured receipt fields, especially when used in finance applications where accuracy is critical (IntuitionLabs, 2025).

One of the key advantages of using Azure Form Recognizer is its domain-specific intelligence. The prebuilt receipt model supports a wide range of formats, including both printed and handwritten receipts which enable seamless automation of financial record keeping tasks such as expense logging and transaction categorization. According to Muniyandi (2025), Azure Form Recognizer not only improves data extraction accuracy but also reduces the reliance on manual input, thereby minimizing errors and enhancing overall processing efficiency (Muniyandi, 2025). Additionally, Azure provides enterprise-grade security, scalability, and compliance features which make it particularly well-suited for managing sensitive financial data within mobile applications.

Furthermore, integration with Azure's cloud-based services allows developers to scale the system easily while maintaining consistent performance. Unlike open-source tools that require multiple separate components for parsing, validation, and error handling, Azure Form Recognizer offers an end-to-end solution with robust APIs and SDKs, reducing development

effort and ensuring maintainability. As outlined in Microsoft's official documentation (2024), the receipt model is purpose-built for expense tracking scenarios and allows rapid deployment without custom machine learning training (Microsoft Learn, 2024). In summary, Azure Form Recognizer was selected over traditional OCR solutions due to its higher accuracy, structured output format, domain-specific optimizations, and secure, scalable architecture which make it ideal for automating receipt processing in a finance-tracking mobile application.

2.5 AI Chatbot Support

The integration of an AI-powered chatbot into the Future Saver mobile application significantly enhances the user experience by providing real-time financial guidance and interactive support. This chatbot is built using the Gemini API, a large language model developed by Google, which enables natural and intelligent conversations with users. Through this API, the chatbot can understand and respond to personal finance-related queries such as "Tips on saving money?" AI chatbots like this have proven effective in improving user engagement and financial habits. According to Manoj (2025), AI chatbots contribute to improved user engagement and financial outcomes by offering timely insights that help users stay on track with their budgeting goals and make informed decisions. The study found that users who engaged with financial chatbots exhibited a 40% increase in budgeting consistency compared to those using conventional finance apps.

Beyond its intelligent responses, the chatbot also offers 24/7 availability, enabling users to access assistance instantly without the need for human intervention. This continuous accessibility not only enhances user satisfaction but also boosts operational efficiency for digital financial platforms (Eustaquio-Jiménez et al., 2024). A systematic review by Eustaquio-Jiménez et al. (2024) concluded that AI chatbots in financial services significantly improve responsiveness and overall customer service performance.

However, despite these benefits, certain limitations persist. Petrecca (2025) observed that financial chatbots may occasionally generate inaccurate responses, particularly when handling complex or ambiguous queries. This issue stems from the AI model's dependence on pattern recognition rather than human reasoning. Therefore, it is important to clearly define the chatbot's scope and incorporate disclaimers indicating that its guidance is not a substitute for professional financial advice. Furthermore, concerns regarding privacy and data security remain relevant. Belen et al. (2021) emphasized the necessity of transparent data usage policies to foster user trust, especially when sensitive financial information is involved.

In summary, the AI chatbot module within the Future Saver application serves as a powerful tool for supporting users in their financial management journey. Through real-time, personalized assistance and intelligent data integration, it enhances the overall functionality and value of the application. With clearly defined boundaries, strong privacy safeguards, and reliable backend systems, the chatbot can significantly contribute to a more informed and empowered user experience.

2.6 Summary

In summary, Future Saver is a comprehensive personal finance management tool that integrate the best features of Goodbudget, YNAB, and Bluecoins while overcoming their limitations. It offers advanced tracking tools, receipt scanning through Azure Form Recognizer, a user-friendly interface, and AI chatbot support to streamline budgeting and expense management. By addressing some of the limitations in the existing apps, such as manual transaction entry, Future Saver provide a more convenient, intuitive, and effective solution for users to manage their finances.

CHAPTER 3: REQUIREMENTS ANALYSIS AND DESIGN

3.1 Introduction

This chapter focuses on the system requirements analysis and design phase of the project. It uses the prototype model throughout the development process. It discusses both the functional and non-functional requirements of the personal finance tracking mobile application. The chapter also provides an overview of the mobile application's design, including its visual appearance and core functionalities. To illustrate interactions between users and the system, a use case diagram visually depicts how users engage with the app's features. An activity diagram is also included to provide a detailed flow of activities within the system. Furthermore, a sequence diagram is included to represent the step-by-step interaction between users and the system’s components.

3.2 Project Development Methodology

This section outlines the project's approach, with the prototype model serving as the methodology used to execute it.

3.2.1 Prototype Model

The prototype model is a software development approach focused on creating an initial version of the system, known as a prototype. This method is used to develop a prototype of Future Saver which incorporating its core features such as budget setting, transaction tracking, report generation, and a chatbot. The prototype allows users to test the app and provide valuable feedback. Based on the feedback, the design is iteratively refined and improved. This process is repeated until the final product meets the required standards and received approval.

3.3 Requirement Analysis of the Proposed System

This section discusses the system requirements analysis phase, which focuses on identifying the essential specifications for the proposed system. The analysis includes a thorough examination of the real-life challenges faced by users in recording daily expenses. It also researches how the proposed system addresses these issues in the current context. To gather relevant insights and define the system's requirements, surveys were conducted using Google Forms to ensure a comprehensive understanding of user needs and preferences.

3.3.1 Analysis of the Current Personal Finance Tracking Situation

The idea for this project comes from the observation that most of the current personal finance tracking mobile applications require users to manually enter each transaction. This process can be time-consuming and inconvenient for some users. Moreover, many of these apps lack advanced features, such as an AI assistant to provide real-time guidance. Without this support, users who are new to budgeting may find it is challenging for them to analyse their spending habits, set budgets, or make informed financial decisions. Additionally, they may struggle to receive instant feedback on their financial choices when needed.

3.3.2 Analysis of the Proposed Mobile Application

After analysing the problem, the proposed mobile application, Future Saver had been designed to solve the problems faced in the current method. The app is available on mobile devices, which makes it easily accessible for users. It allows users to set budgets for each expense and income category. This can help users avoid overspending. Additionally, users can choose to enter transactions manually or use automated expense tracking with receipt scanning through Azure Form Recognizer technology. This gives them the flexibility to select the option that is most convenient for them.

To further enhance financial management, the app generates detailed reports, including Spending by Category and Income vs Spending analysis which enable users to understand their financial habits. The app also includes an AI chatbot assistant to provide real-time guidance. To ensure a user-friendly experience, the app is designed to be intuitive and easy to navigate to make it simple for users to use. Overall, Future Saver would be a comprehensive and user-friendly mobile application that allows users to manage their finances effectively and develop a healthier financial habit.

3.4 System Requirements

The system requirements outline the key features and specifications needed for the successful development and functionality of the personal finance tracking mobile application. It includes both functional and non-functional requirements. Functional requirements describe the system's features, while non-functional requirements focus on aspects such as operational compatibility, system performance, and usability of the system. By establishing a clear system requirement, the project ensure that the mobile app meets the user needs, operates smoothly and achieves its intended purpose.

3.4.1 Functional Requirements

For the functional requirements of Future Saver, users can perform the functionalities as below:

Register and Login module

- The system shall allow new users to register accounts by providing required details such as name, email, and password.
- The system shall allow users to login to their accounts using their registered email and password.

- The system shall validate user credentials (email and password) against the stored database.
- The system shall display an appropriate error message if the email or password is incorrect.
- The system shall allow users to reset their password through “Forgot Password?” link.
- The system shall provide a ‘Remember Me’ feature that enables users to stay signed in, so they do not need to log in repeatedly when accessing the app.

Home Module

- The system shall allow users to access their profile or account settings through a designated edit icon.
- The system shall show the user’s account balance along with a summary of total income and expenses.
- The system shall display the transaction history section that highlights recent transactions.
- The system shall provide a “See All” button for users to view the complete transaction history.
- The system shall provide a notification button for users to access and view all the notifications.

Budget Module

- The system shall allow users to select the month and year for the budget view using a dropdown icon.
- The system shall show the assigned and available amounts for each category in the expense and income sections.

- The system shall provide an “Add” icon to allow users to add new expense or income categories.
- The system shall provide an edit icon for users to update the assigned amounts for existing categories.

Transaction Module

- The system shall allow users to choose between manual entry for expense and income or automated expense tracking using receipt scanning,
- The system shall allow users to enter transaction details manually.
- The system shall allow users to upload a photo of a receipt using camera for scanning.
- The system shall automatically extract the details such as name, amount, and date from the receipt and fill them into the respective fields.
- The system shall allow users to edit or delete transactions from the transaction history.

Report Module

- The system shall allow users to choose to view the report in Spending by Category or Income vs Spending.
- The system shall generate a pie chart report that showing Spending by Category.
- The system shall calculate and display the total spending for the selected date range, along with the percentage and total amount for each category.
- The system shall generate an Income vs Spending report in bar chart.
- The system shall calculate and display the net balance as the difference between total income and total spending for the selected date range.

AI Chatbot Module

- The system shall allow users to interact with the chatbot by typing messages in a text input field.
- The system shall allow users to send messages by pressing a “Send” button
- The system shall be able to process the user’s message and generate an AI-driven response based on the user input.

Profile Module

- The system shall display the user’s profile information, including their profile picture, username, phone number, and email address
- The system shall allow users to edit their profile details.
- The system shall allow users to enable or disable the notifications.
- The system shall allow users to log out of their account

3.4.2 Requirement Elicitation

In this project, questionnaires are used to gather data for the system requirements analysis. The target respondents are the members of the public, particularly individuals who need a simple way to record their expenses and track their daily cash flow. The survey was distributed to a total of 138 respondents. The questionnaire had been carefully designed to ensure it effectively collected the necessary information. It focuses on areas such as financial habits, management practices, and users’ preferences for personal finance tracking apps.

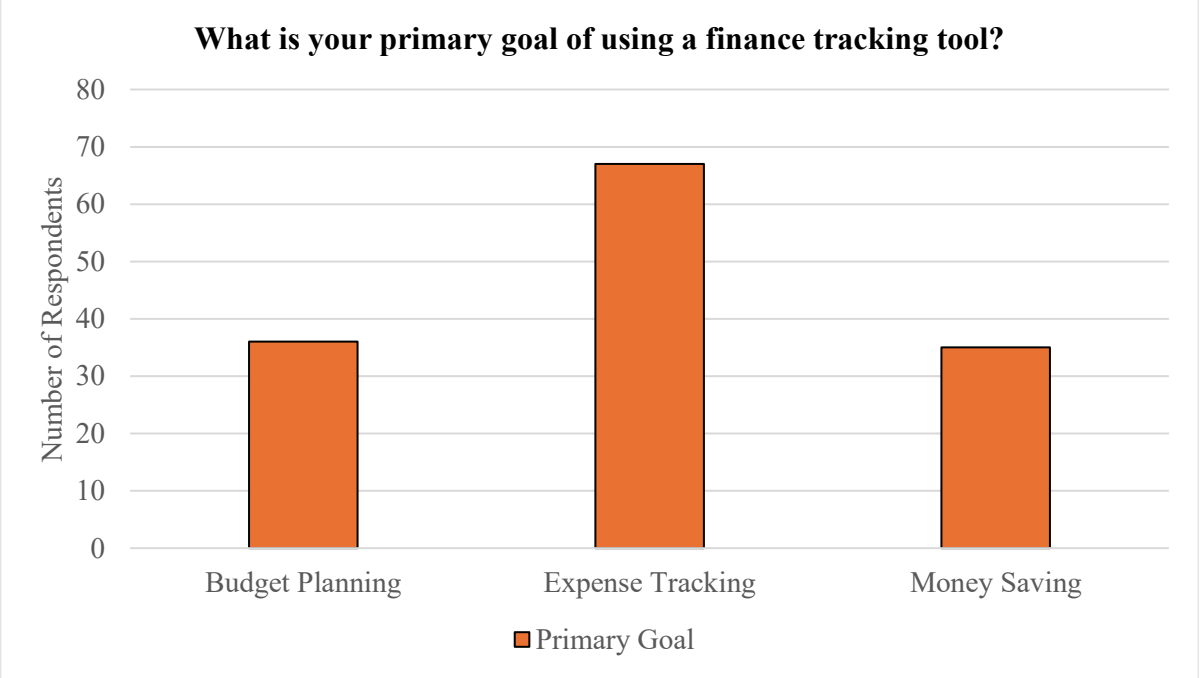


Figure 3.1: The Questionnaires: What is your primary goal of using a finance tracking tool?

The bar chart shows the primary goal of respondents in using a finance tracking tool. The most common purpose is "Expense Tracking," with 67 respondents indicating it as their primary goal, showing a strong preference for keeping tabs on daily spending. This is followed by "Budget Planning," selected by 36 respondents, and "Money Saving," chosen by 35 respondents. The data suggests that while budgeting and saving are important, users prioritize tracking their expenditures as the first step in managing their finances effectively. This highlights a clear focus on gaining visibility and control over spending habits.

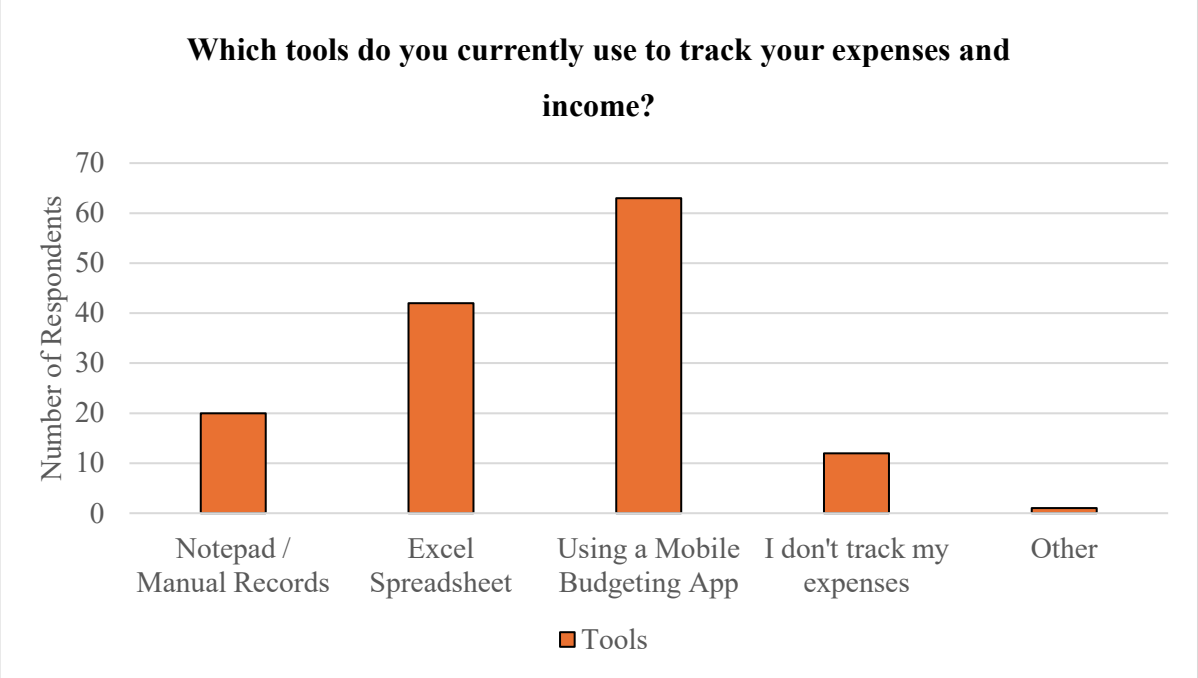


Figure 3.2: The Questionnaires - Which tools do you currently use to track your expenses and income?

The figure above shows the tools that most of the respondents currently use to track the expenses and income. The most popular tool is a mobile budgeting app which used by 63 respondents. This is followed by Excel Spreadsheets which was chosen by 42 respondents. 20 respondents use the notepad or manual records to record the expense while 12 respondents do not track their expenses at all. Only 1 respondent use other tools such as the mobile notepad to track the expense. This data highlights that mobile apps and Excel Spreadsheet are the most preferred tools for financial tracking.

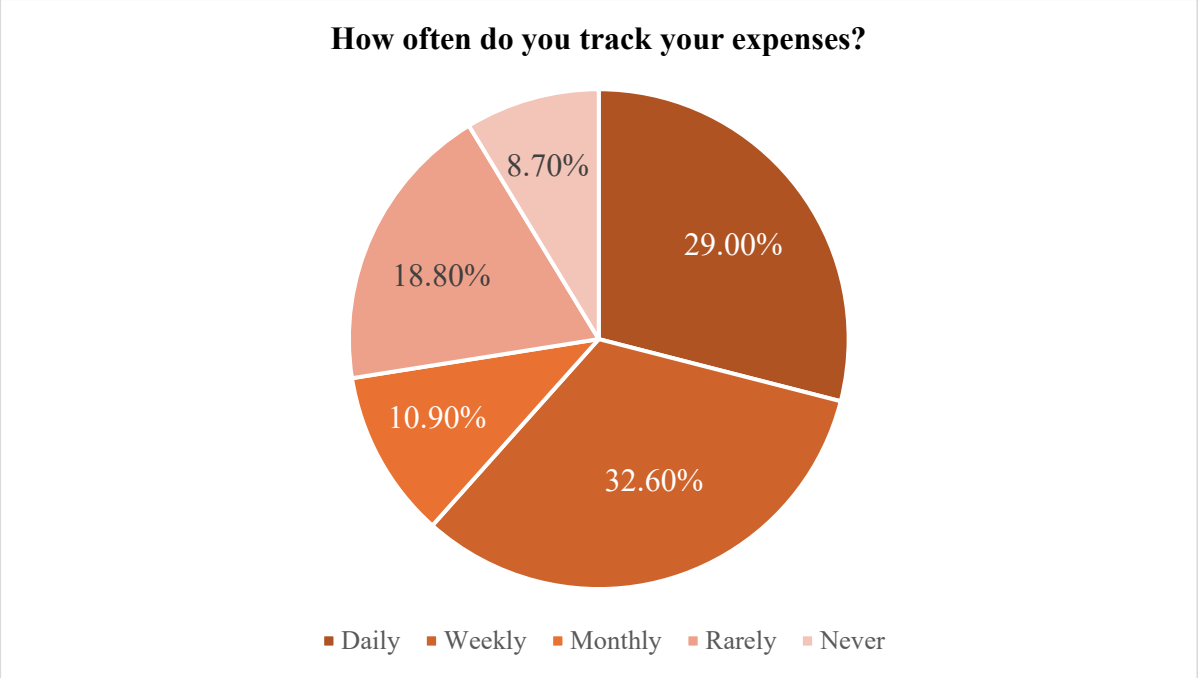


Figure 3.3: The Questionnaires - How often do you track your expense?

The pie chart of the figure above shows how often the respondents tracks their expenses. The majority, 32.60% (45 respondents) track their expenses weekly, followed by 29.00% (40 respondents) who track expenses daily which indicate a consistent habit among many users. A smaller percentage, 18.80% (26 respondents), track expenses rarely, suggesting occasional attention to finances. Meanwhile, 10.90% (15 respondents) review their expenses monthly, and 8.70% (12 respondents) never track their expenses at all. Overall, the data indicates that a majority of respondents maintain a regular tracking habit, with daily and weekly tracking being the most common practices for financial awareness.

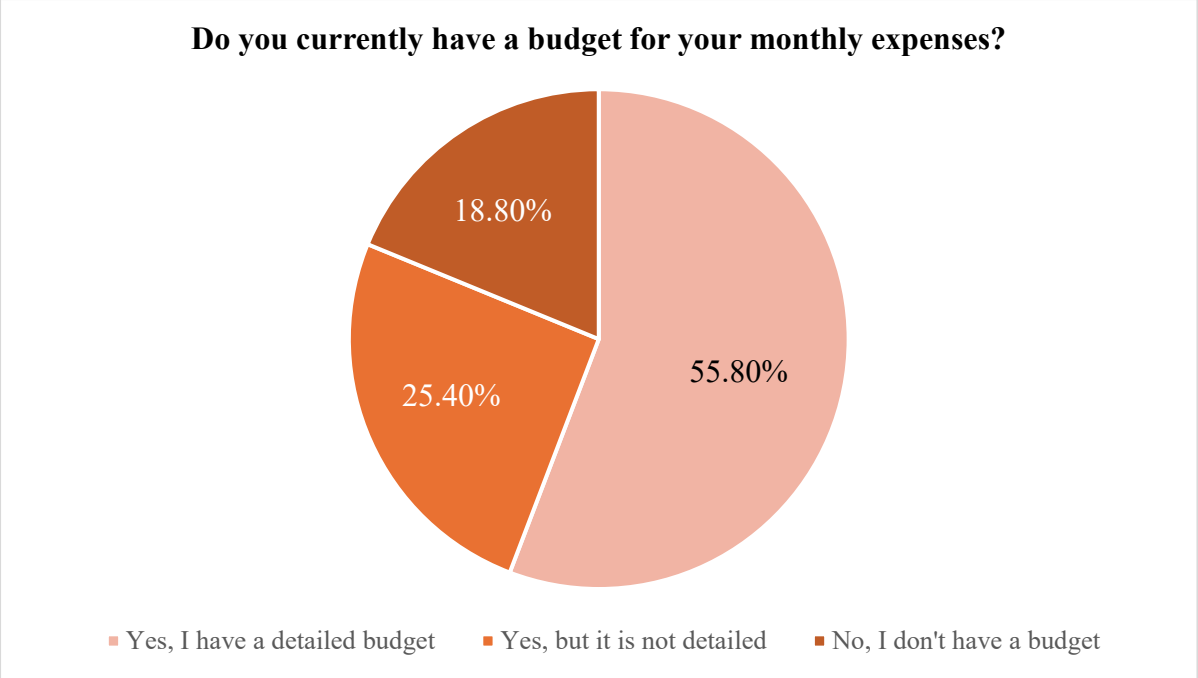


Figure 3.4: The Questionnaires - Do you currently have a budget for your monthly expenses?

The pie chart shows whether the respondents have a budget for their monthly expenses. The majority, 55.80% (77 respondents) have a detailed budget while 25.40% (35 respondents) have a budget but it is not a detailed one. In contrast, 18.80% of respondents do not have any budget in place. This data indicates that a large portion of respondents engage in some form of budgeting, with over half demonstrating a structured and detailed approach, suggesting a strong awareness and effort toward managing their monthly finances.

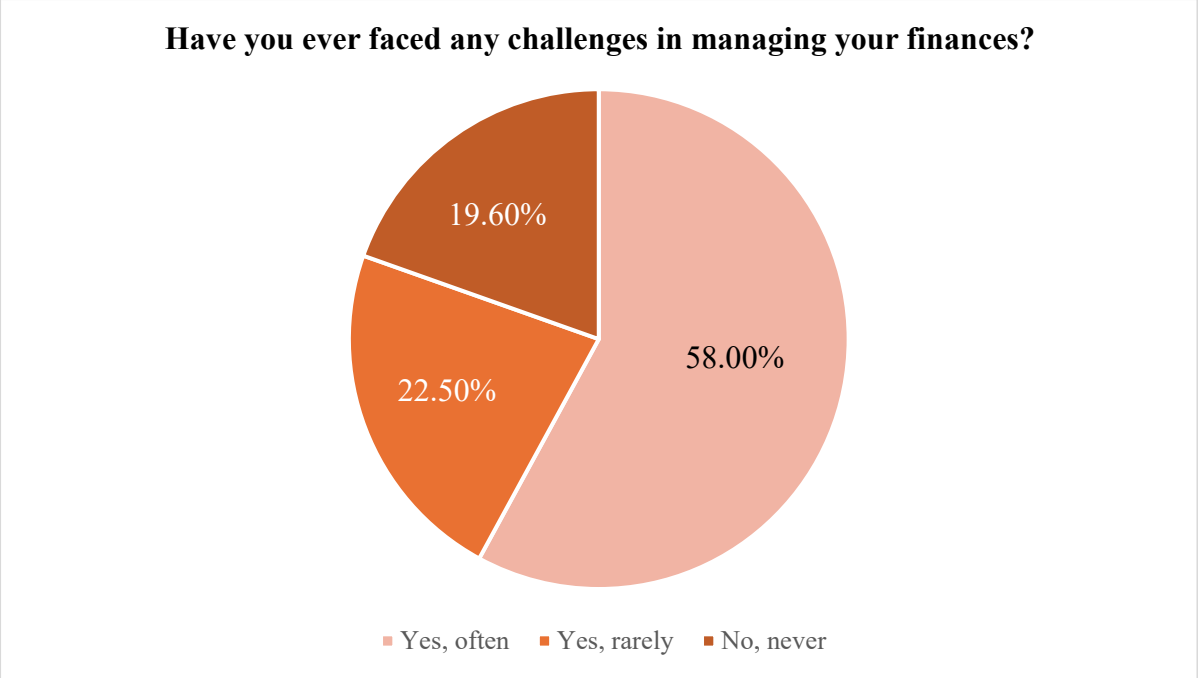


Figure 3.5: The Questionnaires - Have you ever faced any challenges in manging your finances?

The pie chart shows whether respondents have faced any challenges in managing their finances. The majority, 58.00% (80 respondents) reported often facing such challenges, highlighting that financial management is a common struggle. Meanwhile 22.50% (31 respondents) rarely encounter the difficulties in managing their finance. About 19.60% (27 respondents) have never faced challenges in managing their finances. This indicates that a significant portion of respondents frequently face financial difficulties which indicate a widespread need for more effective tools or guidance to support personal financial management.

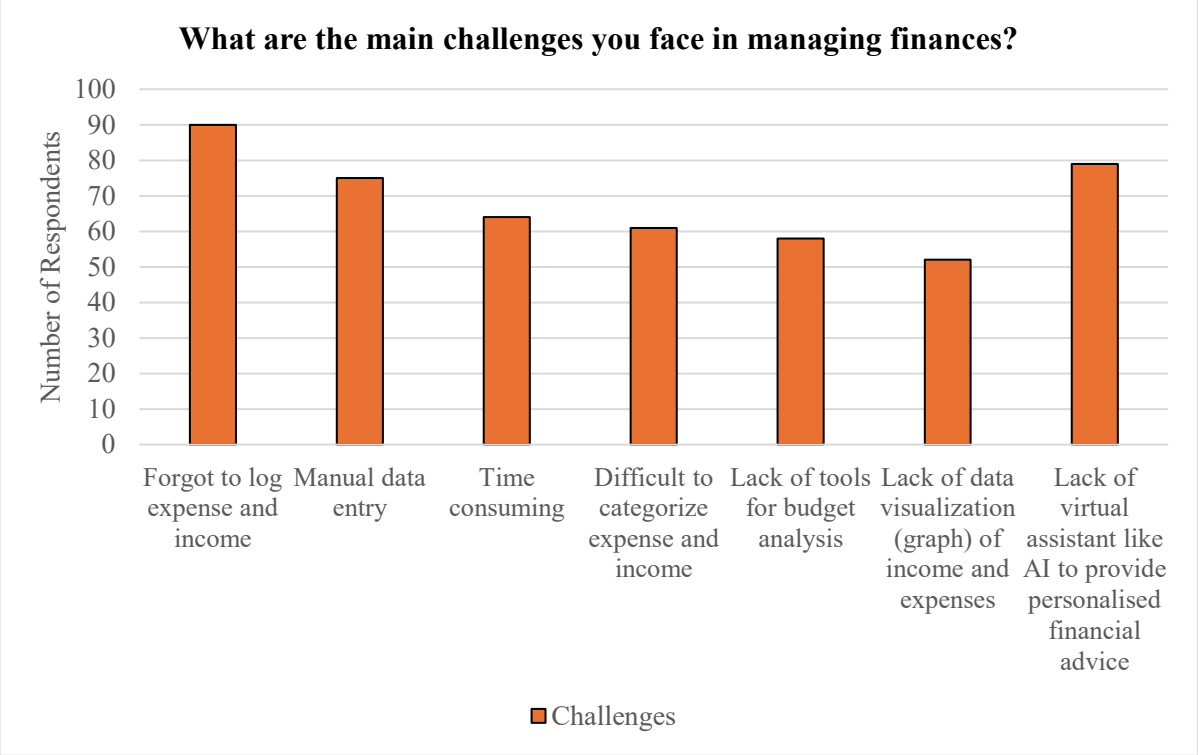


Figure 3.6: The Questionnaires - What are the main challenges you face in managing finances?

The bar charts of the figure above show the main challenges faced by the respondents in managing their finances. The most common challenges are forgetting to log the expense and income which was selected by 90 respondents. This is followed by the lack of virtual assistant like AI for personalised financial advice which has 79 respondents. Manual data entry is another significant challenge which has 75 respondents. Other challenges include time-consuming processes (64 respondents) and difficulty categorizing expenses and income (61 respondents). Additionally, 58 respondents reported a lack of tools for budget analysis while 52 respondents identified lack of data visualization, such as graphs of income and expense. This data highlights the need for automated and AI-driven financial tools.

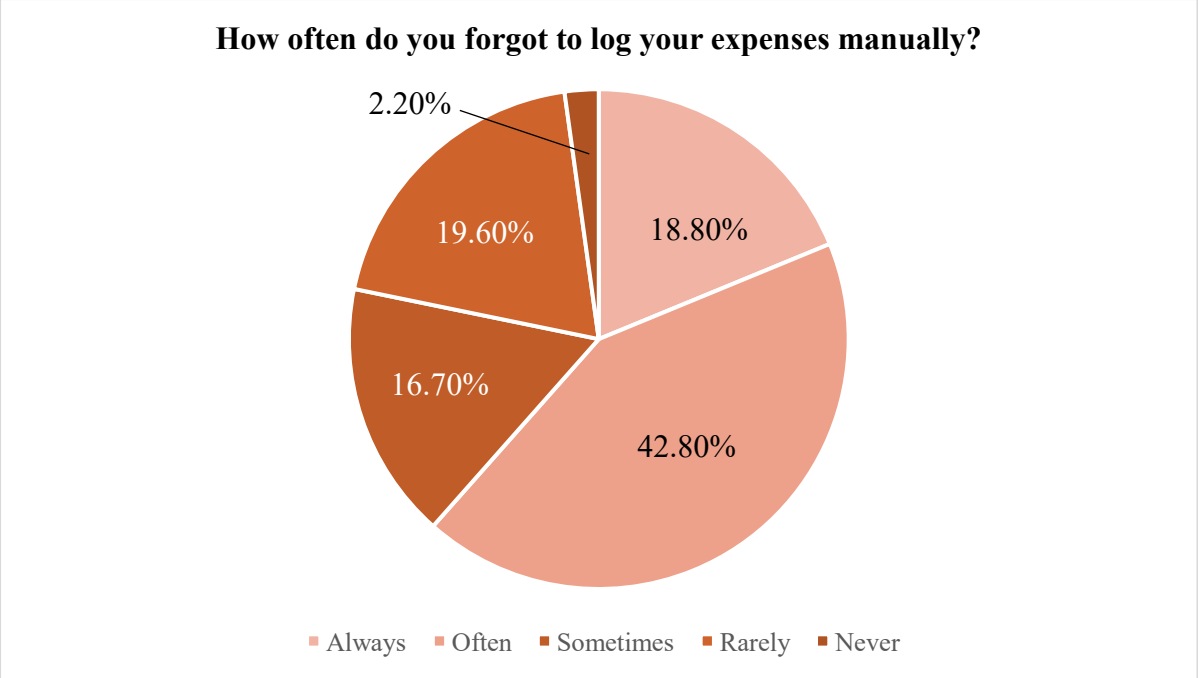


Figure 3.7: The Questionnaires: How often do you forgot to log your expenses manually?

The figure above shows how often the respondents forgot to log their expenses manually. The largest group, 42.80% (59 respondents) often forgets to log their expenses, followed by 19.60% (27 respondents) who rarely forget. 18.80% (26 respondents) who always forget and 16.7% (23 respondents) sometimes forget to log the expenses. A small percentage, 2.20% (3 respondents) never forgets to log expenses. This data highlights that manual logging is a common challenge for many respondents as they often forget to log their expense.

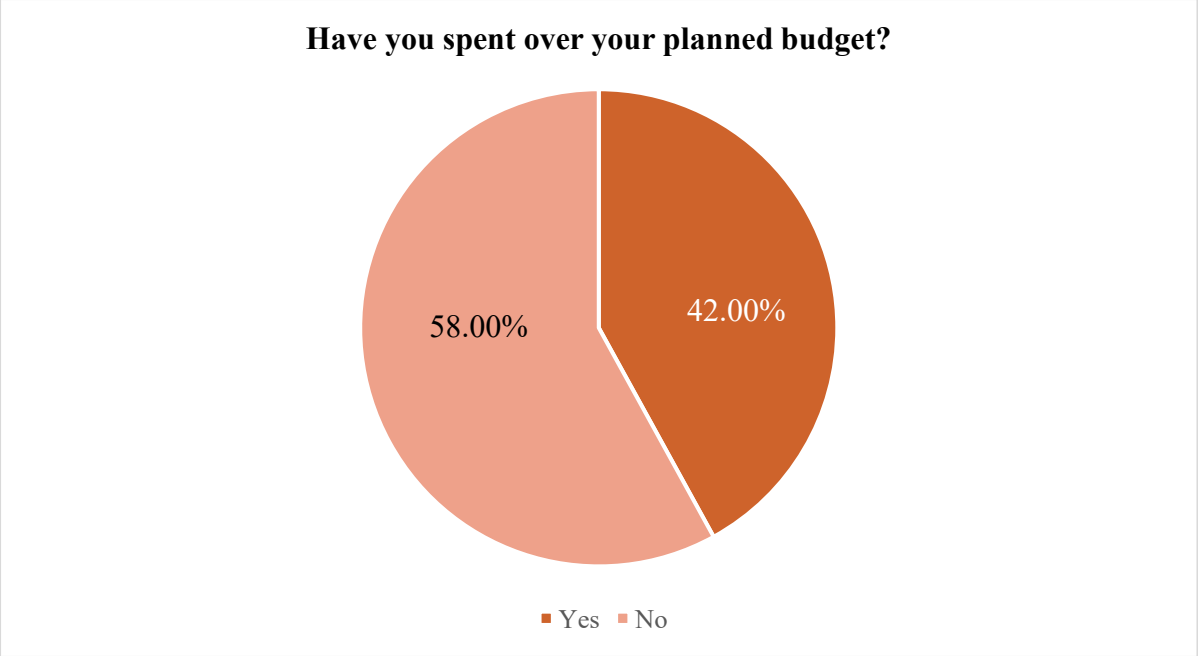


Figure 3.8: The Questionnaires – Have you spent over your planned budget?

The pie chart shows whether respondents have spent over their planned budget. A majority of 42.00% (58 respondents) indicated that they have spent over their budget, while 58.00% (80 respondents) stated that they have not. This data suggests that creating a detailed budget can help to prevent overspending.

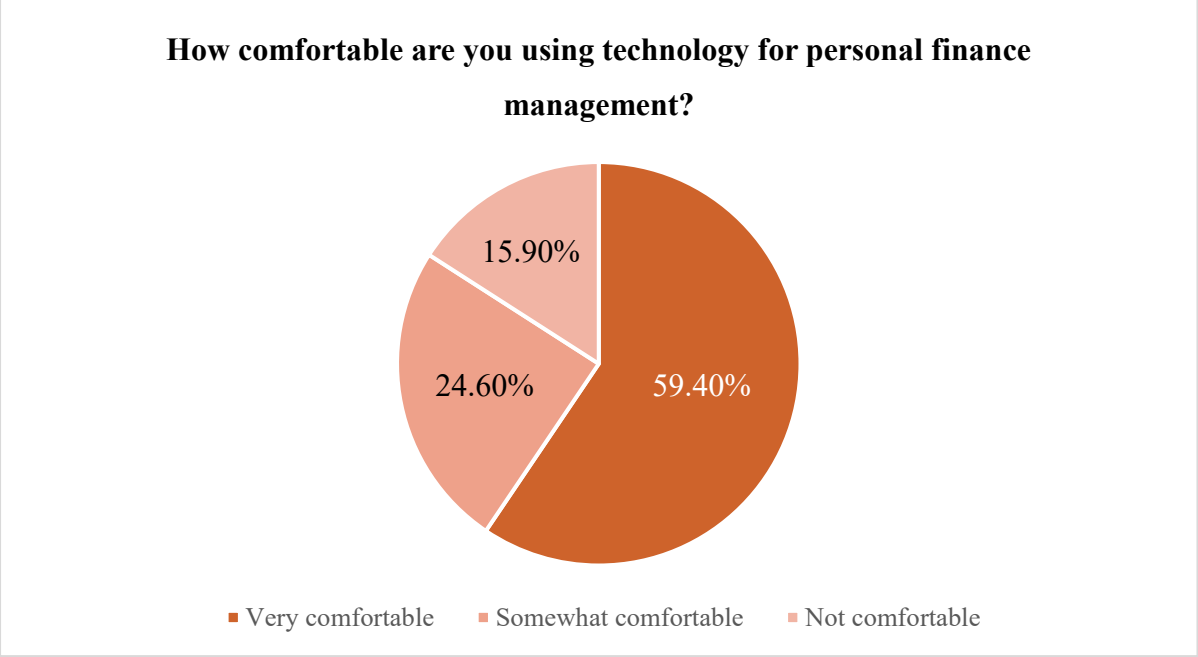


Figure 3.9: The Questionnaires - How Comfortable are you using technology for personal finance management?

The pie chart shows respondents' comfort levels in using technology for personal finance management. The majority, 59.40% (82 respondents) feel very comfortable, while 24.60% (34 respondents) are somewhat comfortable. A smaller group, 15.90% (22 respondents) indicated that they are not comfortable. This data suggests that most respondents are comfortable in using technology for managing their finances.

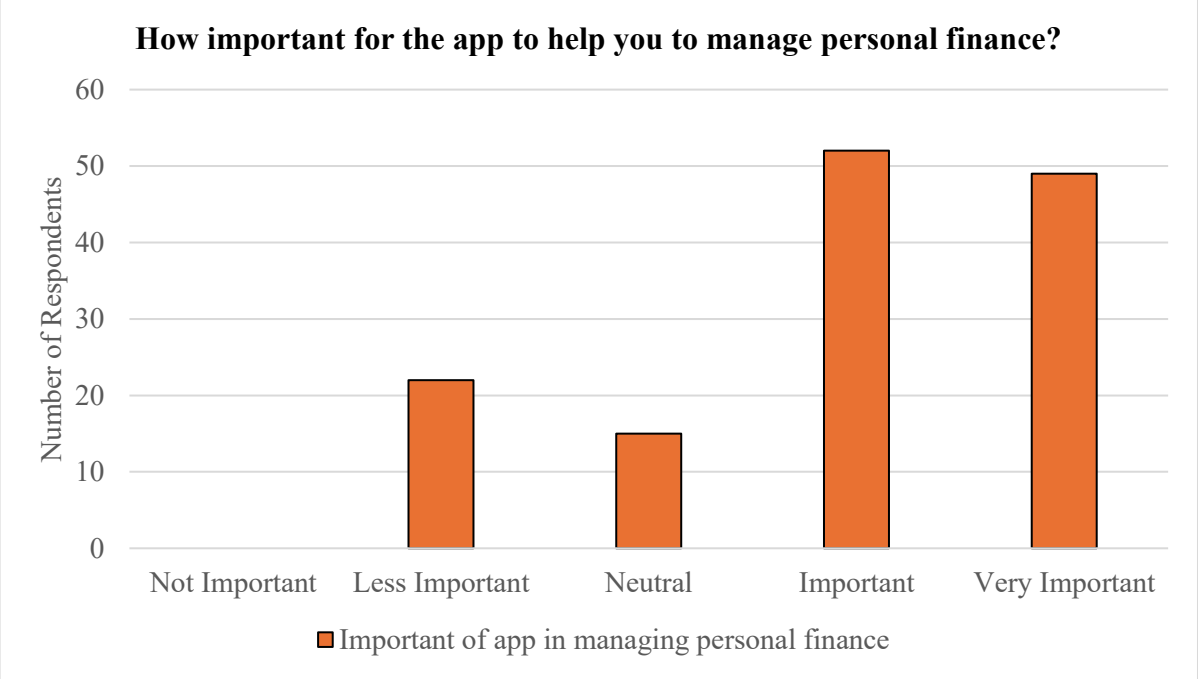


Figure 3.10: The Questionnaires - How important for the app to help you to manage personal finance?

The bar chart shows how respondents rate the importance of an app in managing personal finances. Most respondents consider it either “Important" (52 respondents) or "Very Important" (49 respondents). Around 22 respondents view it as "Less Important," while about 15 respondents remain neutral. None of the respondents found it is "Not Important" of using the mobile app in managing their finances. This data highlights the significant of the finance management apps in helping users to manage their finances.

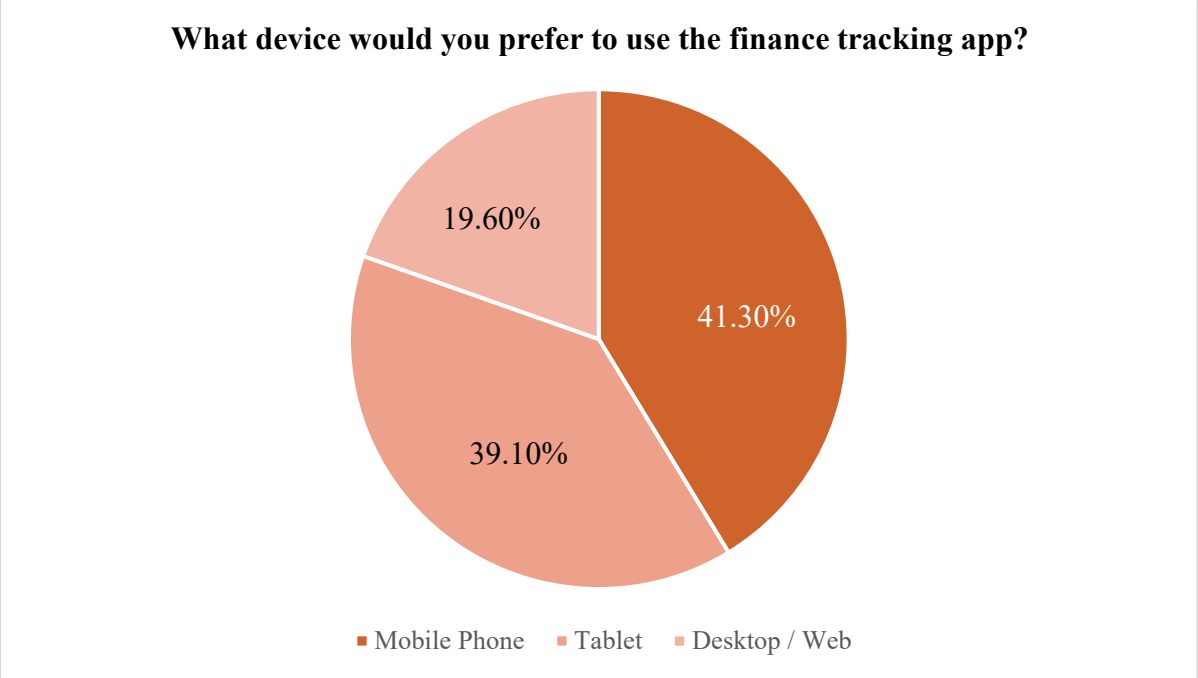


Figure 3.11: The Questionnaires - What device would you prefer to use the finance tracking app?

The pie chart shows which devices is preferred by respondents in using the finance tracking app. Most respondents, 41.30% (57 respondents) prefer using a mobile phone, followed by 39.10% (54 respondents) who prefer to use tablet to track their finances. A smaller group, 19.60% (27 respondents) prefer using desktop or web in finance tracking. This data indicates that mobile phones are the most popular choice for people to manage their finances through an app.

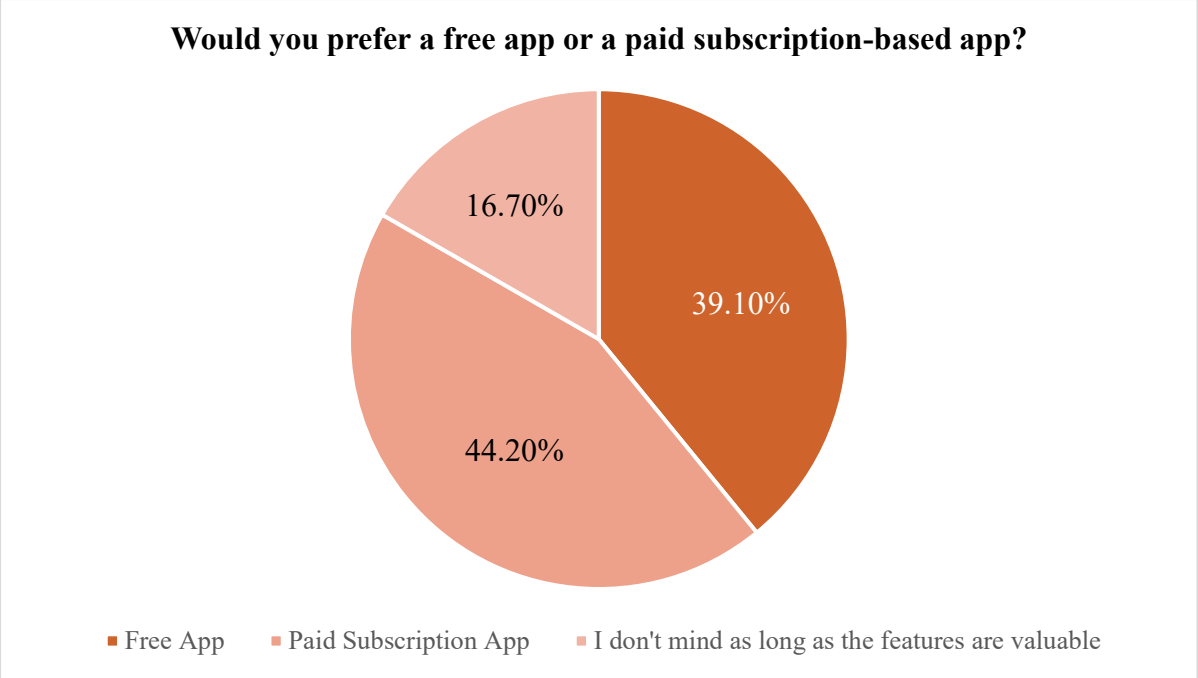


Figure 3.12: The Questionnaires - Would you prefer a free app or a paid subscription-based app?

The pie chart shows the preferences of the respondents for choosing a free or paid subscription-based app. Most respondents, 44.20% (61 respondents) prefer a paid subscription-based app, while 39.10% (54 respondents) prefer a free app. A smaller group of respondents, 16.70% (23 respondents) don't mind whether the app is free or paid as long as it includes the valuable features. This data shows that most of the respondents prefer a paid subscription-based app than a free one as it provide more functional features for users.

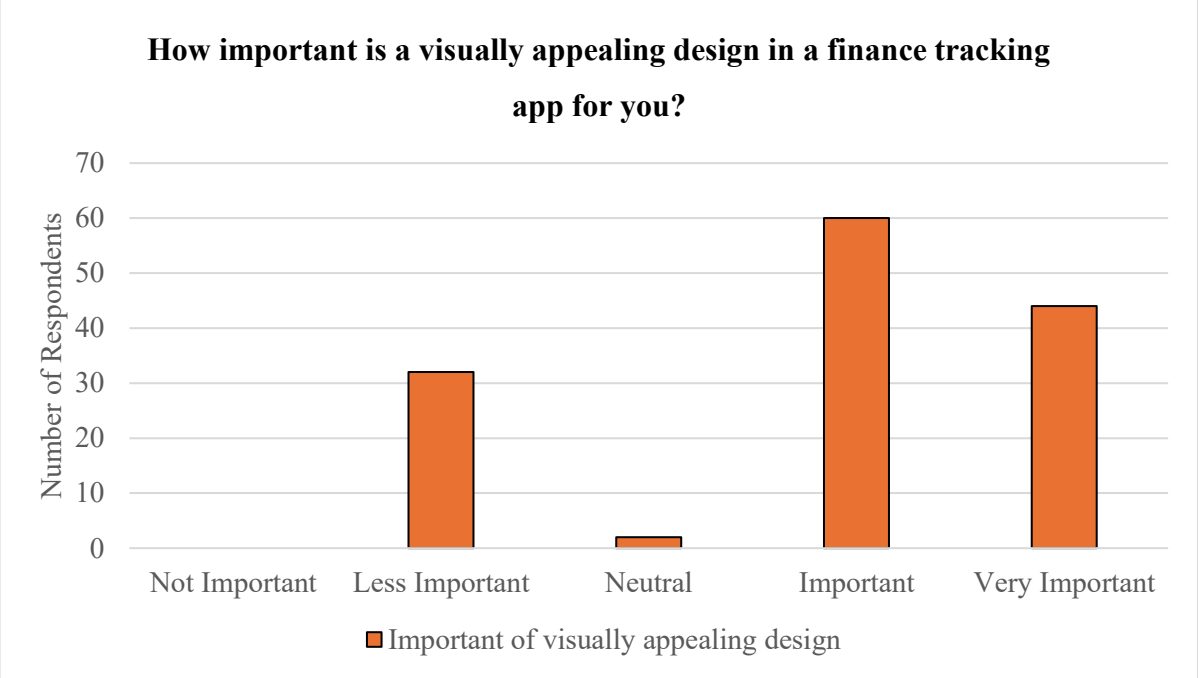


Figure 3.13: The Questionnaires - How important is a visually appealing design in a finance tracking app for you?

The bar chart shows how respondents rate the importance of a visually appealing design in a finance tracking app. The majority, 60 respondents consider it as “Important”, followed by 44 respondents who view it as “Very Important”. Around 32 respondents view is as “Less Important”, while only a small number of respondents (2 respondents) are neutral. None of the respondents consider it as “Not Important”. This data highlights that most of the respondents need a visually appealing design in a finance tracking app to enhance navigation and usability.

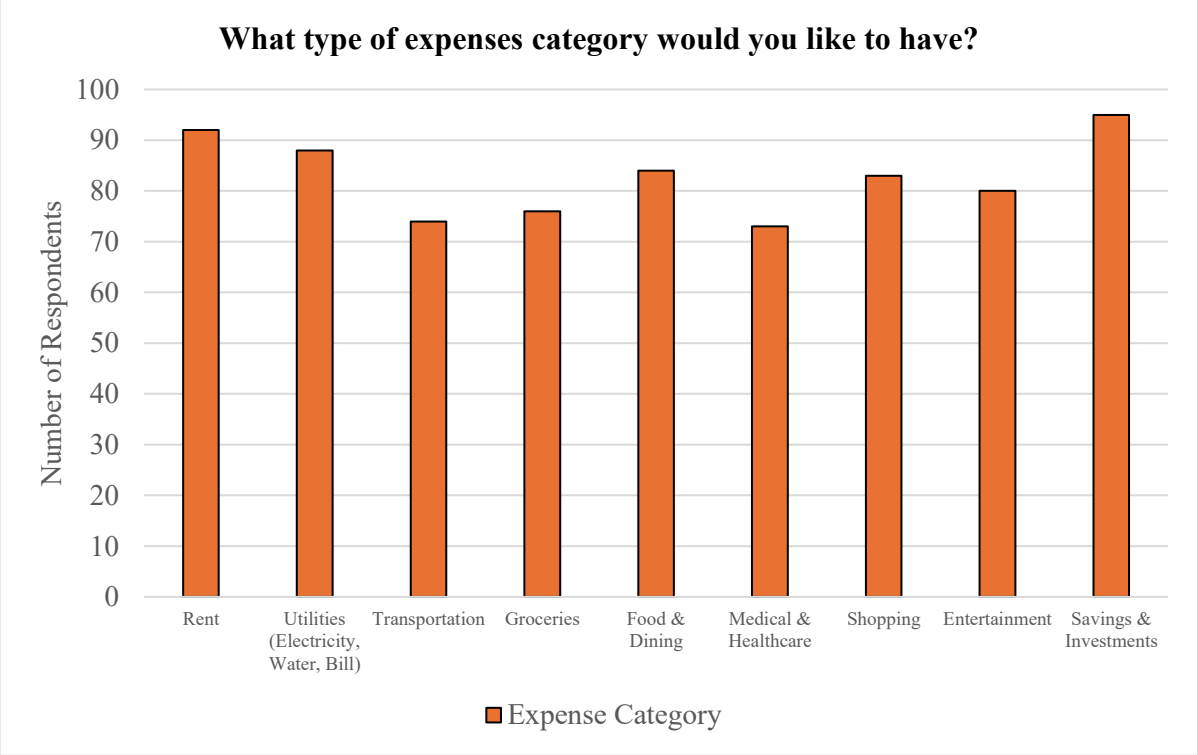


Figure 3.14: The Questionnaires - What type of expenses category would you like to have?

The bar chart shows the type of expense categories that respondents would like to have in a personal finance tracking app. “Savings & Investments” is the most preferred category, selected by 95 respondents. This is followed by the “Rent” and “Utilities” category, each has 92 and 88 respondents respectively. Other popular categories include “Food & Dining” (84 respondents), “Shopping” (83 respondents), “Entertainment” (80 respondents) and “Groceries” (76 respondents). "Transportation" and "Medical & Healthcare" are also preferred, with 74 and 73 respondents selecting these categories. This data indicates that most of the respondents prefer having a variety of expense categories to choose from in their finance tracking app.

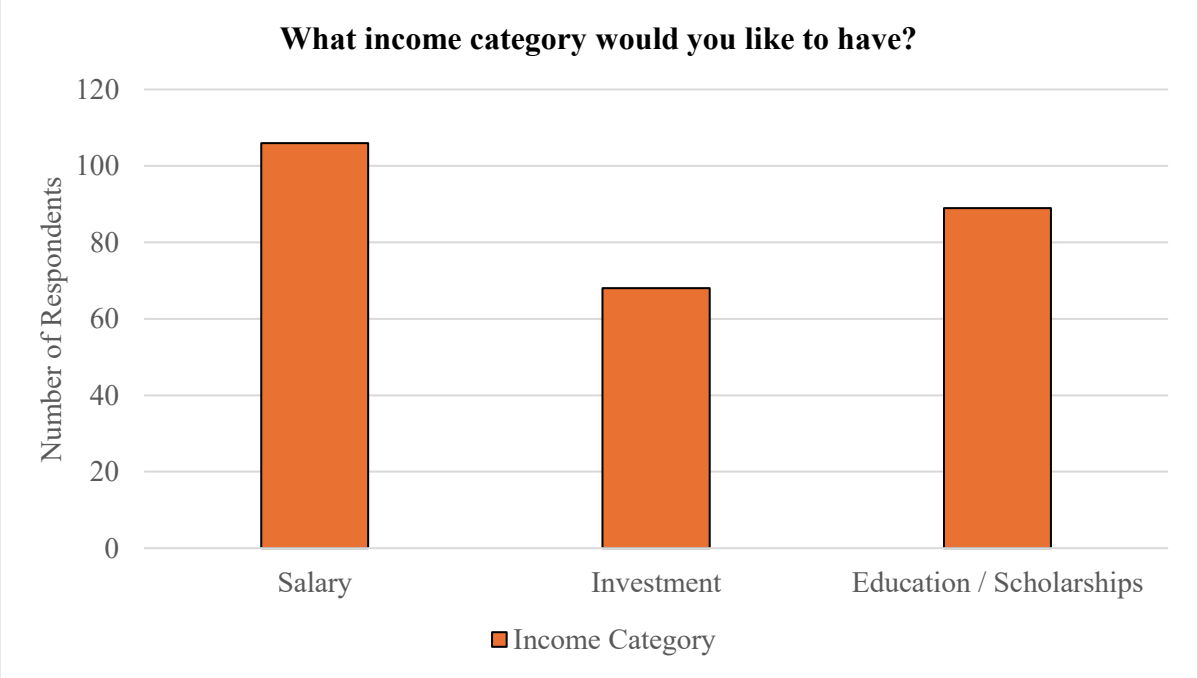


Figure 3.15: The Questionnaires - What income category would you like to have?

The bar chart shows the income categories that the respondents would like to have in a personal finance tracking app. “Salary” is the most preferred category which was chosen by 106 respondents. This is followed by “Education/Scholarships” category which was selected by 89 respondents as many of the respondents are student. The category of “Investment” was chosen by 68 respondents. This data indicates that most respondents prefer having a variety of income categories to choose from.

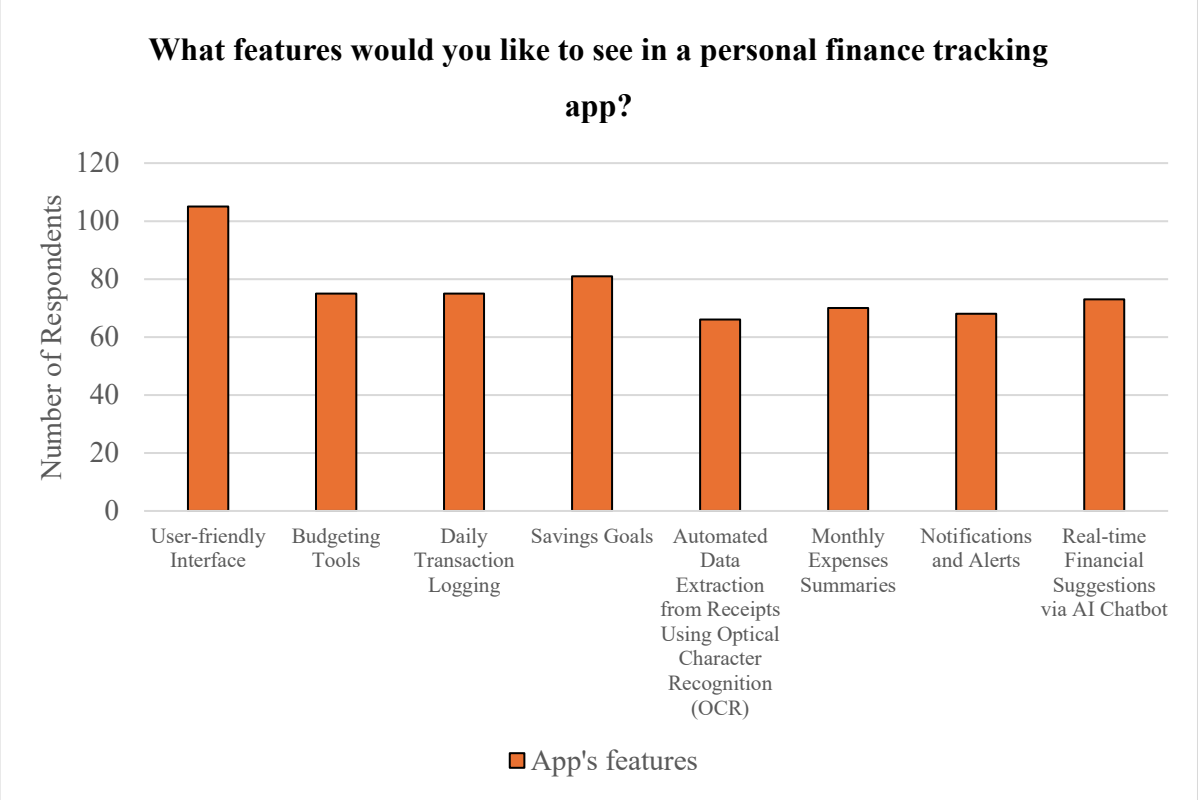


Figure 3.16: The Questionnaires - What features would you like to see in a personal finance tracking app?

The bar chart shows the features that respondents would like to see in a personal finance tracking app. The most requested feature is a “User-friendly Interface” which has 105 respondents. “Savings Goals” follows with 81 respondents. Both “Budgeting Tools” and “Daily Transaction Logging” has 75 respondents. “Real-time Financial Suggestions via AI Chatbot” and “Monthly Expense Summaries” has 73 and 70 respondents respectively. Additionally, 68 respondents want to have notifications and alerts feature, while 66 respondents selected “Automated Data Extraction from Receipts Using Azure Form Recognizer". This data highlights that respondents prioritize the ease of use and advanced features for better financial management.

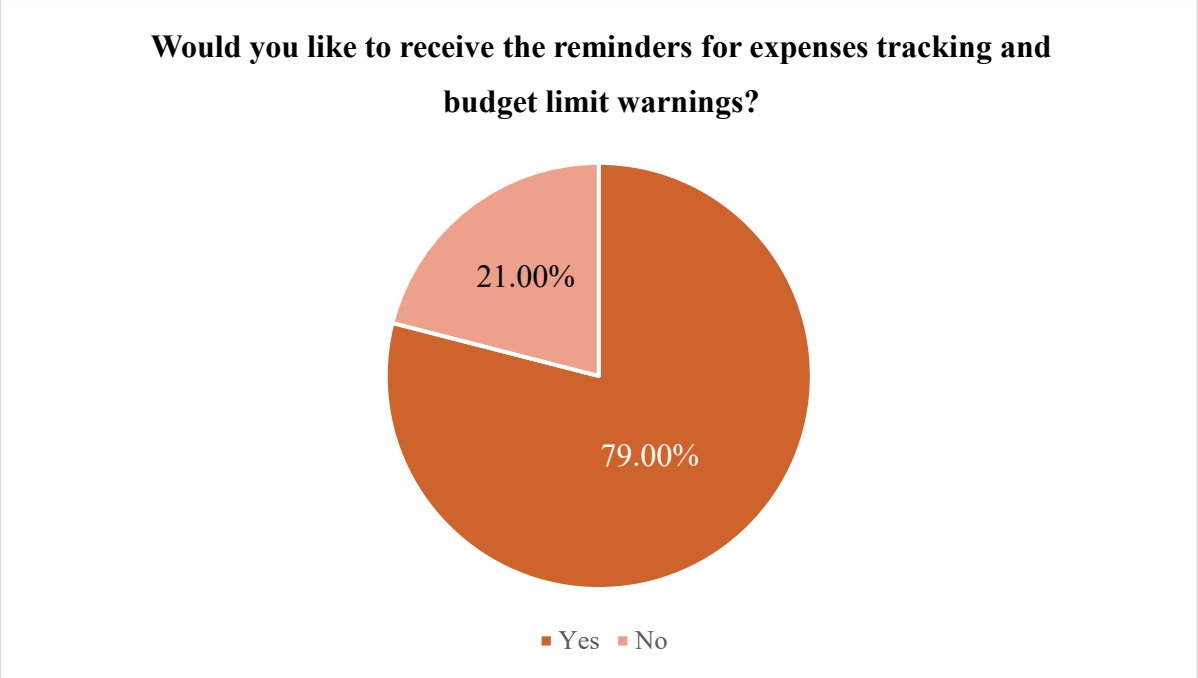


Figure 3.17: The Questionnaires - Would you like to receive the reminders for expenses tracking and budget limit warnings?

The pie chart shows the preference of the respondents in receiving the reminders for expense tracking and budget limit warnings. The majority, 79.00% (109 respondents) would like to receive a reminder, while 21.00% (29 respondents) does not like to receive the reminders. This indicates that most users view reminders as a helpful feature that supports better financial habits by prompting timely actions and helping them stay within their budget. The data highlights the importance of incorporating reminder functionalities in finance tracking apps to enhance user engagement and promote consistent expense management.

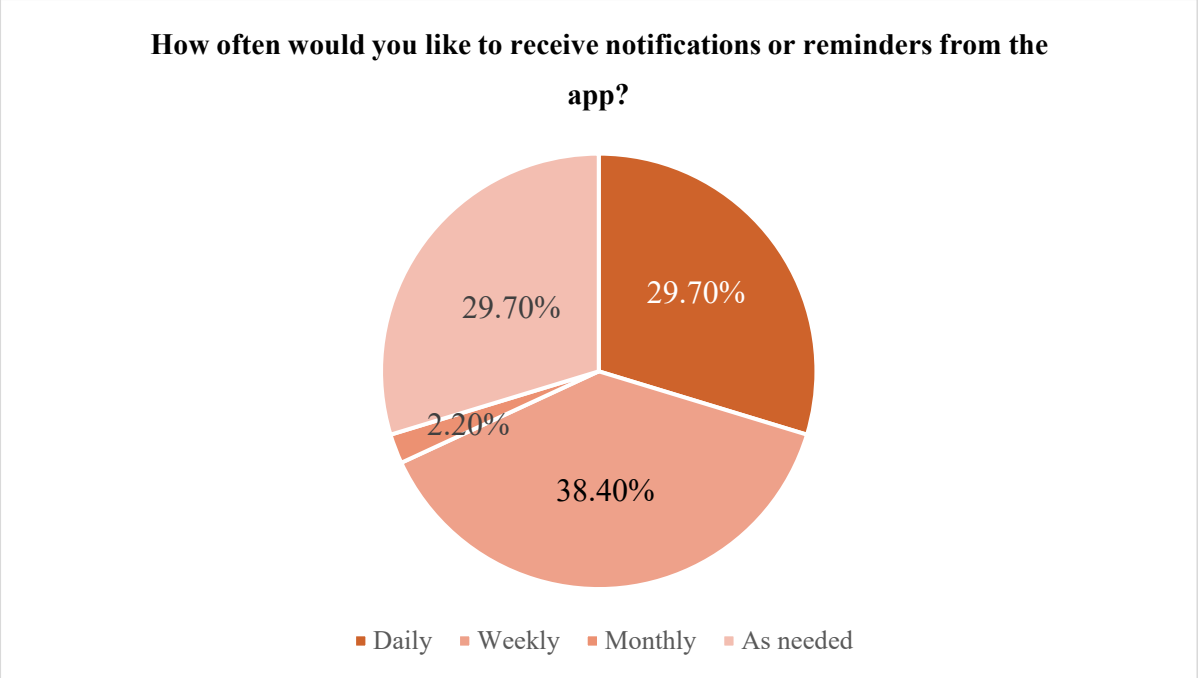


Figure 3.18: The Questionnaires - How often would you like to receive notifications or reminders from the app?

The pie chart shows how often the respondents would like to receive the notifications or reminders from the app. The largest group, 38.40% (53 respondents) prefers to receive reminders weekly, while 29.70% (41 respondents) prefer to receive it daily or only as needed. Only 2.20% (3 respondents) would like to receive monthly reminder. This data indicates that most respondents prefer receiving reminder either daily or weekly in order to help them stay consistent with expense tracking and budget management.

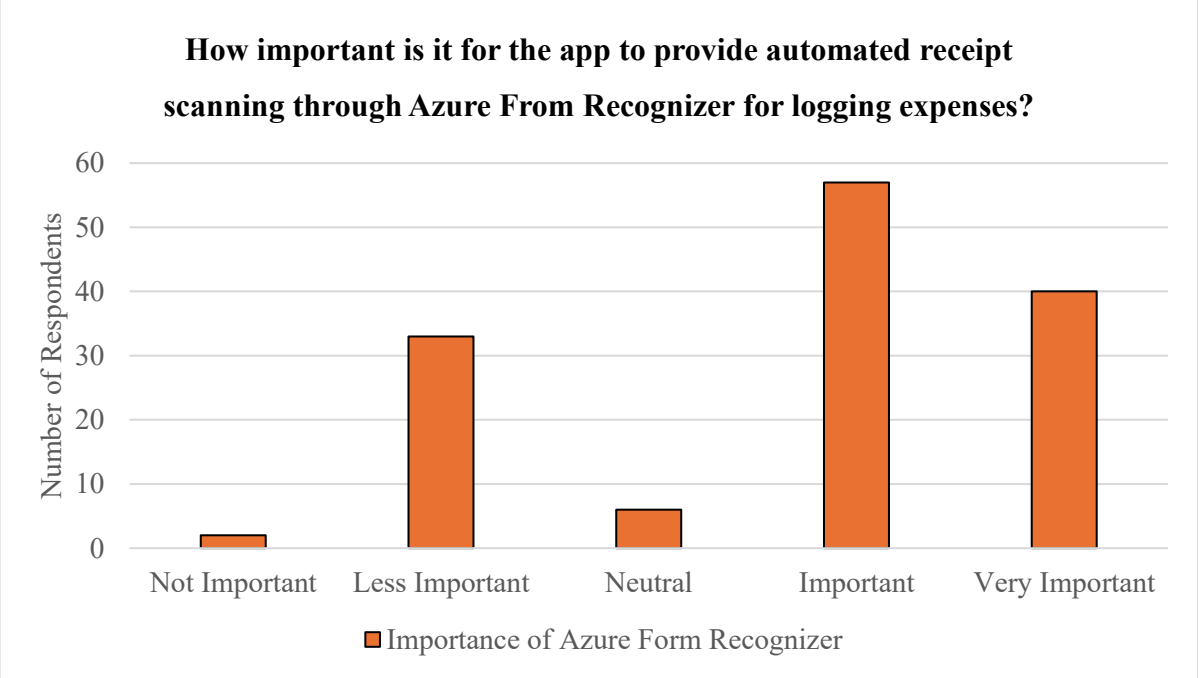


Figure 3.19: The Questionnaires - How important is it for the app to provide automated receipt scanning through Azure Form Recognizer API for logging expenses?

The bar chart shows how respondents rate the importance of having automated receipt scanning using Azure Form Recognizer for logging expenses in an app. Most respondents (57 respondents) consider it is important to have this feature to include in the app, while 40 respondents rate it as "Very Important". Meanwhile, 33 respondents view it is "Less Important", while 6 respondents view it is "Neutral". Only 2 respondents consider it is "Not Important". This data indicates that most of the respondents need the Azure Form Recognizer functionality for easier expense logging.

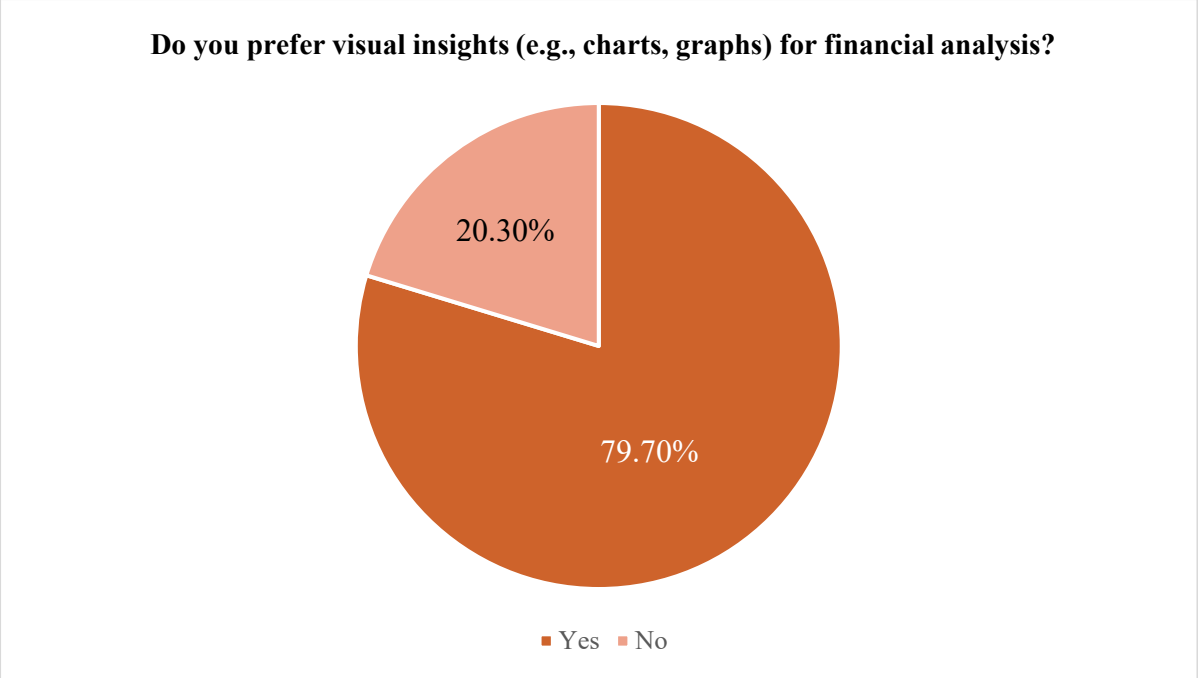


Figure 3.20: The Questionnaires - Do you prefer visual insights (e.g., charts, graphs) for financial analysis?

The pie chart shows the preference of the respondents to have visual insights, such as charts and graphs for financial analysis within a finance tracking app. A majority of 79.70% (110 respondents) prefer visual insights, while 20.30% (28 respondents) do not. This suggests that most users find visual representations helpful in understanding and analysing their financial data more effectively. Visual tools can make complex financial information more accessible and intuitive, supporting better decision-making and financial planning.

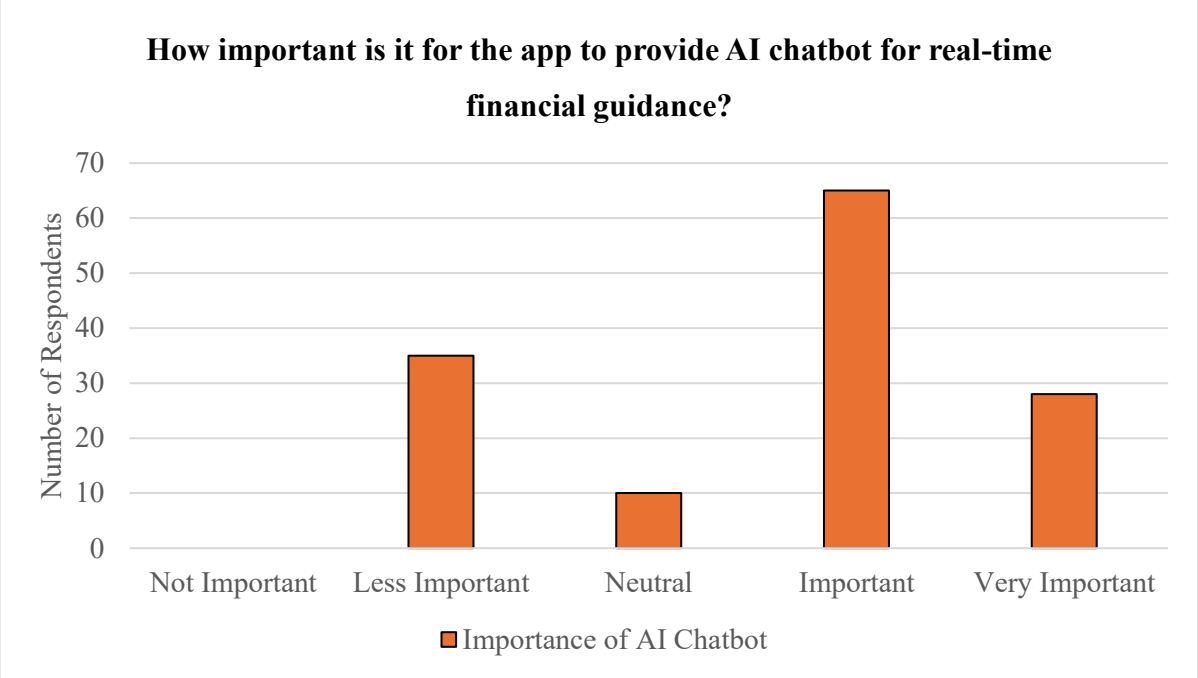


Figure 3.21: The Questionnaires - How important is it for the app to provide AI chatbot for real-time financial guidance?

The bar chart shows how respondents rate the importance of having an AI chatbot for real-time financial guidance within a finance tracking app. The majority of 65 respondents consider it as “Important”. 35 respondents view it as “Less Important”, while 28 responders view it as “Very Important”. Only 10 respondents rate it as "Neutral” and none of the respondents consider it as “Not Important”. This data suggests that most users value the inclusion of an AI chatbot, recognizing its potential to provide timely financial advice and enhance their overall financial management experience.

3.4.3 Non-Functional Requirements

To ensure the performance and efficiency of Future Saver, the following non-functional requirements are included:

Operational Requirements

- The system shall be able to operate on Android devices.
- The system shall require an active internet connection to operate.

Usability Requirements

- The system shall provide an intuitive and user-friendly interface that is easy to use.
- The system shall use clear and consistent icons, labels, and colour schemes to provide a visually appealing interface.
- The system shall ensure that all text and elements are accessible on various screen sizes and resolutions in the mobile app.

Performance Requirements

- The system shall response to user inputs such as tapping buttons or navigation menus within 2 seconds under normal operating conditions.
- The system shall complete the receipt scanning and data extraction within 5 seconds.
- The AI chatbot of the system shall provide responses to user within 5 seconds when connected to the internet.

3.5 System Design

The system design of the Future Saver involves creating a detailed plan for the app's functionality and features. It includes use case diagrams, sequence diagrams, and activity diagrams to provide a clear visualization of the system's workflow and user interactions.

3.5.1 Use Case Diagram

A Use Case Diagram visually represents the interactions between users (actors) and the system. It provides a high-level overview of the system's functionality by showing the different ways users can interact with it.

Figure 3.22: Use Case Diagram

3.5.1.1 Use Case Specifications

Table 3.1: Use Case Specification for Register New User

Use Case	Register
Description	The new user needs to create an account.
Actor	New User
Pre-condition (s)	The new user wants to register and create a new account.
Post-condition (s)	User's account is created and stored in the system's database.
Main Flow	1. The new user navigates to the Registration page. 2. The new user registers a new account [A1]. 3. The new user fills in his/her personal details including name and email address [A2]. • Email address is used as the username for the new account. 4. The new user sets a password for his/her account and complete the registration process. 5. Upon successful registration, the user can log in using their email and password.
Alternative Flow	[A1] The new user cancels registration. [A2] The new user enters invalid input such as invalid email format.

Table 3.2: Use Case Specification for Login Existing User

Use Case	Login
Description	The user needs to log in his/her account.
Actor	Existing User
Pre-condition (s)	The user needs to register an account.
Post-condition (s)	The user successfully logged in or logged out from the account.
Main Flow	1. The user navigates to the Login page. 2. The user fills in his/her email and password that are set on the registration [A1]. 3. The system verifies the user's email and password [A2]. 4. The system redirects the user to the home page.
Alternative Flow	[A1] The user forgot his/her password. • The user can click the "Forgot Password?" link to reset the password. [A2] The user has not registered an account before logging in.

Table 3.3: Use Case Specification for Adding New Budget

Use Case	Add Budget
Description	The user wants to set budget for the expense category.
Actor	New and Existing User
Pre-condition (s)	The user must log into their account.
Post-condition (s)	The new budget is successfully saved and displayed in the budget list.
Main Flow	1. The user navigates to the Budget page. 2. The user selects the option to add a new budget [A1]. 3. The user enters the details for the budget, including the assigned amount for expense and income categories. 4. The system saves the new budget and updates the budget list. 5. Then, the user can edit or delete any existing budget directly from the budget list [A2].
Alternative Flow	[A1] The user cancels the budget adding process. [A2] The user cancels the process of editing and deleting budget.

Table 3.4: Use Case Specification for Adding New Transaction

Use Case	Add Transaction
Description	The user wants to add the new transaction to the system.
Actor	New and Existing User
Pre-condition (s)	1. The user must log into their account. 2. The user must add the budget.
Post-condition (s)	1. The transaction is successfully saved and reflected in the transaction history. 2. The system updates the available amount in the budget accordingly.
Main Flow	1. The user navigates to the Add Transaction page. 2. The user selects between manually entering transactions or using automated transaction tracking via receipt scanning [A1]. For the manual entry transaction: i. The user enters the transaction details, including date, type of transaction, category, name, amount and remarks. For the automated transaction tracking via receipt scanning i. The user scans a photo of the receipt using the camera. ii. The system extracts the details such as name, amount, and date into the respective fields using Azure Form Recognizer. 3. The user clicks the “Save” button to save the transaction and updates the transaction history. 4. The available amount in the budget is also updated.

Alternative Flow	[A1] The user cancels the transaction adding process.

Table 3.5: Use Case Specification for Viewing Transaction History

Use Case	View Transaction History
Description	The user wants to view the transaction history.
Actor	New and Existing User
Pre-condition (s)	1. The user must log into their account. 2. The user must add the transaction.
Post-condition (s)	The user can view a list of transactions.
Main Flow	1. The user navigates to the Transaction History page by clicking the “See All” button from the Home Screen. 2. The system retrieves all the recorded transactions [A1]. 3. The user can filter the transactions by category using the filter icon. 4. The system updates and displays the transaction list based on the selected filter. 5. The user can select a specific transaction to view its detailed information. 6. The user can choose to edit or delete a transaction.
Alternative Flow	[A1] The user has no recorded transactions in the system.

Table 3.6: Use Case Specification for Generating Report

Use Case	Generate Report
Description	The user wants to view a report that has been generated.
Actor	New and Existing User
Pre-condition (s)	1. The user must log into their account. 2. The user must add the transaction.
Post-condition (s)	The user can view a summary of the report, such as Spending by Category or Income vs Spending.
Main Flow	1. The user navigates to the Report page. 2. The user selects the type of report to be generated such as Spending by Category or Income vs Spending. 3. The user specifies the date range of the report [A1]. 4. The system generates a pie chart showing total spending by expense category, along with the percentage and amount for each category in Spending by Category report. 5. The system generates a bar chart that compare the total income and spending and calculates the net balance in Income vs Spending report.
Alternative Flow	[A1] The user has not recorded transactions within the specified date range.

Table 3.7: Use Case Specification for Interacting with AI Chatbot

Use Case	Interact with AI chatbot
Description	The user wants to interact with the AI assistant to receive financial guidance and suggestions.
Actor	New and Existing User
Pre-condition (s)	The user must log into their account.
Post-condition (s)	The chatbot provides an appropriate response or suggestions based on the user's input.
Main Flow	1. The user navigates to the Chatbot page. 2. The user types a message in the text input field [A1]. 3. The user sends the message by clicking the "Send" button [A2]. 4. The system processes the user's input and generates an AI-driven response.
Alternative Flow	[A1] The user provides an unclear message that cannot understand by the AI chatbot. [A2] The user exits or close the chatbot.

Table 3.8: Use Case Specification for Viewing Profile

Use Case	View Profile
Description	The user wants to view their profile information.
Actor	New and Existing User
Pre-condition (s)	The user must log into their account.
Post-condition (s)	The user's profile information is displayed correctly.
Main Flow	1. The user navigates to the Profile page by clicking the edit profile icon on the Home Screen. 2. The system retrieves the user's profile information. 3. The system displays the user's profile details, including name, email, phone number, and profile picture. 4. The user can edit their profile information [A1]. 5. The system allow user to enable and disable the push notifications.
Alternative Flow	[A1] The use can cancel the profile information editing process.

Table 3.9: Use Case Specification for Viewing Notification

Use Case	View Notification
Description	The user wants to view the app's notification.
Actor	New and Existing User
Pre-condition (s)	The user must log into their account.
Post-condition (s)	The user views a list of notifications, such as reminders or alerts.
Main Flow	1. The user navigates to the Notifications page by clicking the notification icon from the Home Screen. 2. The system retrieves all the notifications. 3. The system displays all the notifications to the user.
Alternative Flow	None

3.5.2 Activity Diagram

The activity diagram shows how the system works by outlining the steps involved in completing the tasks. It illustrates the flow of actions, decisions, and interaction between users and the system. The diagram starts at the beginning and ends at the final step to show different paths based on the user choices. This diagram helps to understand how the system works, what triggers certain actions, and how the process can be improved for better user experience.

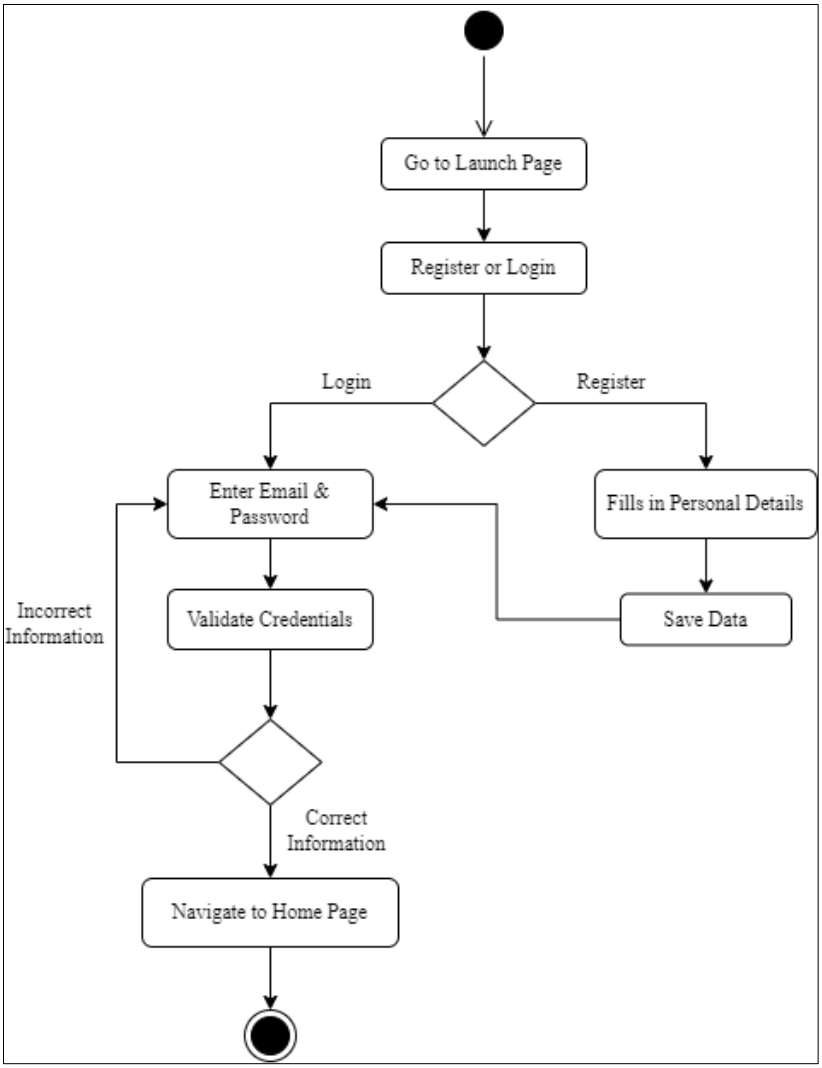


Figure 3.23: Activity Diagram of Register and Login Page

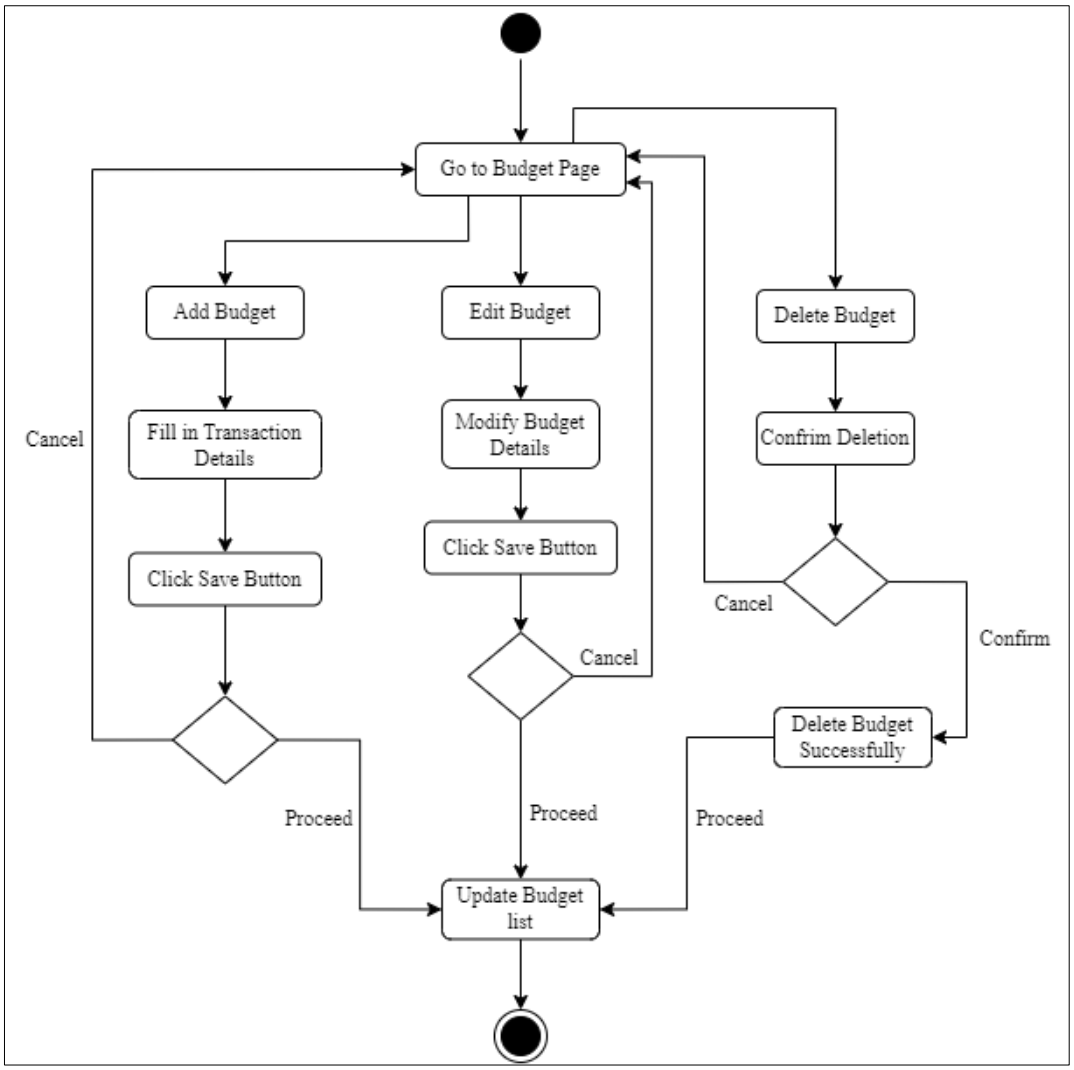


Figure 3.24: Activity Diagram of Budget Page

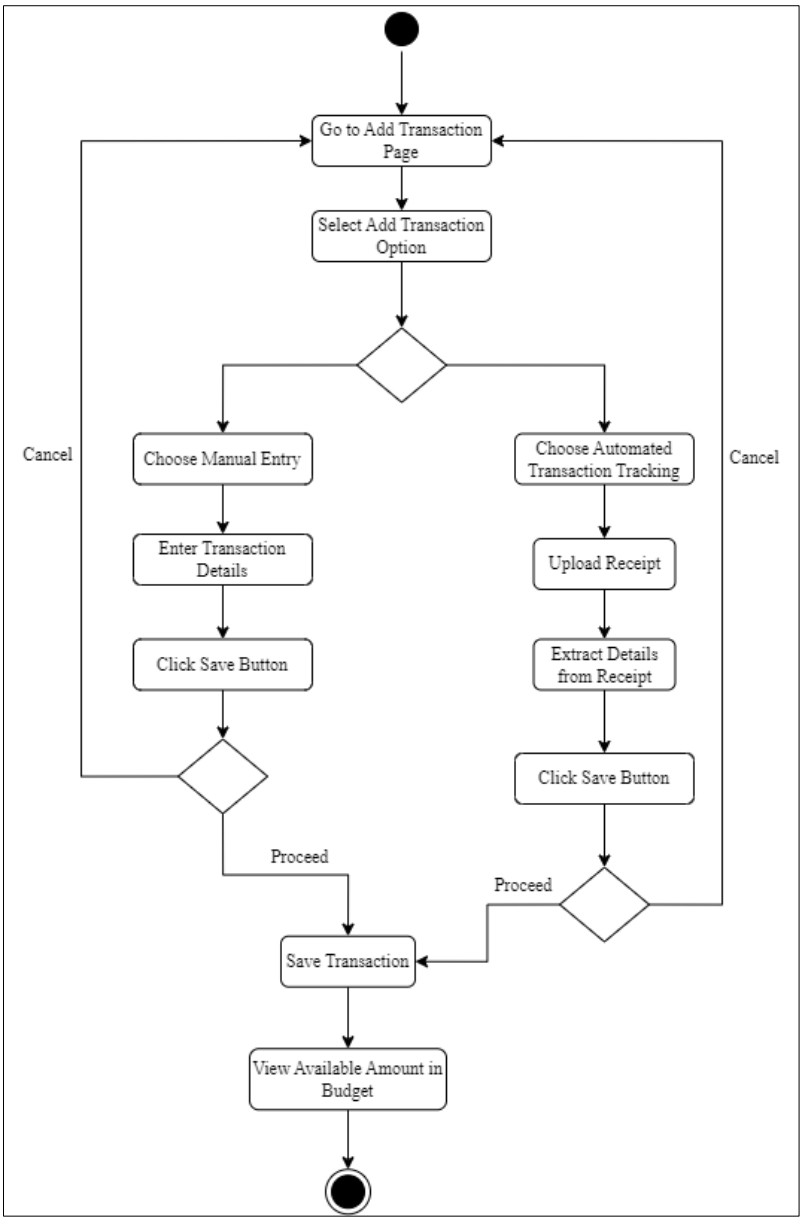


Figure 3.25: Activity Diagram of Add Transaction Page

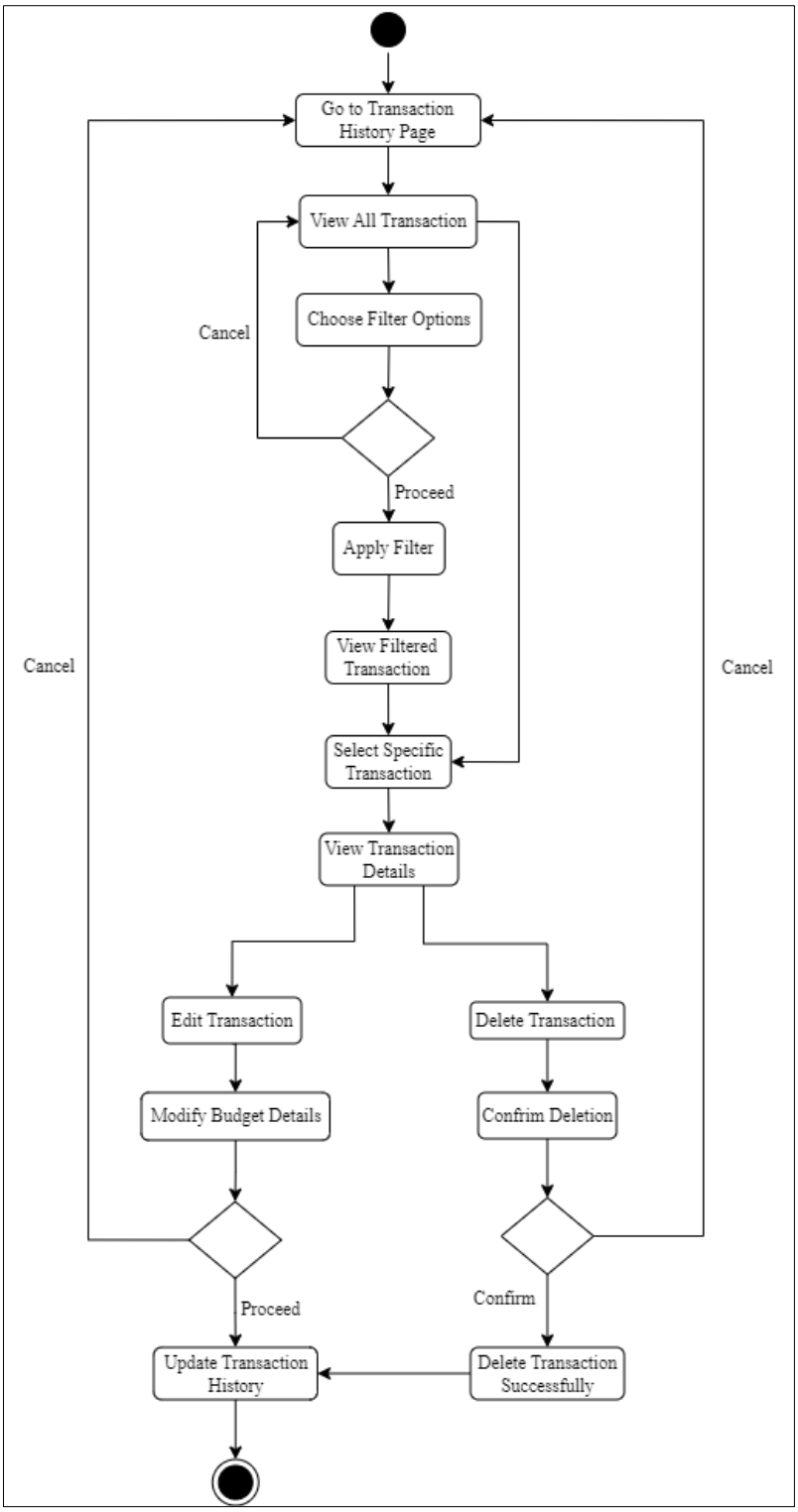


Figure 3.26: Activity Diagram of View Transaction History Page

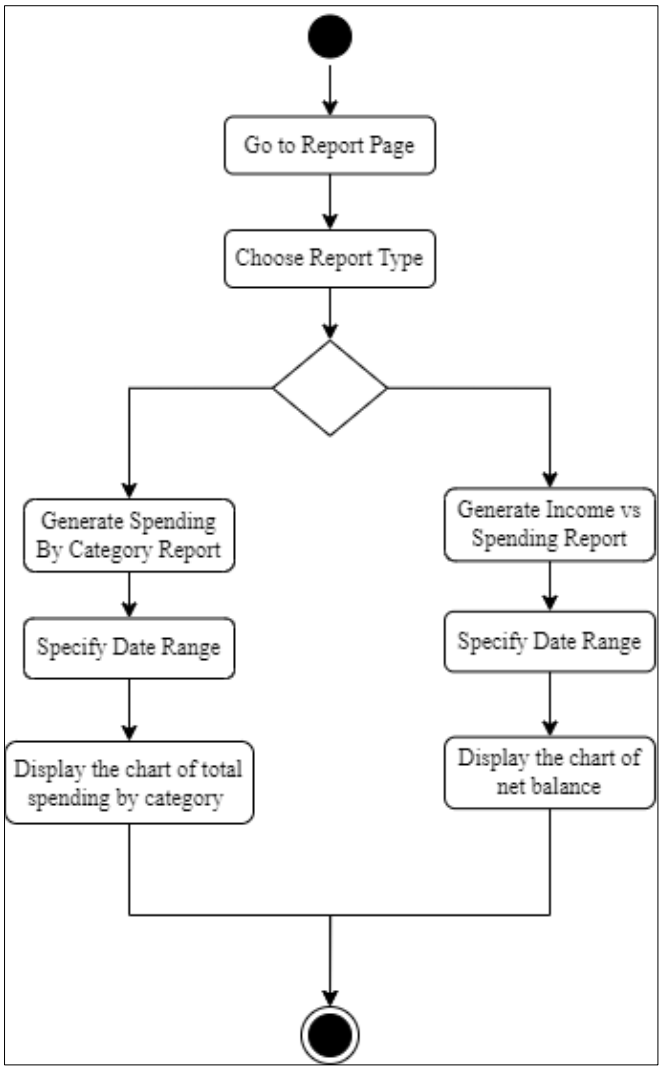


Figure 3.27: Activity Diagram of Report Page

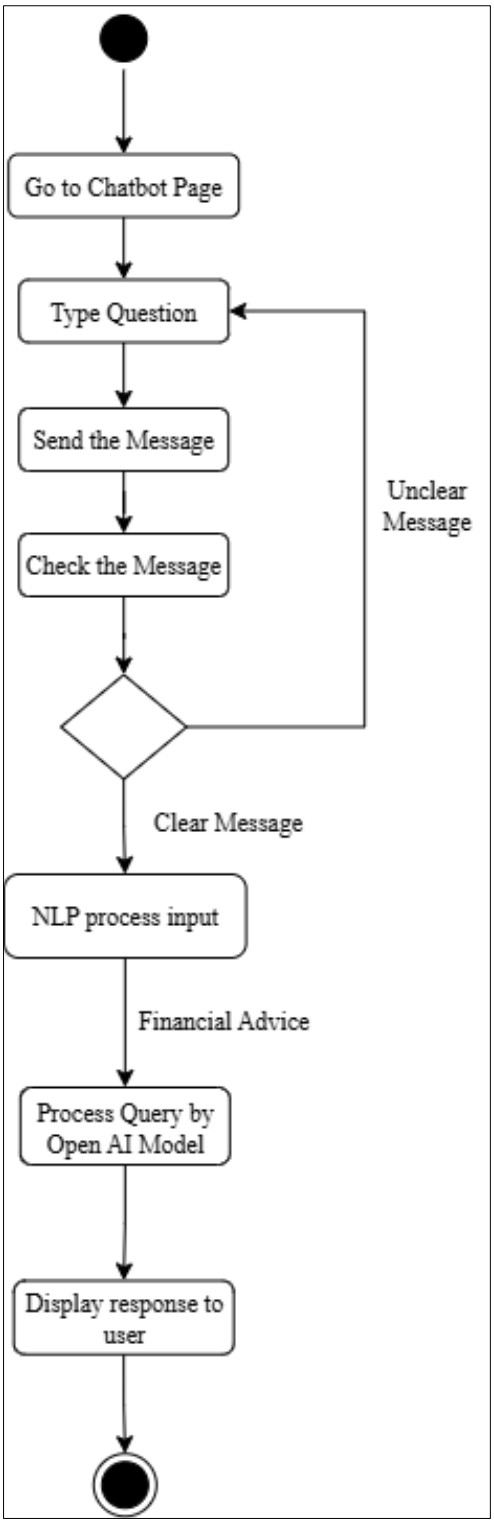


Figure 3.28: Activity Diagram of Chatbot Page

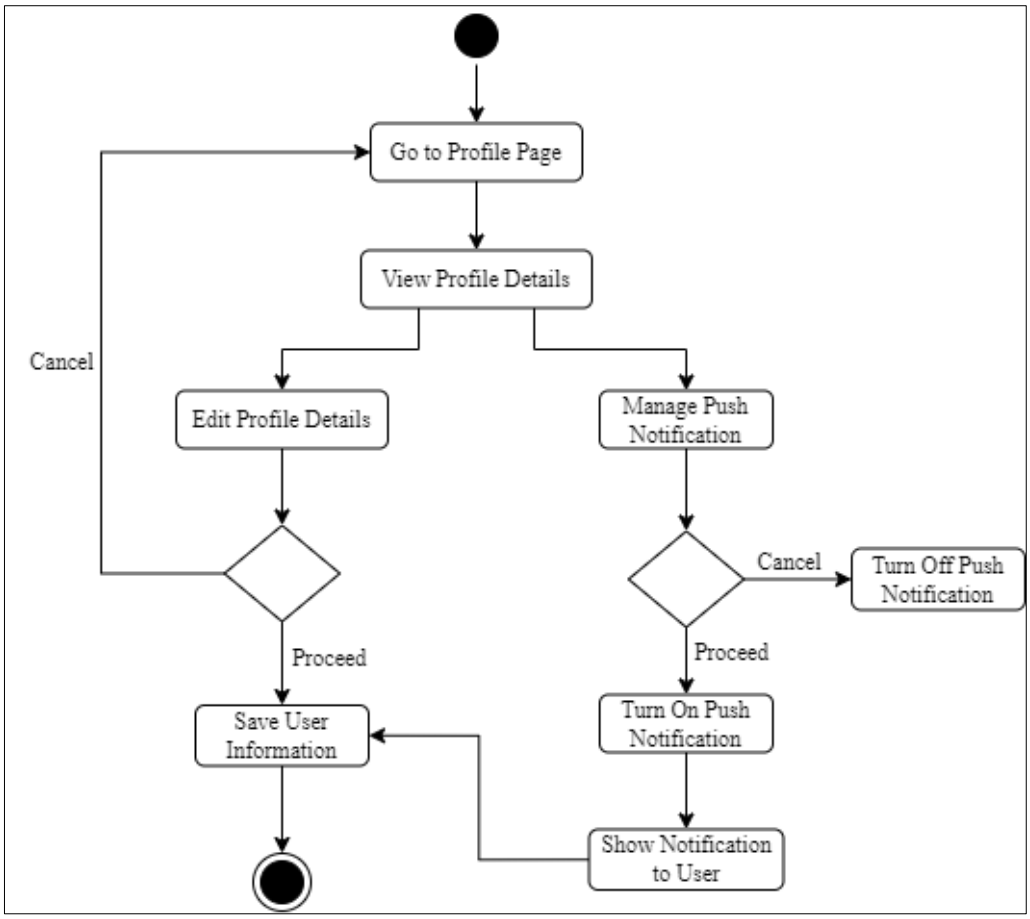


Figure 3.29: Activity Diagram of Profile Page

3.5.3 Sequence Diagram

The sequence diagram visualizes the interaction between users and the system components in a sequential order. It focusses on the timing and order of messages exchanged between different parts of the system elements. The diagram shows how different objects communicate with each other through messages, providing a clear view of the sequence of actions or processes.

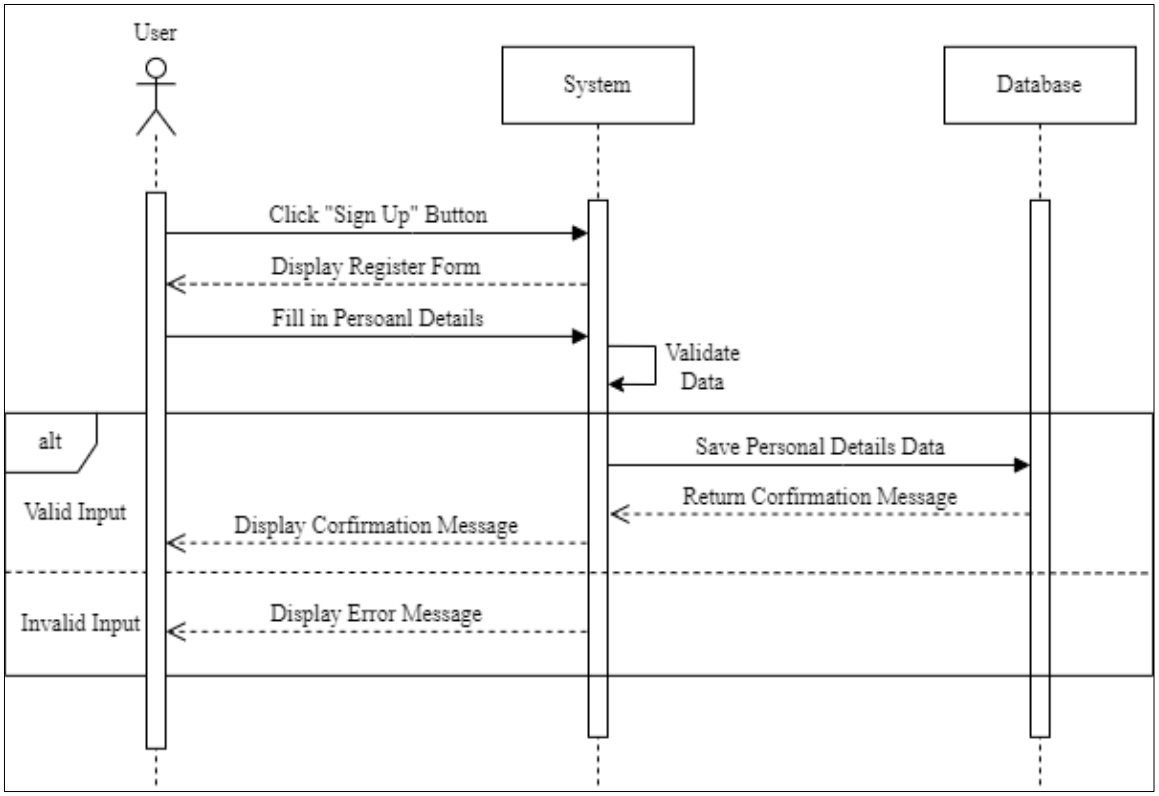


Figure 3.30: Sequence Diagram of Register Page

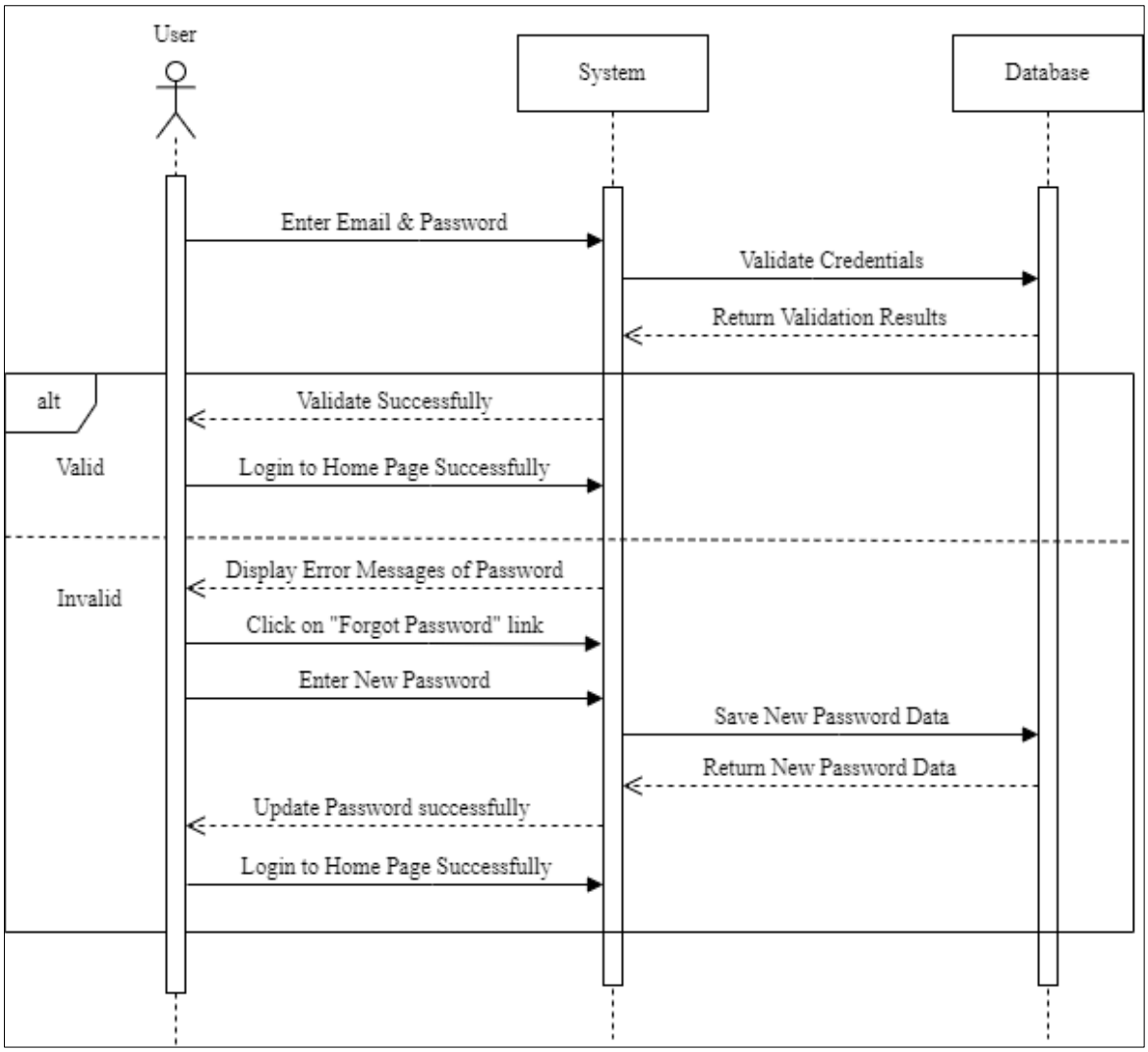


Figure 3.31: Sequence Diagram of Login Page

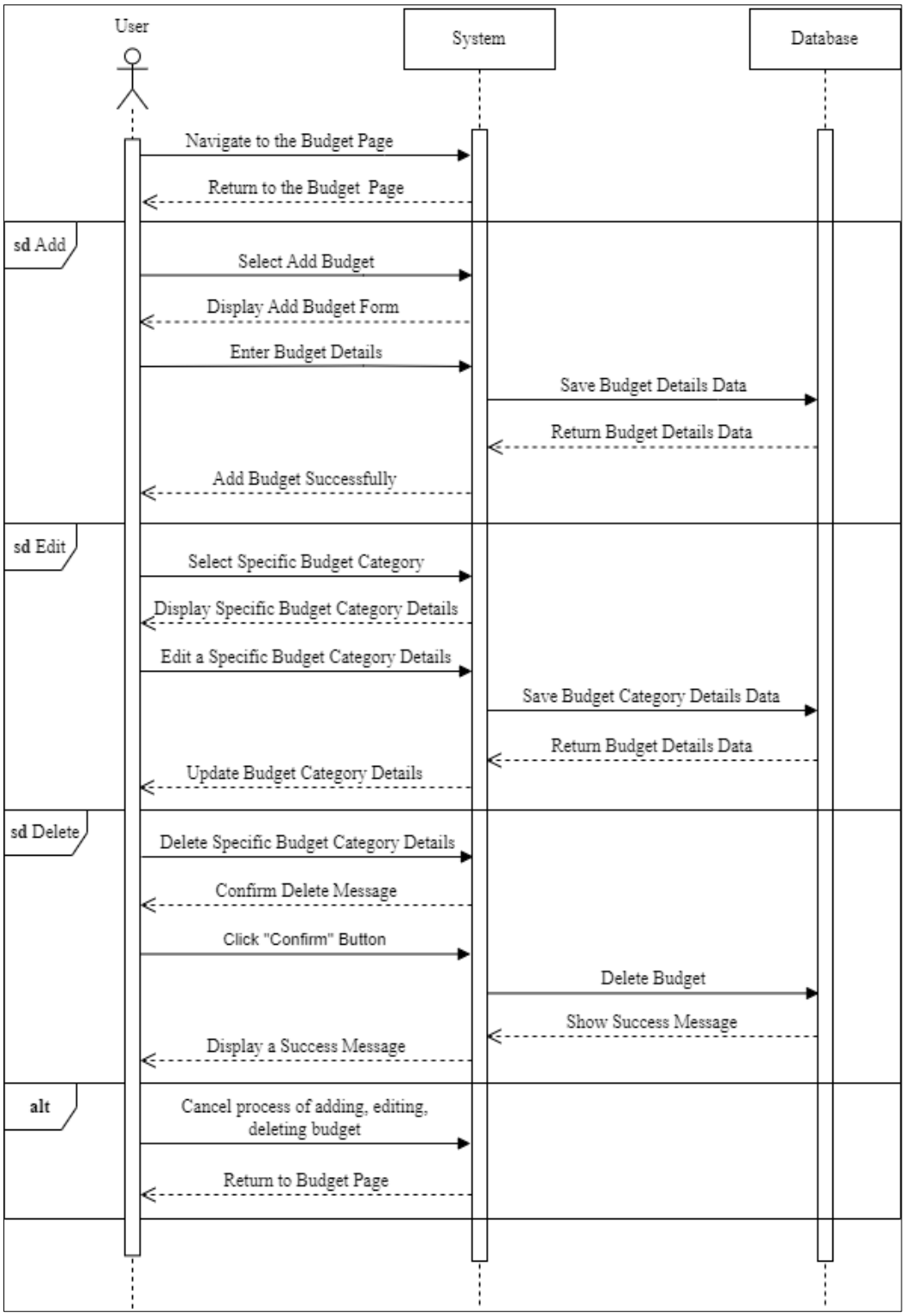


Figure 3.32: Sequence Diagram of Budget Page

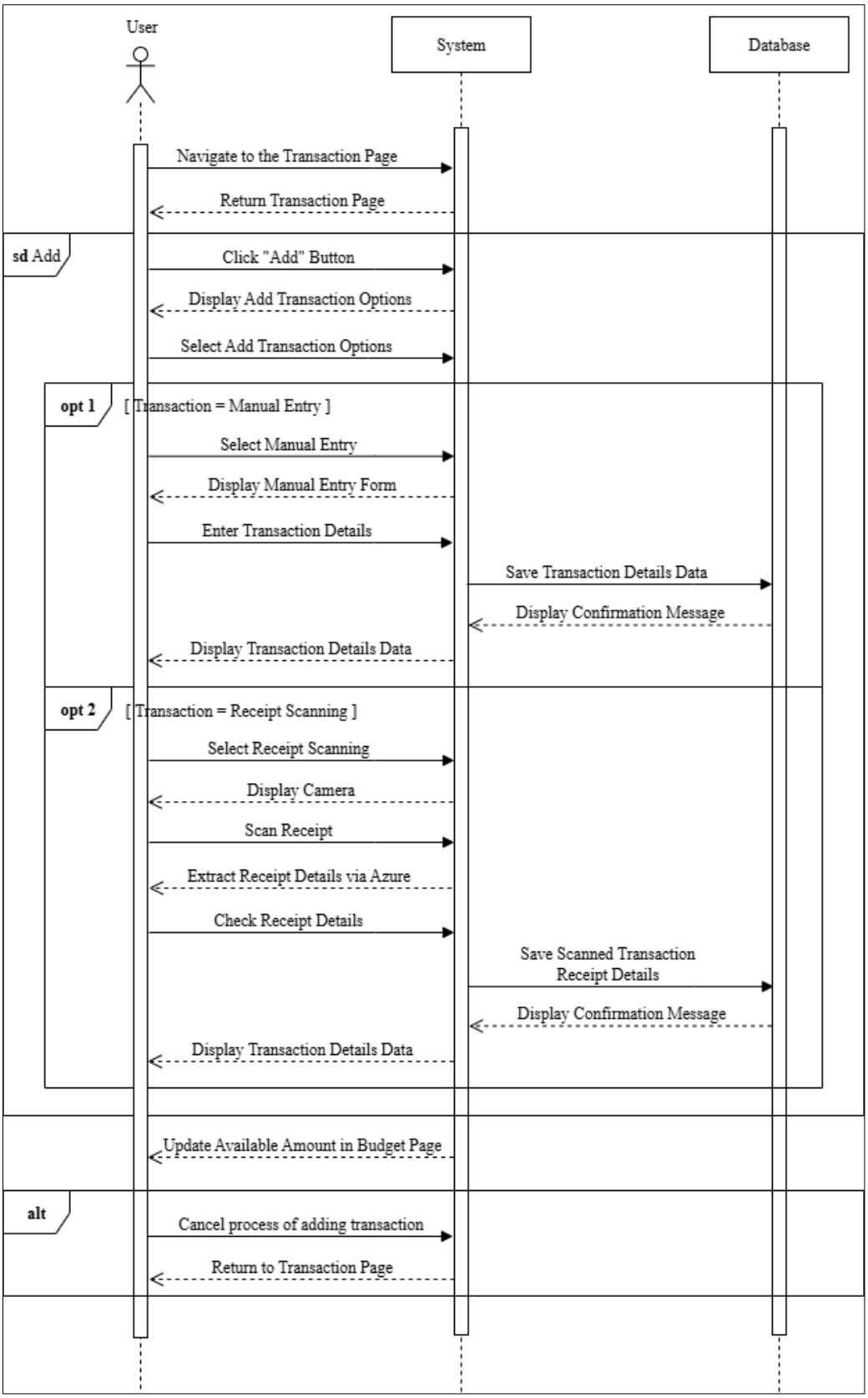


Figure 3.33: Sequence Diagram of Add Transaction Page

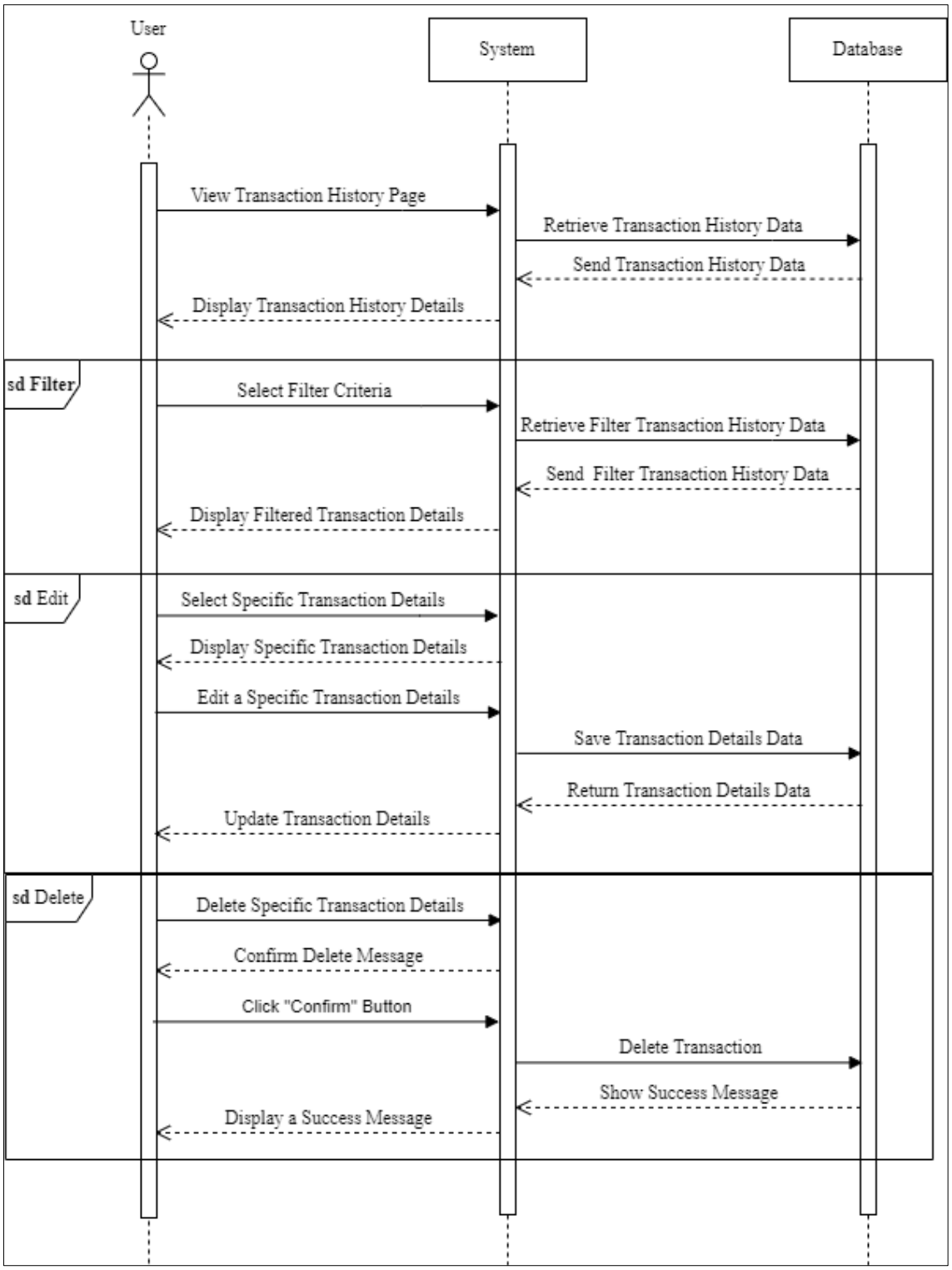


Figure 3.34: Sequence Diagram of View Transaction History Page

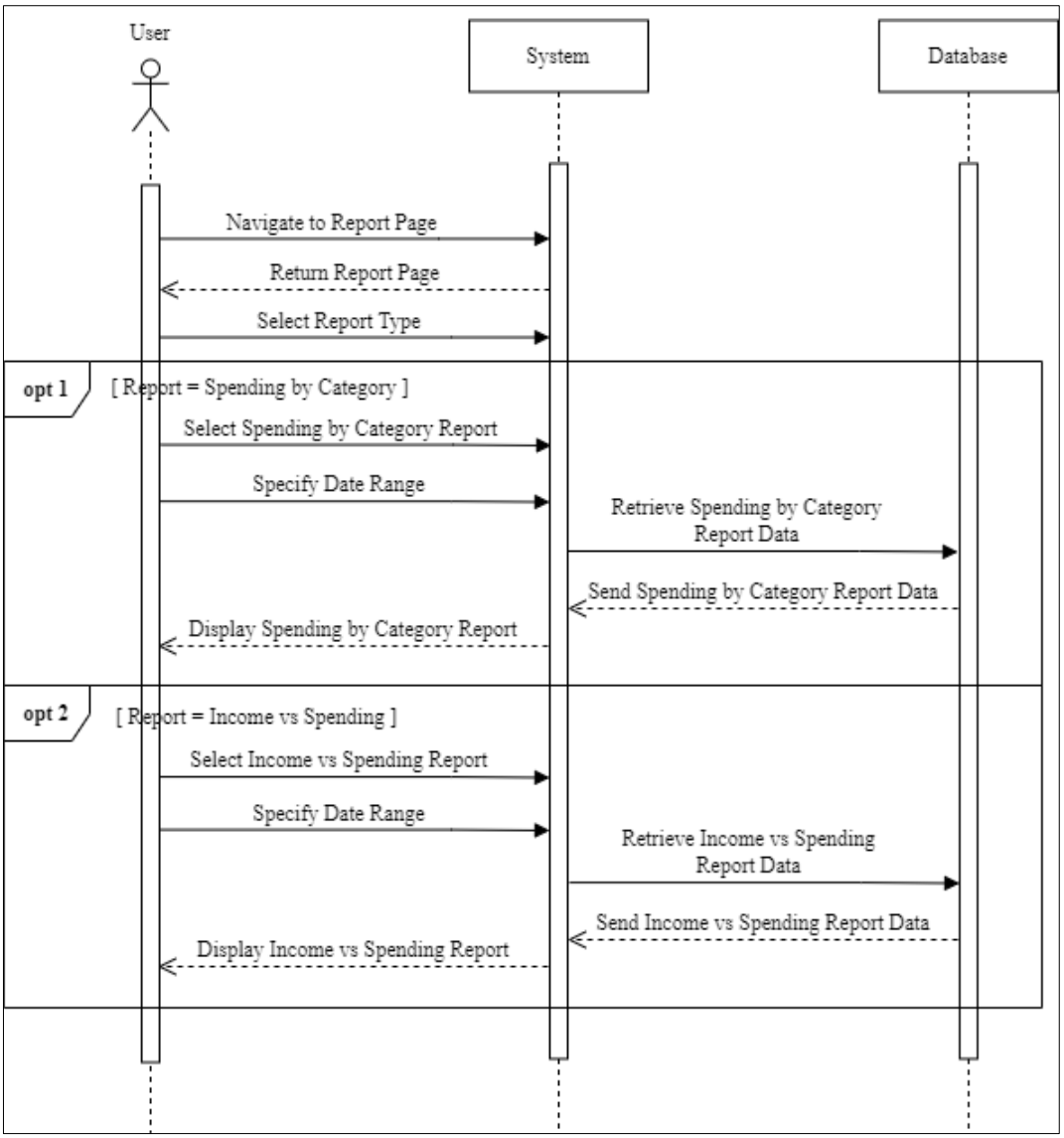


Figure 3.35: Sequence Diagram of Report Page

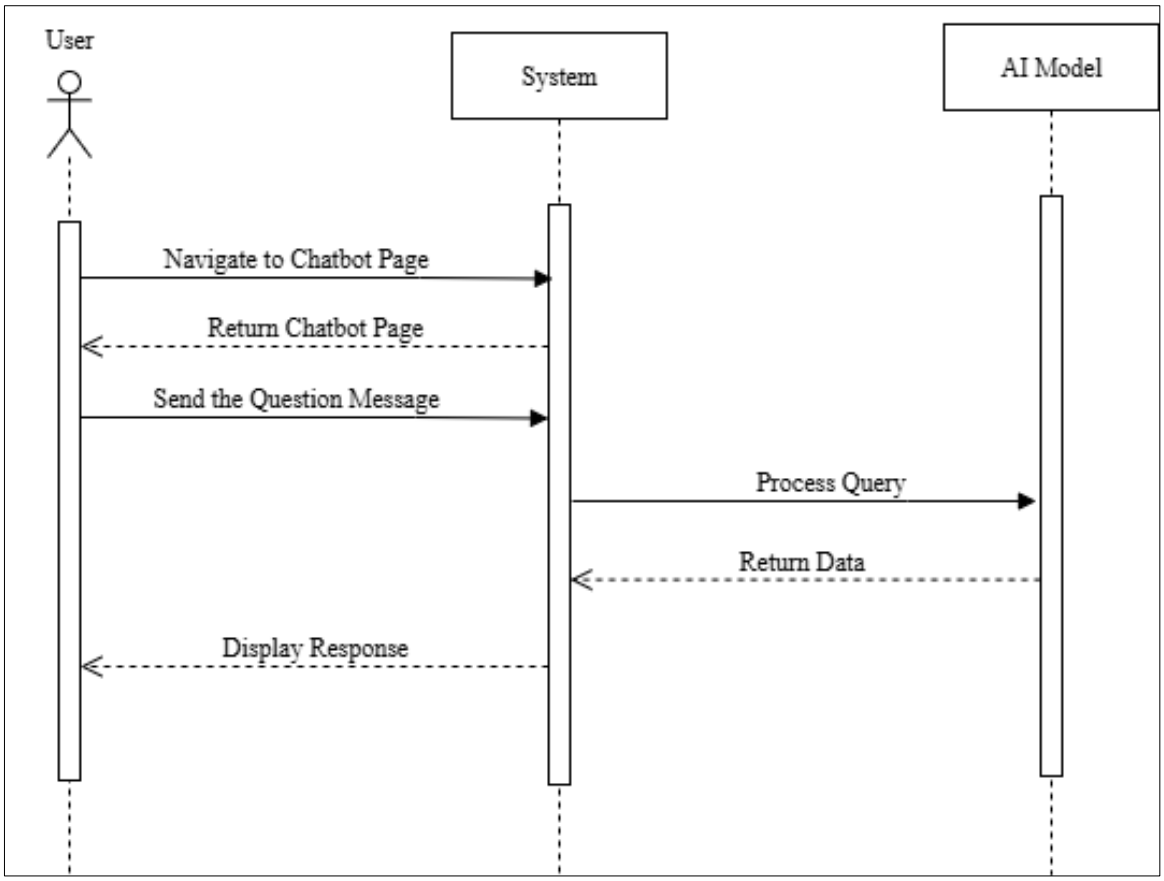


Figure 3.36: Sequence Diagram of Chatbot Page

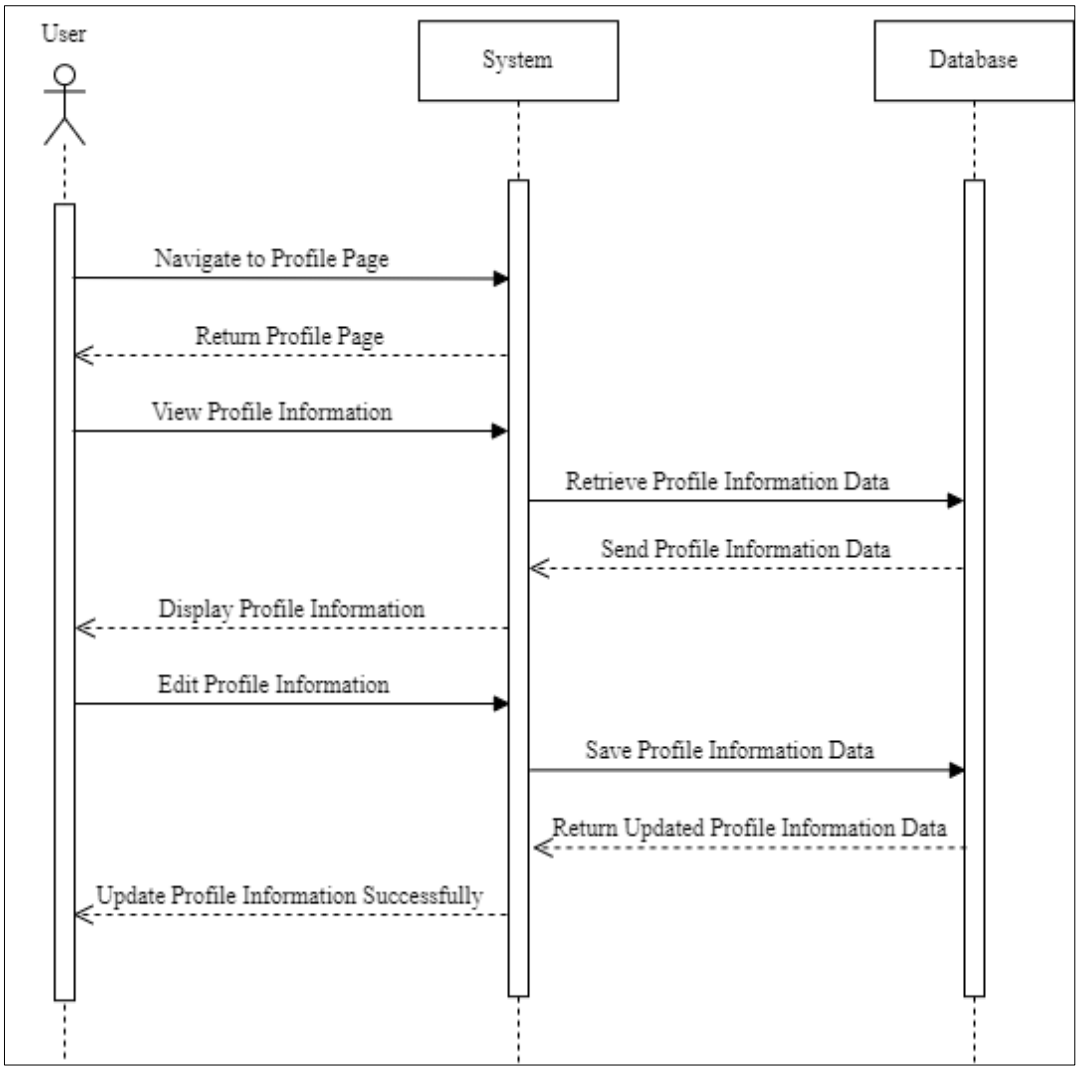


Figure 3.37: Sequence Diagram of Profile Page

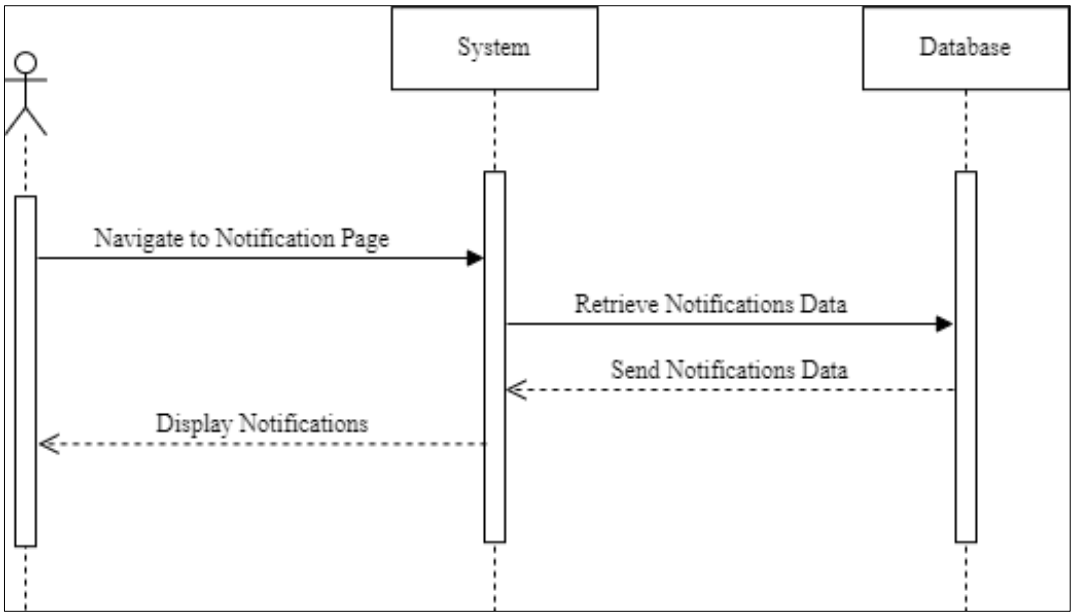


Figure 3.38: Sequence Diagram of Notification Page

3.6 User Interface Design

This section focuses on the user interface (UI) design of the proposed personal finance tracking mobile application. It serves as a blueprint to visualize the system's graphical user interface (GUI) before full implementation. The aim is to create an interface that is intuitive, user-friendly, and visually appealing to enhance the overall user experience. The design covers the app's layout, core functionalities, and navigation flow, ensuring a seamless and efficient interaction for users.

3.6.1 Login & Register Page

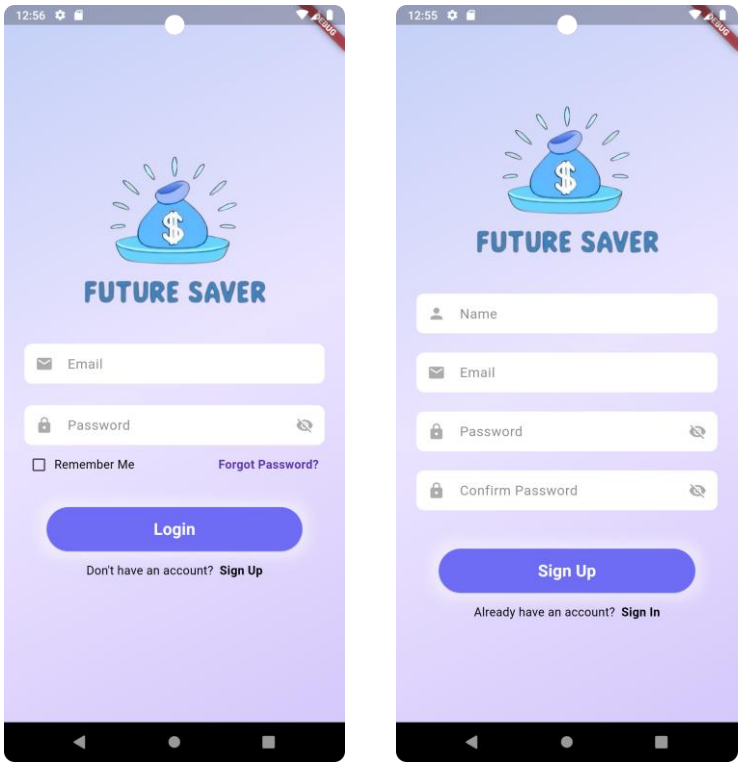


Figure 3.39: Login & Register Page

The figure above shows the login and registers screen of the Future Saver mobile application, where users can sign in with an existing account or register a new one if they are new. If users forget their password, they can click the "Forgot Password" link to initiate the password reset process. The screen also features a "Remember Me" checkbox that allows users to stay signed in for quicker access the next time they open the app.

3.6.2 Home Page

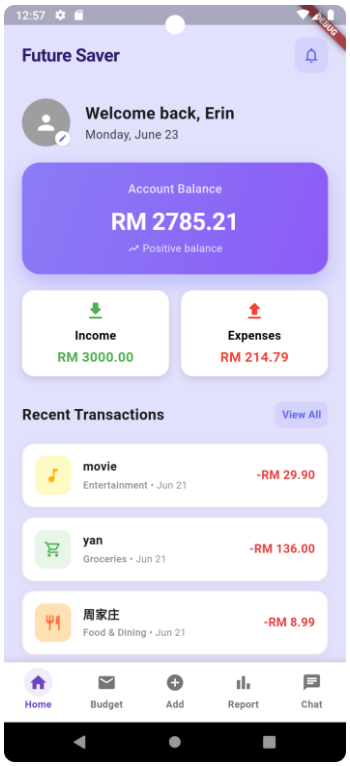


Figure 3.40: Home Page

The home page of the Future Saver mobile application provides users with an overview of their financial status. It shows the current account balance, as well as the total income and expenses for the selected month. Additionally, the page features a transaction history section which displays the recent transaction with details such as date, category, and amount. The home page is divided into four main section which are Budget, Add Transaction, Report and Chatbot at which users can easily navigate to these sections using the menu bar at the bottom of the app.

3.6.3 Budget Page

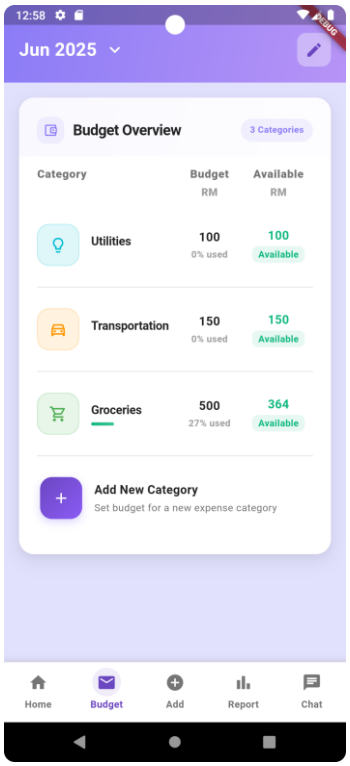


Figure 3.41: Budget Page

The budget page provides a clear overview of the user's financial allocations for each category in expense for the selected month. It displays the assigned budget and remaining balance for each category, with an option to add new categories for a better customization. Users can click the "Edit" icon to update the budget amount for a specific category or click the "Delete" icon to remove a category entirely.

3.6.4 Manual Entry for Expenses or Income Page

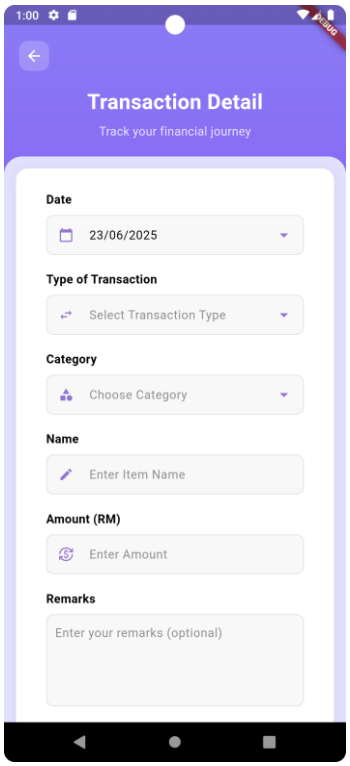


Figure 3.42: Manual Entry for Expenses or Income Page

Users can manually add income or expense through the entry screen. This page includes input fields for key details such as date, type of transaction, category, name, amount and optional remarks. Users can save the transaction by clicking the "Save" button or cancel the entry with the "Cancel" button.

3.6.5 Automated Transaction Tracking Via Receipt Scanning

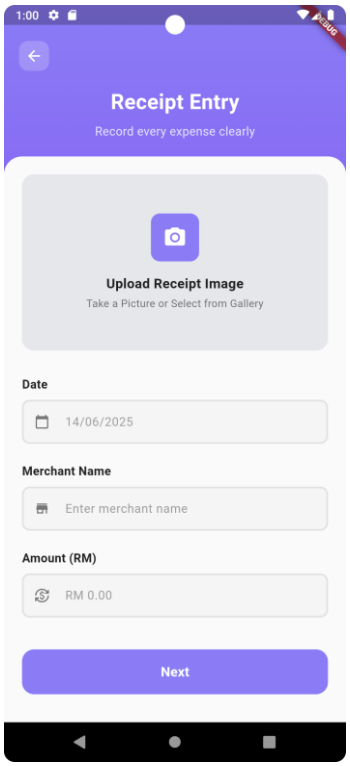


Figure 3.43: Automated Transaction Tracking Via Receipt Scanning

In addition to manual entry, users can automate the transaction tracking by uploading a receipt for scanning. The scanned receipt image is displayed at the top of the screen, and relevant details, such as the merchant's name, amount, and date are automatically extracted and pre-filled into the respective fields. Then, users can review or edit the information before proceeding by clicking the "Next" button.

3.6.6 Transaction History Page

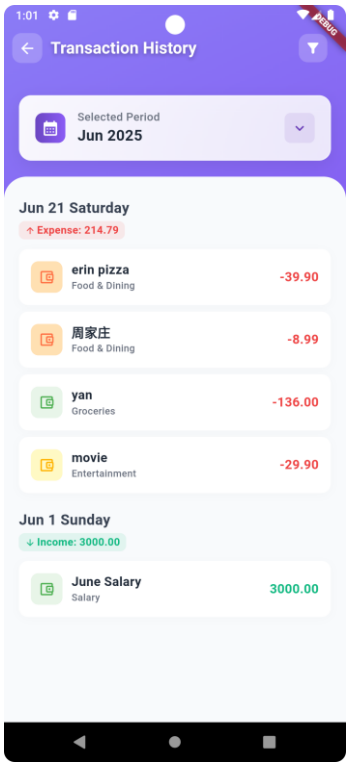


Figure 3.44: Transaction History Page

The transaction history page can be accessed from the home page by clicking the “See All” button. It allows users to view detailed information about their transactions and filter them by category. To edit or delete a transaction, users can simply click on the transaction to view its details and make the necessary changes.

3.6.7 Report Page

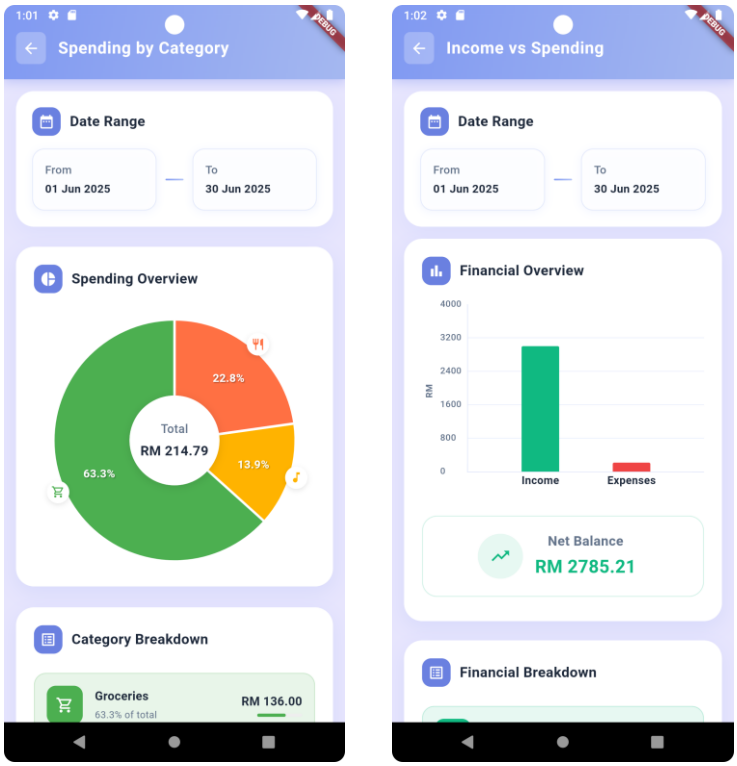


Figure 3.45: Reports Summary Page

There are 2 type of reports shown to the users which are Spending by Category and Income vs Spending. The Spending by Category report is displayed as a pie chart which show the percentage distribution of total spending across various categories. This help users to understand their spending habits and track how their money is distributed across categories. The Income vs Spending is presented as a graph. It provide a comparison between income and expenses, with the net balance is calculated for reference.

3.6.8 Chatbot Page

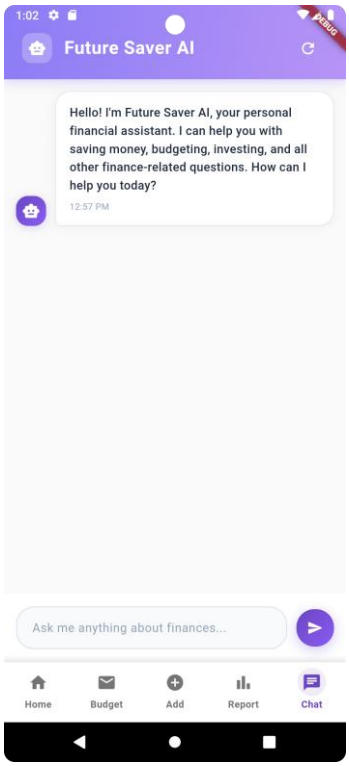


Figure 3.46: Chatbot Page

This figure shows the chatbot page of the Future Saver mobile application. Users can type their questions into the message box, such as asking how much they have spent on the rent category or seeking financial tips to save money and click the “Send” icon to submit the message. Then, the AI assistant will provide answers and offer advice based on the user's query.

3.6.9 Profile Page

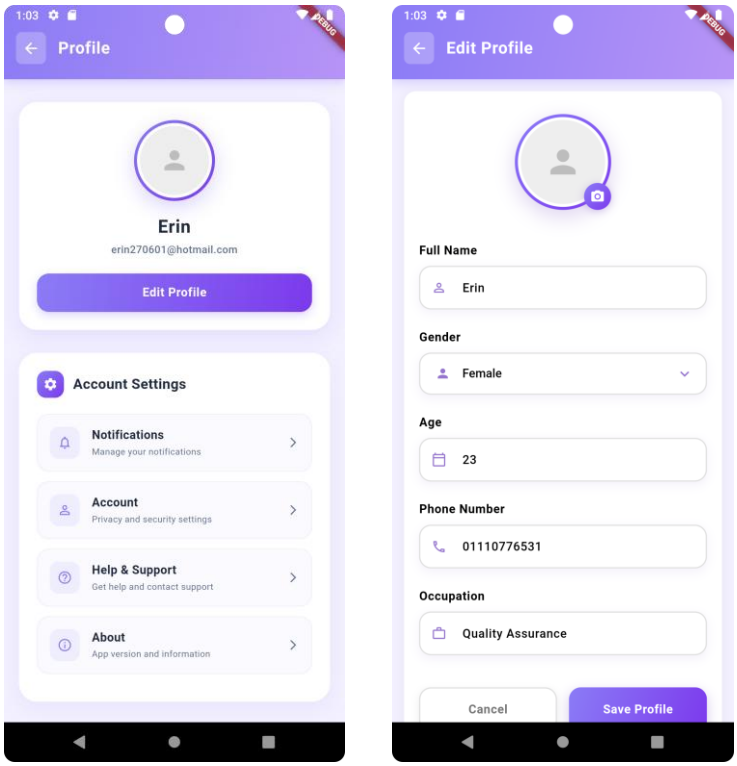


Figure 3.47: User Profile Page

Users can view their profile page by clicking on their profile picture icon from the home page. This page shows their username, phone number, and email address. Users can click the "Edit Profile" button to update their personal information. Additionally, users can also enable push notifications feature in the account settings section to receive alerts in the phone's status bar which can help them to stay on track of their budgeting habits.

3.7 Summary

In conclusion, this chapter identified and defined both the functional and non-functional requirements essential for the development of the Future Saver mobile application. It also introduced key system design elements, including use case, activity, and sequence diagrams, to illustrate user interactions and system behaviour. These design artifacts played a critical role in guiding the development process, ensuring the application is user-friendly, functionally robust, and aligned with the needs and expectations of its target users.

CHAPTER 4: SYSTEM IMPLEMENTATION

4.1 Introduction

This chapter discusses how the Future Saver mobile app is developed. It covers the hardware and software components used in the development process. In addition, it also describes how the Android Studio, Flutter, and Firebase were set up. The purpose of this chapter is to provide an overview of the development process and to show the steps taken to successfully launch Future Saver.

4.2 Hardware and Software Development Tools Requirements

This section outlines the hardware and software tools required to develop the Future Saver mobile application. It includes the minimum hardware specifications needed to run development environments smoothly, as well as the software tools and platforms used throughout the development process.

4.2.1 Hardware Development Tools

The hardware required for developing the proposed Smart Budget Personal Finance Tracking Application includes a laptop and a mobile phone. These devices are essential for coding, testing, and evaluating the application's performance to ensure it meets all functional and user experience requirements.

4.2.1.1 ASUS VivoBook 15

An ASUS VivoBook 15 laptop is used for the development of Future Saver. The table below shows the specifications of the laptop:

Table 4.1: Hardware Requirements of Laptop

Component	Specification
Processor (CPU)	Intel(R) Core (TM) i5-10210U CPU @ 1.60GHz
Graphic (GPU)	NVIDIA GeForce MX250
RAM	20.0 GB
Storage	512GB SSD

The laptop features an Intel Core i5 processor running at 1.60GHz. It provides enough processing power for app development tasks. Besides that, it also includes an NVIDIA GeForce MX250 graphics card that can help to handle graphics-related operations smoothly. The system is equipped with 20.0 GB of RAM, which allows smooth multitasking and run the development tools without lag. A 512 GB SSD is used for storage which offer fast read and write speeds that can enhance system's overall performance. In addition, a comfortable keyboard and mouse are used to enhance user experience and increase productivity during long development sessions.

4.2.1.2 HONOR 50

An Android smartphone which is HONOR 50 is used for testing to ensure the app can perform well on an actual mobile device. The table below shows the specifications of the mobile:

Table 4.2: Hardware Requirements of Mobile Device

Component	Specification
Processor (CPU)	Qualcomm Snapdragon 778G 5G
GPU	Adreno 642L
RAM	8 GB + 2GB (HONOR RAM Turbo)

Storage	256 GB
Display	6.57-inch OLED, 120Hz refresh rate

The HONOR 50 is powered by a Qualcomm Snapdragon 778G 5G processor and an Adreno 642L GPU. It has 8 GB of RAM, plus an additional 2 GB with HONOR RAM Turbo, which help to improve performance. The device has 256 GB of storage and features a 6.57-inch OLED display with a smooth 120Hz refresh rate which make it suitable for app testing and smooth user experience.

4.2.2 Software Development Tools

The development of the Smart Budget Personal Finance Tracking Mobile Application requires the integration of various software tools to ensure seamless functionality and performance. These tools have been carefully chosen for their capabilities and relevance to the project's requirements, providing a strong foundation for efficient and effective development. The table below shows the software requirements for developing the Future Saver mobile application.

Table 4.3: Software Requirements

Component	**Specification**
Operating System	Window 11
System Type	64-bit operating system, x64-based processor
Software	Android Studio, Flutter
Programming Language	Dart
Database	Firebase

The development of the Future Saver app is developed on a Windows 11 system with a 64-bit, x64-based processor. Android Studio is used as the main development tools while

Flutter is used for cross-platform mobile app, with Dart as the programming language. Firebase served as the backend database to store and manage user data.

4.2.2.1 Implementation of Integrated Development Environment (IDE)

Figure 4.1: Android Studio (Google Developers, 2024)

The Integrated Development Environment (IDE) used for developing the Future Saver application is Android Studio. It is a Google's Android Operating System and is built on IntelliJ IDEA and provides a powerful set of tools and features designed specifically for Android app development. It includes essential tools such as a code editor, visual layout designer, emulator and debugging features to help developers efficiently build, test and maintain the Future Saver mobile application. The following are the steps to install Android Studio to set up the development environment for Future Saver:

1. Go to the official Android Studio website at https://developer.android.com/studio.
2. Click the Android Studio Package for windows (64-bit) to download the file and make sure it is .exe file.
3. Once the file is downloaded, double-click it to run the installer and start the setup process. Then the Android Studio Setup Wizard will appear to guide you through the installation.

4. The wizard will ask to choose the components. By Default, the necessary components will be selected so just proceed with the default settings or add additional components if needed.
5. Select the folder path where it should be installed or keep the default location. Click “Next” to proceed.
6. Click the “Install” button, and the installer download and install the necessary files.
7. Once the installation is complete, click “Next” and then “Finish” to close the setup wizard.
8. After the installation, launch the Android Studio from the Start Menu or desktop shortcut.
9. On first launch, Android Studio will prompt to install and set up the Android SDK (Software Development Kit). Click “Next” to download and install the required SDK components.
10. After the SDK setup is complete, Android Studio will configure the environment, and you can create a new project or open an existing one to start developing Android apps.

4.2.2.2 Implementation of Framework (Flutter)

Figure 4.2: Flutter (Flutter, 2024)

Flutter is the framework chosen to develop the app as it enables creating cross-platform mobile applications for both Android and iOS using a single codebase. It uses the Dart

programming language and provides a rich set of prebuilt, customizable widgets that simplify UI design and ensure a native-like experience. Flutter also offers a fast development cycle with features like hot-reload, making it easy to preview changes instantly without restarting the app. The following are the steps to install Flutter:

1. Go to the official Flutter website at https://docs.flutter.dev/get-started/install/windows/mobile.
2. Click the "Download and install" to download the flutter SDK zip file.
3. Extract the flutter SDK zip file into directory (C:\Users\{username}).
4. Open the Start Menu, search for "Edit the system environment variables", and click to open the System Properties to update window path variable.
5. In the System Properties window, click Advanced and then Environment Variables, the environment variables dialog displays.
6. Under User variables, select Path and click Edit.
7. Click New and type %USERPROFILE%\dev\flutter\bin entry.
8. Move the new path entry to the top of the list if needed, then click OK on all dialog boxes to save your changes.
9. Then open Android Studio, install the Flutter and Dart plugins, and restart Android Studio so you can start creating new Flutter projects.

4.2.2.3 Implementation of Database Tool (Firebase)

Figure 4.3: Firebase (Google, 2024)

Firebase is used for the app as the backend service can manage user data, authentication, and real-time synchronization. It provides a scalable and secure cloud database, built-in authentication, and seamless integration with mobile apps. Implementing Firebase simplifies data storage, real-time updates, and user account management across multiple devices. The following are the steps to implement Firebase:

1. Go to the Firebase Console at https://console.firebase.google.com.
2. Click on “Create a Firebase project” and follow the prompts to set up a new Firebase project.
3. Once the project is created, click “Add app”, select Android, then register the app by entering the package name. Follow the instructions to download the google-services.json file and place it in the Flutter project’s android/app/ directory.
4. In the Flutter project, open the pubspec.yaml file and add the required Firebase dependencies (e.g. firebase_core, firebase_auth, cloud_firestore). The latest versions can be found at https://pub.dev/packages.
5. Run flutter pub get in the Android Studio terminal to install the dependencies.
6. In the main.dart file, import firebase_core and initialize Firebase inside the main() function with WidgetsFlutterBinding.ensureInitialized() and await Firebase.initializeApp() before running the app.
7. Configure Firebase Authentication in the console and implement features like user sign up, log in, and password reset in the Flutter app.
8. In the Firebase Console, navigate to Cloud Firestore and set up the database rules so that only authenticated users can read and write their own data.
9. Finally, test the app thoroughly to ensure that Firebase is integrated properly — check sign-in, data read/write, and real-time updates.

4.3 System Architecture

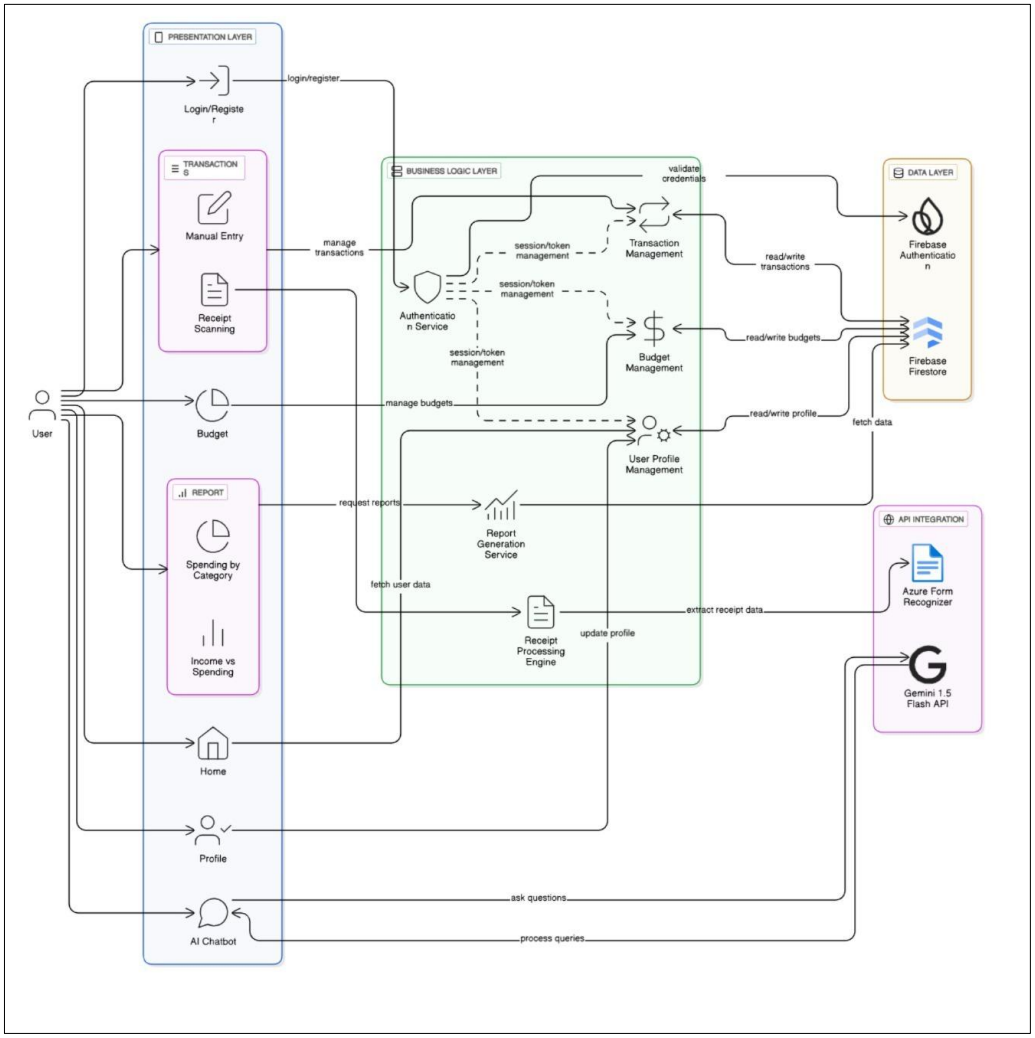


Figure 4.4: Architecture Diagram of Future Saver App

This system architecture is built with three main layers, which are the Presentation Layer, the Business Layer, and the Data Layer. These layers work together to make sure the app is easy to use, stable, and efficient. The Presentation Layer is where the user interacts with the app on a mobile device. It provides the user interface that allows the user to navigate between screens and enter data easily. The Business Layer contains all the important logic of the app. It connects the interface with the data and is responsible for handling requests. In this layer, the app uses Firebase services for authentication and data management and an External API service for getting data or features from other platforms. Finally, the Data Layer is where all the data is stored and retrieved. It uses Firebase Firestore as the primary database and also

keeps local data on the device for quicker access. Together, these three layers help the app to run smoothly, give accurate results, and keep the user experience clear and simple.

4.3.1 Presentation Layer

The presentation layer allows the user to interact directly with the application. It includes all the visual elements and screens of the app, such as the home page, budget page, transaction page, report page, chatbot page, and profile page. This layer is responsible for creating an intuitive and engaging user experience by handling user inputs, gestures, and interface updates. It also ensures consistency and usability across all screens. Importantly, the presentation layer separates the user interface from the underlying business logic and data, enabling easier maintenance and scalability of the app.

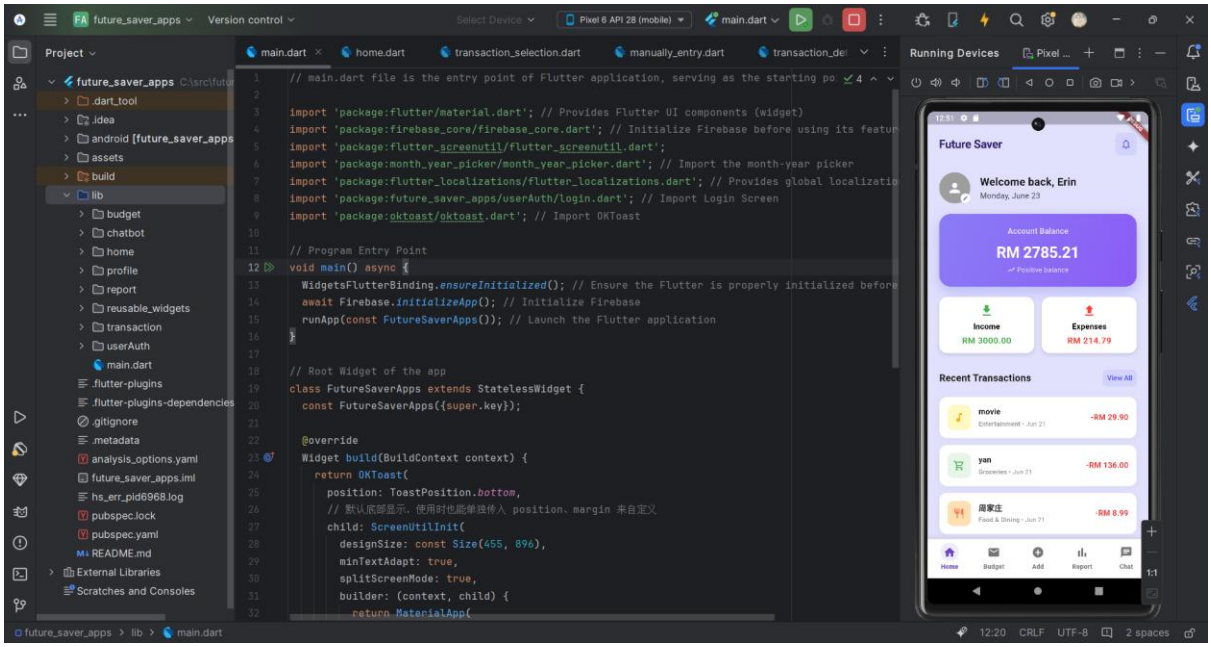


Figure 4.5: Home Page of Future Saver App

Each file in the Future Saver app contains specific code and functions related to a particular task or feature. The code is organized into separate files based on its purpose which make it easier to manage, maintain, and update. The table below illustrates the file structure of the system.

Table 4.4: File Structure of Future Saver App

Folder	**Description**
future_saver_apps/android	This folder contains all the files and settings that are needed to run the app on the Android devices. It includes the setup for the Android platform, like Gradle files and permissions.
future_saver_apps/assets	This folder holds all the images, fonts, and other visual files that the app needs to display properly.
future_saver_apps/build	This folder is created automatically when build or compile the app. It contains temporary files and the final output that can be installed or tested on devices.
future_saver_apps/lib	Core folder that contains the Dart source code for the app, including widgets, logic, and features that make up the core functionality.

4.3.2 Business logic Layer

The Business Logic Layer is responsible for implementing the core functionality and business rules of the application. It acts as an intermediary between the Presentation Layer and the Data Layer, ensuring that all data is properly validated, processed, and formatted before it is saved to or retrieved from the database. This layer handles tasks such as input validation, data transformation, calculations, and decision-making based on business requirements. By centralizing this logic, the Business Logic Layer improves the maintainability and scalability of the application, reduces duplication of code, and simplifies future updates or feature enhancements.

4.3.3 Data Storage Layer

The Data Storage Layer is the part of the architecture that is responsible for storing, retrieving, and managing data. It handles all interactions with databases, local files, or cloud services, ensuring that data is saved securely and can be accessed quickly when needed. By separating data storage from the rest of the application, this layer simplifies maintenance, improves data consistency, and allows the app to scale more easily as the amount of data grows.

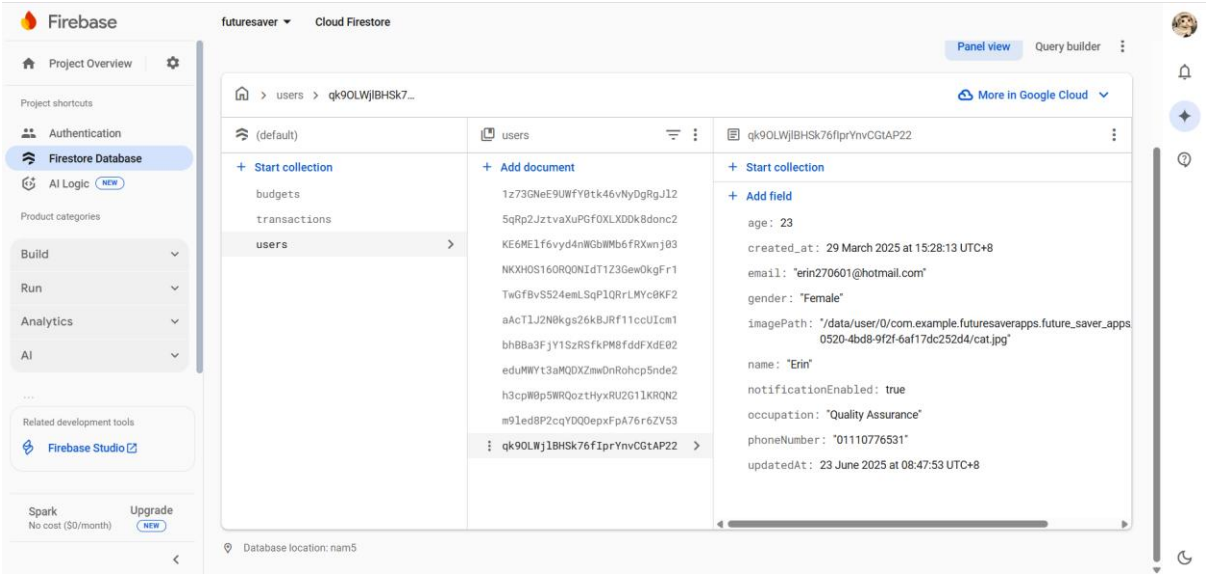


Figure 4.6: Firebase Firestore Database of Future Saver

4.4 Future Saver User's Description

The general users of the Future Saver app are individuals who want to manage their personal finances easily and efficiently. This includes people such as working professionals, students, homemakers, and small business owners who may not have specialized financial knowledge but need a simple and accessible tool to help them to track their daily income, expenses, and savings.

These users typically want to improve their financial habits, avoid overspending, and gain better control of their budgets. They prefer an intuitive interface that allows them to quickly set the budget, record transactions, view spending report summaries, and review their

financial goals without dealing with complicated features. By using the Future Saver app, they can monitor their financial health, identify spending patterns, and make informed decisions that align with their personal financial goals.

4.5 Future Saver Module Implementation

The Future Saver app is built using a modular structure to make the code easier to manage and update. Each module is responsible for a specific set of tasks. This approach allows the app to stay organized and work smoothly as a whole.

Table 4.5: Future Saver App's Module

Module	**Description**
User Authentication	Handles user account creation, login, and password reset using Firebase Authentication to keep the data secure.
Budget	Enables users to set monthly budgets for different categories, track their remaining amounts to avoid overspending, and add, edit, or delete budget entries as needed.
Transaction	Provides two options for adding transactions: • Manual Entry – Users input details like date, type, category, name, and amount. • Receipt Scanning – Users scan a receipt, and the app extracts the transaction details automatically, then pre-fills the form for review before saving. Enables users to add, edit, and delete income and expense transactions. Both methods update account balance, total income, total expenses, and available budget in real time.

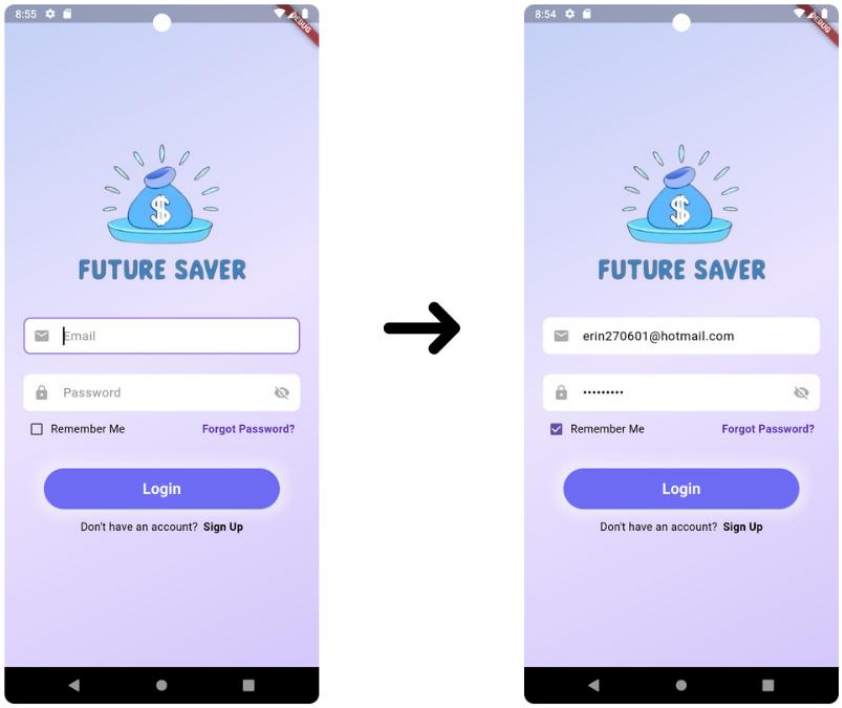
Report	Provides two types of financial summaries: • Spending by Category – Displays all expense categories in a pie chart and shows the percentage and total amount spent for each category in the spending breakdown. • Income vs. Spending – Displays total income and total expenses in a bar chart and calculates the net balance so users can easily see their financial position.
AI Chatbot	Provides financial advice and responds to user questions, powered by a Gemini AI API for personalized guidance.
Profile	Enables users to view and manage their personal information, such as their profile picture, name, gender, age, contact details, and occupation. Users can also update their profile whenever needed to ensure their account stays up to date and personalized. Users can also choose to enable push notifications for timely alerts and reminders.

By separating these features into different modules, the Future Saver app is easier to build, test, and improve. Each module is independent but also works together with the others, making sure the entire app is stable and user-friendly.

4.6 Interface of Future Saver

The following screenshots showcase the interface of Future Saver. Each image highlights a different module in the app, demonstrating its key features, how it functions, and how users can easily navigate and use the app.

4.6.1 Login Page

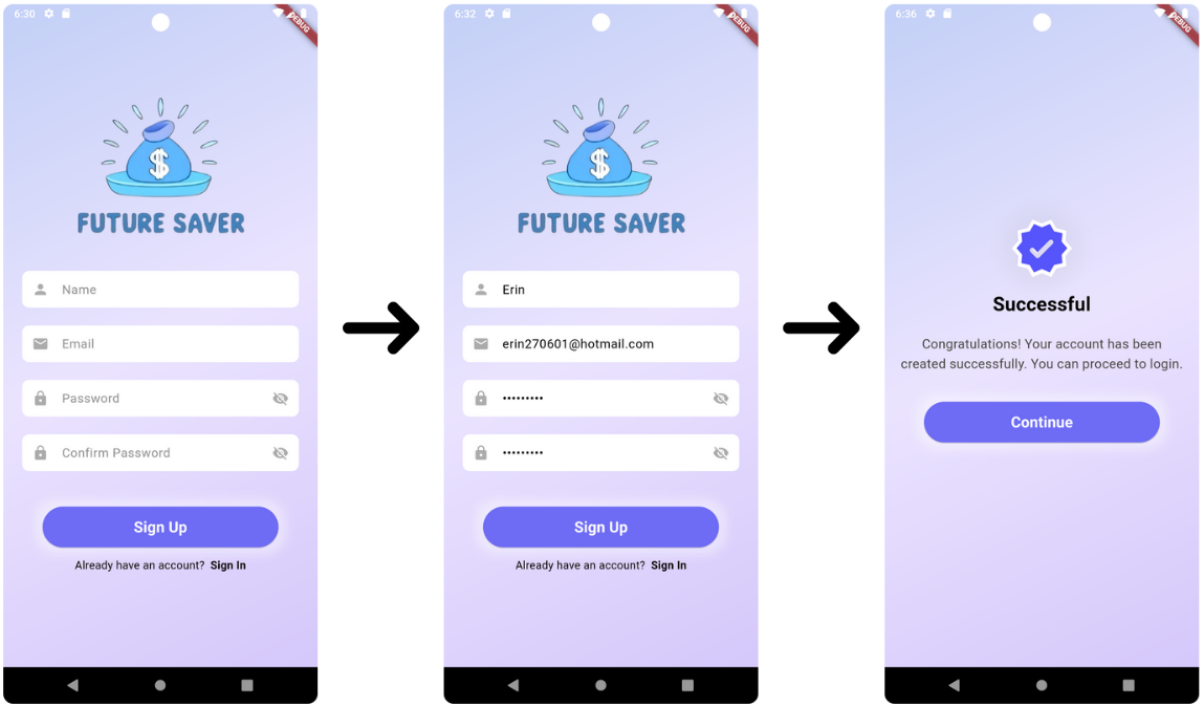


(a) (b)

Figure 4.7: Login Page of Future Saver

Figure 4.4 shows the login page of the Future Saver mobile app. In the initial view (a), users are prompted to enter their email and password to log into the app. The page also provides a "Remember Me" option so that users can stay logged in and do not need to re-enter their credentials every time when they open the app. The "Forgot Password?" link is available for users who need to reset their password, and a "Sign Up" link is provided for new users who do not yet have an account. Once the user enters their login credentials and presses the Login button, the app navigates to the next screen as shown in view (b), allowing the user to access the main features of the Future Saver application.

4.6.2 Register Page



(a) (b) (c)

Figure 4.8: Register Page for Future Saver

Figure 4.5 shows the registration process of the Future Saver mobile app. In the initial view (a), new users are prompted to enter their name, email address, password, and confirm password to create a new account. Once the required information is filled in and the Sign In button is clicked, as shown in view (b), the app validates the credentials and registers the new account. Finally, view (c) displays a confirmation screen indicating that the account has been successfully created. The user can then tap Continue to proceed to the login page and access the main features of the Future Saver app.

4.6.3 Reset Password Page

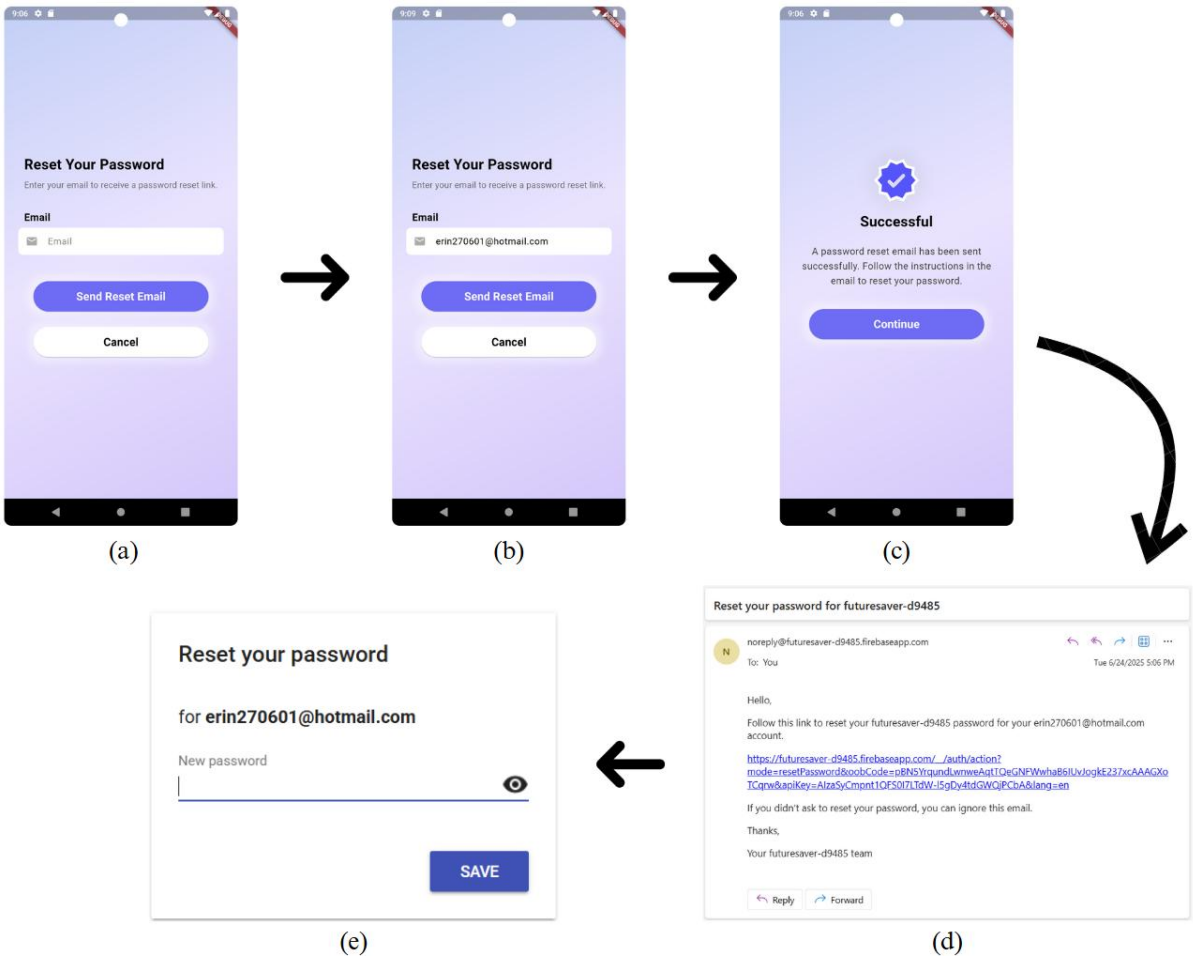


Figure 4.9: Reset Password Page for Future Saver

Figure 4.6 shows the reset password process for the Future Saver mobile app. In view (a), the user is prompted to enter their email address to receive a password reset link. Once the user provides a valid email address and clicks the Send Reset Email button as shown in view (b), the app sends a reset link to the user's email inbox. View (c) confirms that the reset email was successfully sent which allow the user to tap Continue. The user then checks their inbox for the reset message (d), which contains a secure link to reset their password. After clicking the link, as shown in view (e), the user can enter a new password and save the changes. Once completed, the password is successfully updated, allowing the user to log in with their new credentials.

4.6.4 Home Page

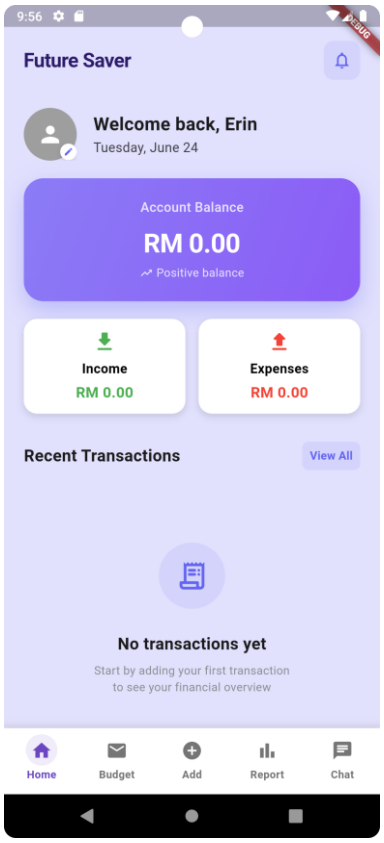


Figure 4.10: Home page of Future Saver

Figure 4.7 shows the Home page of the Future Saver app. This page provides a quick overview of the user's current financial status, including the Account Balance, Total Income, and Total Expenses. Every time a new transaction is added, the app will automatically update the total income and expenses displayed on this screen, and the account balance will also be recalculated in real time. The most recent transactions appear in the Recent Transactions list, allowing users to easily track their spending and income.

4.6.5 Budget Page

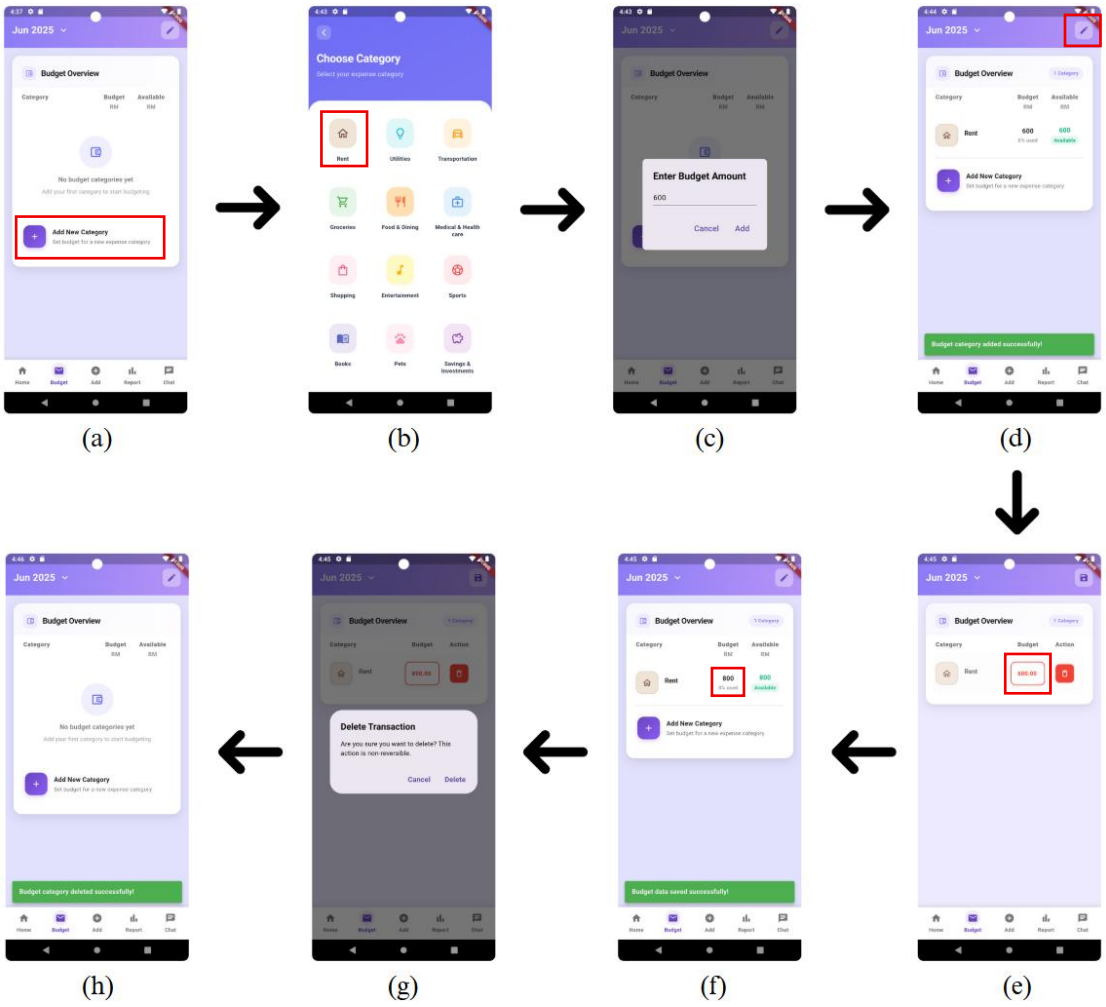


Figure 4.11: Budget Page for Future Saver

Figure 4.8 shows the Budget page in the Future Saver app. In the initial view (a), users can see the Budget Overview, which lists all existing expense categories, their allocated budgets, and the available amounts. Users can click the Add New Category to create a new budget. In view (b), the app displays a list of predefined expense categories to allow user to choose one expense category to set for budget. Once a category is selected, as shown in view (c), the user is prompted to enter a Budget Amount and click Add. View (d) confirms that the new budget category was successfully added and is displayed in the overview list. View (e) and (f) illustrate what happens when the user edits the allocated amount, the updated value is displayed along with a green success message indicating that the data was saved successfully.

Finally, views (g) and (h) demonstrate the process for deleting a category. The user is asked to confirm the deletion in a dialog box, and once deleted, the category is removed, with a green confirmation message shown at the bottom of the screen. This Budget page enables users to easily add, edit, and remove budget categories so they can manage their spending limits and keep track of their finances at a glance.

4.6.6 Manually Entry Transaction Page

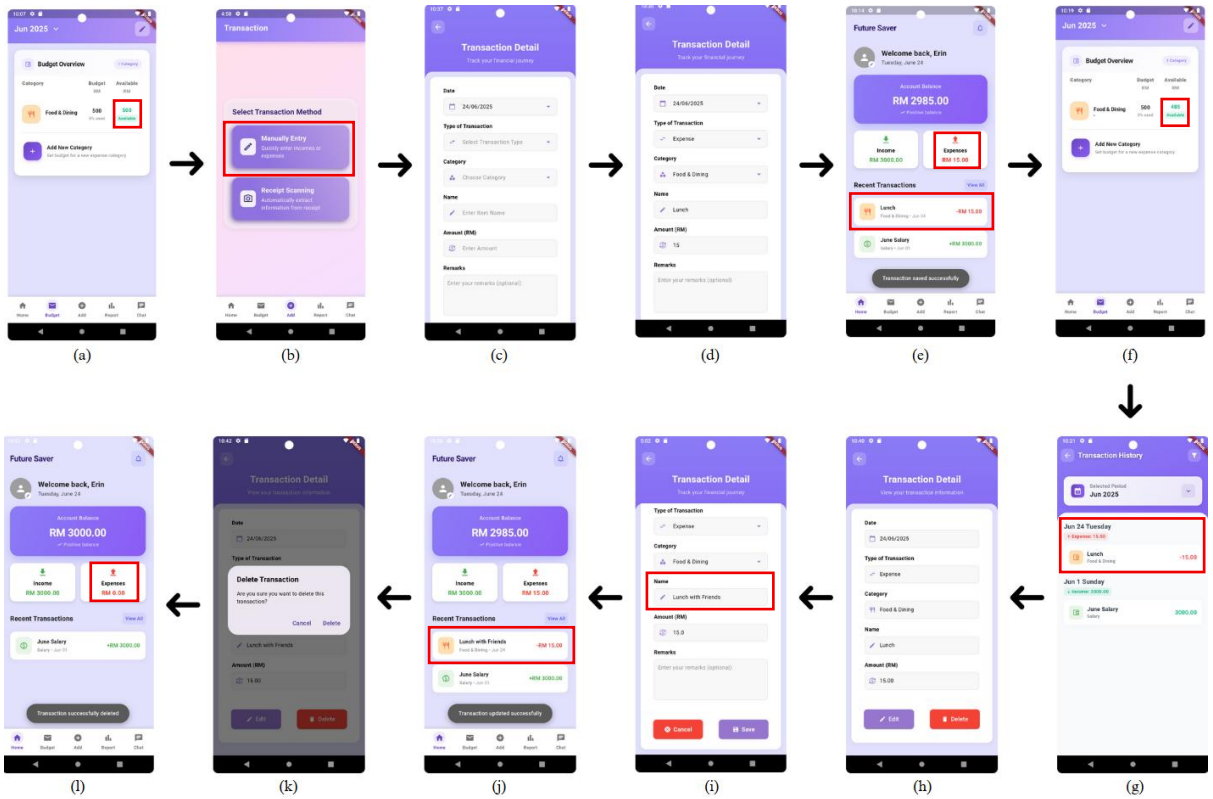


Figure 4.12: Manually Entry Transaction for Future Saver

Figure 4.9 illustrates the process of manually adding a transaction in the Future Saver app. First, in view (a), the user selects Manually Entry as the transaction method. The app then navigates to the Transaction Input page, shown in view (b), where the user can enter transaction details such as the date, transaction type (income or expense), category, name, amount, and any optional remarks. After the user enters all the information and taps Save (view (c)), the transaction is successfully added. The app returns to the Home page in view (d), and a

confirmation message “Transaction saved successfully” is displayed. The new transaction also appears under the Recent Transactions list. When the user taps the See All button on the home screen, they are directed to the Transaction History page shown in view (e), where all past transactions are listed by date and category. If the user taps on one of the transactions, view (f) shows the Transaction Detail page. From here, they can either Edit or Delete the transaction. View (g) illustrates the process of editing a transaction, after making changes and saving, a success message is displayed, as seen in view (h). Similarly, view (i) shows the delete confirmation dialog, and after deleting, the app returns to the home page (j) with the updated transaction list.

4.6.7 Receipt Scanning Transaction Page

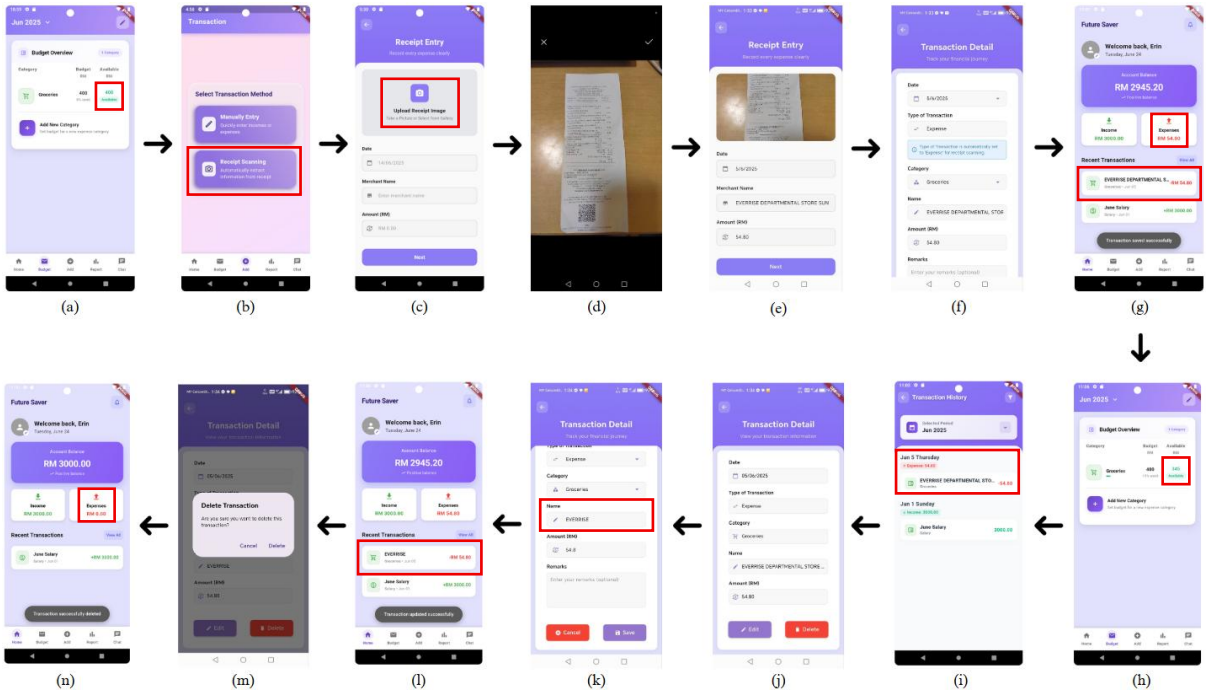
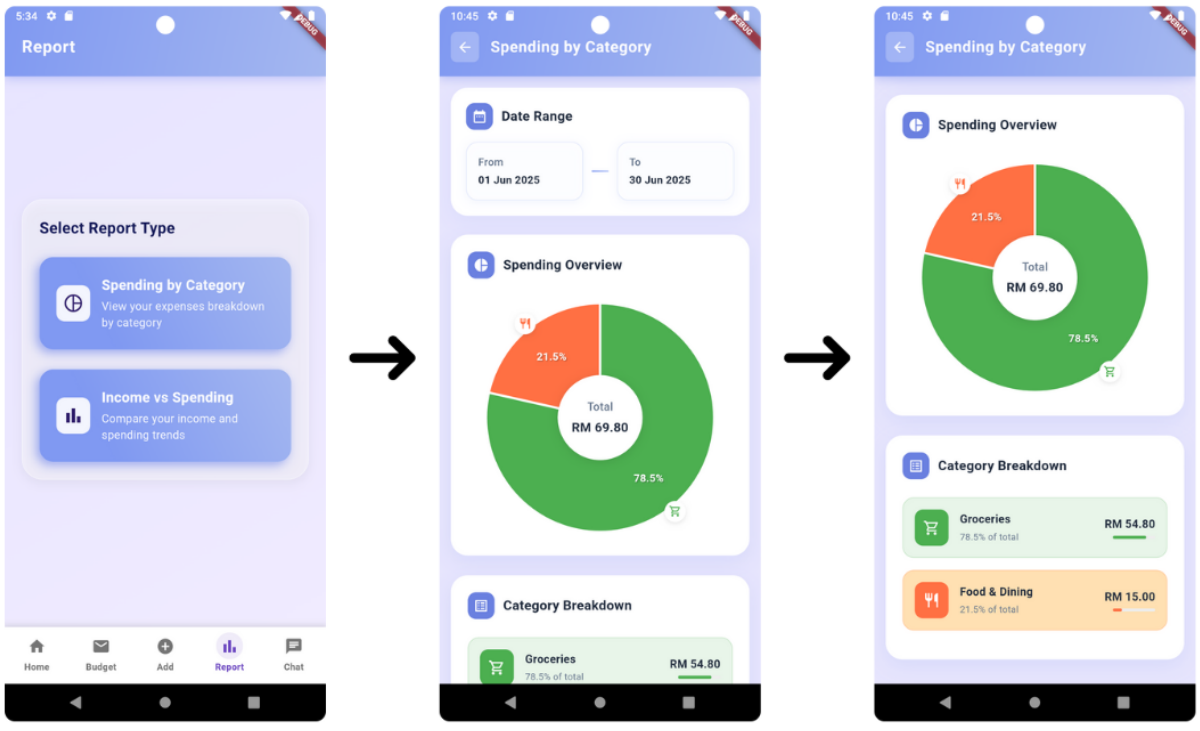


Figure 4.13: Receipt Scanning Transaction for Future Saver

Figure 4.10 illustrates the Receipt Scanning process in the Future Saver app. In view (a), the user selects Receipt Scanning as the transaction method. In view (b), the user is prompted to upload or capture an image of the receipt. After uploading the receipt image as

shown in view (c), the app scans the receipt and automatically extracts the date, merchant name, and amount, which is displayed in view (d). The user can then tap Next to add any other transaction details, such as category and remarks, as shown in view (e). Once all the required information is completed, the user taps Save to store the transaction. The remaining steps (views f through l) follow the same process as the manual transaction entry. The saved transaction appears under Recent Transactions on the Home page (view f), can be viewed in the Transaction History list (view g), and can be edited or deleted if needed (views h–l).

4.6.8 Spending by Category Report Page



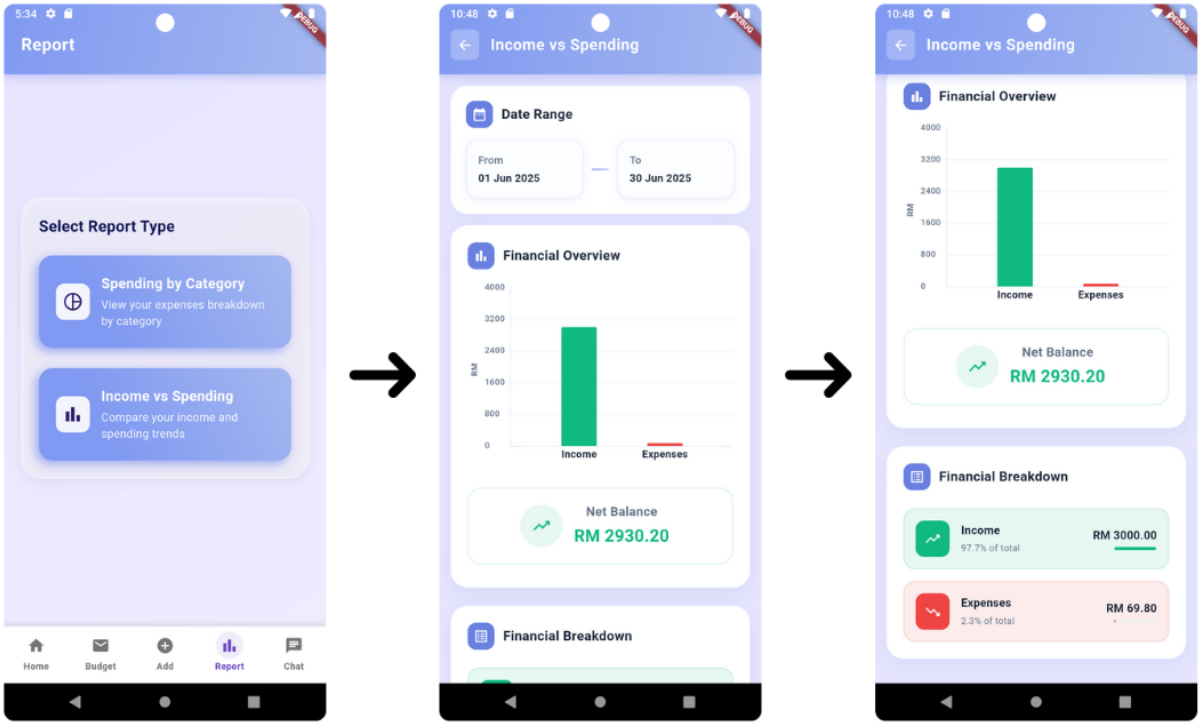
(a) (b) (c)

Figure 4.14: Spending by Category Report for Future Saver

Figure 4.11 illustrates the process for viewing the Spending by Category report in the Future Saver app. In view (a), the user selects the Spending by Category report option. After selecting this report, as shown in view (b), the app automatically displays a pie chart summarizing the user's spending across different categories for the current month. Beneath the

chart is a Category Breakdown that lists each expense category along with its amount and percentage of total spending. In view (c), the report is displayed with a clear summary of each category's contribution to the total expenses. The user can also change the reporting period by tapping the Date Range selector. In view (d), a calendar dialog allows the user to pick a custom date range. Once confirmed, the report updates instantly to reflect the spending breakdown for the new period.

4.6.9 Income vs Spending Report Page



(a) (b) (c)

Figure 4.15: Income vs Spending Report for Future Saver

Figure 4.12 illustrates the process for viewing the Income vs Spending report in the Future Saver app. In view (a), the user selects the Income vs Spending report option. Once selected, as shown in view (b), the app automatically displays a summary of the user's income and expenses for the current month, including a bar chart comparing the two, the calculated net balance, and a financial breakdown. In view (c), the report screen updates instantly when the

data is loaded. The user can also tap on the Date Range selector to view data for a different period. This brings up a date picker, as shown in view (d), allowing the user to select a custom date range. Once the date is confirmed, the report is refreshed to reflect the updated income and spending for the selected period.

4.6.10 Chatbot Page

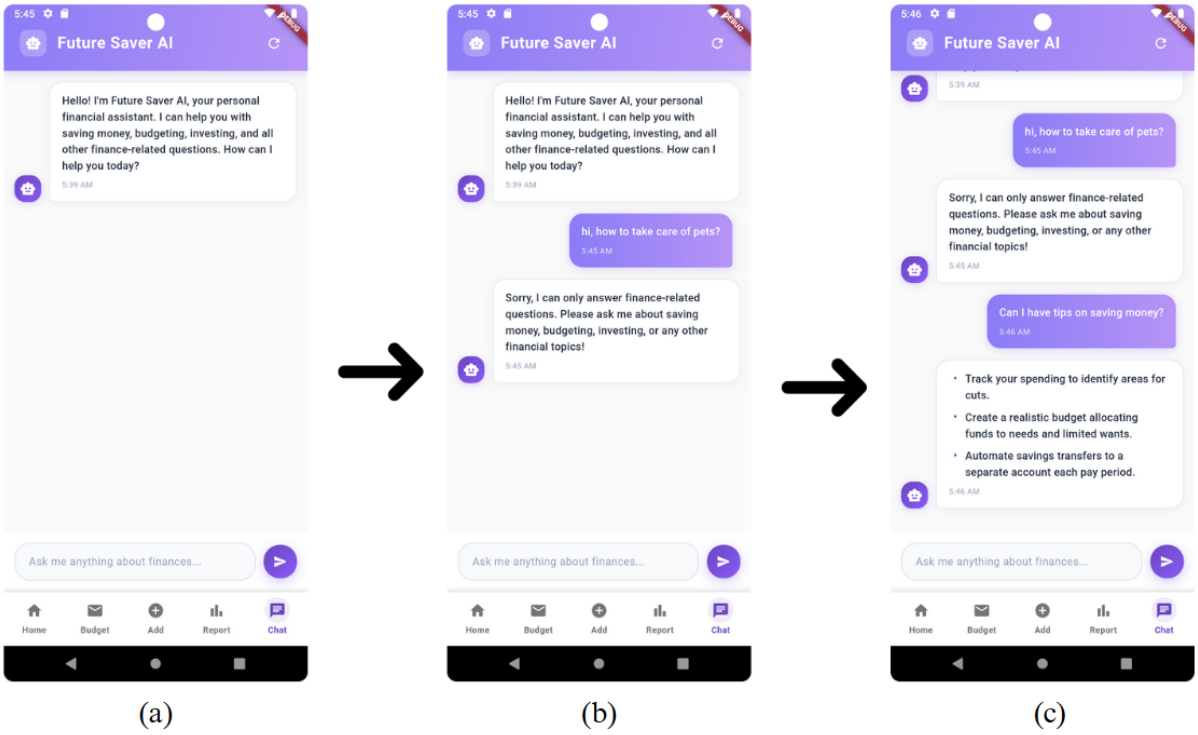
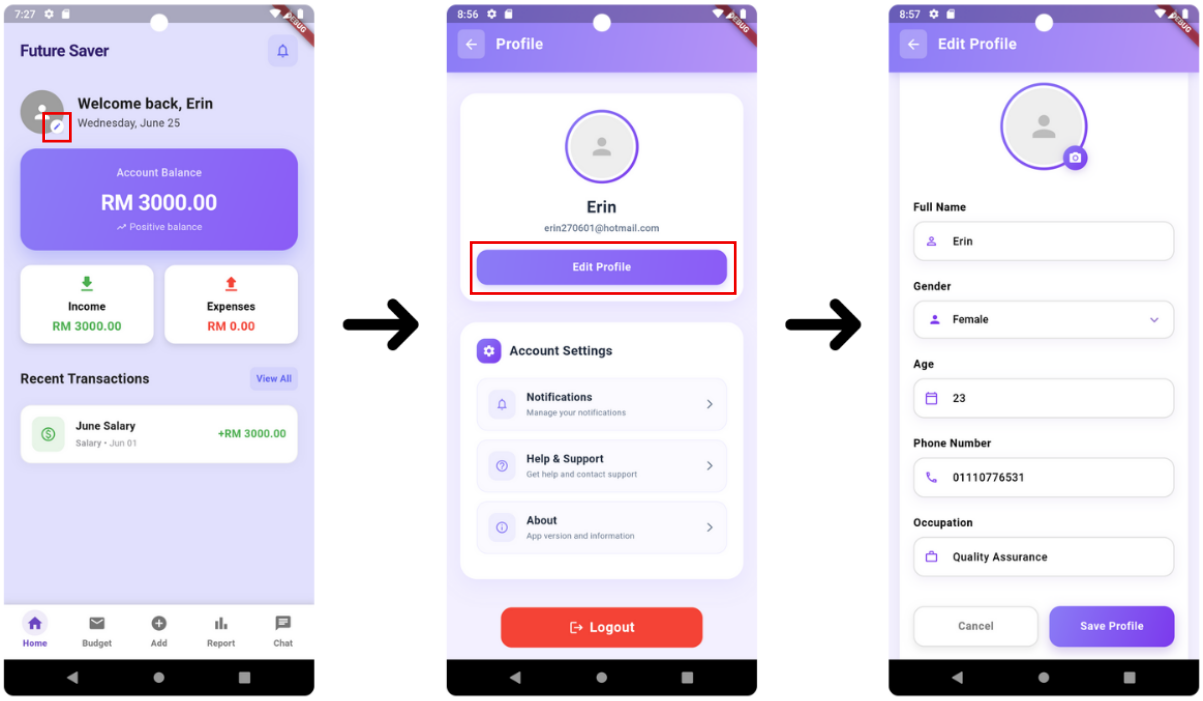


Figure 4.16: Chatbot Page for Future Saver

Figure 4.13 shows the Chatbot feature in the Future Saver app. The chatbot acts as a personal financial assistant that can answer questions related to saving money, budgeting, investing, and other finance-related topics. In view (a), the chatbot welcomes the user and prompts them to ask any financial questions. In view (b), the user asks a question that is not finance-related, and the chatbot responds by politely explaining that it can only help with finance topics. Finally, in view (c), the user asks a financial question, and the chatbot provides helpful, relevant financial tips.

4.6.11 Profile Page



(a) (b) (c)

Figure 4.17: Profile Page for Future Page

Figure 4.14 shows the Profile module in the Future Saver app. In view (a), the user can tap the profile icon on the top-right corner of the home page to navigate to the profile screen. On the profile page shown in view (b), the user's personal details such as their name and email are displayed. Users can also access options for notifications, account settings, help & support, and app information. By selecting the Edit Profile button, the user is taken to the edit screen shown in view (c), where they can update personal information such as their full name, gender, age, phone number, and occupation. The profile picture can also be updated by tapping the edit icon on the profile image at the top of the screen. Once the user is done making changes, they can tap Save Profile to update their information.

4.6.12 Notifications Page

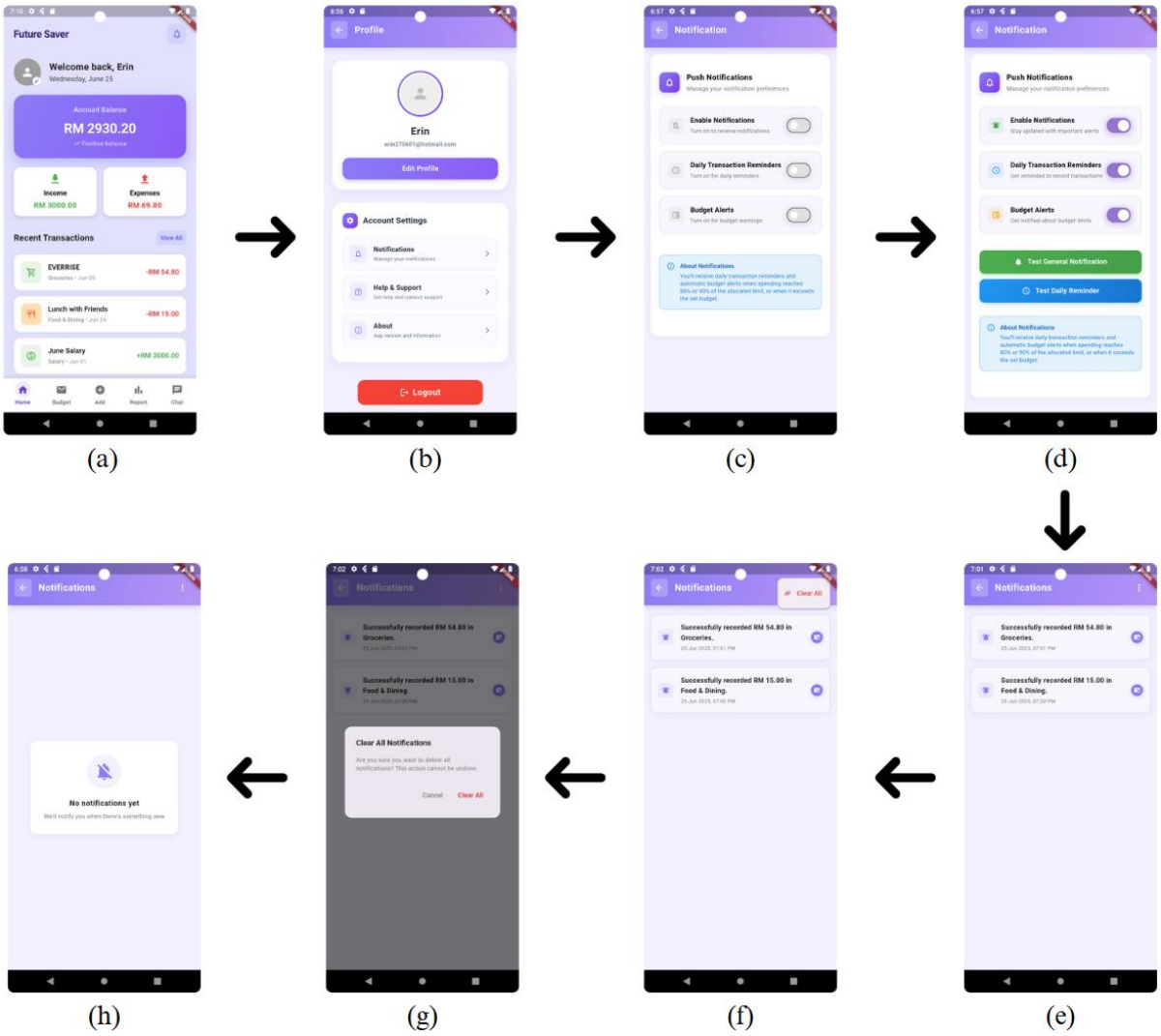


Figure 4.18: Notifications Page of Future Saver

Figure 2.12 illustrates the notification features in the Future Saver app. From the home screen (a), the user can tap the profile icon at the top left corner to navigate to the profile screen shown in view (b), where account options such as Notifications, Help & Support, and About are displayed. Upon selecting the Notifications option, the notification settings page is displayed as in view (c) which allow user to enable or disable daily transaction reminders and budget alerts. Once these options are activated, as shown in view (d), the user can also test the notification features using the provided test buttons. The user can then return to the home screen and tap the notification icon at the top right to view the notifications page shown in view (e),

where a list of recent notifications is displayed. By selecting the “Clear All” button (f), the user can remove all notifications at once. A confirmation dialog then appears as shown in view (g). Once confirmed, the screen updates to show that there are no new notifications as illustrated in view (h), as which the notification list is empty.

4.7 Summary

In conclusion, this chapter discussed the implementation phase of the Future Saver mobile application, including the setup of development tools, chosen frameworks, and database integration. The application was successfully developed using Android Studio, Flutter and Firebase. Each module was implemented in accordance with the system design, ensuring that key features such as budget tracking, transaction logging, AI-powered chatbot assistance, and receipt scanning function effectively. This phase signifies the transition from design to a fully operational application that meets the intended objectives.

CHAPTER 5: SYSTEM TESTING

5.1 Introduction

System testing is important in the development of the Smart Budget Personal Finance Tracking Mobile Application to ensure the mobile app is worked as expected and meet the user requirements identified during the initial stages of the project. It includes functional testing and non-functional testing. The purpose of the testing phase is to identify and resolve any defects before the application is released to users.

5.2 Functional Testing

Functional testing is carried out to ensure that each feature in the application works as expected and that the software performs its intended functions correctly based on defined requirements. The tests are designed based on system requirements and use case scenarios to verify whether the actual output matches the expected result for each given input.

5.2.1 Test Case for Register Functionality

Table 5.1: Test Case for Register Functionality

Test Scenario ID	TS_Register_01	**Test Case ID**	TC_Register_01
Test Title	Successful Registration	**Test Module**	User Authentication - Register Page
Test Case Description	To verify that a new user can successfully register in the app using valid input data.	**Test Priority**	High
Pre-Requisite	The Future Saver app must be installed on the user's mobile device.	**Post-Requisite**	The user is successfully registered and can log in using their email and password.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Open the Future Saver app on the mobile device.	App Icon: Future Saver	Launch the Future Saver app and displays the login page by default.	Successfully launch the Future Saver app and display the login page.	Pass	Launch successful.

2	Click the “Sign Up” button on the login page.	Tap button: "Sign Up"	Navigate to Registration Page.	Navigate to the Registration Page successfully.	Pass	Navigate successful.
3	Enter a valid name, email address, password, and confirm password in the registration form.	Registration Information: • Name: Erin • Email: erin270601@hotmail.com • Password: Erin0627wsy* • Confirm Password: Erin0627wsy*	Name, Email, Password, and Confirm Password fields are filled in successfully.	The Name, Email, Password, and Confirm Password fields are successfully filled in.	Pass	All required fields are entered successfully.

4	Tap the “Sign Up” button to complete the registration.	Tap button: “Sign Up"	The user is redirected to the Sign-Up Success page after completing the registration.	Successfully navigate to the Sign-Up Success page.	Pass	Redirect successful.

5.2.2 Test Case for Register Functionality

Table 5.2: Test Case for Register Functionality

Test Scenario ID	TS_Register_02	**Test Case ID**	TC_Register_02
Test Title	Registration Field Validation Handling	**Test Module**	User Authentication - Register Page
Test Case Description	To verify that the registration process handles both valid and invalid input correctly.	**Test Priority**	High
Pre-Requisite	User has installed the Future Saver app on their mobile device.	**Post-Requisite**	User is either registered successfully or receives proper error messages.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	On the registration page, click the “Sign Up” button without filling in any of the required fields.	Registration Information: • Name: N/A • Email: N/A • Password: N/A	The validation error messages should be displayed.	The validation error messages are displayed.	Pass	Error messages are displayed.

		• Confirm Password: N/A				
2	Fill in some required fields while leaving others empty.	Registration Information: • Name: Erin • Email: N/A • Password: N/A • Confirm Password: N/A	Validation error messages for the Email, Password, and Confirm Password fields should appear.	The app shows validation errors for the Email, Password, and Confirm Password fields.	Pass	
3	Enter an invalid email format in the Email field.	Registration Information: • Name: Erin • Email: erin.com	The error message of “Enter a valid email address” should be displayed.	The error message “Enter a valid email address” is displayed.	Pass	

		• Password: Erin0627wsy* • Confirm Password: Erin0627wsy*				
4	Fill in the registration form with valid name and email, and a weak password.	Registration Information: • Name: Erin • Email: erin270601@hotmail.com • Password: Erin123* • Confirm Password: Erin123*	A helpful message, such as "Suggest a strong password," should be display when a weak password is entered by the user.	The app successfully displays a prompt such as "Suggest a strong password" when a weak password is entered.	Pass	Helpful message is displayed.

5	Enter an email address that is already registered in the app.	Registration Information: • Name: Erin • Email: erin01@gmail.com • Password: Erin0627wsy* • Confirm Password: Erin0627wsy*	The error message of "This email is already registered. Please log in." should be displayed.	The error message "This email is already registered. Please log in." is displayed.	Pass	
6	Enter valid inputs for the Name, Email, Password, and Confirm Password fields.	Registration Information: • Name: Erin	Name, Email, Password, and Confirm Password fields are entered	All fields are entered correctly, and the user account is created successfully.	Pass	

		• Email: erin270601@hotmail.com • Password: Erin0627wsy* • Confirm Password: Erin0627wsy*	successfully, and the user account is created.			

5.2.3 Test Case for Login Functionality

Table 5.3: Test Case for Login Functionality

Test Scenario ID	TS_Login_02	**Test Case ID**	TC_Login_02
Test Title	Successful login	**Test Module**	User Authentication - Login Page
Test Case Description	To verify that a user can successfully log in to the app using valid input data.	**Test Priority**	High
Pre-Requisite	User is already registered in the Future Saver app.	**Post-Requisite**	User is logged in and redirected to the home page.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Open Future Saver app.	App Icon: Future Saver	Launch the Future Saver app and displays the login page by default.	Successfully launch the Future Saver app and display the login page.	Pass	

2	Enter valid registered email and password.	• Email: erin270601@hotmail.co • Password: Erin0627wsy*	All inputs are valid, and no validation errors occur.	All inputs are valid, and no validation errors are shown.	Pass	
3	Click on the “Login” button.	Tap button: “Login”	The user should be redirected to the Home Page.	The user is successfully redirected to the Home Page.	Pass	

5.2.4 Test Case for Login Functionality

Table 5.4: Test Case for Login Functionality

Test Scenario ID	TS_Login_02	**Test Case ID**	TC_Login_02
Test Title	Login Field Validation – Invalid Input Handling	**Test Module**	User Authentication - Login Page
Test Case Description	To verify that the login process handles invalid input correctly.	**Test Priority**	High
Pre-Requisite	User is already registered in the Future Saver app.	**Post-Requisite**	User is either login successfully or receives proper error messages.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	On the login page, click the "Login" button without filling in any of the required fields.	• Email: N/A • Password: N/A	The validation error messages should be displayed.	Both email and password fields return validation errors due to invalid input.	Pass	Error message is displayed.

2	Fill in some required fields while leaving others empty.	• Email: erin270601@hotmail.com • Password: N/A	The password field should display the validation error message: “Please enter your password.”	A validation message “Please enter your password.” is shown in the password field when left empty.	Pass	
3	Enter an invalid email format or an email that is not registered in the app.	**Test Data Set 1:** • Email: erin.com • Password: Erin0627wsy* **Test Data Set 2:** • Email: erin270601@hotmail.com • Password: Erin0627wsy*	The error message of “Incorrect email or password. Please try again.” should be displayed.	The error message of “Incorrect email or password. Please try again.” is displayed.	Pass	

4	Enter wrong password.	• Email: erin270601@hotmail.com • Password: Erin	The error message of “Incorrect email or password. Please try again.” should be displayed.	The error message of “Incorrect email or password. Please try again.” is displayed.	Pass	

5.2.5 Test Case for Reset Password Functionality

Table 5.5: Test Case for Reset Password Functionality

Test Scenario ID	TS_ResetPassword_01	**Test Case ID**	TC_ResetPassword_01
Test Title	Verify Password Reset with Email Handling and Login	**Test Module**	User Authentication – Reset Password Page
Test Case Description	To verify that a user can reset their password with a valid email and log in using the new password or receive an error if the email is invalid.	**Test Priority**	High
Pre-Requisite	The user must have a registered account.	**Post-Requisite**	A password reset email is sent successfully.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	On the Login Page, click the "Forgot Password" button.	Tap button: “Forgot Password”	Navigate to the Forgot Password page.	Navigate to the Forgot Password page successfully.	Pass	

2	Leave the email field empty.	• Email: N/A	The app should display an error message: “Please enter your email”.	An error message of “Please enter your email” is displayed.	Pass	
3	Enter an invalid email format.	• Email: erin.com	The app should display an error message: “Enter a valid email format.”	An error message of “Enter a valid email format” is displayed.	Pass	
4	Enter a valid email address in the email field.	• Email: erin270601@hotmail.com	A valid email address is entered successfully.	The email field is filled in successfully.	Pass	
5	Click the “Sent Reset Email” button.	Tap button: “Sent Reset Email”	Redirect to the Reset Password Success Page.	Successfully navigate to the Reset Password Success page.	Pass	

6	Check the email inbox for the password reset email.	• Email: erin270601@hotmail.com	User should receive the password reset email in their inbox.	The password reset email is received successfully.	Pass	
7	Click the password reset link provided in the email.	Link of Reset Password	Go to the Reset Password Page from the link provided to reset the password	User is redirected to the Reset Password page successfully.	Pass	
8	Enter the new password.	• New password: Erin01wsy$	Enter the new password.	Input the new password successfully.	Pass	
9	Proceed to log in with the updated password.	• Email: erin270601@hotmail.com • Password: Erin01wsy$	The user should be able to log in and be redirected to the Home Page.	The user log in successfully and is redirected to the Home Page.	Pass	

5.2.6 Test Case for Budget Functionality

Table 5.6: Test Case for Budget Functionality

Test Scenario ID	TS_Budget_01	**Test Case ID**	TC_Budget_01
Test Title	Verify Budget Module Operations	**Test Module**	Budget Page
Test Case Description	To test that the Budget Module correctly supports all the core operations (add, display, edit, delete) and updates budget data accurately after expense transactions.	**Test Priority**	High
Pre-Requisite	The user must be logged into the app.	**Post-Requisite**	The new budget record is successfully added and reflected in both the UI and Firebase.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Log in to the user account.	• Email: erin270601@hotmail.com	The user should be able to log in and be	The user log in successfully and is	Pass	Launch successful.

		• Password: Erin01wsy$	redirected to the Home Page.	redirected to the Home Page.		
2	Click the "Budget" button to proceed to the Budget page.	Tap button: “Budget” button	Navigate to the Budget page.	Navigate to the Budget page successfully.	Pass	Navigate successful.
3	Add New Category to Budget.	• Category: Rent • Amount: 600	A success message is displayed confirming that the budget is added successfully and the newly added budget record appears on the budget page with the correct category and amount.	The success message is showing the "Budget data saved successfully!" and the new budget record display correctly.	Pass	

4	View Available Budget After Adding, Editing, and Deleting a Transaction	Test Data Set 2 from the manual entry function.	The available budget amount should be correctly updated after adding, editing or deleting the expense record.	The available budget amount is updated correctly after adding, editing or deleting the expense record.	Pass	
5	Edit an existing budget entry.	Changes the amount of Rent Category from RM600 to RM800.	The app displays a success message, and the budget data is updated successfully and displayed correctly on the budget page.	The app displays a success message. The updated budget data is saved and accurately reflected on the budget page without any errors.	Pass	
6.	Delete existing budget data.	Delete the Rent Category Budget	The app displays a success message, and the budget page	The app displays a success message, and the deleted budget	Pass	

			updates correctly to reflect the removal of the deleted budget entry.	entry is removed and no longer appears on the budget page.		

5.2.7 Test Case for Budget Functionality

Table 5.7: Test Case for Budget Functionality

Test Scenario ID	TS_Budget_02	**Test Case ID**	TC_Budget_02
Test Title	Budget Entry Validations and Error Handling	**Test Module**	Budget Page
Test Case Description	To verify that the budget entry process correctly handles both valid inputs and errors.	**Test Priority**	High
Pre-Requisite	The user must be logged into the app.	**Post-Requisite**	The user can either successfully add a new budget record or receive appropriate error messages.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	On the budget page, add same category of	**First Budget Record:**	The app should display the error message: "Cannot add the same	The app successfully displays the error message: "Cannot add	Pass	

	budget in the same month.	• Category: Shopping • Amount: 800 **Second Budget Record:** • Category: Shopping • Amount: 1000	category budget again!".	the same category budget again!"		
2	Enter the amount using the test data provided.	Amount: N/A	The app should show error message "Amount cannot be empty".	The app shows an error message of “Amount cannot be empty.”.	Pass	
		Amount: 0 (zero)	The app should show error message	The app shows an error message of “Amount cannot be zero”.	Pass	

			“Amount cannot be zero”.			
		Amount: abc	The app should show an error message "Please enter a valid amount".	The app shows an error message "Please enter a valid amount".	Pass	
		Amount: -100	The app should show an error message of “Amount cannot be negative”.	The app shows an error message of “Amount cannot be negative”.	Pass	
		Amount: 200	The amount input filled should be filled in successfully.	The amount input filled is filled in successfully.	Pass	

5.2.8 Test Case for Manually Entry Functionality

Table 5.8: Test Case for Manually Entry Functionality

Test Scenario ID	TS_ManuallyEntry_01	**Test Case ID**	TC_ManuallyEntry_01
Test Title	Test Manual Entry, Viewing, Editing, and Deletion of Income and Expenses	**Test Module**	Transaction – Manually Entry Page
Test Case Description	To verify that the users can manually record, view, edit, and delete income and expenses correctly.	**Test Priority**	High
Pre-Requisite	The user must be logged into the app.	**Post-Requisite**	The new transaction record is successfully saved and reflected in both the transaction history UI and Firebase.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**

1	Log in to the user account.	• Email: erin270601@hotmail.com • Password: Erin01wsy$	The user should be able to log in and be redirected to the Home Page.	The user log in successfully and is redirected to the Home Page.	Pass	
2	Click the "+" button to proceed to the Transaction Selection page.	Tap button: “+” button	Navigate to the Transaction Selection page.	Navigate to the transaction selection page successfully.	Pass	
3	Select “Manually Input” on the Transaction Selection page.	Select transaction: “Manually Input”	Navigates to Manually Input screen.	Navigates to Manually Input page successfully.	Pass	
4	Fill in all required fields with valid data	**Test Data Set 1:** • Date: 03/06/2025	All required fields have been filled in.	All required fields are filled in correctly and successfully.	Pass	

	according to the test inputs.	• Type of transaction: Income • Category: Salary • Name: June Salary • Amount (RM): 3000 • Remarks: Bonus included **Test Data Set 2:** • Date: 03/06/2025 • Type of transaction: Expense				

		• Category: Food & Dining • Name: Lunch • Amount (RM): 15.50 • Remarks: N/A				
5	Click on the “Save” button.	Tap button: “Save” button	Transaction data should be stored in Firebase and accurately displayed in the UI, sorted by date in the transaction history.	The transaction data is successfully saved in Firebase and reflected in the UI, with correct sorting by date in the transaction history.	Pass	
6	Click the “See All” button to navigate to	Tap button: “See All”	Navigate to the Transaction History page.	Navigate to the Transaction History page successfully.	Pass	

	the transaction history page.					
7	Click on a transaction record to view its details.	The newly added expense record (test data set 2).	The new transaction is shown in the transaction history list and can be opened to view its full details.	The newly added transaction appears in the transaction history list and the full transaction details are shown correctly.	Pass	
8	Edit the transaction record details.	Change the item name from Lunch to Lunch with friends	The transaction should be successfully updated with the new item name and reflected in both the UI and firebase.	The item name of the transaction is updated successfully and reflected in both UI and firebase.	Pass	

9	Delete the transaction record.	The newly added expense record (test data set 2).	The transaction should be successfully removed from the list, and the total income or expense value on the home page should be updated accordingly. The available amount should also be updated to reflect the change.	The transaction is removed from the list, and the total income or expense value on the home page is updated accordingly. The available amount is also updated.	Pass	

5.2.9 Test Case for Manually Entry Functionality

Table 5.9: Test Case for Manually Entry Functionality

Test Scenario ID	TS_ ManuallyEntry_02	**Test Case ID**	TC_ ManuallyEntry_02
Test Title	Manually Entry Field Validation Handling	**Test Module**	Transaction – Manually Entry Page
Test Case Description	To verify that the manually entry process handles both valid and invalid input correctly.	**Test Priority**	High
Pre-Requisite	The user must be logged into the app.	**Post-Requisite**	The user can either successfully add a new transaction manually or receive appropriate error messages.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	On the Manually Entry page, select the date based on the test data.	Date: N/A	The app should show an error message: "Please enter the date".	An error message of “Please enter the date” is displayed.	Pass	

		Date: 06/22/2025	The app should show an error message: "Please follow the format: DD/MM/YYYY".	An "Invalid format" error message is displayed.	Pass	The error message is different, but the app still handles the input correctly.
		Date: 01/01/1999	The app should show an error message: "Please select a date between 01/01/2000 and 31/12/2099".	The "Out of range" error message is displayed.	Pass	The error message is different, but the app still handles the input correctly.
		Date: 01/01/2100	The app should show an error message: "Please select a date	The "Out of range" error message is displayed.	Pass	The error message is different, but the app still

			between 01/01/2000 and 31/12/2099".			handles the input correctly.
		Date: 22/06/2025	The date input field should be filled in successfully.	The date input field is filled in successfully.	Pass	
2	Select the category only after the transaction type has been chosen.	Category: N/A	An inline error message: “Please select the transaction type first before choosing a category.” Should be displayed.	The inline error message “Please select the transaction type first before choosing a category” is displayed when the user attempts to select a category without first selecting a transaction type.	Pass	

		Type of Transaction: Expense	The category should be selectable, and no inline error message should be displayed.	The category is selectable, and no inline error message is displayed.	Pass	
3	Enter the name using the test data provided.	Name: N/A	The app should display the error message: “Please enter an item name”.	An error message of "Please enter a name" is shown.	Pass	
		Name: Bubble Tea	The spaces should be trimmed, and the name should be saved correctly in the transaction details.	Spaces are trimmed, and the name is saved correctly in the transaction details.	Pass	
		Name: This item name is too long and should be rejected	An error message of "Name cannot exceed	An error message of "Name cannot exceed	Pass	

		by the system for exceeding 100 characters. (137 characters)	100 characters" should be displayed	100 characters" is shown.		
		Name: Teatime with Friends (20 characters)	The name input field should be filled in successfully.	The name input field is filled in successfully.	Pass	
4	Input the amount based on the test data.	Amount (RM): N/A	An error message of “Please enter an amount” should be shown.	An error message of “Please enter an amount” is shown.	Pass	
		Amount (RM): 1.599	The app should display the error message: “Enter a valid amount: positive numbers only,	An error message of "Enter a valid amount: positive numbers only, with a maximum of two	Pass	

			with a maximum of two decimal places”.	decimal places" is displayed.		
		Amount (RM): 20.5.0	The app should display the error message: “Enter a valid amount: positive numbers only, with a maximum of two decimal places”.	An error message of "Enter a valid amount: positive numbers only, with a maximum of two decimal places" is displayed.	Pass	
		Amount (RM): 1,000.00	The app should display an error message: “Commas are not allowed in the amount”.	An error message of “Commas are not allowed in the amount” is shown.	Pass	
		Amount: -100	Error message of "Negative amount is	An error message of “Negative amount is not allowed” is shown.	Pass	

			not allowed" should be displayed.			
		Amount (RM): “abc@12”	The app should display the error message: “Enter a valid amount: positive numbers only, with a maximum of two decimal places”.	An error message of "Enter a valid amount: positive numbers only, with a maximum of two decimal places" is displayed.	Pass	
		Amount (RM): 99999999.99	An error message of “Amount cannot exceed 7 digits” should be shown.	An error message of “Amount cannot exceed 7 digits” is shown.	Pass	
		Amount (RM): 50.80	The amount input field should be filled in successfully.	The amount input field is filled in successfully.	Pass	

5	Enter the remarks based on the test data.	Remarks: N/A	The remarks input field should be optional for the user to fill in.	The remarks input field is optional for the user to fill in.	Pass	
		Remarks: Starbucks	The remarks input field should be filled in successfully.	The remarks input field is filled in successfully.	Pass	
		Remarks: This is a very long remark intended to test the system's ability to handle excessive input beyond the expected limit. The user continues to write unnecessary details just to see	An error message of "Remarks cannot exceed 255 characters" should be displayed.	An error message of "Remarks cannot exceed 255 characters" is displayed.	Pass	

		whether the application can catch this overflow or not, which is crucial for maintaining data integrity and performance. (298 characters)				

5.2.10 Test Case for Receipt Scanning Functionality

Table 5.10: Test Case for Receipt Scanning Functionality

Test Scenario ID	TS_ReceiptScanning_01	**Test Case ID**	TC_ ReceiptScanning_01
Test Title	Track Expenses with Receipt Scanning	**Test Module**	Transaction – Receipt Scanning
Test Case Description	To verify that the users can scan and record receipt expenses correctly.	**Test Priority**	High
Pre-Requisite	The user must be logged into the app.	**Post-Requisite**	The transaction record is successfully saved and reflected in both the transaction history UI and Firebase.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Log in to the user account.	• Email: erin270601@hotmail.com	The user should be able to log in and be redirected to the Home Page.	The user log in successfully and is redirected to Home Page.	Pass	

		• Password: Erin01wsy$				
2	Click the "+" button to proceed to the Transaction Selection page	Tap button: "+" button	Navigate to the Transaction Selection page.	Navigate to the Transaction Selection page successfully.	Pass	
3	Select "Receipt Scanning" on the Transaction Selection page.	Select transaction: "Receipt Scanning"	Navigate to Receipt Scanning page.	Navigate to Receipt Scanning page successfully.	Pass	
4	Tap the camera icon to capture a picture or select a receipt image from the gallery.	• Receipt: (Grocery Receipt Image) • Date: 04/06/2025	The pre-filled fields match the data from the scanned receipt accurately.	The data is successfully extracted from the receipt, and all pre-filled fields such as date, name, and amount	Pass	

		• Name: SuperMart • Amount (RM): 50		matched the receipt information correctly.		
5	Then, click the “Next” button to proceed to the transaction detail page and fill in the remaining fields.	• Tap button: “Next” button • Category: Groceries	All required fields have been filled in.	All required fields are filled in correctly and successfully.	Pass	
6	Click on the “Save” button	Tap button: “Save” button	Transaction data should be saved in Firebase and presented correctly in the UI, arranged by date in the transaction history.	The transaction data is successfully saved in Firebase and displayed in the UI, correctly sorted by date in the transaction history.	Pass	

5.2.11 Test Case for Receipt Scanning Functionality

Table 5.11: Test Case for Receipt Scanning Functionality

Test Scenario ID	TS_ReceiptScanning_02	**Test Case ID**	TC_ ReceiptScanning_02
Test Title	Receipt Scanning Handling for Invalid or Blurred Images	**Test Module**	Transaction – Receipt Scanning
Test Case Description	To test the app's handling of Receipt Scanning failures for invalid or blurred receipts.	**Test Priority**	High
Pre-Requisite	The user must be logged into the app.	**Post-Requisite**	The user can either successfully add a transaction via receipt scanning or receive appropriate error messages.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Upload an invalid receipt image.	Receipt: Invalid receipt	An error message of “Invalid receipt image” should be displayed,	The error message “Invalid receipt image”	Pass	

			and all fields should remain empty.	is displayed, and all fields are left empty.		
2	Upload a blurred receipt image.	Receipt: Blurred grocery receipt	An error message “The receipt may be blurry. Please try again” should be displayed, and some of the fields should remain empty.	The error message “The receipt may be blurry. Please try again” is displayed, and some fields are left empty.	Pass	

5.2.12 Test Case for Spending by Category Report Functionality

Table 5.12: Test Case for Spending by Category Report Functionality

Test Scenario ID	TS_SpendingbyCategory_01	**Test Case ID**	TC_SpendingbyCategory_01
Test Title	Verify Expense Category Data Display in Pie Chart Based on Date Range Filter	**Test Module**	Report – Spending by Category
Test Case Description	To verify that the expense category data is correctly displayed in the pie chart according to the selected date range filter.	**Test Priority**	High
Pre-Requisite	The user must be logged into the app.	**Post-Requisite**	The expense category is correctly displayed in the pie chart in the spending overview section after applying the date range filter.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**

1	Log in to the user account.	• Email: erin270601@hotmail.com • Password: Erin01wsy$	The user should be able to log in and be redirected to the Home page.	The users log in successfully and is redirected to the Home page.	Pass	
2	Click the “Report” button to proceed to the Report Selection page.	Tap button: “Report” button	Navigate to the Report Selection page.	Navigate to the Report Selection page successfully.	Pass	
3	Select “Spending by Category” report.	Select transaction: “Spending by Category”	Navigate to the Spending by Category page.	Navigate to the Spending by Category page successfully.	Pass	
4	Select a date range to filter the data.	• From: 01 Jun 2025 • To: 30 Jun 2025	The date range filter should be successfully applied, and the data should be updated to	The date range filter successfully applies, and the data is updated to reflect the selected date range.	Pass	

			reflect the selected date range.			
5	Check the expense data (from test data set 2 via the manual entry function) in the spending overview and category breakdown section after the expense is added.	Test Data Set 2 from the manual entry function	The percentage for the Food & Dining category should be correctly updated in the spending overview section, and the detailed information for that category should be accurately displayed in the category breakdown section.	The pie chart updates the percentage for the Food & Dining category of the total expense, and the detailed information is accurately updated and displayed after adding the new transaction records.	Pass	

6	Check the expense data (from test data set 2 via the manual entry function) in the spending overview and category breakdown section after the expense is edited or deleted.	Test Data Set 2 from the manual entry function	The percentage for the Food & Dining category should be correctly updated in the spending overview section, and the detailed information for that category should be accurately displayed in the category breakdown section.	The pie chart updates the percentage for the Food & Dining category of the total expense, and the detailed information is accurately updated and displayed after editing or deleting transaction records.	Pass	

5.2.13 Test Case for Income vs Spending Report Functionality

Table 5.13: Test Case for Income vs Spending Report Functionality

Test Scenario ID	TS_IncomeVsSpending_01	**Test Case ID**	TC_ IncomeVsSpending _01
Test Title	Verify Income and Expense Category Data Display in Bar Chart Based on Date Range Filter	**Test Module**	Report – Income vs Spending
Test Case Description	To verify that the income and expense category data is correctly displayed in the bar chart according to the selected date range filter.	**Test Priority**	High
Pre-Requisite	The user must be logged into the app.	**Post-Requisite**	The income and expense category is correctly displayed in the bar chart in the financial overview section after applying the date range filter.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**

1	Log in to the user account.	• Email: erin270601@hotmail.com • Password: Erin01wsy$	The user should be able to log in and be redirected to the Home page.	The users log in successfully and is redirected to the Home page.	Pass	
2	Click the “Report” button to proceed to the Report Selection page.	Tap button: “Report” button	Navigate to the Report Selection page.	Navigate to the Report Selection page successfully.	Pass	
3	Select “Spending by Category” report.	Select transaction: “Income vs Spending”	Navigate to the Income vs Spending page.	Navigate to the Income vs Spending page successfully.	Pass	
4	Select a date range to filter the data.	• From: 01 Jun 2025 • To: 30 Jun 2025	The date range filter should be successfully applied, and the data should be updated to	The date range filter successfully applies, and the data is updated to reflect the selected date range.	Pass	

			reflect the selected date range.			
5	Check the income and expense data (from test data sets 1 & 2 via the manual entry function) in the financial overview and financial breakdown section after the transaction is added.	Test Data Set 1 and 2 from the manual entry function	The bar chart should correctly update the totals for both income and expenses after the transactions are added. The net amount should also be accurately recalculated and displayed.	The bar chart displays the updated totals for both income and expenses accurately. The net amount is also correctly recalculated and updated after the income and expense transactions are added.	Pass	
6	Check the income and expense data (from test data sets 1 & 2 via the manual entry function) in the financial	Test Data Set 1 and 2 from the manual entry function	The bar chart should correctly update the totals for both income and expenses after the transactions are edited	The bar chart displays the updated totals for both income and expenses accurately. The net amount is also	Pass	

	overview and financial breakdown section after the transaction is edited or deleted.		or deleted. The net amount should also be accurately recalculated and displayed.	correctly recalculated and updated after editing or deleting the income and expense transactions.		

5.2.14 Test Case for Chatbot Functionality

Table 5.14: Test Case for Chatbot Functionality

Test Scenario ID	TS_Chatbot_01	**Test Case ID**	TC_Chatbot_01
Test Title	Verify Chatbot Accuracy for Finance Tasks	**Test Module**	Chatbot
Test Case Description	To ensure that the chatbot functions correctly for finance-related tasks and provides relevant responses.	**Test Priority**	High
Pre-Requisite	The user must be logged into the app.	**Post-Requisite**	The

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Log in to the user account.	• Email: erin270601@hotmail.com • Password: Erin01wsy$	The user should be able to log in and be redirected to the Home page.	The users log in successfully and is redirected to the Home page.	Pass	

2	Click the “Chatbot” button to proceed to the Chatbot page.	Tap button: “Chatbot” button	Navigate to the Chatbot page.	Navigate to the Chatbot page successfully.	Pass	
3	Ask for recent transactions, assigned amount, spending, and remaining balance for a specific category.	• Query: "What is my last transaction?" • Query: "How much I have spent for Food & Dining Category?" • Query: "How much do I have left for Groceries this month?"	The chatbot provides accurate transaction details, as well as the assigned, spent, and remaining amounts based on the requested category.	The chatbot correctly displayed the transaction details, assigned, spending and remaining amount for the query.	Fail	

4	Inquire about budgeting tips.	Query: "Can you give me tips to save money?"	The chatbot should provide useful financial advice such as "Track your expenses and set savings goals."	The chatbot successfully answers user queries about budgeting tips.	Pass	
5	Ask non-financial related questions.	Query: "What is the weather today?"	The chatbot should reply with "Sorry, I can only answer finance-related questions. Please ask me about saving money, budgeting, investing, or any other financial topics ".	Chatbot handled the query correctly and reply with "Sorry, I can only answer finance-related questions. Please ask me about saving money, budgeting, investing, or any other financial topics".	Pass	

5.2.15 Test Case for Profile Functionality

Table 5.15: Test Case for Profile Functionality

Test Scenario ID	TS_Profile_01	**Test Case ID**	TC_Profile_01
Test Title	Verify Profile Display, Edit, and Logout Redirection	**Test Module**	Profile
Test Case Description	To verify that the user data is displayed correctly, updates after editing, and the user is redirected to the login page after logging out.	**Test Priority**	High
Pre-Requisite	The user must be logged into the app.	**Post-Requisite**	The user data is correctly displayed on the profile page and successfully updated after editing. After logging out, the user is redirected to the login page.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**

1	Log in to the user account.	• Email: erin270601@hotmail.com • Password: Erin01wsy$	The user should be able to log in and be redirected to the Home page.	The users log in successfully and is redirected to the Home page.	Pass	
2	Click the “Edit” icon button at the top right of the image in the home page to proceed to the Chatbot page.	Tap button: “Edit” icon button	Navigate to the Profile page.	Navigate to the Profile page successfully.	Pass	
3	Check user profile information.	• Name: Erin • Email: erin270601@hotmail.com	The profile information should be displayed correctly.	The profile information is displayed correctly.	Pass	

4	Click “Edit Profile” button to update Profile.	Tap button: “Edit Profile” button	Navigate to the Edit Profile page.	Navigate to the Edit Profile page successfully.	Pass	
5	Edit the gender, age, phone number and occupation.	• Gender: Female • Age: 23 • Phone Number: 01110776531 • Occupation: Quality Assurance	Able to edit and update the profile information.	Successfully edit and update the profile information in the designated text fields.	Pass	
6	Click “Save Profile” button.	Tap button: “Save Profile” button	All the changes should be saved and updated.	All the changes are saved and updated on Profile page and firebase.	Pass	

7	Click “Logout” button.	Tap button: “Logout” button	User can log out and is redirected to the login page.	The user is logged out and redirected to the login page.	Pass	

5.2.16 Test Case for Profile Functionality

Table 5.16: Test Case for Profile Functionality

Test Scenario ID	TS_Profile_02	**Test Case ID**	TC_Profile_02
Test Title	Profile Entry Field Validation Handling	**Test Module**	Profile
Test Case Description	To verify that the profile entry process handles both valid and invalid input correctly	**Test Priority**	High
Pre-Requisite	The user must be logged into the app.	**Post-Requisite**	The user can either successfully update the profile page or receive appropriate error messages.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	On the Profile Page, select the gender.	Gender: N/A	The gender input field should be optional.	The gender input field is optional.	Pass	

		Gender: Female	The gender input field should be selected successfully.	The gender input field is selected successfully.	Pass	
2	Enter the age.	Age: abc	An error message of “Age must be a valid number” should be shown.	The “Age must be a valid number” error message is shown.	Pass	
		Age: 125	An error message of “Age must be between 1 and 120” should be shown.	The “Age must be between 1 and 120” error message is shown.	Pass	
		Age: N/A	The age input field should be optional.	The age input field is optional.	Pass	
		Age: 23	The age input field should be filled in successfully.	The age input field is filled in successfully.	Pass	

3	Enter the phone number.	Phone number: abc123	An error message of “Phone number must contain only digits” should be displayed.	The “Phone number must contain only digits” error message is displayed.	Pass	
		Phone number: 0111077	An error message of “Phone number must be at least 10 digits” should be displayed	The “Phone number must be at least 10 digits” error message is displayed.	Pass	
		Phone number: 0111077653132	An error message of “Phone number cannot exceed 11 digits” should be displayed.	The “Phone number cannot exceed 11 digits” error message is displayed.	Pass	
		Phone number: N/A	The phone number input field should be optional.	The phone number input field is optional.	Pass	

		Phone number: 01110776531	The phone number input field should be filled in successfully.	The phone number input field is filled in successfully.	Pass	
4	Fill in the occupation.	Occupation: N/A	The occupation input field should be optional.	The occupation input field is optional.	Pass	
		Occupation: Quality Assurance	The occupation input field should be filled in successfully.	The occupation input field is filled in successfully.	Pass	
		Occupation: Senior Software Engineer in Artificial Intelligence and Machine Learning Technologies. (86 characters)	An error message of “Occupation cannot exceed 50 characters” should be displayed.	The “Occupation cannot exceed 50 characters” error message is displayed	Pass	

5.3 Non-Functional Testing

Non-functional testing focuses on evaluating the performance, usability, security, and reliability of the system. It ensures that the application meets the required standards and performs effectively under various conditions, not just for its core features.

5.3.1 Performance Testing

Performance testing measures how well an application performs under different conditions. It checks how the system handles load, how fast it responds, and how it behaves under stress, ensuring it works smoothly even when many users are accessing it at once.

Table 5.17: Test Case of Performance Testing

Test Scenario ID	TS_Performance_01	**Test Case ID**	TC_Performance_01
Test Title	System Performance Under Load	**Test Module**	Overall System
Test Case Description	To test Performance and Responsiveness of Future Saver App Under Varying Load Conditions	**Test Priority**	High
Pre-Requisite	• Test environment set up with production-like data. • Application is running and stable.	**Post-Requisite**	Performance test report generated with analysis of system responsiveness and behaviour under load.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**
1	Load Testing	• Number of concurrent users: 10 • Simulate adding, editing or loading budget or transactions	The application should handle the expected load without significant delays or crashes. Budget and transaction data should be accurately recorded.	The system handles the load and responses within acceptable response time which less than 3 seconds and no crashes or significant delays occurs. Average CPU and memory usage	Pass	

		data with different categories		stay within acceptable limits.		
2	Stress Testing	• Number of concurrent users: 50 • Simulate adding, editing or loading budget or transactions data.	The system should maintain performance until reaching its limit. Upon exceeding capacity, the system returns error messages or gracefully fails. Errors must be properly logged. The system should attempt to auto-restart if it crashes and recovers to normal performance after testing.	The system's response time increase beyond 5 seconds under stress, and errors or crashes which are acceptable past its capacity. Error messages and logs is generated as expected. After the test, the system is fully recovered and return to stable operation with no data loss.	Pass	
3	Response Time Testing	• Add a transaction • Edit a budget entry	Each action should complete quickly and return data without noticeable delays. User	Response time for each action is within acceptable limits which is under 2–3 seconds.	Pass	

		• Load the budget overview, transaction history, and reports	interface should always remain responsive.	No timeouts, UI freezes, or errors observed during the operations.		
4	Scalability	• Increase the number of concurrent users from 10 to 50 • Simulate a high volume of transactions and report generation during each increment	The application should scale gracefully with increased load. Performance degradation must be minimal and system resources should scale appropriately. The system must not crash under peak load.	Response times and throughput remain acceptable as user load grows. The system can accommodate increasing workloads without errors, crashes, or unacceptable slowdowns.	Pass	
5	Endurance Testing	• Number of concurrent users: 10 • Simulate continuous	The application should remain stable and responsive throughout the entire test. No gradual performance	Response times remain consistent and within acceptable limits during the full test period. No crashes, errors, or	Pass	

		usage of the application (e.g. viewing, adding, editing budget or transactions) over an extended period of 8–24 hours	degradation or resource leaks (e.g. memory, CPU, database connections). Data integrity must be preserved.	abnormal resource consumption. Post-test performance returns to baseline with no manual intervention.		
6	Database Performance Testing	• Execute common database-intensive operations under varying loads (e.g. insert transactions, query budget reports, update budgets).	The database must process queries and updates efficiently without significant delays. Indexes and queries should be optimized for smooth data retrieval and updates.	Query response times remain within acceptable thresholds which is less than 2 seconds for typical queries. Insert or update operations scale without timeouts or errors. No data loss or integrity issues occur under heavy read or write loads.	Pass	

5.3.2 Usability Testing

Usability testing evaluates how easily, and efficiently real users can navigate and complete tasks in the application. It ensures the interface is intuitive, accessible, and provides a smooth user experience.

Table 5.18: Test Case for Usability Testing

Test Scenario ID	TS_Usability_01	**Test Case ID**	TC_Usability_01
Test Title	User-Friendly Interface Test	**Test Module**	Overall System
Test Case Description	To verify that the application's interface is intuitive and easy to navigate which allow users to access all primary features quickly with minimal effort and without confusion.	**Test Priority**	High
Pre-Requisite	• The application is installed and running on a test device. • The user is logged in successfully.	**Post-Requisite**	Detailed usability testing report with recommendations and improvements based on findings.

Test Execution Steps:

S.No	**Action**	**Test Inputs**	**Expected Output**	**Actual Output**	**Test Result**	**Test Comments**

1	Test the ease of navigation in the Future Saver App	Navigating through different module (e.g., Home, Budget, Transaction History, Report, Chatbot, Profile)	User should be able to navigate smoothly between each module with minimal loading time which is less than 2 seconds.	The user can easily navigate between all main module with minimal taps. Navigation is smooth and responsive, and all menu items are clearly labelled.	Pass	
2	Verify the consistency in layout and design across modules	• Navigating through different pages (e.g., Home, Budget, Transaction History, Report,	The layout, fonts, colours, and button styles should remain consistent across all screens.	All the UI elements are consistent across all the pages.	Pass	

		Chatbot, Profile)				
3	Error validation messages are clearly visible and app responsiveness to the action.	• Input invalid data in budget and transaction page • Click add, edit cancel or delete button in budget and transaction page	The app should display clear validation messages for each required field when saving without required data. App should prompt user to confirm cancellation or deletion if click yes.	Validation messages appear clearly below the relevant fields when input is missing or incorrect. The app responds quickly to user actions like add, edit or delete.	Pass	

5.3.3 Reliability Testing

Reliability testing evaluates whether the application can consistently perform its intended functions without failure over time. It ensures the system handles regular and peak usage without crashing, data corruption, or unexpected behaviour, thereby providing a stable and dependable user experience.

Table 5.19: Test Case of Reliability Testing

Test Scenario ID	TS_Reliability_01	**Test Case ID**	TC_Reliability_01
Test Title	Continuous Operation Reliability Test	**Test Module**	Overall System
Test Case Description	To verify that the application operates reliably and without errors or crashes during prolonged use under typical load conditions.	**Test Priority**	High
Pre-Requisite	• Future Saver App is installed and running on the test device. • Test environment (network, database) are set up and stable.	**Post-Requisite**	Test logs and performance metrics are saved for review and analysis.
Test Execution Steps:			

S.No	Action	Test Inputs	Expected Output	Actual Output	Test Result	Test Comments
1	Verify the API response time under both normal and heavy load conditions.	Perform a simulated API request to the Azure Form Recognizer and Chatbot services.	The API should respond within 2 seconds for standard requests and 5 seconds under heavy load.	The API successfully returned responses within 2 seconds for standard requests and 5 seconds during heavy load testing.	Pass	
2	Assess the app's response to network disconnections and its behaviour once the connection is restored.	• Simulate network loss during app usage (e.g., adding a transaction) • Restore the network after disconnection	The app should handle the network loss by showing an error message, then automatically reconnect and sync data once the network is restored, without	No error message is displayed, and the app is automatically reconnected and updated the data automatically once the network is restored, without requiring any manual intervention.	Pass	

			requiring any manual intervention.			
3	Verify that the app data stays synchronized across multiple devices	• Add a transaction on Device 1 and sync with Device 2	Add a transaction on Device 1 and sync with Device 2	Changes made on one device are immediately reflected on the other device without any delay.	Pass	
4	Verify system behaviour after restart	• Restart the device after performing key actions (e.g., adding a transaction)	The app should open correctly, and any unsaved data should not be lost. The app should resume from where the user left off.	The app opened, but the unsaved data was lost, and it did not resume from where the user left off.	Fail	

5.4 User Acceptance Testing (UAT)

User Acceptance Testing (UAT) had been conducted among 25 respondents to assess the overall usability, functionality, and user satisfaction of the Future Saver application. The purpose of this questionnaire is to gather real-world feedback from the end-users, ensuring the app meets their expectations and is intuitive to use. Insights from this UAT help identify any remaining usability issues, validate whether all core features work as intended, and highlight areas for further enhancement before the final deployment.

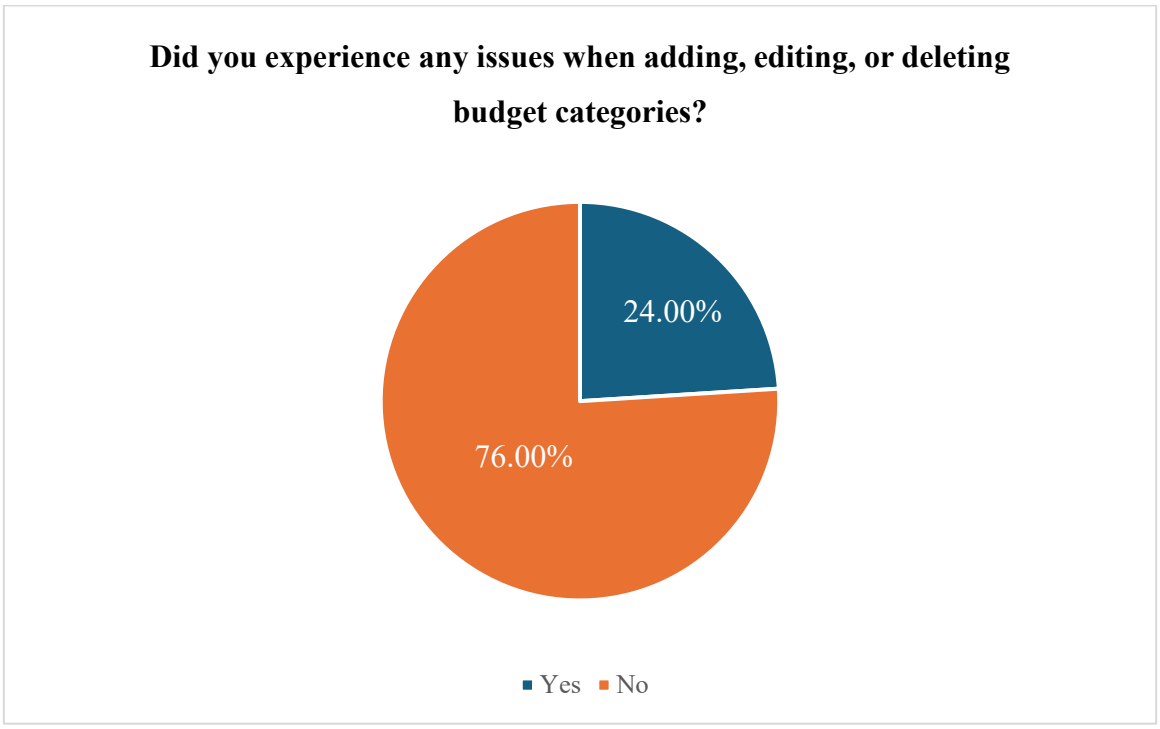


Figure 5.1: The Questionnaires - Did you experience any issues when adding, editing, or deleting budget categories?

The pie chart above shows whether the respondents experienced any issues when adding, editing, or deleting the budget categories. The largest group, 76% (19 respondents), reported that they did not face any issues. Meanwhile, 24% (6 respondents) indicated that they had encountered problems. This data suggests that most respondents found the process of managing budget categories to be smooth and issue-free.

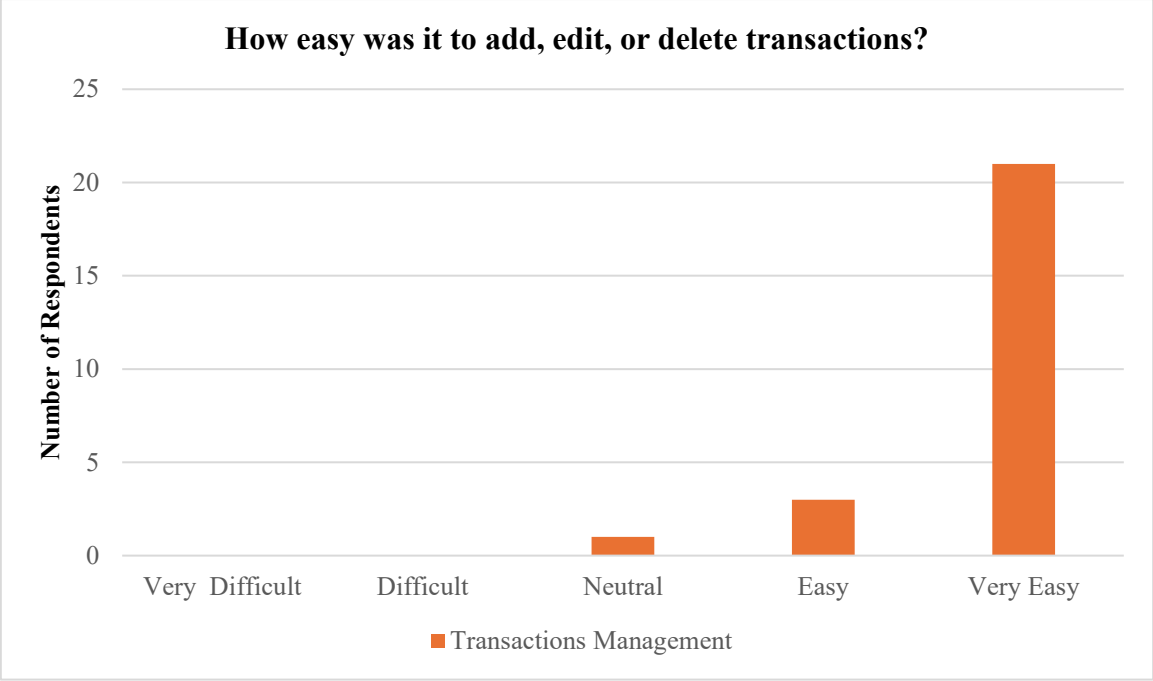


Figure 5.2: The Questionnaires - How easy was it to add, edit, or delete transactions?

The bar chart illustrates respondents' feedback on how easy it was to add, edit, or delete transactions in the app. The largest group, 21 respondents, rated the process as Very Easy, followed by 3 respondents who found it Easy. Only 1 respondent selected Neutral, and no respondents chose Difficult or Very Difficult. This feedback highlights the app's intuitive and user-friendly interface, suggesting that the transaction management features are well-designed and accessible which allow users to perform essential financial tasks with minimal effort or confusion.

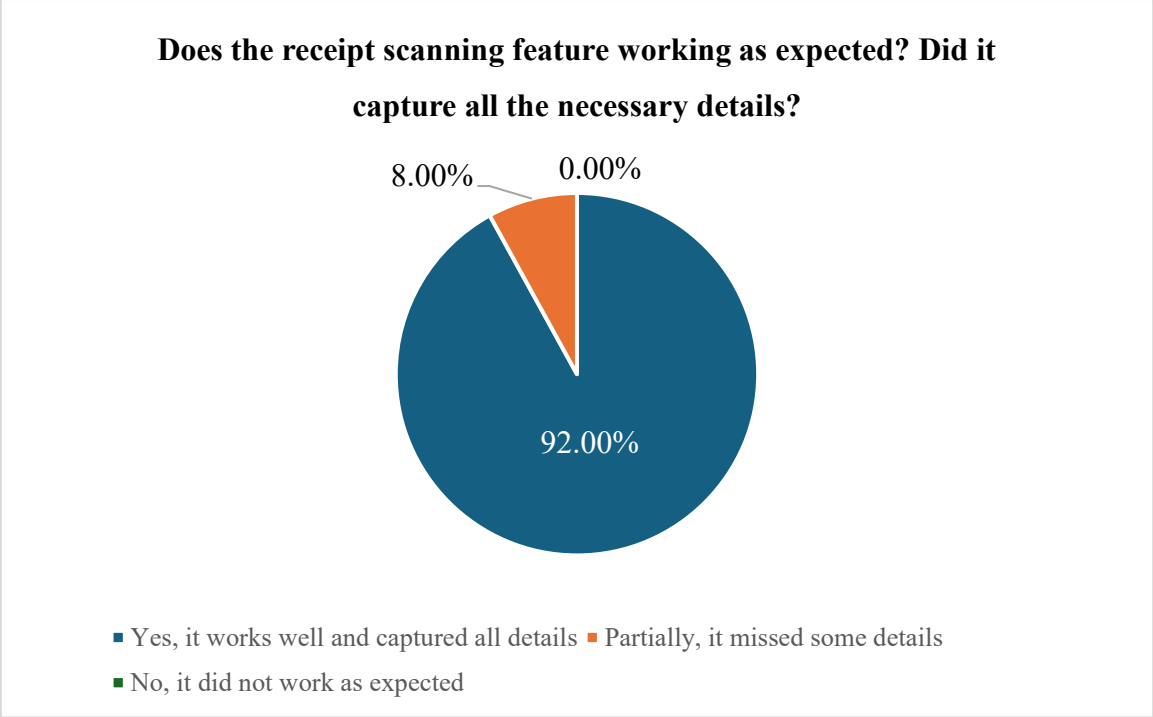


Figure 5.3: The Questionnaires – Does the receipt scanning feature working as expected? Did it capture all the necessary details?

The pie chart shows respondents' feedback on the receipt scanning feature. The vast majority, 92% of respondents, stated that the feature worked well and captured all details. In contrast, only 8% of respondents indicated that it partially missed some details. None of the respondents reported that the feature failed completely. This data suggests a high level of user satisfaction with the receipt scanning functionality which reflect its accuracy and reliability in automating expense entry and reducing the need for manual input.

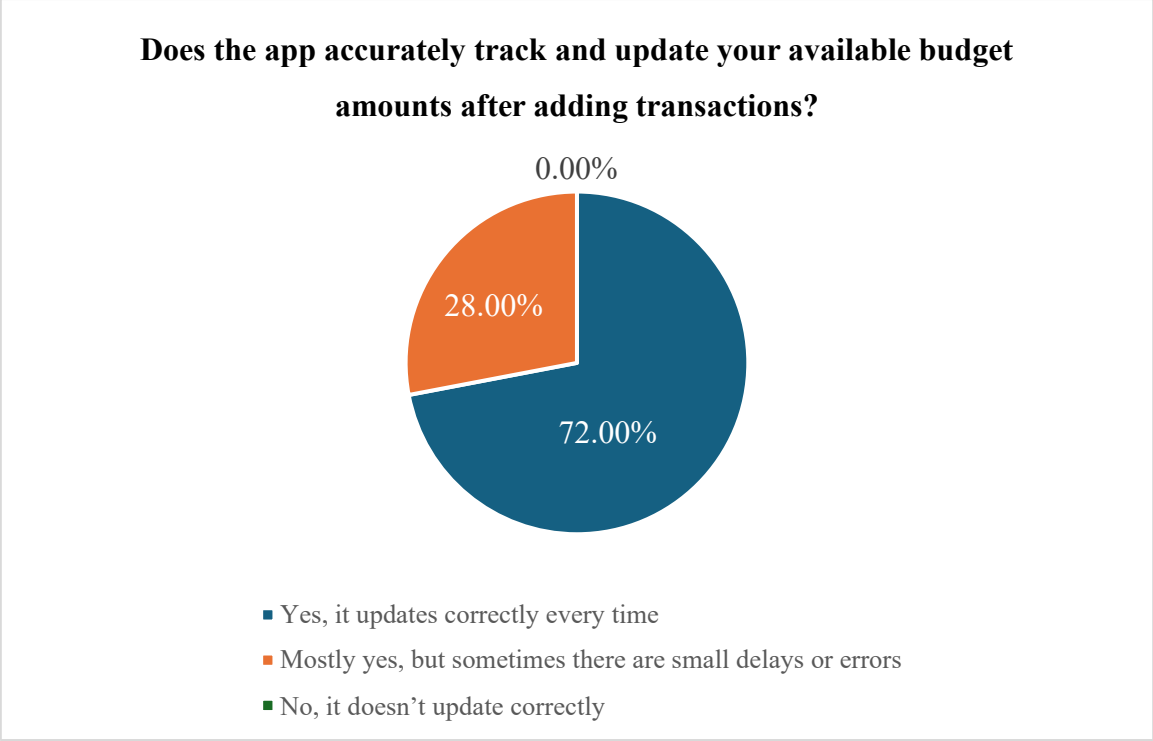


Figure 5.4: The Questionnaires – Does the app accurately track and update your available budget amounts after adding transactions?

The pie chart illustrates respondents' feedback on how accurately the app updates available budget amounts after adding transactions. The majority, 72%, reported that the app updates correctly every time. Meanwhile, 28% indicated that it mostly updates correctly, although they experienced occasional small delays or errors. No respondents reported that the app failed to update the budget correctly. This indicates that most users found the app to be reliable in tracking their available budget.

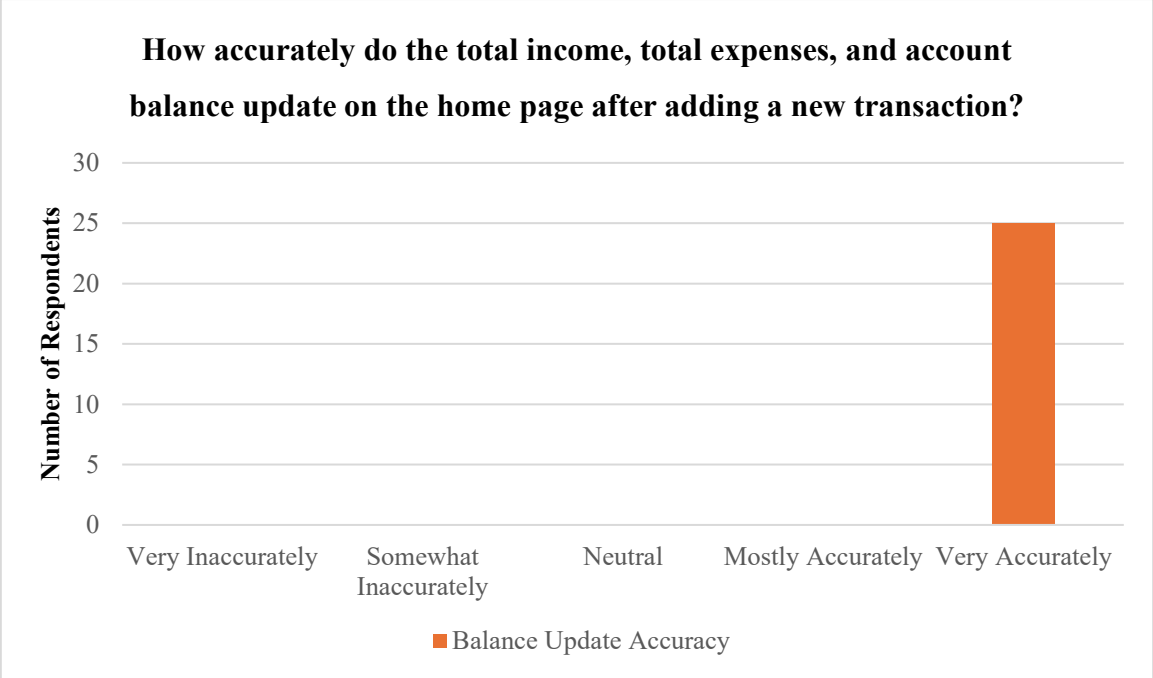


Figure 5.5: The Questionnaires – How accurately do the total income, total expenses, and account balance update on the home page after adding a new transaction?

The bar chart shows respondents' feedback on the accuracy of the updates to total income, total expenses, and account balance on the home page after adding a new transaction. All 25 respondents indicated that these amounts were updated very accurately. No respondents selected the other options such as mostly accurately, neutral, somewhat inaccurately, or very inaccurately. This indicates that all respondents were highly satisfied with the app's accuracy in updating financial information on the home page.

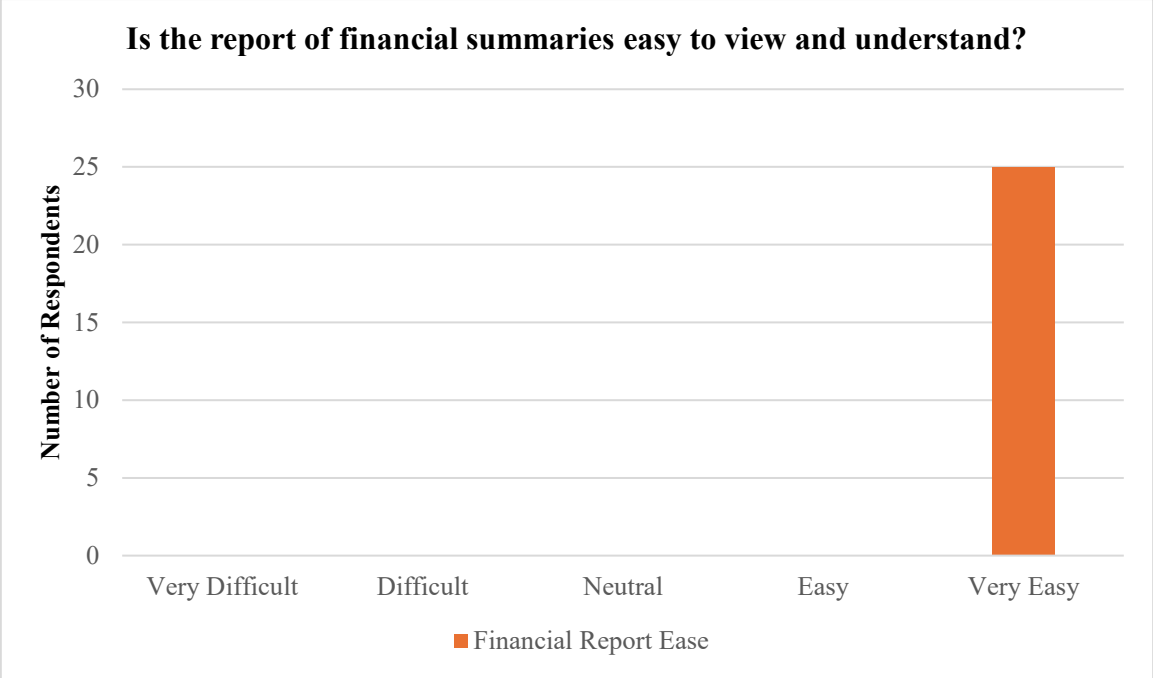


Figure 5.6: The Questionnaires - Is the report of financial summaries easy to view and understand?

The bar chart illustrates respondents' feedback on the ease of viewing and understanding the report of financial summaries. All 25 respondents rated the report as Very Easy to view and understand, with no one selecting Easy, Neutral, Difficult, or Very Difficult. The uniform response from all users highlights the app's effectiveness in presenting financial reports in a clear and intuitive way. The results reflect strong user satisfaction and indicate that the app successfully communicates financial information in a way that is easy to interpret and navigate.

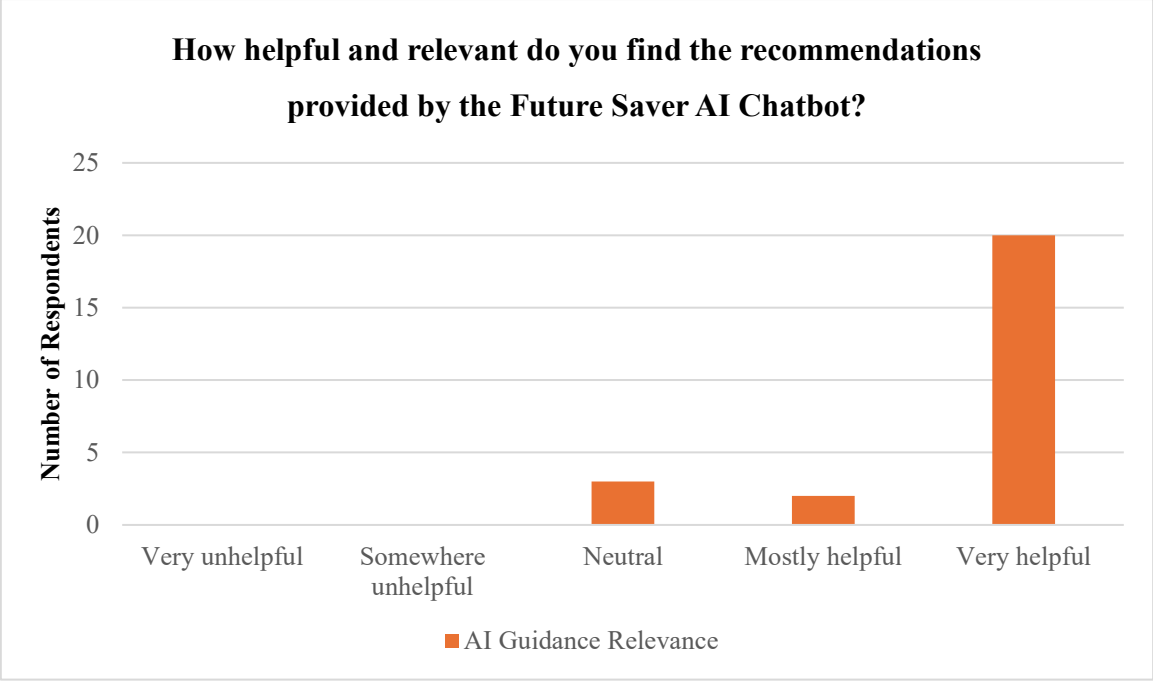


Figure 5.7: The Questionnaires - How helpful and relevant do you find the recommendations provided by the Future Saver AI Chatbot?

The bar chart illustrates respondents' feedback on how helpful and relevant they found the recommendations provided by the Future Saver AI Chatbot. The majority of respondents, 20 in total, found the recommendations Very Helpful. Three respondents selected Neutral, and two respondents rated the recommendations as Mostly Helpful. None of the respondents found the recommendations to be unhelpful. This indicates that most respondents were satisfied with the quality and relevance of the AI Chatbot's recommendations.

Figure 5.8: The Questionnaires - How clear and helpful were the error messages when erros occurred?

The bar chart illustrates respondents' ratings of the clarity and helpfulness of error messages when errors occurred. All 25 respondents rated the error messages as Very Helpful, while no respondents selected Mostly Helpful, Neutral, Somewhat Helpful, or Very Unhelpful. This indicates that all respondents found the error messages extremely clear and useful.

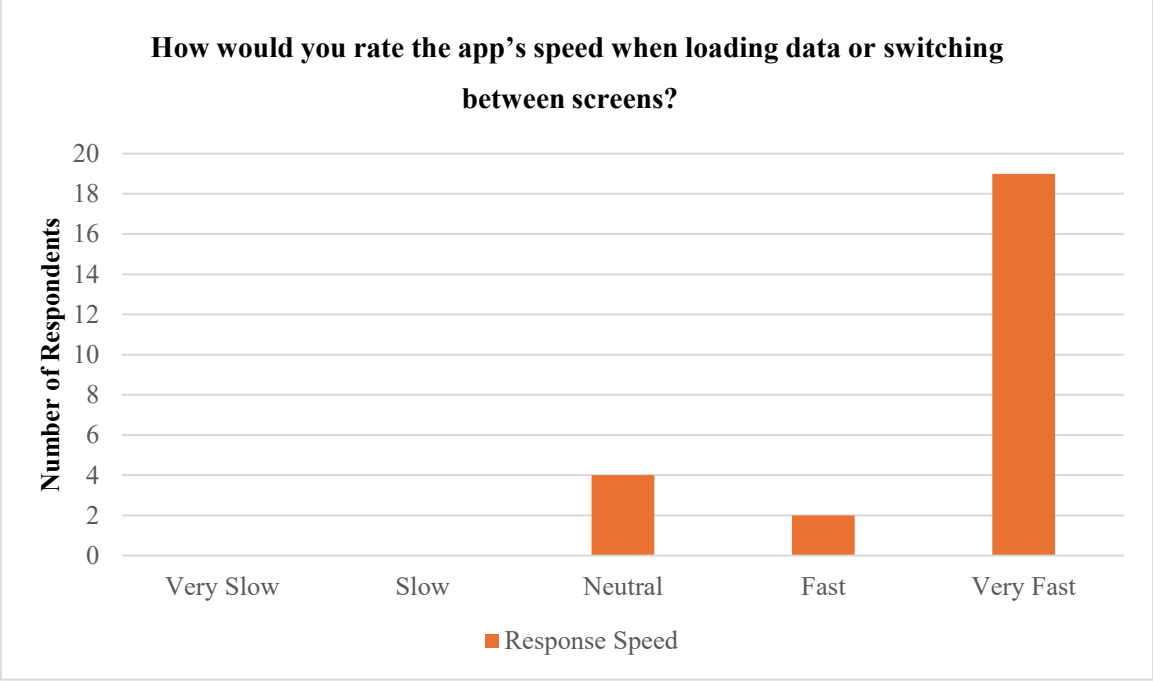


Figure 5.9: The Questionnaires - How would you rate the app's speed when loading data or switching between screens?

The bar chart illustrates respondents' ratings of the app's speed when loading data or switching between screens. The majority, 19 respondents, rated the app as Very Fast, while 2 respondents found it Fast. A smaller group of 4 respondents selected Neutral, and none of the respondents chose Slow or Very Slow. This data indicates that most respondents experienced a smooth and responsive app, with minimal delays in data loading and screen transitions.

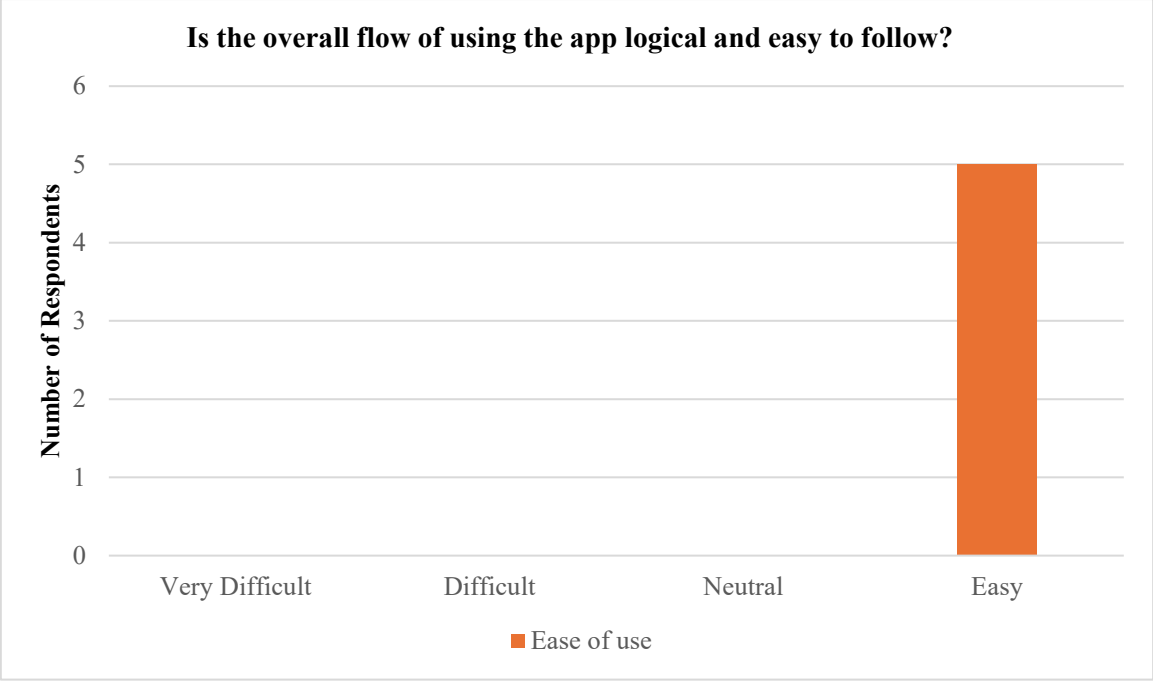


Figure 5.10: The Questionnaires - Is the overall flow of using the app logical and easy to follow?

The bar chart shows respondents' ratings on whether the overall flow of using the app is logical and easy to follow. All 25 respondents selected Easy, while none chose Very Difficult, Difficult, or Neutral. This consistent feedback suggests that users found the app well-structured, with a smooth and coherent progression from one function to another. The results reflect a strong sense of usability, indicating that the app's design effectively supports intuitive interaction and minimizes confusion throughout the user journey.

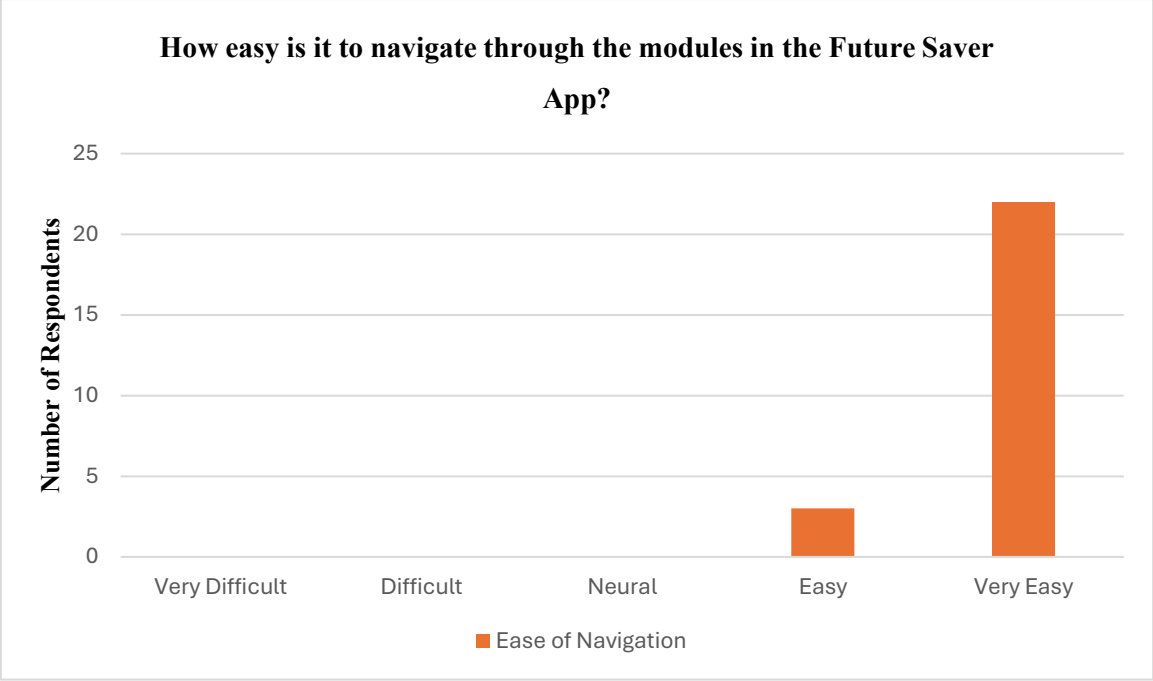


Figure 5.11: The Questionnaires - How easy is it to navigate through the modules in the Future Saver App?

The bar chart illustrates respondents' ratings of how easy it is to navigate through the modules in the Future Saver App. A large majority, 22 respondents, rated the navigation as Very Easy, while 3 respondents rated it as Easy. None of the respondents selected Neutral, Difficult, or Very Difficult. This strong approval suggests that the app's layout and structure are thoughtfully designed, allowing users to move between features effortlessly. The results highlight the app's success in delivering a smooth and intuitive user experience, which is essential for encouraging regular use and engagement.

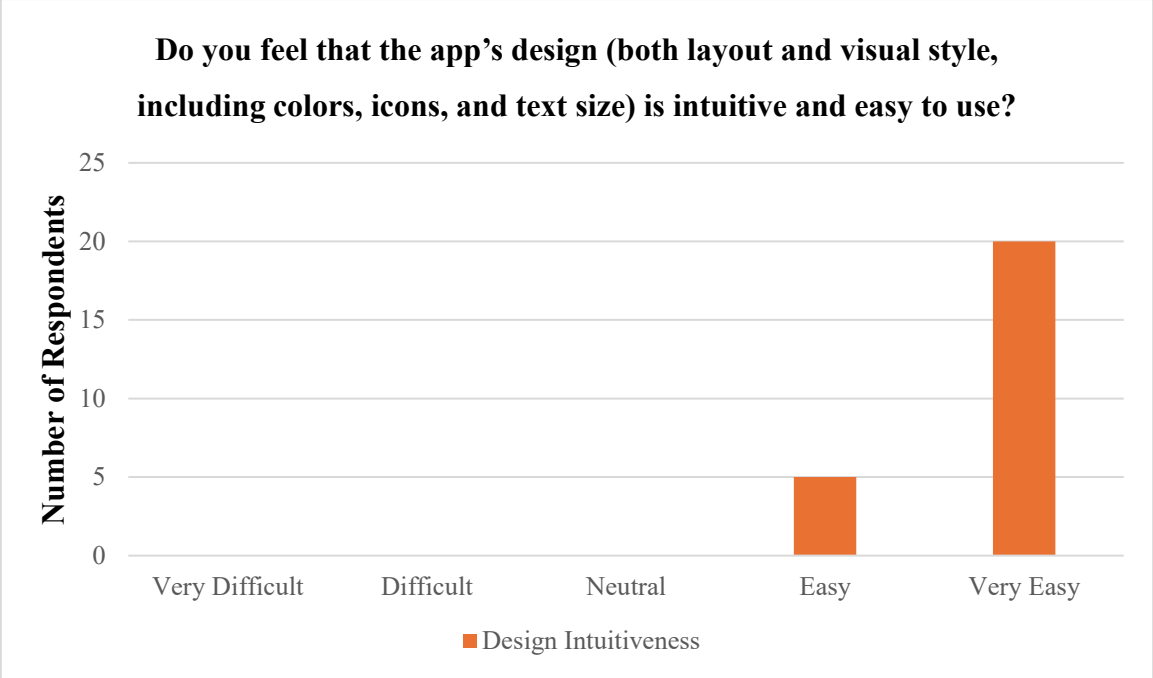


Figure 5.12: The Questionnaires – Do you feel that the app's design (both layout and visual style, including colours, icons, and text size) is intuitive and easy to use?

The bar chart shows respondents' ratings of the app's design in terms of layout, visual style, colors, icons, and text size. The majority, 20 respondents, rated the design as Very Easy and 5 respondents rated it as Easy. None of the respondents chose Neutral, Difficult, or Very Difficult. This consistent feedback suggests that the app's visual and structural design is well-executed, offering an intuitive and aesthetically pleasing user experience that effectively supports ease of navigation and accessibility.

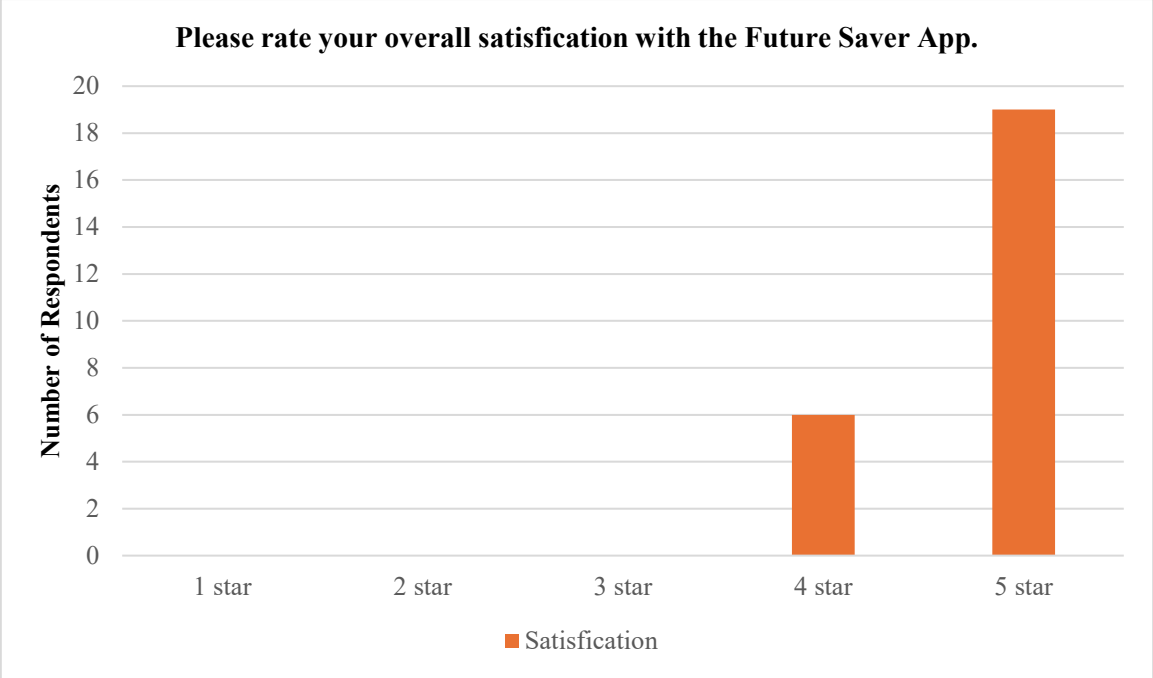


Figure 5.13: The Questionnaires - Please rate your overall satisfaction with the Future Saver App

The bar chart shows respondents' overall satisfaction with the Future Saver App. The majority of respondents, 19 in total, gave the app a 5-star rating, while 6 respondents rated it 4 stars. No respondents gave a rating lower than 4 stars. This indicates a very high level of satisfaction with the app among the respondents.

5.5 Summary

This chapter demonstrated how both functional and non-functional testing were conducted to evaluate the reliability and usability of the Future Saver application. The results confirmed that the core features performed as intended and met the defined user requirements. Additionally, feedback gathered during the testing phase identified areas for future enhancement, reinforcing the application's aim to provide a stable, efficient, and user-friendly personal finance management tool.

CHAPTER 6: CONCLUSION AND FUTURE WORK

6.1 Conclusion

The Smart Budget Personal Finance Tracking Mobile Application is developed to provide users with a practical and accessible solution for managing their personal finances more effectively. By addressing common challenges such as difficulty in tracking expenses, budgeting, and understanding financial habits, the app offers a comprehensive solution through features like category-based monthly budgeting, manual transaction logging, receipt scanning with Azure Form Recognizer, and AI-powered financial guidance using GerminiAI's API to respond to user financial queries

Furthermore, the integration of visual reports, including spending by category and income versus expenses, enables users to gain clear insights into their financial behaviour over time. Real-time synchronization and secure data storage via Firebase Firestore further enhance the app's reliability and usability. Overall, the application delivers meaningful support for users aiming to improve their financial awareness and take greater control of their spending habits.

6.2 Achievement of Objectives

The development of the Smart Budget Personal Finance Tracking Mobile Application has successfully met its primary objectives, as outlined at the start of the project. It has delivered a comprehensive and user-friendly financial management solution, aligning well with the initial goals and assumptions.

Key achievements of this project include:

- Monthly Budgeting: Users can set budgets for different expense categories on a monthly basis. The system calculates the remaining budget dynamically based on logged expenses, providing warnings when nearing limits.

- Income and Expense Recording: Users can manually input income or expense transactions, specifying category, date, and amount. The interface is intuitive and includes validation to prevent errors.
- Receipt Scanning: The app allows users to upload or capture receipt images. Using Azure Form Recognizer, the system extracts key information such as total amount, date, and merchant name, automatically filling these into the transaction form.
- Report Generation: The app includes automatic generation of visual reports based on user data. It provides:
    1. Spending by Category Report: A clear breakdown of how the user's expenses are distributed across different categories (e.g., Rent, Food, Transportation), helping users identify major spending areas.
    2. Income vs. Spending Report: A monthly comparison that shows whether the user is spending within their means or overspending, supporting better financial planning.
- AI Chatbot Assistance: The integrated AI chatbot responds to user queries related to financial status, such as "How much have I spent on food this month?" or "Any suggestions to save money?" The chatbot uses natural language processing to understand the input and returns relevant responses based on stored financial data.
- Real-time Data Management: The system uses Firebase Firestore for real-time cloud-based data storage, ensuring that transactions, user profiles, and budget information are consistently updated.

6.3 Limitations

Although the Smart Budget Personal Finance Tracking Mobile Application has successfully achieved its primary objectives, certain limitations and constraints were encountered throughout its development and are present in its current version. These challenges have influenced the overall functionality, feature scope, and general usability of the app.

- No User Authentication via Social Login: Only email-password authentication is supported. Login options such as Google, Apple or Facebook were excluded to maintain focus on the core functionality. Although email-password login fulfils basic access control requirements, the lack of social login options may affect user convenience and the onboarding experience. Social authentication is widely preferred for its speed and reduced friction, as it allow users to log in without creating and remembering additional credentials.
- Azure Form Recognizer Limitations: The Azure Form Recognizer performs best with structured and clearly printed receipts. Handwritten, blurry, or poorly formatted receipts may result in inaccurate or incomplete extraction of key data fields such as transaction name, amount, or date. Additionally, the prebuilt model used in the application is designed specifically for expense-type receipts and cannot reliably interpret income-related receipts, such as salary slips or payment confirmations. Scanning income receipts would require custom model training to accurately extract relevant data. As a result, users are currently unable to scan income receipts, which limits the comprehensiveness of financial tracking.
- Internet Dependency: Several core features of the application, including the AI chatbot, data synchronization, and receipt scanning, are dependent on a stable internet connection. Without internet access, users are unable to interact with the AI chatbot for financial tips, sync transaction data to the cloud (Firestore), or use the receipt scanning

feature. This lack of offline support can be a problem for users in areas with poor or no internet which make the app harder to use in those situations. Offline capabilities were not implemented due to time and resource constraints.

6.4 Recommendations and Future Work

Several additional enhancements could be implemented to further improve the application's functionality and enhance the overall user experience:

- Implementing security enhancements such as biometric authentication (e.g., fingerprint or facial recognition) provides an additional layer of security by verifying the user's unique physical traits which can make unauthorized access more difficult. Additionally, integrating social media login options (e.g., Google or Facebook) strengthens user privacy, improves data protection, and offers a more convenient login experience. By allowing users to quickly access their accounts without remembering additional credentials, social media login reduces friction during sign-in. Together, these features enhance the overall user experience by making the app both more secure and user-friendly.
- Supporting multi-currency and localization, including multiple language options would make the app more accessible to a broader range of users across different regions. By allowing users to select their preferred language and currency, the app can cater to diverse cultural and financial contexts, making it easier for people from different countries to manage their finances without confusion. This feature would enhance the user experience by providing personalized content, accurate currency conversion, and appropriate formatting for dates, numbers, and financial data, ultimately encouraging greater adoption and usability of the application worldwide.

- Expanding the receipt scanning feature to support income entries could further enhance the app's automation. Currently, the receipt scanner powered by Azure Form Recognizer is limited to extracting expense details from receipts and cannot process income sources such as salary slips. Future improvements could involve training a custom model to recognize various income documents so that users can scan both income and expense receipts. This would help provide a more complete financial record and simplify personal finance tracking.
- Incorporating offline support by enabling local data storage and automatic synchronization upon reconnection would allow users to manage their finances without internet access, improving convenience and ensuring uninterrupted usability. This feature would let users record transactions, view past data, and set budgets even when offline, and then seamlessly update the cloud database once a stable connection is available. By reducing dependence on continuous connectivity, the app would offer a more reliable and flexible experience, especially for users in areas with limited or unstable internet access.

References

About us:YNAB. (2024). YNAB. https://www.ynab.com/about-us

Adepegba, O., Fayemiwo, M. A., Oduwole, O., Onamade, A.A. (2019) An Android Based Mobile Application for Tracking Daily Expenses. *ResearchGate*. http://dx.doi.org/10.22624/AIMS/iSTEAMS-2019/V21N1P9

Agarwal, V., Varghese, N., Ray, R. (2024). An AI-Powered Personal Finance Assistant: Enhancing Financial Literacy and Management. *ResearchGate*. http://dx.doi.org/10.13140/RG.2.2.10706.57280

Belen Saglam, R., Nurse, J. R.C., & Hodges, D. (2021). Privacy concerns in chatbot interactions: When to trust and when to worry. *Communications in Computer and Information Science*, 391–399. https://doi.org/10.1007/978-3-030-78642-7_53

Best AI Models for OCR and PDF-to-Text Conversion. (2025, May 28). IntuitionLabs. https://intuitionlabs.ai/articles/ai-ocr-models-pdf-structured-text-comparison

Bhatt, P., Nutheti, S. C., Mamidipaka, G., Kondapally, U. K., Lokineni, H. (2024). Expense zTracker: A Smart Approach to Track Daily Expense. *Tuijin Jishu/Journal of Propulsion Technology, 45*(1). https://doi.org/10.52783/tjjpt.v45.i01.5664

Bluecoins Finance App Review: An incredibly simple finance app. (2021, June 23). Appedus. https://appedus.com/bluecoins-finance-app-review/

Document Intelligence receipt model. (2024). Microsoft Learn. https://learn.microsoft.com/en-us/azure/ai-services/document-intelligence/prebuilt/receipt?view=doc-intel-4.0.0

Eustaquio-Jiménez, R., Durand-Azurza, M., Gamboa-Cruzado, J., Morales, M. E. L., Ruiz, N. P., Montoya, R. L., & Céliz, N. M. O. (2024). Chatbots for customer service in financial entities —A comprehensive systematic review. *Journal of Infrastructure, Policy and Development, 8*(16), 10122. https://doi.org/10.24294/jipd10122

Goodbudget. (2021, February 16). Dayspring Partners. https://www.dayspringpartners.com/portfolio/ios-android-app-goodbudget/

Luthi, B. (2022, August 30). *Goodbudget Review: A Digital Envelope Budgeting System*. Experian. https://www.experian.com/blogs/ask-experian/goodbudget-budgeting-app-review/

Manoj, A. (2025). AI-Powered Personal Finance Apps: Are they Effective for Students? *International Journal of Research Publication and Reviews*, *6*(3), 2901–2905. https://doi.org/10.55248/gengpi.6.0325.11102

Muniyandi V. (2025). AI-Powered Document Processing with Azure Form Recognizer and Cognitive Search. *Journal of Computational Analysis and Applications. 33*(5), 1884-1902. https://www.researchgate.net/publication/393333689_AI-Powered_Document_Processing_with_Azure_Form_Recognizer_and_Cognitive_Search

Petrecca, L. (2025, July 15). *Can AI Help With Your Finances?*. Kiplinger. https://www.kiplinger.com/personal-finance/can-ai-help-with-your-finances

Software prototyping model and phases. (2023). GeeksforGeeks. https://www.geeksforgeeks.org/software-prototyping-model-and-phases/

Taylor, P. (2023). *Forecast number of mobile users worldwide from 2020-2025*. Statista. https://www.statista.com/statistics/218984/number-of-global-mobile-users-since-2010/

Tindall, T., & Neidel C. (2024, November 24). Review: *We Tested the YNAB App.* Nerdwallet. https://www.nerdwallet.com/article/finance/ynab-app-review

Wong, R. Y., & Tan, P. P. (2024). Receipt image extraction in budgeting mobile application. *Trends in Undergraduate Research*, *7*(1), c16-25. https://doi.org/10.33736/tur.6110.2024

Appendix A: Questionnaire

Questionnaire for Final Year Project (FYP): Smart Budget Personal Finance Tracking Mobile Application

Dear Respondent,

My name is Erin Wong Sing Ying (79332), a final year student from Bachelor of Software Engineering (Honors) at the Faculty of Computer Science and Information Technology (FCSIT), Universiti Malaysia Sarawak (UNIMAS). I am currently conducting this survey as part of my Final Year Project titled: "Smart Budget Personal Finance Tracking Mobile Application".

The goal of this project is to develop a mobile application called *Future Saver*, enabling users to record the transactions and track their daily cash flow easily for a clear and organized overview of their finances. The app integrates automated receipt scanning using the Azure Form Recognizer to effortlessly extract the transaction details, along with the real-time support from an AI chatbot which empowering users to manage their finances more effectively.

The survey consist of three sections:

- Section A: Demographic Information
- Section B: Financial Habits and Management
- Section C: Preferences for Personal Finance Tracking App

This survey will take approximately 5-10 minutes to complete.

Your responses are highly valued, and all information provided will be kept strictly confidential.

If you have any questions or concerns, please contact me at 79332@siswa.unimas.my.

Thank you for taking time to participate in this survey.

erin270601@hotmail.com Switch accounts

Not shared

Section A: Demographic Information

This section gathers general demographic information to provide better insights into participant backgrounds and support effective analysis of the survey results.

Gender *

- ◯ Male
- ◯ Female

Age *

- ◯ Below 18
- ◯ 18 - 24
- ◯ 25 - 34
- ◯ 35 - 44
- ◯ 45 and above

What is your occupation? *

- ◯ Student
- ◯ Full Time Employment
- ◯ Part Time Employment
- ◯ Unemployed
- ◯ Retired

What is your income level? *

- ◯ Less than RM1000
- ◯ RM1000 - RM3000
- ◯ RM3001 - RM5000
- ◯ RM5001 and above

Section B: Financial Habits and Management

This section aims to explore your current financial habits and practices related to managing personal finances. By understanding how you track expenses, set budgets, and the challenges you face, this section will help identify key areas for improvement and better financial decision-making.

What is your primary goal of using a finance tracking tool? *

- ◯ Budget planning
- ◯ Expense Tracking
- ◯ Money Saving
- ◯ Other: _____

Which tools do you currently use to track your expenses and income? *

- ◯ Notepad / Manual records
- ◯ Excel Spreadsheet
- ◯ Using a mobile budgeting app
- ◯ I don't track my expenses
- ◯ Other: _____

How often do you track your expenses? *

- ◯ Daily
- ◯ Weekly
- ◯ Monthly
- ◯ Rarely
- ◯ Never

Do you currently have a budget for your monthly expenses? *

- ◯ Yes, I have a detailed budget
- ◯ Yes, but it is not detailed
- ◯ No, I don't have a budget

Have you ever faced any challenges in managing your finances? *

- ◯ Yes, often
- ◯ Yes, rarely
- ◯ No, never

What are the main challenges you face in managing finances? *

- [ ] Forgot to log expense and income
- [ ] Manual data entry
- [ ] Time consuming
- [ ] Difficult to categorize expense and income
- [ ] Lack of tools for budget analysis
- [ ] Lack of data visualization (graph) of income and expenses
- [ ] Lack of virtual assistant like AI to provide personalised financial advice
- [ ] Other: _____

How often do you forgot to log your expenses manually? *

- ( ) Always
- ( ) Often
- ( ) Sometimes
- ( ) Rarely
- ( ) Never

Have you spent over your planned budget? *

- ( ) Yes
- ( ) No

How comfortable are you using technology for personal finance management? *

- ◯ Very comfortable
- ◯ Somewhat comfortable
- ◯ Not comfortable

How important for the app to help you to manage personal finance? *

	1	2	3	4	5	
Not important	◯	◯	◯	◯	◯	Extremely important

Section C: Preferences for Personal Finance Tracking App

This section seeks to gather your preferences and expectations for a personal finance tracking app. Your responses will help to identify the most important features and functionalities that would enhance your experience in managing finances effectively.

What device would you prefer to use the finance tracking app ? *

- ◯ Mobile Phone
- ◯ Tablet
- ◯ Desktop / Web
- ◯ Other: _____

Would you prefer a free app or a paid subscription-based app? *

- ◯ Free app
- ◯ Paid subscription app
- ◯ I don't mind as long as the features are valuable

How important is a visually appealing design in a finance tracking app for you? *

	1	2	3	4	5	
Not important	◯	◯	◯	◯	◯	Extremely important

What type of expenses category would you like to have? *

- ☐ Rent
- ☐ Utilities (Electricity, Water, Bill)
- ☐ Transportation
- ☐ Groceries
- ☐ Food & Dining
- ☐ Medical & Healthcare
- ☐ Shopping
- ☐ Entertainment
- ☐ Savings & Investments
- ☐ Other: _____

What income category would you like to have? *

- [ ] Salary
- [ ] Investment
- [ ] Education / Scholarships
- [ ] Other: _____

What features would you like to see in a personal finance tracking app? *

- [ ] User-friendly interface
- [ ] Budgeting tools
- [ ] Daily transaction logging
- [ ] Savings goals
- [ ] Automated data extraction from receipts (Azure Form Recognizer API)
- [ ] Monthly expenses summaries
- [ ] Notifications and alerts
- [ ] Real-time financial suggestions via AI chatbot
- [ ] Other: _____

Would you like to receive the reminders for expenses tracking and budget limit warnings? *

- ( ) Yes
- ( ) No

How often would you like to receive notifications or reminders from the app? *

- ◯ Daily
- ◯ Weekly
- ◯ Monthly
- ◯ As needed

How important is it for the app to provide automated receipt scanning through Azure Form Recognizer API for logging expenses? *

	1	2	3	4	5	
Not important	◯	◯	◯	◯	◯	Extremely important

Do you prefer visual insights (e.g., charts, graphs) for financial analysis? *

- ◯ Yes
- ◯ No

How important is it for the app to provide AI chatbot for real-time financial guidance? *

	1	2	3	4	5	
Not important	◯	◯	◯	◯	◯	Extremely important

Do you have any additional feedback or suggestions for personal finance tracking app?

Your answer

Appendix B: User Acceptance Testing (UAT) Questionnaire for Future Saver

User Acceptance Testing (UAT) Questionnaire of Future Saver Mobile Application

Dear respondents,

I am Erin Wong Sing Ying (79332), a final year student from Bachelor Degree of Software Engineering (Honors) at the Faculty of Computer Science and Information Technology (FCSIT), Universiti Malaysia Sarawak (UNIMAS). I would like to invite you to participate in my final year project titled **"Smart Budget Personal Finance Tracking Mobile Application"**

The purpose of this questionnaire is to gather feedback from the users regarding the design, functionality, usability, reliability, and overall experience of the Future Saver mobile application. Your responses will help evaluate how well the app meets user needs and identify areas for improvement before its final release.

If you are an android user, please download the Future Saver App using the following link: https://drive.google.com/file/d/1z8KZZy7499P2nzJq_Imk9tunTt2RB8yM/view?usp=sharing

Please note that the app is only available for Android devices at the moment.

Thank you for participating in the User Acceptance Testing (UAT) for Future Saver. Your feedback is extremely valuable and will help me improve the app to better meet the needs of future users. If you have any questions or require further information, feel free to contact me at **79332@siswa.unimas.my**.

erin270601@hotmail.com Switch accounts

Not shared

* Indicates required question

Did you experience any issues when adding, editing, or deleting budget categories? *

- ◯ Yes
- ◯ No

If yes, please write down the issues and bugs that you have encounter.

Your answer

How easy was it to add, edit, or delete the transactions? *

	1	2	3	4	5	
Very Difficult	◯	◯	◯	◯	◯	Very Easy

Does the receipt scanning feature working as expected? Did it capture all the necessary details? *

- ◯ Yes, it works well and captured all details
- ◯ Partially, it missed some details
- ◯ No, it did not work as expected

Does the app accurately track and update your available budget amounts after adding transactions? *

- ◯ Yes, it updates correctly every time
- ◯ Mostly yes, but sometimes there are small delays or errors
- ◯ No, it doesn't update correctly

How accurately do the total income, total expenses, and account balance update on the home page after adding a new transaction? *

	1	2	3	4	5	
Very inaccurately	◯	◯	◯	◯	◯	Very accurately

Is the report of financial summaries easy to view and understand? *

	1	2	3	4	5	
Very Difficult	◯	◯	◯	◯	◯	Very Easy

How helpful and relevant do you find the recommendations provided by the Future Saver AI Chatbot? *

- ◯ Very helpful and relevant
- ◯ Mostly helpful and relevant
- ◯ Neutral
- ◯ Somewhat unhelpful or irrelevant
- ◯ Very unhelpful or irrelevant

How clear and helpful were the error messages when errors occurred? *

- ◯ Very clear and helpful
- ◯ Mostly clear and helpful
- ◯ Neutral
- ◯ Somewhat unclear
- ◯ Very unclear and not helpful

How would you rate the app's speed when loading data or switching between screens? *

	1	2	3	4	5	
Very Slow	◯	◯	◯	◯	◯	Very Fast

Is the overall flow of using the app logical and easy to follow? *

- ◯ Very logical and easy to follow
- ◯ Mostly logical and easy to follow
- ◯ Neutral
- ◯ Somewhat confusing
- ◯ Very confusing

How easy is it to navigate through the modules in the Future Saver app?

	1	2	3	4	5	
Very Difficult	◯	◯	◯	◯	◯	Very Easy

Do you feel that the app's design (both layout and visual style, including colors, icons, and text size) is intuitive and easy to use?

	1	2	3	4	5	
Very Difficult	○	○	○	○	○	Very Easy

Please rate your overall satisfication with the Future Saver App. *

1	2	3	4	5
☆	☆	☆	☆	☆

Would you recommend this app to others? *

- ○ Yes
- ○ No
- ○ Maybe

Any additional suggestions for Future Saver App?

Your answer