SPLITPAY: A BILL SPLITTING MOBILE APPLICATION SYSTEM

MOHD RAMIZAN BIN ROSLAN

FACULTY OF COMPUTING AND INFORMATICS

UNIVERSITI MALAYSIA SABAH

2025

SPLITPAY: A BILL SPLITTING MOBILE APPLICATION SYSTEM

MOHD RAMIZAN BIN ROSLAN

THESIS SUBMITTED IN PARTIAL FULFILLMENT FOR THE DEGREE OF BACHELOR OF COMPUTER SCIENCE WITH HONOURS (SOFTWARE ENGINEERING)

FACULTY OF COMPUTING AND INFORMATICS

UNIVERSITI MALAYSIA SABAH

2025

NAME : MOHD RAMIZAN BIN ROSLAN

MATRIC NUMBER : BI18110321

TITLE : SPLITPAY: A BILL SPLITTING MOBILE APPLICATION SYSTEM

DEGREE : BACHELOR OF COMPUTER SCIENCE WITH HONOURS (SOFTWARE ENGINEERING)

CERTIFIED BY:

1. SUPERVISOR **SIGNATURE**

TS DR NORAZLINA BINTI KHAMIS

DR. NORAZLINA KHAMIS
Senior Lecturer
Faculty of Computing and Informatics
Universiti Malaysia Sabah

DECLARATION

I hereby declare that the material in this thesis is my own except for quotations, equations, summaries and references, which have been duly acknowledged.

27 JUNE 2025

MOHD RAMIZAN BIN ROSLAN
BI18110321

ACKNOWLEDGEMENT

First and foremost, with heartfelt gratitude and humility, I would like to thank the Almighty Allah SWT for His endless blessings, particularly the strength, health, and clarity of mind granted to me throughout the course of completing my Final Year Project 1. It is through His grace and guidance that I have been able to reach this important milestone.

I am deeply thankful to my supervisor, Dr. Norazlina Binti Khamis, whose unwavering support, insightful feedback, and thoughtful guidance have been instrumental to the progress and development of this project. Her commitment to excellence and mentorship has been a constant source of motivation and inspiration.

I would also like to extend my sincere appreciation to all the lecturers and staff at the Faculty of Computing and Informatics (FKI). Their dedication to teaching and their willingness to share knowledge have significantly enriched my academic experience and personal growth during my time here.

My deepest gratitude goes to my beloved parents and family members. Their unconditional love, continuous prayers, and steadfast encouragement have been the foundation of my strength throughout this academic journey. I am truly blessed to have them by my side.

Lastly, to my friends and fellow classmates—thank you for your camaraderie, support, and the many moments of shared learning and motivation. Your presence and kindness have made this journey not only manageable but meaningful.

MOHD RAMIZAN BIN ROSLAN

27 JUNE 2025

ABSTRACT

SplitPay is a bill-splitting mobile application designed to address the inefficiencies and challenges associated with managing shared expenses in group settings. Traditional methods of splitting bills, such as manual calculations, messaging apps, or verbal agreements, often lead to confusion, disputes, and miscalculations. This project aims to provide a user-friendly solution by automating the bill-splitting process, enabling real-time debt tracking, and incorporating advanced features like receipt scanning using Optical Character Recognition (OCR) and QR code payment facilitation. Developed for the Android platform using the waterfall methodology, SplitPay focuses on key modules such as user management, expense logging, split logic, and payment facilitation. The application targets university students, young adults, and other groups who frequently engage in shared financial activities. By streamlining expense management and promoting transparency, SplitPay reduces errors, disputes, and delays in payments, ultimately enhancing financial clarity and collaboration among users. This project aligns with Sustainable Development Goal (SDG) 8 by promoting financial responsibility and economic awareness among individuals.

ABSTRAK

SPLITPAY: SISTEM APLIKASI MUDAH ALIH UNTUK PEMBAHAGIAN BIL

SplitPay ialah sebuah aplikasi mudah alih untuk pembahagian bil yang dibangunkan bagi menangani ketidakefisienan dan cabaran dalam mengurus perbelanjaan bersama dalam situasi berkumpulan. Kaedah tradisional pembahagian bil seperti pengiraan secara manual, penggunaan aplikasi mesej, atau perjanjian lisan sering kali mengakibatkan kekeliruan, pertikaian, dan kesilapan pengiraan. Projek ini bertujuan untuk menyediakan satu penyelesaian yang mesra pengguna dengan mengautomasikan proses pembahagian bil, membolehkan penjejakan hutang secara masa nyata, serta menggabungkan ciri-ciri lanjutan seperti pengimbasan resit menggunakan teknologi Pengecaman Aksara Optik (OCR) dan kemudahan pembayaran melalui kod QR. Aplikasi ini dibangunkan khusus untuk platform Android menggunakan metodologi waterfall, dan memberi tumpuan kepada modul-modul utama seperti pengurusan pengguna, pencatatan perbelanjaan, logik pembahagian, dan pemudahan pembayaran. Sasaran utama aplikasi ini adalah pelajar universiti, golongan dewasa muda, serta kumpulan lain yang sering terlibat dalam aktiviti kewangan secara bersama. Dengan memperkemas pengurusan perbelanjaan dan menggalakkan ketelusan, SplitPay dapat mengurangkan kesilapan, pertikaian, dan kelewatan dalam pembayaran, sekaligus meningkatkan kejelasan kewangan dan kerjasama dalam kalangan pengguna. Projek ini juga selari dengan Matlamat Pembangunan Mampan (SDG) 8 dengan mempromosikan tanggungjawab kewangan dan kesedaran ekonomi dalam kalangan individu.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1:

PROJECT INTRODUCTION

1.1 Chapter Overview

This chapter introduces SplitPay, a bill-splitter mobile application that mainly function to help users split bills in a shared group expense automatically. This chapter contains the problem with current manual methods, the project objectives, and the definition of the project's scope, its limitations, and the expected outcomes.

1.2 Problem Background

Splitting bills in group settings, such as dining out with friends or sharing expenses for events, is often a frustrating and error-prone process. Many individuals still rely on manual calculations, messaging apps, or verbal agreements to track shared expenses, leading to confusion, disputes, and miscalculations. In some cases, users struggle to remember who paid for what, how much each person owes, and whether payments have been settled. Traditional methods of bill-splitting often lack transparency and efficiency, making the process inconvenient for users. Without a structured system, individuals must repeatedly remind friends to make payments, leading to potential awkwardness and delays. Additionally, when multiple items are involved, manually distributing the costs fairly among participants becomes even more complex and time-consuming. With the increasing adoption of digital payments, there is a need for a modern and automated solution that streamlines the bill-splitting process. A mobile application that allows users to log expenses, calculate individual shares, track outstanding debts, and share QR codes for quick repayments would significantly improve efficiency. By integrating features such as real-time expense tracking, multiple split methods, and receipt scanning using Optical Character Recognition (OCR), users can manage shared expenses seamlessly. This project aims to provide a user-friendly solution that eliminates the need for manual calculations and promotes smooth financial interactions among friends, families, and colleagues.

1.3 Problem Statement

In social gatherings, group hangouts, or shared living arrangements, it is common for individuals to split bills informally—often through messaging platforms like WhatsApp, using calculator apps for manual computations, and relying on mutual trust to ensure everyone pays their share. While seemingly straightforward, this approach frequently leads to problems. Miscalculations are prevalent, especially when expenses involve varied items, unequal contributions, or discounts (Gneezy et al., 2004). Disagreements and confusion can easily arise over who paid for what and how much each participant owes. Often, one person is burdened with doing the calculations, sending reminders, and managing non-payers, which introduces tension and awkward dynamics within the group ("Split payment solutions: Innovations for hassle free bill splitting", 2024).

Despite the widespread adoption of smartphones and digital wallets, the process of splitting bills remains largely manual, fragmented, and inefficient. Users typically switch between apps for calculations, communication, and payments, increasing friction and reducing accountability (European Central Bank, 2023). While some tools like Splitwise attempt to streamline this, they often demand significant manual input and fail to accommodate real-life complexities such as uneven contributions, shared recurring expenses, or multi-party debts (Nidhibhat, 2023). Moreover, the discomfort around initiating conversations about money further exacerbates the problem, as people often avoid such discussions to maintain social harmony, leading to forgotten debts or unbalanced repayments (Splitkaro Marketing Team, 2024).

1.4 Project Objectives

1. To identify the requirements for developing a bill-splitting mobile application by using surveys and interviews among the students and working adults.
2. To develop SplitPay mobile application by using waterfall methodology.
3. To evaluate the system's usability through user testing and survey-based feedback evaluation.

1.5 Project Scopes

SplitPay is a mobile application that will mainly focus on providing an automated, user-friendly solution for bill-splitting and shared expense management in group settings. This project is divided into eight development modules.

1.5.1 Development Modules

Table 1: Development Modules for SplitPay

Module	Description
User Management	• Allows users to register, log in, and manage their accounts. • Enables users to update profile details, including name, contact information, and profile picture. • Manages authentication and security features such as password recovery. • Restricts access to non-registered users and ensures only verified accounts can interact with the system.
Circle Management	• Enables users to add and manage friends within the system. • Allows users to create and join expense groups for bill splitting. • Displays a list of added friends and existing groups for easy selection.

	• Restricts group creation and friend requests to registered users only. • Add roles & permissions (e.g., Admin can edit expenses, Members can only view). • Introduce temporary groups (auto-delete when all payments are settled).
Expense Management	• Allows users to log expenses manually by entering item details such as name, price, quantity, date, tags, and remarks. • Supports OCR receipt scanning to automatically extract and input expense details. • Enables users to review and edit OCR-extracted data before confirming. • Allows users to assign specific items to friends within an expense group. • Provides a history of expenses within each group for tracking and reference.
Split Logic & Debt Calculation	• Automates the bill-splitting process based on different logic (equal split, custom split, itemized split). • Calculates each user's share and determines who owes whom. • Updates balances dynamically when expenses are added or modified.
Payment Facilitation	• Enables users to save and share their QR PayNow code for quick payments. • Allows users to generate a pre-filled payment request message including the amount owed and QR code for non-registered users. • Displays the saved QR code for easy access when requesting payments.
Notifications & Alerts	• Sends reminders and updates when a new expense is added, or payments are due. • Notifies users when they are added to an expense group.

	• Alerts users when a payment request is sent or received. • Displays notification history for users to track past updates.
Reports & Analytics	• Provides an overview of spending patterns, total expenses, and outstanding debts. • Displays breakdowns of expenses by category, date, and group. • Helps users track payment history and pending debts.
IOU Manager	• Enables users to keep track of informal money exchanges between individuals, such as when a friend borrowed money from other friend. • Allows users to record key details, including who lent or borrowed, the amount, the date, and payment status. • Allows users to add IOU expense.

1.5.2 User Scope

The primary target users for SplitPay are university and college students who frequently engage in shared expenses such as group meals, accommodation, and subscriptions. These users often face difficulties managing and tracking shared costs due to reliance on manual methods and informal agreements. Secondary user groups include young working adults, housemates, and travel companions who experience similar challenges in managing group-related financial interactions.

1.5.3 Scope Limitation

This project focuses solely on developing a functional Android mobile application for the purpose of facilitating bill-splitting and shared expense tracking among users. Several limitations have been identified to ensure the project remains feasible within the allocated timeframe and resources:

1. The application will only be developed for the Android platform. iOS and web-based versions are beyond the scope of this project.

2. Real-time integration with banking systems or payment gateways is not included. Payment facilitation will be limited to displaying user-provided QR codes for manual settlement.
3. Optical Character Recognition (OCR) will be implemented to extract text from receipts; however, its accuracy may vary depending on receipt quality and format.
4. The system is designed primarily for online use. Offline functionality will be minimal and limited to local data storage.
5. Authentication is limited to basic user registration and login. Social login or advanced identity verification features will not be included.
6. The application is intended for personal finance management among peers and is not compliant with financial regulatory standards.
7. Security measures will be limited to standard practices such as session management and input validation; advanced security features such as end-to-end encryption and biometric access are excluded.

1.5.4 Expected Outcome

This project is expected to produce a fully functional Android mobile application, SplitPay, designed to streamline the process of bill-splitting and shared expense tracking. The application will address the key challenges identified in the problem background, including the lack of transparency, inefficiencies in manual calculations, and the absence of structured debt tracking in group financial interactions.

Through the integration of core features such as automatic bill-splitting, real-time debt tracking, receipt scanning via Optical Character Recognition (OCR), and QR code payment facilitation, and multiple split methods, users will be able to manage group expenses more effectively and simplify the user experience.

By eliminating the need for manual calculations and fragmented communication, SplitPay is expected to significantly reduce errors, disputes, and delays in payments among friends, families, and colleagues. Users will benefit from improved financial clarity, better organization of shared costs, and a more seamless method of settling debts. Ultimately, this project will provide a practical and accessible digital solution that enhances collaboration and accountability in everyday financial interactions.

1.6 Sustainable Developmental Goals (SDG)

The SplitPay project aligns with Sustainable Development Goal (SDG) 8, which aims to "promote sustained, inclusive and sustainable economic growth, full and productive employment and decent work for all." While the project does not directly create employment, it indirectly supports the SDG by promoting financial responsibility, accountability, and economic awareness among individuals, particularly students and young adults who frequently engage in shared financial activities.

By providing a structured system for managing shared expenses, SplitPay encourages users to adopt transparent, organized, and fair financial practices. These behaviors contribute to a more informed and financially literate population, which is essential for sustained and inclusive economic growth. Moreover, by reducing the friction and conflict associated with informal financial agreements, the app fosters healthier interpersonal dynamics and financial trust—key components for productive collaboration in both social and professional settings.

In essence, SplitPay empowers users to handle their financial obligations more effectively, promoting habits that support economic efficiency, fairness, and sustainability—all of which are foundational to achieving the broader goals of SDG 8.

CHAPTER 2:

LITERARURE REVIEW

2.1 Chapter Overview

This chapter presents a comprehensive literature review that supports the development of the SplitPay mobile application. It describes the fundamental concepts behind bill-splitting systems and the role such tools play in facilitating shared expense management. The chapter begins by outlining the concept of SplitPay and the key benefits it brings to users. This is followed by an discussion of the challenges associated with traditional methods of bill-splitting, which this project aims to address. The justification for selecting a mobile platform over other alternatives is also included. Finally, a comparative review of existing systems with similar functionality is presented to identify gaps and opportunities that SplitPay intends to fill. This literature review serves as the foundation for understanding the necessity and relevance of the proposed system within the context of current technological and social environments.

2.2 Bill Splitting

Bill splitting refers to the process of dividing a shared expense among multiple individuals, ensuring that each person contributes a fair portion of the total cost (Zhang, Zhao, & Xu, 2017). This practice is common in various group scenarios such as dining out with friends, group travel, shared subscriptions, event planning, and household expenses among roommates (Venmo, 2021; Belk, 2014). While seemingly straightforward, bill splitting can often become a source of confusion, miscommunication, and inconsistency, especially when done manually or without a structured system (Zhou & Rau, 2020). These issues can lead to interpersonal friction and financial ambiguity, highlighting the need for more reliable and user-friendly solutions (Zhang et al., 2017).

An example that has showed how bill-splitting affects the dynamic between the shared group expense can be observed from a study by The Economic Journal (Gneezy, Haruvy, & Yafe, 2004). The study investigates the psychological effects between having to split the bill between the grouped diners or having them to pay individually, and it was proven that when the bill is being split among the diners, the total of consumption are bigger. A study by Roos and Hahn (2019) showed that collaborative consumptions is influenced by 2 factors: egoistic motives – such as cost savings, resource usage efficiency, and community engagement. The other is normative motive – for instance, altruistic and biospheric intentions.

SplitPay is a mobile application developed to address these issues by offering an automated, user-friendly platform for managing and dividing shared expenses. Unlike traditional approaches that rely on manual calculations, messaging apps, or informal verbal agreements, SplitPay streamlines the entire process by enabling users to input expenses, assign payees, calculate individual shares, and track payments in one cohesive environment. The system supports various split methods, recurring bills, debt tracking, and receipt scanning via Optical Character Recognition (OCR), making it a comprehensive tool for personal and group financial management.

The role of SplitPay is to act as a digital intermediary that simplifies financial interactions among peers. It promotes transparency, reduces disputes, and eliminates the awkwardness of reminding others to repay debts. By automating complex calculations and maintaining clear records of who owes what, SplitPay fosters responsible financial behavior and builds trust among users. The application is especially beneficial for students, young adults, and housemates who frequently engage in shared financial activities but may lack the tools to manage them effectively. In essence, SplitPay serves not only as a bill-splitting tool but also as a lightweight financial collaboration platform designed for everyday convenience.

2.3 Current Method

In the absence of structured digital solutions, bill-splitting is often conducted using manual and informal methods. This typically involves verbal agreements, handwritten notes, messaging apps, or basic calculator tools to determine individual shares (Shen, 2019). While such approaches may work in simple scenarios, they frequently lead to complications as the number of participants or expense items increases (Anderson et al., 2018). One common problem is inaccuracy. Manual calculations are prone to human error, especially when splitting uneven costs, applying discounts, or dealing with multi-item bills (Böhringer & Roth, 2020). In group settings like dining or travel, it becomes difficult to determine who paid for what, how much each person owes, and whether any repayments have been completed. This leads to confusion and disputes, often resulting in awkward conversations or unpaid debts (Gibson, 2017).

Another issue is lack of transparency and record-keeping. Traditional methods do not provide a shared history of transactions. If someone forgets the agreed amount or disputes a share, there is rarely any objective reference to resolve the disagreement (Belk, 2014). This becomes more problematic in long-term arrangements like shared housing or recurring group subscriptions (Karlan et al., 2016). Additionally, coordination becomes time-consuming when split bills need to be calculated and communicated individually. Users must repeatedly remind others to make payments and manually follow up, creating unnecessary friction in social interactions. In large groups, this can be stressful and inefficient (Shen, 2019).

Overall, the traditional approach lacks structure, scalability, and convenience. It fails to support features like real-time debt tracking, smart categorization, and receipt scanning, which are now increasingly relevant in the age of digital payments (Anderson et al., 2018). SplitPay is designed to fill this gap by providing a reliable, transparent, and automated solution for managing shared expenses with minimal friction.

2.4 Findings from User Survey on Bill Splitting

Based on the data collected from the Google Form survey, the results provide a compelling quantitative foundation to support the problem definition of SplitPay, particularly in identifying both the relevance of bill-splitting practices in daily life and the inefficiencies in current approaches.

The demographic profile of the 42 respondents reveals that the majority are young adults, specifically aged 22 (33.3%), 26 (26.2%), and 23 (23.8%) (**Figure 1**), with the overwhelming majority being students (73.8%) and a smaller group comprising working adults (14.3%) (**Figure 2**). This demographic is particularly relevant as it represents a population that frequently engages in communal activities and shared financial responsibilities, but may not yet have established structured financial tools or habits. Their financial behaviors are likely to be informal and spontaneous, which aligns with the nature of situations where bill-splitting occurs.

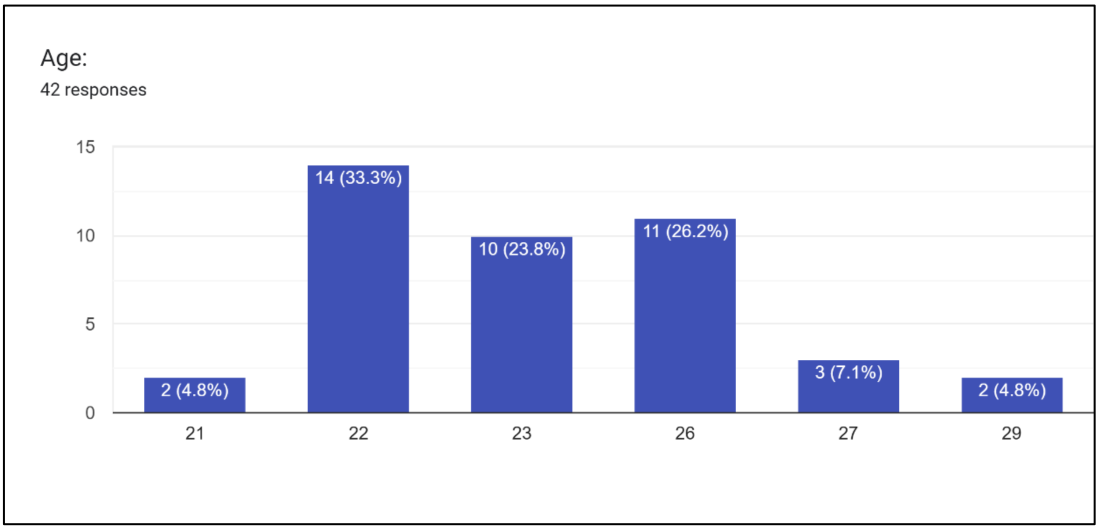


Figure 1: Age Distribution of Respondents

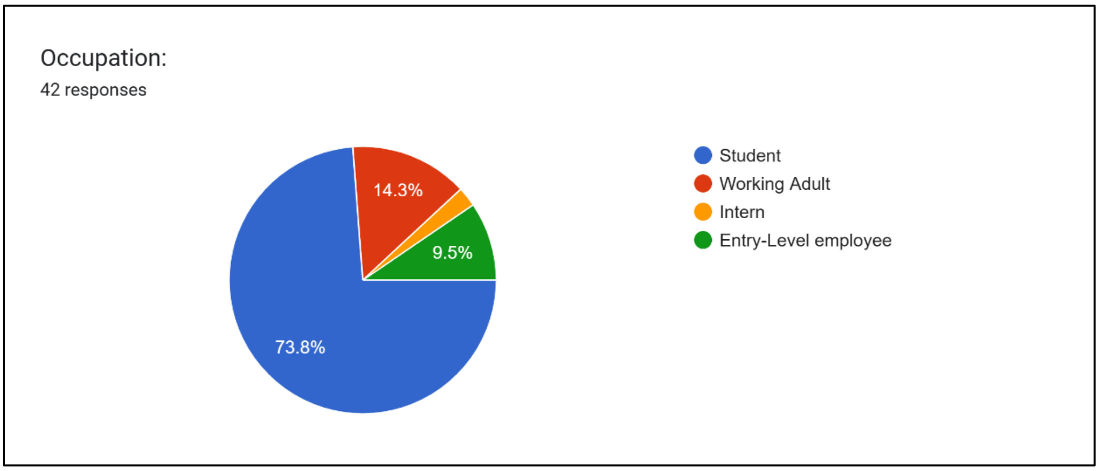


Figure 2: Occupation of Respondents

Regarding the frequency of shared expenses, 40.5% of respondents report needing to split costs multiple times a week, while another 21.4% do so at least weekly (**Figure 3**). This high frequency indicates that splitting bills is not an occasional issue but a recurring aspect of their lifestyle. Therefore, inefficiencies in this process could result in compounding frustrations or financial mismanagement over time.

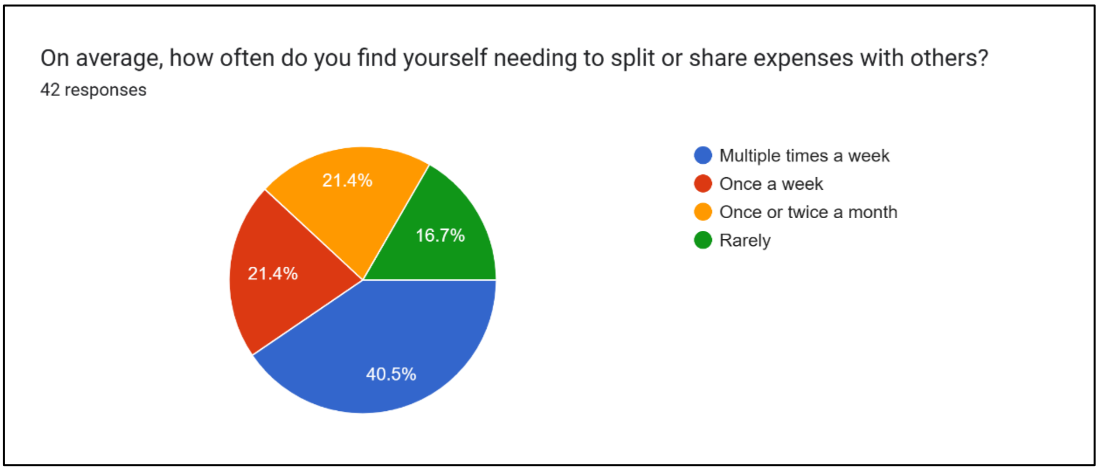
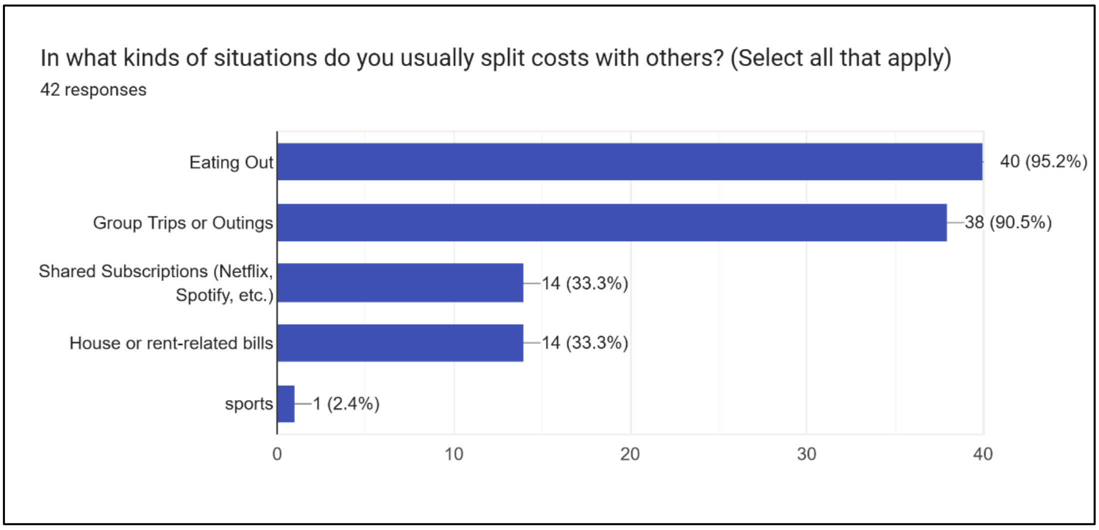


Figure 3: Frequency of Expense-Splitting Among Respondents

In exploring the contexts where shared spending occurs, the most commonly cited situations include eating out (95.2%) and group outings or trips (90.5%) (**Figure 4**). A third of the respondents also reported splitting costs for shared subscriptions

(33.3%) and rent or utility bills (33.3%). These responses suggest that shared expenses extend across both casual social interactions and more formal, recurring financial commitments, reinforcing the need for a flexible solution that can accommodate a range of scenarios.


In what kinds of situations do you usually split costs with others? (Select all that apply)

42 responses

Figure 4: Common Situations Where Respondents Split Bills

The methods currently used to manage bill-splitting are predominantly informal. A vast majority (83.3%) manually calculate and communicate cost breakdowns via messaging apps, while 95.2% rely on one person paying upfront and settling payments later. Only a small portion (7.1%) reported using specialized apps such as Splitwise (**Figure 5**). These findings point to a general lack of structured tools being adopted, possibly due to accessibility, usability, or awareness issues. This heavy reliance on manual methods introduces a significant risk of human error and lack of transparency, which directly correlates with the problems identified in the next question.

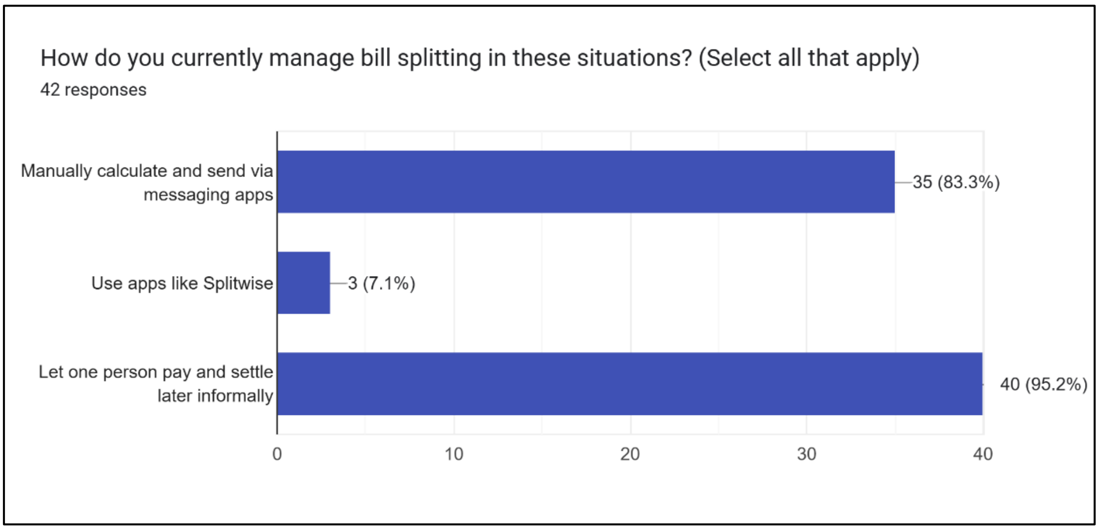


Figure 5: Current Methods Used for Bill Splitting

Respondents reported experiencing multiple common issues: delay in payments (76.2%), forgotten debts (66.7%), lack of a clear payment record (66.7%), and miscommunication (61.9%) were particularly prominent (**Figure 6**). Almost half had also experienced unfair splits (45.2%) or difficulty managing group expenses (47.6%). Only a marginal 2.4% reported no issues at all. These statistics clearly indicate that the existing bill-splitting practices are inadequate and frequently lead to confusion, tension, or financial discrepancies.

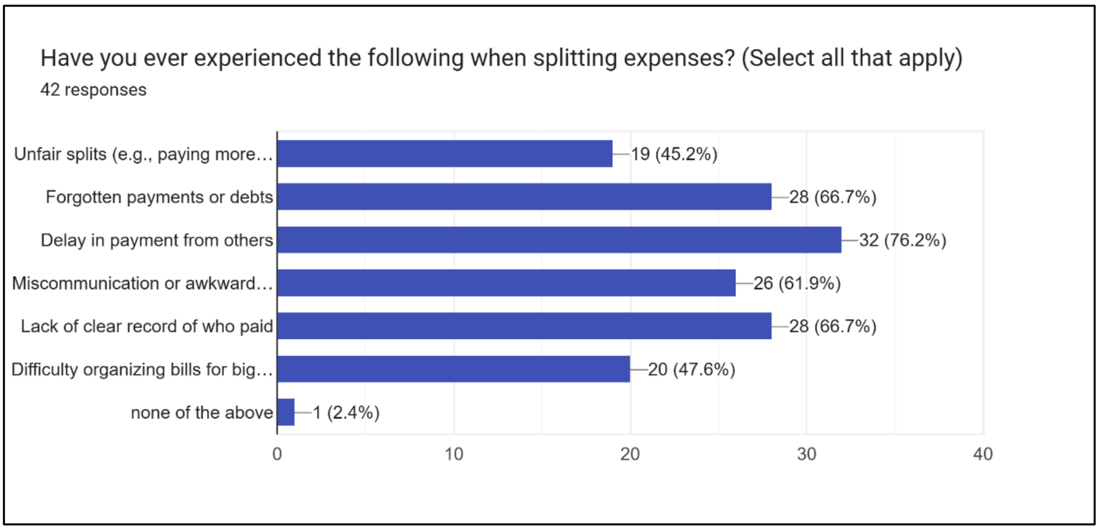


Figure 6: Problems Encountered When Splitting Bills

Lastly, when asked whether these challenges had influenced their social behavior, a majority admitted that it had — 45.2% responded "sometimes" and 26.2% said "yes," indicating that the inefficiencies and discomfort surrounding shared expenses can deter individuals from engaging in group activities (**Figure 7**). This suggests that the problem is not just logistical but also social, with implications on participation in communal settings.

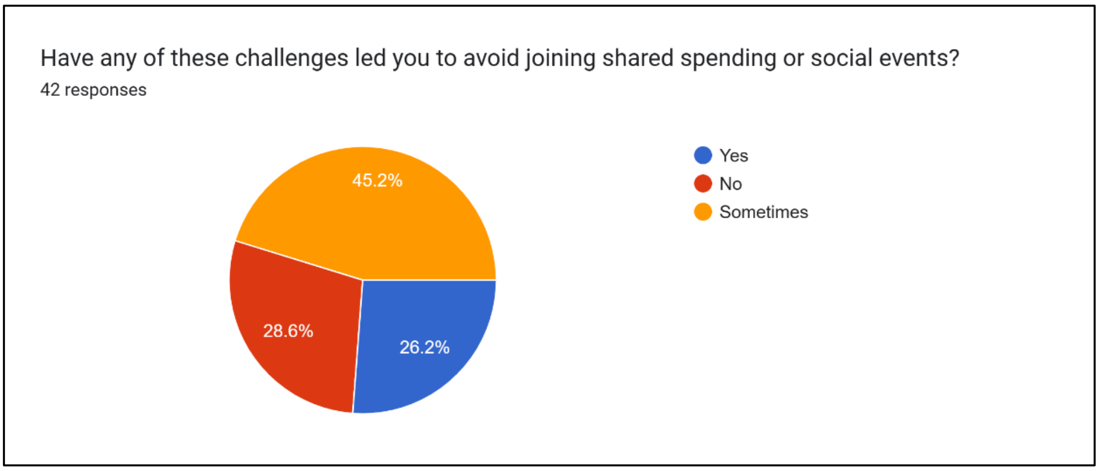


Figure 7: Impact of Expense-Splitting Issues on Social Participation

Overall, the survey data affirms the presence of a consistent, widespread problem in how shared expenses are managed, particularly among younger demographics. These findings support the need for a dedicated solution like SplitPay that simplifies, structures, and reduces the friction involved in bill-splitting scenarios.

2.4 Mobile Platform

The decision to develop SplitPay as a mobile application is based on the nature of the use cases it is intended to support. Bill-splitting commonly occurs in on-the-go scenarios such as dining out, shopping, traveling, or social gatherings — situations where users need quick and easy access to the app in real time. A mobile platform provides the most practical and efficient solution in these contexts.

With the widespread adoption of smartphones and the increasing reliance on mobile apps for daily financial tasks, developing SplitPay as a mobile app ensures maximum accessibility and convenience. Users can input expenses, scan receipts, and settle debts via QR code all within a few taps, without needing a computer. This aligns with user expectations in the current digital landscape, where mobile-first interactions are standard.

Moreover, mobile platforms enable integration with native features such as the camera for OCR (Optical Character Recognition), notifications for payment reminders, and e-wallet apps for instant settlements — all of which enhance the utility and responsiveness of the system. These features are either limited or significantly less convenient on desktop and web platforms.

The mobile approach also supports real-time collaboration, as users in a group can receive immediate updates, track contributions live, and interact within the app as financial events unfold. This responsiveness is essential for resolving expenses on the spot and avoiding delays in payment coordination.

In contrast, a web or desktop platform may introduce accessibility limitations, as users may not always have access to a laptop or desktop during group events. Such platforms are also less effective in leveraging real-time tools like OCR scanning or QR payment functionalities, which are integral to SplitPay's features.

Therefore, the mobile platform is the most suitable and strategic choice for SplitPay, ensuring seamless usability, maximum feature integration, and alignment with the app’s core mission to simplify shared expense management in real-world social contexts.

2.5 Review of Existing Systems

As digital financial tools become increasingly integrated into daily life, several mobile applications have emerged to assist users with bill-splitting and expense management. These systems aim to simplify the process of tracking shared expenses among groups and reduce the friction associated with manual calculations and informal tracking methods.

This section reviews a selection of widely used bill-splitting applications that share conceptual similarities with the proposed project, SplitPay. The purpose of this review is to identify the strengths, limitations, and design patterns of existing systems, which will serve as a foundation for defining the unique value proposition of SplitPay.

Three established systems — Splitwise, Tricount, and Settle Up — are examined in terms of their features, user experience, technical limitations, and suitability for various user scenarios. This analysis will highlight both the current industry standards and the opportunities for innovation that SplitPay seeks to address.

2.5.1 Splitwise

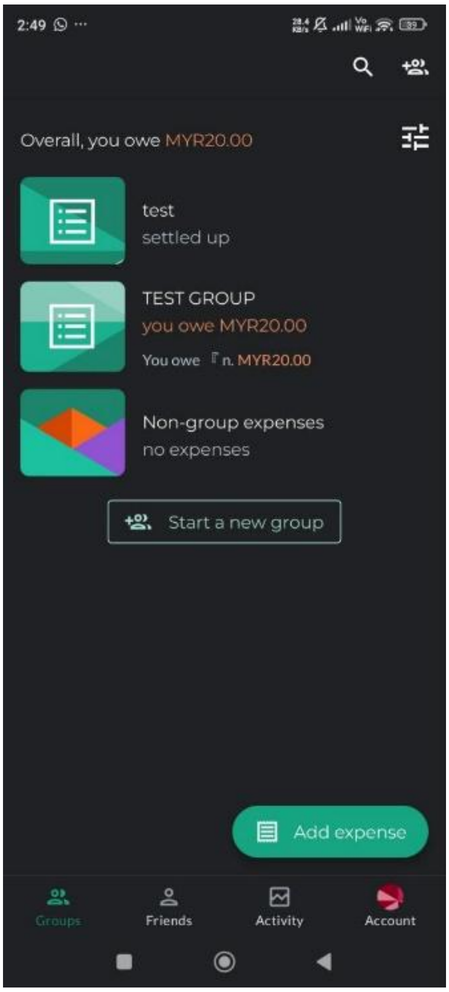
Splitwise is one of the most widely recognized mobile and web-based applications for managing shared expenses among friends, roommates, travel groups, and families. Its core functionality revolves around logging expenses, creating grouped and non-grouped expenses, automatically calculating who owes whom, and allowing users to settle debts through linked payment methods: venmo and paypal (for the United States only), Paytm (India only) or manually.

One of Splitwise's major strengths lies in its user-friendly interface, which allows for quick entry of expenses and supports various split methods (equal split, percentage-based, share-based). It also provides a running balance for each group member and a summary of debts, which helps users understand their financial position in the group at any time.

Additionally, Splitwise offers group management, recurring bill tracking, and support for multiple currencies — making it suitable for international travel or cross-border payments. It also integrates with third-party services like PayPal and Venmo for simplified settlements (where supported).

However, Splitwise has notable limitations. While it supports receipt attachment, it lacks native OCR (Optical Character Recognition), meaning users must manually input each expense item. The free version is limited in functionality, with advanced features such as detailed charts, search capabilities, and currency conversions being locked behind a paid "Pro" subscription. It also limits the daily expense addition. It is primarily focused on general expense tracking, and does not differentiate between casual group spending and structured loans or recurring debts (e.g., personal loans between two users).

In the context of SplitPay, Splitwise serves as a foundational reference for effective group expense tracking. However, SplitPay aims to go beyond Splitwise by incorporating smart OCR receipt scanning, enhanced loan tracking, and QR-based payment facilitation, all in a system optimized for the local payment ecosystem.



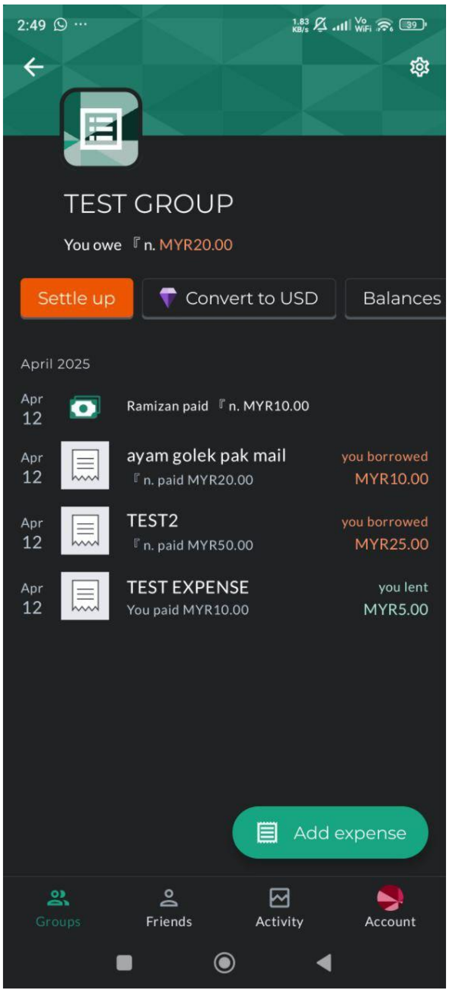
Overall, you owe MYR20.00

test
settled up

TEST GROUP
you owe MYR20.00
You owe 「n. MYR20.00

Non-group expenses
no expenses

Figure 8: Group Expense Page on Splitwise



TEST GROUP
You owe 「n. MYR20.00

April 2025

Apr 12	Ramizan paid 「n. MYR10.00	
Apr 12	ayam golek pak mail 「n. paid MYR20.00	you borrowed MYR10.00
Apr 12	TEST2 「n. paid MYR50.00	you borrowed MYR25.00
Apr 12	TEST EXPENSE You paid MYR10.00	you lent MYR5.00

Figure 9: Group Expense Page on Splitwise

2.5.2 Tricount

Tricount is another popular bill-splitting application designed to simplify shared expenses in group contexts such as trips, events, and household budgeting. It allows users to create a shared "tricount" where all group members can add expenses and view balances in real-time.

The strength of Tricount lies in its simplicity and collaboration features. Each group member can view and edit shared expenses, making it ideal for informal group scenarios. Users can split costs equally or unequally, assign custom weights to participants, and track partial payments. Tricount also supports offline data entry, which is useful when traveling without internet access.

Unlike Splitwise, Tricount emphasizes a summary-based approach, focusing on net balances between participants rather than detailed transaction logs. It provides a clear "who owes whom" breakdown and even offers suggestions to minimize the number of repayments required — a feature many users find efficient.

However, Tricount has its own set of limitations. It lacks integrated payment options such as QR payments or e-wallet support. Users must settle payments manually outside the app. There is no OCR capability for digitizing receipt data, requiring users to manually input every line item. The application does not support personal loans or recurring expenses, limiting its utility beyond one-off group expenses.

While useful for travel and group events, its scope is relatively narrow, and analytics or financial summaries are minimal. Tricount informs the development of SplitPay by showcasing how a lightweight, intuitive interface can facilitate expense sharing. However, SplitPay intends to expand on this model by introducing automated expense recognition, debt categorization (e.g., bills vs. loans), and payment facilitation mechanisms, making it more suitable for both casual and structured financial interactions.

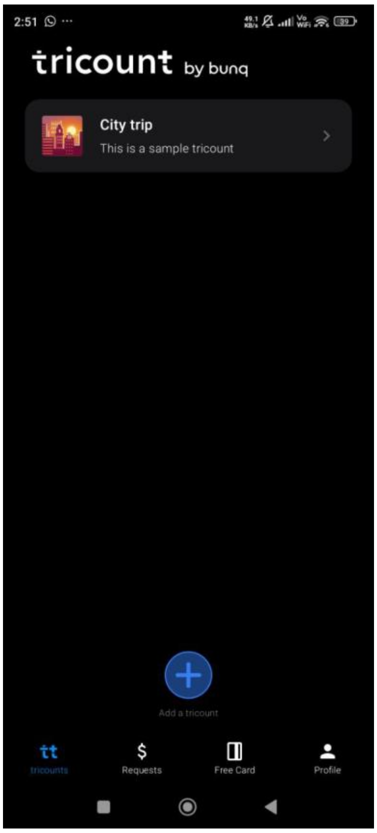


Figure 8: Expense Group Page on Tricount

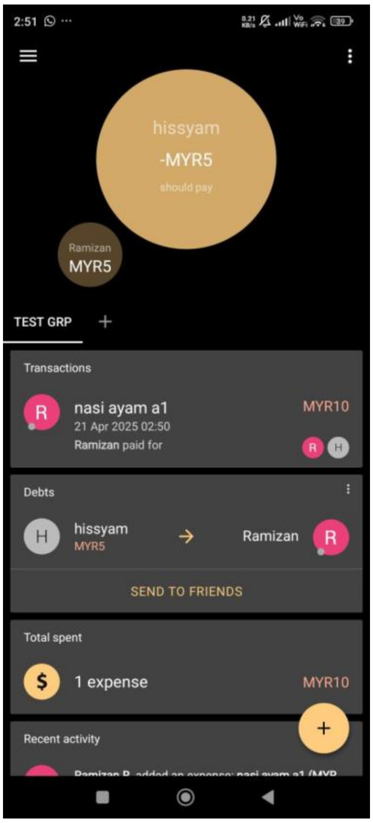
2.5.3 Settle Up

Settle Up is a cross-platform expense-sharing application designed for both short-term and long-term group usage. Its design philosophy centers around helping users track and reconcile shared spending efficiently, with a strong focus on clarity of debt relationships and flexible expense input methods.

Settle Up supports various types of splitting — equally, by percentage, by shares, or even custom amounts — and allows for multiple payers and split types per transaction, making it suitable for more complex bill scenarios. It offers group management, supports offline mode, and enables data synchronization across devices, which is particularly useful for groups traveling or managing shared budgets over time.

In addition, Settle Up includes features such as, exporting reports in CSV or PDF formats, expense filtering and search, and optional PIN code protection for privacy. However, Settle Up is not without limitations. Like many of its competitors, it does not offer OCR capabilities, which makes receipt itemization a manual process. Although the app supports multiple currencies and conversion, it lacks integration with localized payment methods like QR-based e-wallets, bank transfers, or regional fintech APIs. The system focuses more on tracking rather than automation, offering no features like payment reminders.

Settle Up is a solid reference point for SplitPay in terms of handling complex group splits and offering detailed exports. However, SplitPay seeks to address broader user needs by incorporating automated receipt scanning, local QR payment compatibility, and smart categorization of expenses to improve financial clarity and user convenience.



hissyam
-MYR5
should pay

Ramizan
MYR5

Transactions

nasi ayam a1 MYR10
21 Apr 2025 02:50
Ramizan paid for

Debts

hissyam MYR5 → Ramizan

Total spent

1 expense MYR10

Recent activity

Ramizan R. added an expense: nasi ayam a1 (MYR

Figure 9: Example of Group Expense in Settle Up

2.6 Comparison Summary

Table 2: Comparison Between the Reviewed Systems and SplitPay functionalities

Feature	Splitwise	Tricount	Settle Up	SplitPay
Platform	Web, iOS, Android	Web, iOS, Android	Web, iOS, Android	Android
Expense Splitting Options	Equal, shares, percent	Equal, weighted	Equal, percent, shares, custom	Equal, percent, shares, custom
OCR Receipt Scanning	Available in "Pro" version	Not Available	Not Available	Available
Recurring Expense Support	Available in "Pro" version	Not Available	Not Available	Available
Loan Tracking	Not Available	Not Available	Not Available	Available
Payment Integration	Limited to the US (venmo, paypal) and India (Paytm)	Not Available	Not Available	DuitNow QR Code
Group Management	Available	Available	Available	Available
Analytics & Reports	Available in "Pro" version	Limited availability	Available in basic	Available
Offline Mode	Available	Available	Available	Available

2.7 Gap Analysis and Justification for SplitPay

After evaluating the existing systems — Splitwise, Tricount, and Settle Up — several key functional and usability gaps have been identified. While each application offers valuable features for tracking shared expenses, they largely rely on manual data input and do not cater to localized payment ecosystems, especially in regions where QR codes and e-wallets are the norm.

The major gaps included:

1. **Manual Receipt Entry**

All reviewed systems require users to manually input expense details, which is time-consuming and error-prone. No native support exists for automated OCR-based receipt scanning, despite its relevance for accurate and quick bill entry.

2. **Lack of Structured Loan Handling**

None of the systems provide a formal way to track non-group loans (e.g., borrowing money from a friend), which are common in personal and social contexts. This limitation reduces the system’s relevance for managing recurring debts or informal lending.

3. **No Support for QR-Based Payment Systems**

Most solutions either have limited third-party payment integration (only for select countries), or require users to settle manually. In regions like Malaysia or Southeast Asia, where QR code payments and e-wallets are dominant, this is a major gap in accessibility and convenience.

4. **Premium Feature Walls**

Core features such as analytics, data exports, and full features availibility are often locked behind paid plans, making them less accessible to students or budget-conscious users.

SplitPay is designed to address these shortcomings by introducing OCR-enhanced receipt scanning to reduce manual entry, Loan management capabilities alongside group expense tracking, Integration with local QR code system, and a fully free and open-access platform aimed at inclusivity for students, young adults, and casual users.

CHAPTER 3:

METHODOLOGY

3.1 Chapter Overview

This chapter outlines the systematic approach used to develop the SplitPay mobile application. The project adopts the Waterfall software development methodology, a linear and sequential approach that progresses through defined stages: Planning, Analysis, Design, Implementation, and Testing. This method ensures thorough documentation and clarity at each of the developmental phase.

3.2 Project Methodology

The development of the SplitPay system adopts the Waterfall model, a traditional software development methodology known for its structured, sequential approach. This model divides the development process into distinct phases — Planning, Analysis, Design, Implementation, and Testing — each of which must be completed before moving on to the next. The linear flow of the Waterfall model makes it particularly effective for projects with clearly defined requirements and limited scope for changes once development begins.

The Waterfall methodology is widely used in academic and industry settings for its clarity, discipline, and simplicity. The progression from one phase to another ensures that every aspect of the system is systematically addressed and adequately documented. This structure is especially beneficial for small-scale projects such as this, where resource constraints and limited team size demand a well-organized development workflow.

In contrast to agile or iterative models, which encourage flexibility and repeated cycles of refinement, the Waterfall model is more suitable for this project because the requirements are largely fixed and well-understood from the beginning. This ensures that all necessary considerations are addressed early in the process and incorporated into the planning and design stages.

Furthermore, the Waterfall model aligns well with academic expectations, where clear documentation, well-defined deliverables, and systematic progress are often required at various checkpoints. With its emphasis on linear development and phase completion, the methodology helps the developer remain organized, efficient, and focused — all while producing high-quality documentation and a maintainable final product.

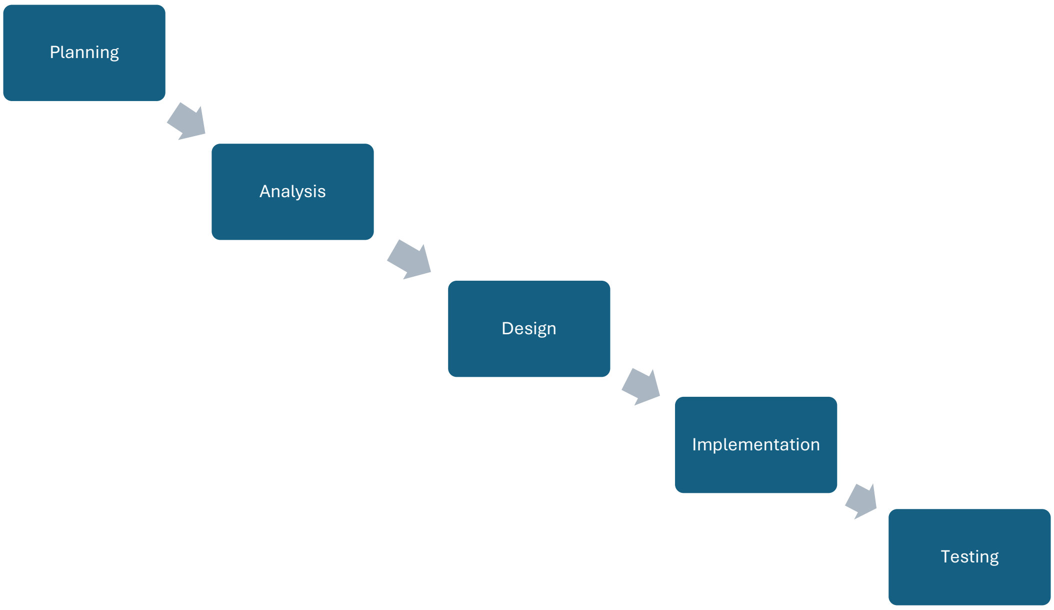


Figure 10: Waterfall Model

3.3 Planning Phase

The planning phase represents the foundation upon which the entire SplitPay project is built. This phase began with the identification of a real-world problem through informal consultations with peers and fellow students. In casual discussions with colleagues, a recurring issue was highlighted — the difficulty and awkwardness involved in splitting bills manually in group settings. One particular scenario described by a friend involved the lack of a reliable tool to fairly and efficiently divide expenses among friends after social outings. The reliance on manual calculations, often conducted through messaging apps or mental math, not only introduced frequent errors but also led to delays, misunderstandings, and discomfort in requesting payment from others. This anecdotal evidence aligned closely with broader pain points commonly observed in social expense sharing, thus motivating the exploration of a potential solution.

To validate that this issue was not anecdotal or isolated, a structured online survey was distributed to a broader group of potential users. The aim of this survey was to assess the relevance, frequency, and impact of the problem among target users such as university students and young working adults. The responses confirmed that the issue was widespread and that many respondents would welcome a system that could streamline the bill-splitting process and reduce the complications associated with it.

In parallel with problem validation, a review of existing systems was conducted. Several applications that offer similar services — such as Splitwise, Tricount, and Settle Up — were analyzed to understand their strengths and limitations. This comparative review helped identify key gaps in the current market, such as limited support for receipt-based expense entry, inflexible splitting methods, or inadequate support for tracking personal loans and debt settlements. This gap analysis directly informed the conceptual direction of SplitPay, ensuring the project is not a redundant solution but one that meaningfully addresses unmet user needs and offers improved usability and feature scope.

Having clearly defined the problem and verified its significance, the project proceeded with the formulation of its objectives, scope, and stakeholder roles, ensuring a structured and measurable development approach. This phase also involved the drafting of a comprehensive project plan, which includes a Gantt chart outlining the project timeline, key milestones, and deliverables. These planning tools serve as a roadmap to monitor progress and maintain accountability throughout the development cycle, ensuring each phase contributes meaningfully to the intended outcomes.

By the end of the planning phase, the project has established a strong foundation — with a validated problem, a clearly articulated set of objectives, an understanding of the stakeholder environment, and a project timeline that facilitates structured development. This thorough groundwork ensures that subsequent phases in the Waterfall model can proceed with clarity, purpose, and alignment to the project's core goal: to deliver a usable, efficient, and relevant bill-splitting solution through a well-managed development process.

3.4 Analysis Phase

The analysis phase serves as a critical step in establishing a thorough understanding of what the system must achieve in order to effectively address the real-world challenges identified during the planning stage. In this phase, efforts are concentrated on analysing user needs, outlining precise functional and non-functional requirements, and identifying any constraints that may affect the development of the system. The success of this phase is pivotal to the system's eventual usability and relevance, ensuring that development is grounded in actual user expectations and behaviour patterns.

To carry out this investigation, a structured online survey was designed and distributed via Google Forms. The survey was shared among university peers, colleagues, and social circles in order to reach a wide range of respondents who fall within the intended user demographic for SplitPay. The design of the questionnaire was carefully segmented to ensure logical flow and clarity. The survey sections included: respondent demographics, current practices in handling bill-splitting, specific issues encountered with manual or informal methods, expectations for a digital solution like SplitPay, UI/UX design preferences, and an open-ended section for additional feedback or suggestions.

The responses gathered were then analysed to extract meaningful insights into how users currently manage shared expenses and what functionalities they would expect from a mobile application designed to solve these issues. This aligns directly with Objective 1 of the project: To identify the requirements of the proposed system using surveys and interviews with potential users. Through this feedback, it became possible to validate the existence and impact of the problem, affirm the relevance of the proposed solution, and prioritize which features are most essential for the initial version of the application.

Overall, the analysis phase ensures that the development of SplitPay remains grounded in practical user expectations. It provides a solid base for the design phase by clarifying what the system must accomplish and by whom it will be used. In doing so, this phase reinforces the project's user-centred development approach and ensures that future work will remain aligned with the needs identified from the study.

3.5 Design phase

The design phase is a crucial part of the development process, where the requirements identified during the analysis phase are translated into a clear and structured blueprint for the system's implementation. This phase involves the formulation of the system's architecture, the development of a well-structured database schema, and the construction of intuitive user interface designs—each of which plays a pivotal role in ensuring that the proposed system, SplitPay, will be both functional and user-friendly.

The system design process begins with the creation of a Context Diagram, which outlines the overall boundaries of SplitPay and illustrates how it interacts with external entities such as the users. This provides a high-level understanding of the system's scope and its interactions with the environment. Subsequently, Data Flow Diagrams (DFDs) are developed to represent the movement of data throughout the system. These diagrams help to visualise how data is input, processed, and output within various modules of SplitPay, and how different modules such as group management, bill logging, and debt tracking interconnect to support the overall functionality.

Parallel to system design, the database design is constructed using Entity Relationship Diagrams (ERDs). These diagrams define the relationships between entities such as Users, Groups, Bills, Transactions, and Items. Key aspects such as attributes, cardinality, data types, and primary-foreign key relationships are carefully considered to ensure data integrity and efficient access. A sound database structure is essential for the performance and reliability of core operations like bill splitting, debt calculations, and user history tracking, particularly as the app scales.

In tandem with architectural planning, the User Interface (UI) design is also developed during this phase. Mockups and wireframes are designed using tools such as Figma to model the layout, navigation, and interactivity of the mobile application. The design emphasizes simplicity and clarity to accommodate the general user base, especially students and young adults who demand straightforward and responsive experiences. User flow, accessibility considerations, and aesthetic consistency are prioritized to ensure that users can navigate the app intuitively. Feedback gathered during the analysis phase also influences UI decisions, especially regarding preferences on theme, iconography, layout, and ease of use.

By the conclusion of the design phase, SplitPay will have a finalized system design specification, a normalized and efficient database schema, and a complete set of UI prototypes. These outputs collectively serve as a concrete reference for the implementation phase, ensuring that the system is built in alignment with both user expectations and the project's technical objectives. The design phase also supports Objective 2, which is to develop the system methodically using the waterfall model, by preparing all the necessary technical groundwork required to begin system construction with minimal ambiguity.

3.6 Implementation Phase

The implementation phase marks the transition from design to actual development, during which all previously defined requirements and blueprints are translated into a working application. For this project, implementation is executed in a modular fashion, aligned with the system's architecture and the waterfall methodology. Each module—such as user authentication, group management, expense tracking, OCR scanning, and reporting—is developed individually and is integrated incrementally. This approach is used to ensure maintainability, enable isolated testing of functionalities, and adhere to the project's second objective: to develop SplitPay using the waterfall model.

The system is developed as a native Android mobile application, given the platform's accessibility and alignment with the preferences of the target user demographic. The Integrated Development Environment (IDE) used is Android

Studio, recognized as the official and most robust development environment for Android. Kotlin, a modern, expressive, and officially supported programming language for Android, is employed for its concise syntax and enhanced safety features.

To handle the user interface (UI) layer, Jetpack Compose is utilized—a modern declarative UI toolkit that simplifies UI development and allows for more reactive, readable, and maintainable code. Screen navigation within the app is managed using Navigation Compose, which enables seamless transitions between features such as the dashboard, bill creation, and group overview.

The backend infrastructure of SplitPay is implemented using a serverless architecture powered by Firebase, which provides seamless integration with Android and significantly accelerates development by reducing the overhead of traditional server-side setup. Firebase Authentication is used for secure login and registration, while Cloud Firestore, a scalable NoSQL cloud-hosted database, is adopted as the primary datastore for user data, bills, items, transactions, and groups. For storing image receipts—particularly relevant to the OCR functionality—Firebase Storage is utilized.

SplitPay's OCR capability is achieved using Firebase ML Kit, specifically the on-device text recognition API, which supports receipt scanning even in offline conditions. This functionality directly supports the application's goal of eliminating manual data entry when splitting bills, thereby improving accuracy and user convenience.

Automation and background tasks such as scheduling reminders for unpaid bills or recurring payments are handled via Firebase Cloud Functions, further enhancing the system's interactivity and automation. Additionally, Firebase Cloud Messaging is employed to send real-time push notifications, including alerts about due payments or bill creation, thus supporting user engagement and usability evaluation in line with the third objective.

To ensure consistency and intuitive design, all user interface components are designed in accordance with Material Design Guidelines. Prototypes and design mockups are created using Figma, an industry-standard tool for collaborative UI design. This design-first approach is adopted to ensure that each user interaction is intuitive and aligned with expectations gathered during the analysis phase.

Finally, to maintain project discipline and track progress, version control is implemented using Git in combination with GitHub, while Word is used as the primary platform for documentation, planning, and task management. Milestones and development tasks are organized to match the deliverables defined during the planning phase.

Through this structured and modular implementation process, the project not only is aligned with the principles of the waterfall methodology but also is ensured to remain consistent with the system’s defined scope and objectives. Each module is independently developed, integrated, and validated, thereby contributing to a stable, responsive, and user-centered mobile application.

3.7 Testing phase

This phase encompasses the validation and refinement of the system to ensure it meets all defined requirements, functions correctly under various conditions, and provides a satisfactory user experience. It also covers ongoing maintenance strategies and the preparation of system documentation to support future scalability, usability, and reliability. All these activities align directly with the third objective of this project, which is to evaluate the system's functionality and usability through comprehensive testing and user feedback mechanisms.

The testing process is structured across four levels: unit testing, integration testing, system testing, and usability testing. Unit testing is performed first to validate individual components such as data processing functions, user authentication logic, and bill calculation modules. This ensures that each module behaves as intended in isolation.

Following this, integration testing is carried out to verify that inter-module communication works correctly. For example, tests are conducted to ensure that bills created in the expense module are correctly recorded in the database, reflected in the debt tracking interface, and trigger appropriate push notifications. This phase helps confirm that all modules interact seamlessly within the application's ecosystem.

Subsequently, system testing is performed to assess the end-to-end behavior of the application in a simulated real-world environment. This involves executing complete user flows such as creating a group, scanning a receipt via OCR, assigning expenses, and QR Code. The goal is to ensure that the system meets its functional and non-functional requirements as a whole.

The final stage, usability testing, is critical in evaluating how effectively users can interact with the system and whether it fulfills the users' expectations and needs, as explored in the earlier analysis phase. Users are invited to interact with a working version of the app, after which they provide feedback through survey forms and verbal interviews. These insights are instrumental in identifying points of confusion, interface friction, or any unmet user expectations. The results of this testing stage not only validate the user-centric design of the system but also guide any final refinements prior to deployment.

The collected feedback informs iterative updates to improve the UI/UX design, integrate frequently requested features, and revise or deprecate components that may cause confusion or reduce usability. This responsive maintenance approach ensures that the system evolves in tandem with user needs and continues to address the original problem identified during the planning phase.

Lastly, comprehensive system documentation is prepared throughout the development and testing stages. This includes documentation for code structure, API references, database schema, installation and appropriate guides. These documents not only support debugging and future enhancements but also ensure the project remains sustainable and transferable should additional developers or stakeholders become involved later.

3.8 Hardware and Software Requirements

The development of SplitPay involves a combination of both hardware and software tools to support system design, development, testing, and project management. Below are the essential resources utilized throughout the development lifecycle:

Table 3: Hardware Requirements

Hardware	Description
Laptop	Used for coding, compiling, testing, and managing the entire development workflow.
Smartphone	Essential for real-device testing to ensure app performance and responsiveness in real-world usage.

Table 4: Software Requirements

Software	Description
Android Studio	Official IDE for Android development; provides all tools necessary to build, test, and debug the Kotlin-based mobile app.
Firebase	serves as the backend for SplitPay, providing user authentication, real-time database, cloud storage, and app performance monitoring.
ML Kit (OCR)	Provides on-device optical character recognition, enabling accurate receipt scanning even offline.
Figma	Used for designing and prototyping user interfaces; facilitates feedback and design iteration.
Git + GitHub	Manages source code versioning and facilitates collaboration and code tracking.
Word	Centralizes documentation, project planning, retrospectives, and task management.

3.9 Summary

The SplitPay system is developed using the Waterfall model, a linear and structured software development methodology that progresses through defined phases—

Planning, Analysis, Design, Implementation, and Testing. This model suits the project due to its fixed requirements and the need for clear documentation, which aligns with academic expectations. The planning phase began by identifying a real-world problem—manual bill splitting—and validating it through informal discussions and a structured online survey targeting university students and young adults. A comparative review of existing apps like Splitwise and Tricount revealed key limitations, guiding the conceptual design of SplitPay to address gaps such as receipt scanning, personal loan tracking, and flexible expense splitting. These insights led to the formation of clear project objectives, a defined scope, and a development timeline via a Gantt chart.

The analysis phase translated user feedback into functional and non-functional requirements, which directly informed the system's design. The design phase produced a complete blueprint, including context diagrams, data flow diagrams, and a normalized database schema. User interfaces were prototyped using Figma, guided by user preferences. Implementation was done using Android Studio with Kotlin and Jetpack Compose, supported by Firebase for authentication, database, storage, and machine learning (OCR). Testing was performed across four levels—unit, integration, system, and usability—to validate system functionality and user satisfaction. Feedback from testing informed refinements, and comprehensive documentation was created to ensure maintainability and scalability. The process ensured that SplitPay was developed systematically, efficiently, and with a strong user-centered focus.

CHAPTER 4:

SYSTEM ANALYSIS & DESIGN

4.1 Chapter Overview

System analysis involves understanding what the system must do to meet user needs, mainly through requirement elicitation. For this project, an online survey was conducted to gather insights from target users of the SplitPay app, helping identify both functional and non-functional requirements. These requirements then guided the system design phase, which focused on planning the system's structure and components. Design tools such as context diagrams, data flow diagrams, UI/UX prototypes, and an ERD were used to ensure the system is organized, user-friendly, and ready for development.

4.2 System Analysis

System analysis refers to the process of understanding what the system must do to meet the needs and expectations of its stakeholders. For this project, the main activity in system analysis is requirement elicitation, which involves identifying, gathering, and clarifying what users expect the system to do. This was done using an online survey aimed at the target users of the SplitPay app. The responses collected provided valuable insights into users' preferences and pain points when splitting bills. These insights were then analyzed and structured into the system's functional and non-functional requirements, which form the foundation for the system's design and development.

4.2.1 Requirement Elicitation Methodology – Survey, Interview

The requirement elicitation process was carried out to understand the real-world challenges, preferences, and expectations of potential users regarding bill-splitting, which serves as the foundation for developing the SplitPay system. Two primary methods were employed: an online survey and semi-structured interviews. These methods were selected to ensure both quantitative breadth and qualitative depth in gathering user insights, in alignment with the project's first objective.

The online survey was conducted using Google Forms and received responses from 42 participants, primarily comprising university students and working adults aged 21 to 30. The survey was structured into five sections: (1) Demographics, (2) Current Practices and Problems, (3) Expectations and Feature Needs, (4) UI/UX Design Insights, and (5) Open Comments. It employed a combination of multiple-choice questions, Likert scale ratings, and open-ended prompts. These questions aimed to uncover common pain points in current bill-splitting practices, gather feedback on existing apps (such as Splitwise), and identify desirable features for a more efficient and user-friendly alternative. The use of this survey method allowed the collection of statistically significant trends and patterns in user behavior and preferences.

In addition to the survey, three interview sessions were conducted to simulate in-depth conversations with targeted user personas. These included a student who shares rental expenses, a working adult who frequently split bill friends, and a classmate. The interviews were designed to explore the participants' lived experiences with bill-splitting, their frustrations with current tools or manual methods, and their expectations for functionality, usability, and accessibility in a proposed solution. These interviews followed a semi-structured format, that combines both pre-planned questions and open conversation, to allow flexibility while still covering key thematic areas relevant to the system's scope.

These combined elicitation methods provided a comprehensive foundation for defining the system's functional and non-functional requirements, ensuring that the final solution aligns closely with the actual needs and expectations of the target user base. The results from both the survey and interviews are presented in detail in the subsequent section.

4.2.2 Findings from Requirement Elicitation -Survey

The survey results provided valuable insights into user expectations, preferred features, and interface preferences for a mobile application that simplifies bill-splitting and payments. This section presents findings from three key areas of the questionnaire: Expectations and Feature Needs, UI/UX Design Insights, and Open Comments. For reference, the results in illustration of diagrams can be found in the Appendix section.

Expectations and Feature Needs

The majority of respondents demonstrated a strong interest in a bill-splitting mobile application. Out of 43 responses, 74.4% indicated they would find such an app helpful, while the remaining 25.6% expressed openness to the idea, suggesting a favorable outlook towards adoption. Notably, no respondents considered the app unhelpful, reinforcing the system's relevance and necessity.

In terms of desired functionalities, the most highly requested features included Payment reminders (86%), automatic splitting, receipt scanning, and payment link generation (each 81.4%), and item tagging and personal loan tracking (74.4%). While expense history reports (58.1%) and recurring shared expenses tracking (53.5%) were selected less frequently, over half of respondents still expressed interest in these functionalities, indicating their supplementary value. When asked about preferred payment methods, 100% of respondents selected QR code integration, specifically referencing Malaysia's DuitNow national QR standard. Additionally, 53.5% favored recording manual cash settlements, suggesting that the app should accommodate non-digital payment scenarios for tracking purposes.

These results clearly indicate that users are not only interested in a digital bill-splitting solution but also expect it to integrate seamlessly with existing Malaysian payment systems while providing smart automation features and record-keeping capabilities.

UI/UX Design Insights

User interface and usability preferences were also captured in the survey. Over 53.5% of respondents preferred a simple and minimalistic interface, while 44.2% preferred a balanced design—neither too simple nor overly detailed. This suggests that the interface should focus on clarity and ease of use, avoiding cluttered layouts or overly complex workflows. Respondents were also asked to rate the importance of various app aspects on a scale of 1 to 5. The highest-rated aspects, receiving the majority of "5 - Very Important" responses, were Ease of use, Privacy and security, Fast performance, and Clean interface. QR code integration and offline functionality were also rated highly, although slightly more responses were distributed across the "4" and "3" levels for these features. Nevertheless, this indicates that QR support is seen as critical, while offline support is appreciated but not an absolute requirement.

Overall, these findings highlight the importance of intuitive navigation, strong performance, and trustworthy data handling as key design pillars for the app. Simplicity and speed must be prioritized to ensure high user satisfaction and broad adoption.

Open Comments and Qualitative Insights

Open-ended responses provided further context on user frustrations, desires, and practical challenges with current bill-splitting practices. Approximately 25% of respondents had never used an expense-splitting app, while those who had shared the following recurring pain points: Confusing or cluttered interfaces, Advertisement overload, Limited functionality, particularly in tracking shared expenses or customizing item splits. When asked about must-have features, automatic expense splitting, itemized tagging, and QR payment integration were the most frequently mentioned. These features are seen as critical to removing the friction of manual calculation and follow-up.

In terms of broader expectations, users emphasized:

- Automation: Integration of receipt scanning, smart item tagging, and tax/service charge calculation to avoid disputes and manual entry.
- Payment management: Inclusion of reminders, clear transaction history, and QR payments for easier settlements.

- Ease of adoption: A clean UI and the ability to share bill information via messaging platforms to encourage group participation.
- Trust and privacy: Several respondents highlighted concerns about complex apps and data privacy, pointing to the need for transparent and secure design.

One unique suggestion was social media integration to follow or connect with friends within the app—although niche, it points toward potential future enhancements in terms of personalization and group engagement.

4.2.3 Findings from Requirement Elicitation - Interview

To complement the quantitative insights gathered through the online survey, a series of interviews were conducted with three distinct personas representing typical user groups of the proposed SplitPay application. These included: a student who shares rental expenses, a working adult who frequently splits costs with friends, and a classmate involved in group activities. The goal was to uncover deeper qualitative insights into real-life scenarios, pain points, and expectations that users have regarding bill-splitting practices and digital solutions. The findings have been grouped into key recurring themes as follows:

1. **Frustrations with Current Practices**

All interviewees expressed frustration with the limitations of existing informal or semi-digital bill-splitting methods. For instance, one student currently uses a shared Google Sheet with housemates but noted issues like forgotten entries and disputes about who paid what. Similarly, the classmate highlighted the confusion that arises during group meals or projects when manual calculations are done via messaging apps like WhatsApp. Inaccuracies and lack of accountability were common concerns.

The working adult mentioned past experience with Splitwise but cited low adoption among friends due to the app’s complexity, which led them to abandon it. This suggests that even when tools exist, usability and social acceptance are major barriers to consistent usage.

2. **Strong Demand for Automation and Accuracy**

All participants emphasized the need for an app that minimizes manual effort and enhances accuracy. They expressed interest in features such as:

- Recurring expense support, especially for rent and utilities
- Automatic item-based splitting to avoid unfair equal splits
- Receipt uploading or scanning to document purchases accurately
- Real-time calculation of individual shares, including tax and service charges

The classmate explicitly requested an app that can split by item rather than evenly, while the student renter noted that being able to upload receipts or photos would improve transparency during monthly settlements.

3. Usability and Simplicity

Ease of use emerged as a top priority. Interviewees consistently asked for a clean, minimal interface that reduces the steps required to complete tasks. The classmate explicitly mentioned the desire to "tap and split instantly", highlighting the importance of intuitive interaction flows.

The working adult emphasized that apps should feel lightweight and responsive, and also raised concerns about excessive permissions and unnecessary complexity. Features like offline access and dark mode were mentioned as contributing positively to usability and accessibility.

4. Local Payment Integration

One of the most critical expectations was the inclusion of local payment support, particularly QR code-based systems like DuitNow. The working adult interviewee, despite appreciating Splitwise's core tracking features, was dissatisfied with the need to switch to separate apps for actual payment. Integrating local payment mechanisms would not only improve convenience but also make the app more relevant to Malaysian users.

Additionally, both the student and working adult participants mentioned the importance of debt reminders to help prompt settlements without awkward confrontations.

5. Trust, Privacy, and Onboarding Experience

Trust and privacy were mentioned across the interviews as essential for adoption. Users were generally wary of apps that request bank access or force account registration from the start. The classmate suggested allowing guest access or trial use without full account setup, to reduce friction during the onboarding stage. Transparency about data handling policies and minimal permissions were also cited as factors that would influence users' willingness to retain and regularly use the app.

Overall, the interviews revealed a clear and recurring set of user expectations: automation, local payment integration, simplicity, and transparency. The current methods—manual calculations, spreadsheets, and messaging apps—are deemed unreliable and inconvenient. Existing apps fall short in areas like ease of use, local relevance, and adoption by social circles. These qualitative findings reinforce the need for a tailored, Malaysia-friendly solution like SplitPay, which should prioritize intuitive design, QR-based payments, item-level splitting, and low-friction onboarding.

4.2.4 System Requirements

To ensure that the SplitPay application aligns with user needs and system goals, both functional and non-functional requirements were defined. These requirements were derived from the findings of the requirement elicitation process, including surveys and interviews conducted with the target user base. Functional requirements (FRs) describe the specific behaviors, features, and services the system must support, while non-functional requirements (NFRs) define the quality attributes and system constraints necessary to ensure reliability, usability, and performance. The functional requirements are categorized according to the major system modules, whereas the non-functional requirements are grouped by their respective operational themes.

Each functional and non-functional requirement has been assigned a priority level using the MoSCoW method (Must have, Should have, Could have, Won't have). This prioritization is based on user feedback obtained from surveys and interviews, as well as technical considerations regarding development feasibility and logical system dependencies. The priority classification enables efficient planning of development phases and ensures that the most critical features are implemented first.

4.2.4.1 Functional Requirements

The table below outlines the functional requirements of the system, organized by their respective modules:

Table 5: SplitPay Functional Requirements

Requirement IDs	Requirement Statements	Priority
FR1	**User Management Module**	
FR1.1	Users shall be able to register an account with email, password, and username.	Must Have
FR1.2	Users shall be able to log in and log out securely.	Must Have
FR1.3	Users shall be able to manage their profile, including name, phone number, and bank QR code image.	Should Have
FR1.4	Users shall be able to reset their password using a recovery process.	Should Have
FR1.5	Users shall be able to upload, update, or delete their personal bank QR code.	Must Have
FR2	**Group Management Module**	
FR2.1	Users shall be able to create a new expense group.	Must Have
FR2.2	Users shall be able to create a new expense group.	Must Have
FR2.3	Users shall be able to view group details and member lists.	Must Have
FR2.4	Users shall be able to leave or delete a group (based on ownership role).	Should Have
FR3	**Expense Management Module**	
FR3.1	Users shall be able to add new expenses within a group.	Must Have
FR3.2	Users shall be able to assign payers and split costs among selected members.	Must Have
FR3.3	Users shall be able to categorize or tag expenses.	Could Have
FR3.4	Users shall be able to view expense history for each group.	Should Have

FR3.5	Users shall be able to edit or delete existing expenses.	Should Have
FR4	**Split Logic & Debt Calculation Module**	
FR4.1	The system shall auto-calculate each user's share based on selected split method (equal, percentage, custom).	Must Have
FR4.2	The system shall display a summary of who owes whom and how much.	Must Have
FR4.3	The system shall support recalculation when an expense is edited or deleted.	Must Have
FR4.4	The system shall track settlement status of each individual debt.	Should Have
FR5	**Payment Facilitation Module**	
FR5.1	Users shall be able to display their stored bank QR code for others to scan.	Must Have
FR5.2	The system shall allow users to access their own QR code from within the app without needing to use external apps.	Must Have
FR5.3	The QR code shall be accessible from group debt summaries or user profile.	Must Have
FR6	**Notifications & Alerts Module**	
FR6.1	The system shall notify users when they are added to a group or expense.	Should Have
FR6.2	The system shall send reminders for unsettled debts based on due dates.	Must Have
FR6.3	The system shall alert users when someone marks a payment as completed.	Should Have
FR6.4	Users shall be able to view past notifications within the app.	Could Have
FR6.5	Users shall be able to configure which types of notifications to receive.	Could Have
FR6.6	The system shall send email reminders for outstanding balances periodically.	Should Have
FR6.7	The system shall generate and email monthly summaries to users.	Should Have
FR6.8	Users shall be able to configure the frequency and type of notifications (email, push).	Should Have

FR7	**Reports & Analytics Module**	
FR7.1	Users shall be able to view total spending, debt, and repayments over time.	Should Have
FR7.2	Users shall be able to filter analytics by date, group, or category.	Could Have
FR7.3	Users shall be able to view group-specific summaries and member contributions.	Should Have
FR7.4	The system shall provide visual representations such as bar, pie, and line charts.	Could Have
FR8	**IOU Manager**	
FR8.1	Users shall be able to manually record IOUs owed or lent.	Should Have
FR8.2	Users shall be able to enter description, amount, date, due date, and person involved.	Should Have
FR8.3	Users shall be able to mark IOUs as paid or partially paid.	Should Have
FR8.4	Users shall be able to view a history of IOUs, filter by person or status.	Could Have
FR8.5	Users shall be able to edit or delete IOUs.	Could Have

4.2.4.2 Non-Functional Requirements

The following non-functional requirements specify the quality criteria the system must meet:

Requirement IDs	Requirement Statement	Priority
NFR1	**Performance**	
NFR1.1	The system shall respond to user interactions within 2 seconds under normal network conditions.	Must Have
NFR1.2	The app shall load group and expense data with less than 1-second delay for datasets under 500 entries.	Must Have
NFR2	**Security**	
NFR2.1	All user authentication data shall be encrypted in transit and at rest.	Must Have

NFR2.2	Only authenticated users shall access their personal and group data.	Must Have
NFR2.3	Uploaded QR codes shall be securely stored and accessible only by the user.	Must Have
NFR2.4	Notification delivery (push/email) shall occur within 30 seconds of the triggering event.	Should Have
NFR2.5	Email summaries shall be delivered reliably by the first day of the following month.	Should Have
NFR3	**Usability**	
NFR3.1	The user interface shall follow mobile-first design principles with intuitive navigation.	Must Have
NFR3.2	The app shall provide confirmation prompts for deletion or major changes.	Should Have
NFR3.3	The app shall include accessibility labels for screen readers.	Could Have
NFR4	**Availability & Reliability**	
NFR4.1	The app shall function offline with cached data where possible (excluding dynamic features like QR code updates or real-time notifications).	Should Have
NFR4.2	The system shall ensure data consistency even in the event of temporary disconnection.	Should Have
NFR5	**Data Backup and Recovery**	
NFR5.1	All user data shall be backed up to the cloud in real-time or at scheduled intervals	Should Have
NFR5.2	Users shall be able to restore their account and data after reinstalling the app.	Should Have
NFR6	**Maintainability**	
NFR6.1	The system shall be modular to allow independent updates to each module.	Must Have
NFR6.2	Error logs shall be tracked and stored for debugging purposes.	Must Have

4.2.5 Requirement Modelling

This section presents the requirement modelling approach used to capture and describe the functional behavior of the SplitPay system from the users' perspective.

The system adopts the Object-Oriented (OO) modelling methodology in accordance with the project design strategy. OO modelling is chosen because it supports modularity, reusability, and aligns well with the component-based architecture of mobile applications.

Among the various OO modelling techniques, the Use Case Diagram is used as the primary tool to visually represent the system's interactions with its external actor (user) and to identify the core functionalities offered by the system. It helps define the system's scope and ensures that all intended features align with stakeholder needs and previously identified functional requirements.

4.2.5.1 Use Case

The use case diagram developed for the SplitPay application provides a visual representation of the system’s functional requirements, focusing on the interactions between users and the system. It defines the application’s operational boundaries and clarifies the roles of various users by outlining how each actor engages with specific functionalities.

At the core of the diagram is the system boundary labeled “SplitPay System,” which encapsulates all relevant use cases. These use cases reflect both user-initiated actions and internal system operations that contribute to the app’s overall functionality. The actors identified include the registered user, group member, guest user, and the system itself. While all users have access to core features, group members operate in a shared financial context and therefore interact with more collaborative functions. The guest actor represents individuals without an account who are permitted to perform limited tasks, such as QR-based payments. The system actor, on the other hand, reflects background processes that are triggered automatically in response to certain conditions or user actions

The diagram illustrates how regular users interact with features like account registration, profile viewing, preference editing, QR generation for payments, and personal loan tracking. Additionally, they are able to receive system-generated notifications and access financial reports or spending analytics. Group members extend this interaction by managing group-related tasks such as creating or joining groups, recording shared expenses, monitoring debts, and settling payments.

Notably, the process of adding a shared expense is tightly coupled with an automatic expense splitting mechanism, which is modeled using the <<include>> relationship. This indicates that the auto-splitting functionality is a necessary sub-process of every shared expense action. Similarly, settling debts is inherently linked to sending payment reminders, which the system handles automatically as part of that process.

The diagram also includes a conditional behavior involving recurring expenses. While users can view general notifications, the system may optionally trigger additional alerts regarding upcoming or repeating expenses. This optionality is depicted using the <<extend>> relationship, signifying that the notification of recurring expenses is not always present, but is available under specific conditions.

Overall, the use case diagram captures the core functionality of SplitPay while distinguishing between user roles and system responsibilities. It provides a foundational understanding of how the system is expected to behave from a user's perspective and supports further modeling tasks such as sequence diagrams, activity flows, and database design.

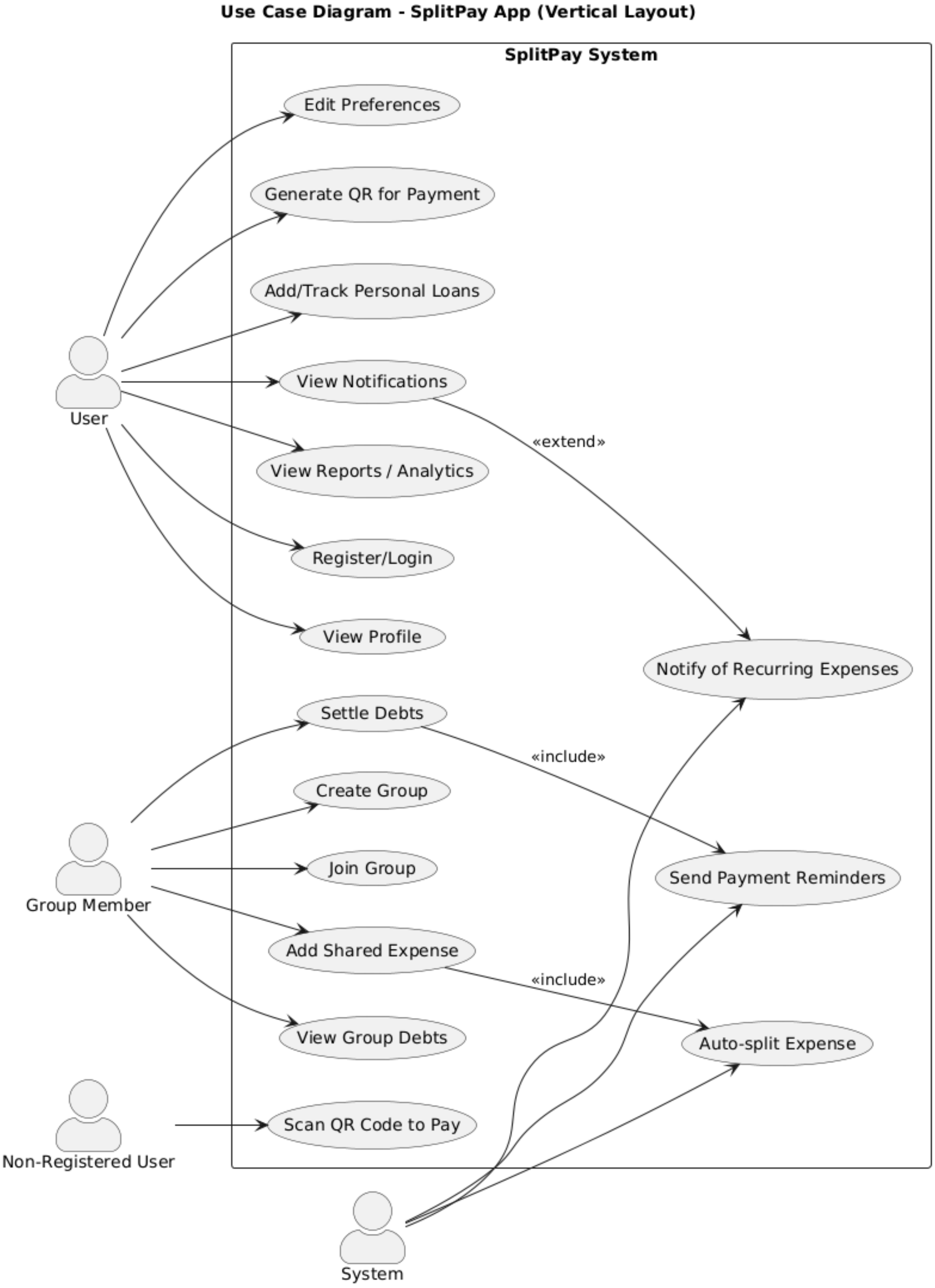


Figure 11: Use Case Diagram for SplitPay

4.2.5.2 Use Case Description

This section provides detailed descriptions of selected core use cases derived from the SplitPay system's overall functionality. These descriptions outline how users interact with the system to achieve specific goals, including the flow of actions, required conditions, and alternative scenarios. Each use case description is organized in a structured tabular format to enhance clarity and traceability of system requirements.

Table 6: Use Case 1 - User Registration

Element	Description
Use Case Name	Register Account
Actor	Registered User
Description	Allows a new user to register for an account with email, username, and password.
Preconditions	User is not already registered in the system.
Postconditions	User account is created and stored.
Basic Flow	1. User selects "Register" option. 2. System prompts for user details. 3. User submits the form. 4. System validates and creates the account. 5. User is redirected to login.
Alternative Flows	• Email already exists → error message. • Missing fields → validation errors.

Table 7: Use Case 2 - Add/Edit/Delete Expense

Element	Description
Use Case Name	Add/Edit/Delete Expense
Actor	Registered User
Description	Users can add, modify, or delete expense entries within a group.
Preconditions	User is logged in and a member of a group.
Postconditions	Expense list is updated.

Basic Flow (Add)	1. User selects "Add Expense" 2. Enters expense details. 3. Submits form. 4. System saves the expense and recalculates debts.
Alternative Flows	• User adds receipt image (optional). • Network failure → prompt to retry.
Edit/Delete Flow	1. User selects an existing expense. 2. Edits fields or confirms deletion. 3. System updates/deletes the entry.

Table 8: Use Case 3 - Split Expense

Element	Description
Use Case Name	Split Expense (Equal / Custom)
Actor	Registered User
Description	Applies a method to divide costs among participants of an expense.
Preconditions	Expense must have at least two members.
Postconditions	Split shares are saved to the database.
Basic Flow	1. User selects split method (equal/custom/percentage). 2. System validates total. 3. User confirms the split. 4. System stores the breakdown.
Alternative Flows	• Total doesn't match amount → system prompts for correction.

Table 9: Use Case 4 - Display Bank QR Code

Element	Description
Use Case Name	Display Bank QR Code
Actor	Registered User
Description	Displays the user's uploaded bank QR code for others to scan.
Preconditions	User has uploaded a bank QR code in their profile.
Postconditions	QR code is displayed temporarily.
Basic Flow	1. User selects "Show My QR Code" 2. System fetches the stored image.

	3. QR code appears in a pop-up.
Alternative Flows	• QR code not uploaded → system prompts upload.

Table 10: Use Case 5 - Manage IOUs

Element	Description
Use Case Name	Manage IOUs
Actor	Registered User
Description	Enables manual recording and tracking of personal debts (IOUs).
Preconditions	User is logged in.
Postconditions	IOU is created, updated, or deleted.
Basic Flow	1. User navigates to IOU Manager. 2. Adds new IOU (with person, amount, date). 3. System saves entry and displays it.
Alternative Flows	• Mark IOU as paid → update status. • Delete IOU → confirm before removal.

Table 11: Use Case 6 - View Reports and Analytics

Element	Description
Use Case Name	View Reports and Analytics
Actor	Registered User
Description	Allows users to visualize their expense and debt data over time.
Preconditions	User must be logged in and have recorded data.
Postconditions	Visual summaries and reports are generated.
Basic Flow	1. User opens "Reports" section. 2. Filters by date, group, or category. 3. System generates visual analytics.
Alternative Flows	• No data → system displays friendly "no data" message. • Filters return empty → show empty chart notice.

4.2.5.3 Sequence Diagram

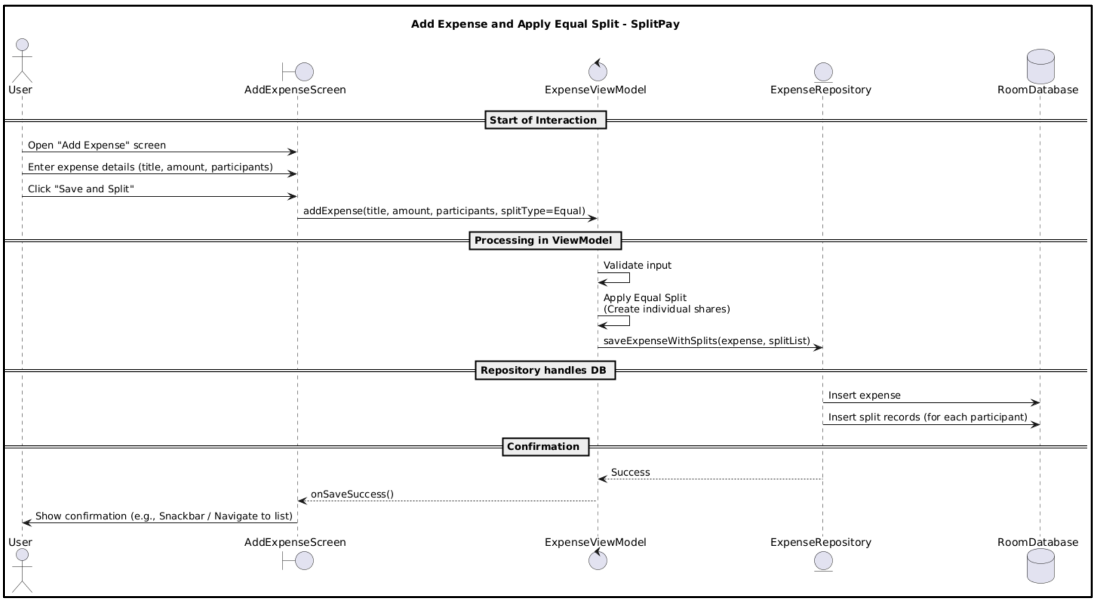

Add Expense and Apply Equal Split - SplitPay
User
AddExpenseScreen
ExpenseViewModel
ExpenseRepository
RoomDatabase
Start of Interaction
Open "Add Expense" screen
Enter expense details (title, amount, participants)
Click "Save and Split"
addExpense(title, amount, participants, splitType=Equal)
Processing in ViewModel
Validate input
Apply Equal Split
(Create individual shares)
saveExpenseWithSplits(expense, splitList)
Repository handles DB
Insert expense
Insert split records (for each participant)
Confirmation
Success
onSaveSuccess()
Show confirmation (e.g., Snackbar / Navigate to list)
User
AddExpenseScreen
ExpenseViewModel
ExpenseRepository
RoomDatabase

Figure 12: Add Expense and Splitting Bill Sequence Diagram

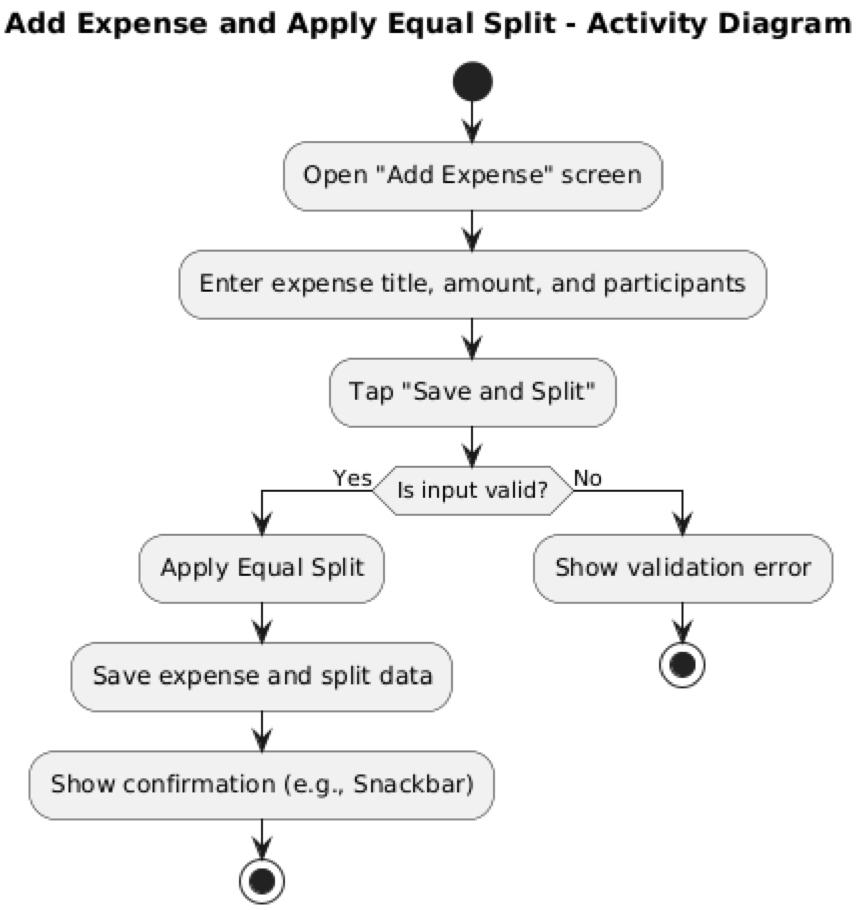
The sequence diagram presented illustrates the "Add Expense and Apply Equal Split" use case within the SplitPay mobile application. It depicts the flow of interactions among the user, the user interface (AddExpenseScreen), the logic layer (ExpenseViewModel), the intermediary data handler (ExpenseRepository), and the local persistence mechanism (RoomDatabase). This use case is initiated when a user navigates to the expense logging interface and inputs details such as the title, amount, and list of participants involved in the expense. Once the user confirms the entry by selecting the "Save and Split" action, the process is forwarded to the ViewModel layer.

At the ViewModel level, the system first validates the input data to ensure it meets the necessary conditions. Following successful validation, the ViewModel applies the equal split logic, wherein the total amount is evenly divided among the selected participants. This results in a list of individual split records that represent each participant's financial responsibility. The ViewModel then proceeds to call the repository's saveExpenseWithSplits() function, passing both the main expense entry and the list of split data for storage.

Subsequently, the ExpenseRepository performs two key database operations. First, it inserts the primary expense record into the Room database, followed by the insertion of each participant's individual split entry. While these operations are shown as sequential in the diagram, they are logically encapsulated within a single transactional context to ensure data consistency and atomicity. Upon successful completion of the database interactions, the repository returns a confirmation to the ViewModel, which then triggers a success callback to update the UI. The interface subsequently notifies the user of the successful operation—either through a visual confirmation such as a Snackbar or by redirecting the user back to the expense summary screen.

Overall, the diagram captures a complete local workflow that aligns with SplitPay's offline-first architecture. It emphasizes the modular responsibilities of each layer and avoids external network interactions, consistent with the current system design. Error handling and failure states are not explicitly modeled in this diagram, in order to maintain clarity and focus on the nominal successful path of execution. The inclusion of confirmation feedback and clear separation of concerns between UI, business logic, and data storage contributes to a maintainable and user-responsive architecture.

4.3.5.4 Activity Diagram



Add Expense and Apply Equal Split - Activity Diagram
Open "Add Expense" screen
Enter expense title, amount, and participants
Tap "Save and Split"
Yes
Is input valid?
No
Apply Equal Split
Show validation error
Save expense and split data
Show confirmation (e.g., Snackbar)

Figure 13: Add Expense and Apply Equal Split - SplitPay

This activity diagram illustrates the workflow for the “Add Expense and Apply Equal Split” use case within the SplitPay application. It focuses on the interaction between the user and the system at the UI layer, emphasizing the key decisions and actions involved in logging an expense and applying equal split logic.

The process begins with the user opening the "Add Expense" screen, where they are prompted to enter essential details such as the expense title, total amount, and the list of participants. Once the user submits this data by tapping the “Save and Split” button, the system performs input validation to ensure all required fields are correctly filled and logically sound.

If the input is invalid, the system immediately halts the process and displays a validation error message. The user must then rectify the input before proceeding. If the input is valid, the system proceeds to apply the Equal Split logic. This step involves dividing the total expense evenly among the listed participants and generating individual share records accordingly.

Following the split, the application persists both the main expense record and the individual split details to the local database. Upon successful saving, the user receives a visual confirmation—typically in the form of a Snackbar or toast message—indicating the completion of the operation. The activity concludes at this point.

This diagram emphasizes the necessity of applying split logic as a mandatory step prior to saving. It also reflects a sequential, decision-driven UI behavior pattern, ensuring both data integrity and user clarity throughout the process. All validation occurs prior to computation, and confirmation feedback marks the successful conclusion of the flow.

4.3 System Design

This section presents the technical design blueprint for the SplitPay system, outlining how the system will be implemented based on the requirements gathered during the analysis phase. It covers the architectural structure, database schema, and user interface layout to ensure the system is scalable, maintainable, and user-friendly. The design aims to support the system's core functionalities such as group expense management, automated bill splitting, debt tracking, and QR-based payment facilitation. By detailing the architecture, data design, and interface components, this chapter translates the system's conceptual requirements into a concrete plan for development.

4.3.1 Architecture Design

The SplitPay system adopts a two-tier client-based architecture designed primarily for mobile platforms. This architectural choice ensures responsiveness, local performance, and offline usability—key priorities identified during the requirements phase. The application is developed using Jetpack Compose for the presentation layer, ViewModel with LiveData/Flow for managing UI logic and state, and Room as the local database engine. The architecture is modular and scalable, allowing future enhancements such as internet-based features or an admin web portal without major restructuring.

The system is organized into two primary layers:

1. Presentation Layer (Client/UI Tier):

Built using Jetpack Compose, this layer handles all user interactions. It includes screens for login, registration, expense logging, group management, QR display, analytics, and more. The UI interacts with ViewModels to manage state and trigger business logic.

2. Application & Data Layer (Logic and Storage Tier):

This layer includes:

- ViewModels: Mediates between the UI and data layer using LiveData or Flow. It holds UI state and business rules (e.g., expense splitting logic).
- Repositories: Abstracts data access, managing calls to Room and potentially SharedPreferences.
- Room Database: Manages structured data storage for users, groups, expenses, debts, and IOUs.
- Local File Storage: Used to persist user-uploaded QR code images securely on the device.

The app is fully operational offline, with data persistence and retrieval done locally. However, it is designed to support future upgrades including Gmail login, cloud backup, or admin portal integration, thereby making it hybrid-capable.

A two-tier mobile architecture is ideal for this type of expense management application because:

- It ensures quick response times and offline accessibility, aligning with the non-functional requirements for performance and availability.
- It minimizes dependency on external services, increasing reliability in low-connectivity scenarios (e.g., at restaurants or during travel).
- The use of ViewModel and repository patterns improves maintainability and testability.

- The modular structure allows scaling towards a three-tier architecture in the future (e.g., by adding a cloud backend or a web-based admin panel).

Below is the architecture diagram representing the interaction between layers and key components.

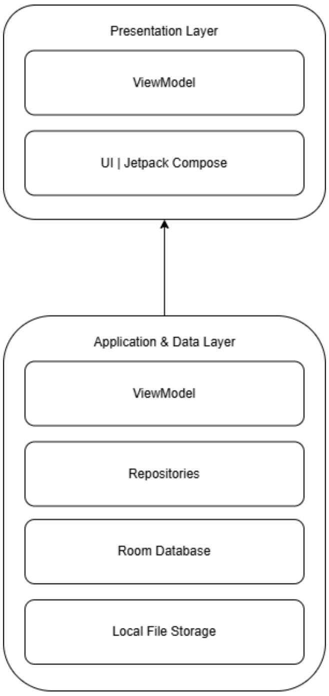


Figure 14: Architecture Design Diagram

4.3.2 Database Design

This section outlines the structure of the SplitPay system’s local database, designed using Room Persistence Library for Android. The database is fully local but designed in a modular and scalable way to support potential future features such as cloud sync and admin dashboard integration. It consists of multiple interrelated entities representing users, groups, expenses, IOUs, notifications, and QR code information. The database is normalized to reduce redundancy while remaining optimized for mobile performance.

4.3.2.1 ERD

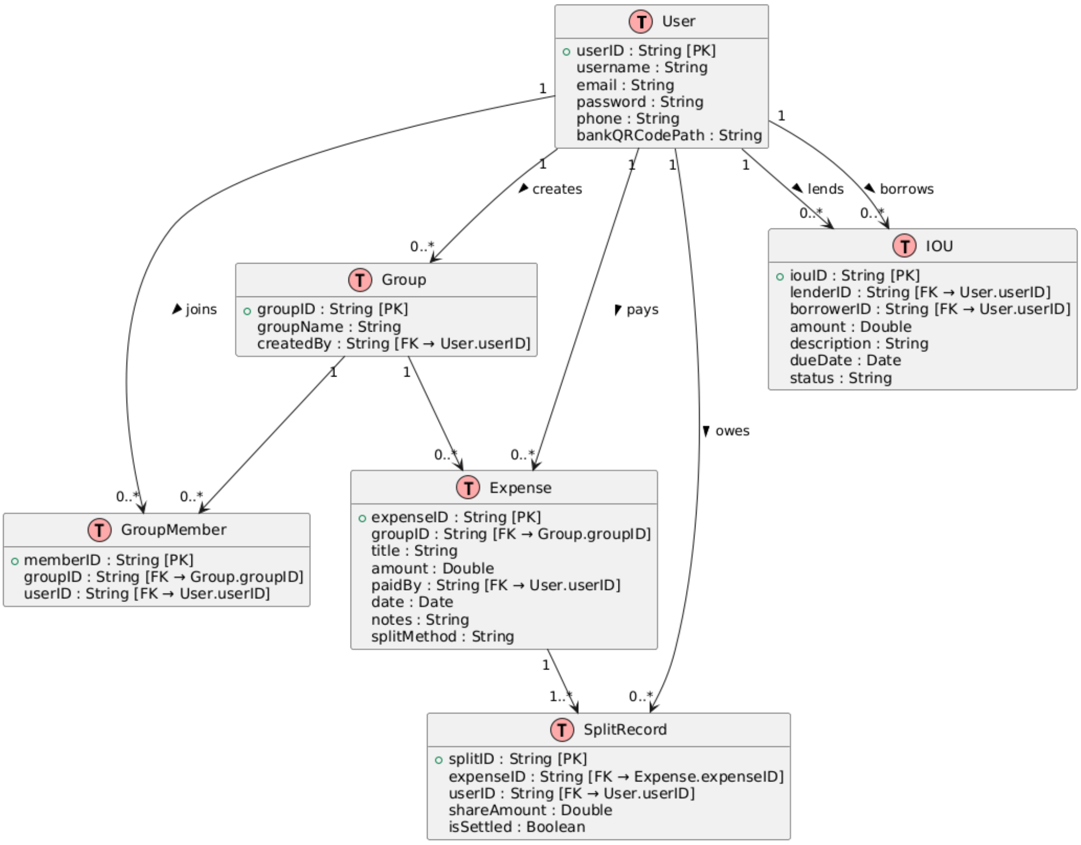


Figure 15: SplitPay ERD

4.3.2.2 Entity Descriptions

1. User

- Represents each registered user in the system. Stores personal information and QR details for payment representation.
- One user can join multiple groups.
- One user can create multiple expenses and IOUs.

2. Group

- Represents a shared expense group (e.g., housemates, trip, event).
- Created by a user.
- Linked to many users through GroupMember.

3. GroupMember

- A join table that connects users to groups.
- Contains roles (e.g., creator vs participant).
- Enables many-to-many relation between users and groups.

4. Expense

- Represents an individual bill or payment to be split among group members.
- Belongs to a group.
- Created by a user.

5. SplitDetail

- Defines how an expense is divided.
- Links users to expenses and specifies who owes how much.
- Supports equal or custom splits.

6. IOU

- Manual debt records outside of group expenses (e.g., "Lent RM20 to John").
- Independent of groups or split logic.

7. QRInfo

- Optional table for bank QR representation. Can alternatively be stored inside User.
- Allows for flexibility in separating sensitive or binary data.

4.3.2.3 Data Dictionary

Table 12: User Table Data Dictionary

Field Name	**Type**	**Description**	**Constraints**
userId	INT	Unique user ID	Primary Key, Auto Inc

email	TEXT	Email address	Unique, Not Null
password	TEXT	Hashed password	Not Null
username	TEXT	Chosen display name	Not Null
phoneNumber	TEXT	Optional phone number	
profileImage	TEXT	URI of profile image (if any)	Nullable
qrImagePath	TEXT	Local path to stored bank QR	Nullable

Table 13: Group Table Data Dictionary

Field Name	**Type**	**Description**	**Constraints**
groupId	INT	Unique group ID	Primary Key, Auto Inc
groupName	TEXT	Name of the group	Not Null
createdBy	INT	User ID of group creator	Foreign Key → User(userId)
createdAt	DATETIME	Group creation timestamp	Default CURRENT_TIMESTAMP

Table 14: GroupMember Table Data Dictionary

Field Name	**Type**	**Description**	**Constraints**
memberId	INT	Unique ID for group member record	Primary Key, Auto Inc
userId	INT	ID of the user in the group	Foreign Key → User(userId)

groupId	INT	ID of the group	Foreign Key → Group(groupId)
role	TEXT	Role in group ("admin", "member")	Default 'member'

Table 15: Expense Table Data Dictionary

Field Name	**Type**	**Description**	**Constraints**
expenseId	INT	Unique ID for the expense	Primary Key, Auto Inc
groupId	INT	Group to which the expense belongs	Foreign Key → Group(groupId)
paidBy	INT	User ID of the payer	Foreign Key → User(userId)
amount	REAL	Total amount of the expense	Not Null
description	TEXT	Description of the expense	Nullable
createdAt	DATETIME	Timestamp of expense	Default CURRENT_TIMESTAMP

Table 16: SplitDetail Table Data Dictionary

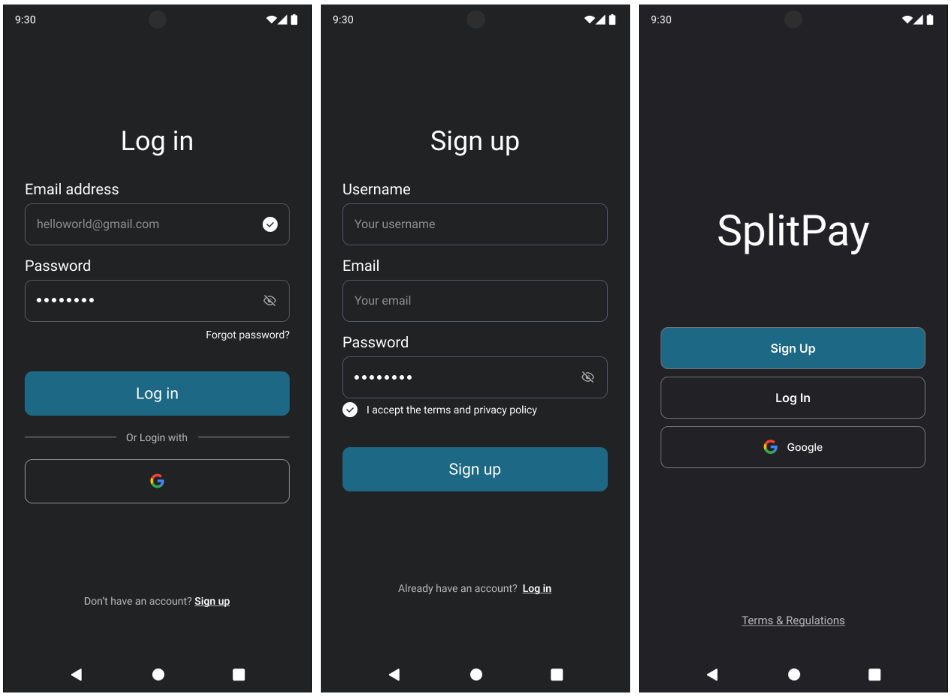
Field Name	**Type**	**Description**	**Constraints**
splitId	INT	Unique ID for split record	Primary Key, Auto Inc

expenseId	INT	Linked expense	Foreign Key → Expense(expenseId)
userId	INT	User who owes part of the expense	Foreign Key → User(userId)
amountOwed	REAL	Amount owed by the user	Not Null
isSettled	BOOLEAN	Whether this share has been paid	Default false

Table 17: IOU Table Data Dictionary

Field Name	**Type**	**Description**	**Constraints**
iouId	INT	Unique IOU record ID	Primary Key, Auto Inc
lenderId	INT	User who lent the money	Foreign Key → User(userId)
borrowerId	INT	User who borrowed the money	Foreign Key → User(userId)
amount	REAL	Amount of the IOU	Not Null
reason	TEXT	Description/reason for the IOU	Nullable
dueDate	DATE	When repayment is expected	Nullable
isSettled	BOOLEAN	Whether the IOU is repaid	Default false

4.3.3 UI/UX Prototype



helloworld@gmail.com

••••••••

Don't have an account? Sign up

Your username

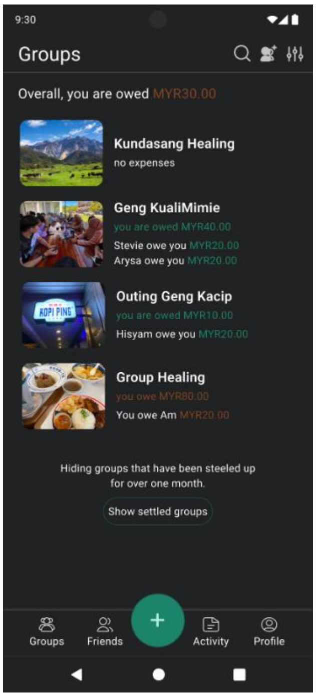
Your email

••••••••

✓ I accept the terms and privacy policy

Already have an account? Log in

Figure 16: User Registration Page



Overall, you are owed MYR30.00

Kundasang Healing
no expenses

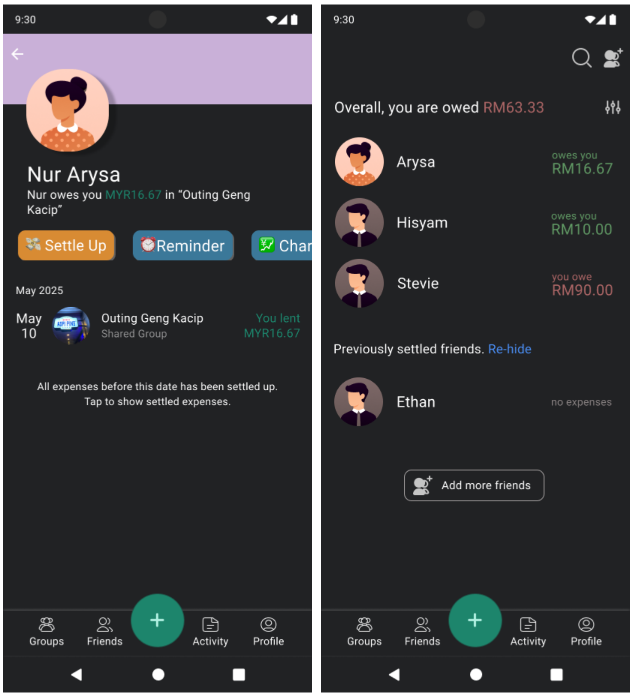
Geng KualiMimie
you are owed MYR40.00
Stevie owe you MYR20.00
Arysa owe you MYR20.00

Outing Geng Kacip
you are owed MYR10.00
Hisyam owe you MYR20.00

Group Healing
you owe MYR80.00
You owe Am MYR20.00

Hiding groups that have been steeled up for over one month.

Figure 17: Homepage



Nur Arysa

Nur owes you MYR16.67 in "Outing Geng Kacip"

May 2025

May 10 Outing Geng Kacip
Shared Group
You lent MYR16.67

All expenses before this date has been settled up.
Tap to show settled expenses.

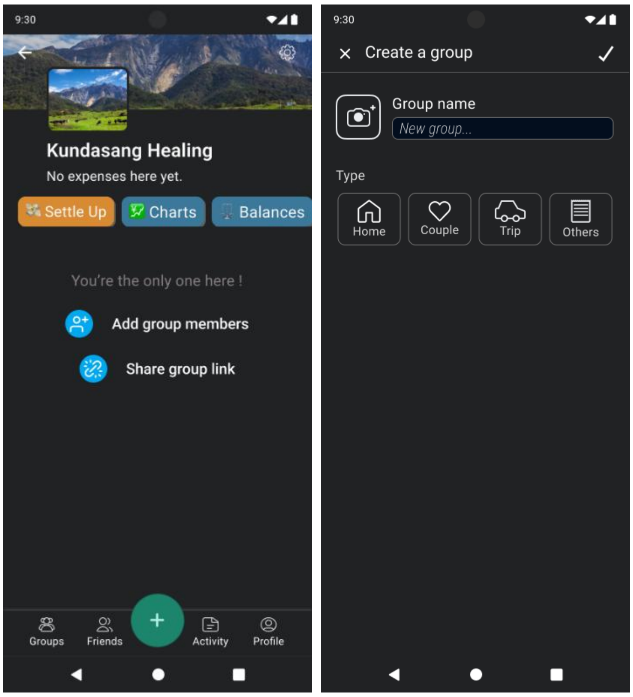
Overall, you are owed RM63.33

Arysa	owes you RM16.67
Hisyam	owes you RM10.00
Stevie	you owe RM90.00

Previously settled friends. Re-hide

Ethan	no expenses

Figure 18: Friends Profile & Friends Page



Kundasang Healing

No expenses here yet.

You're the only one here !

Add group members

Share group link

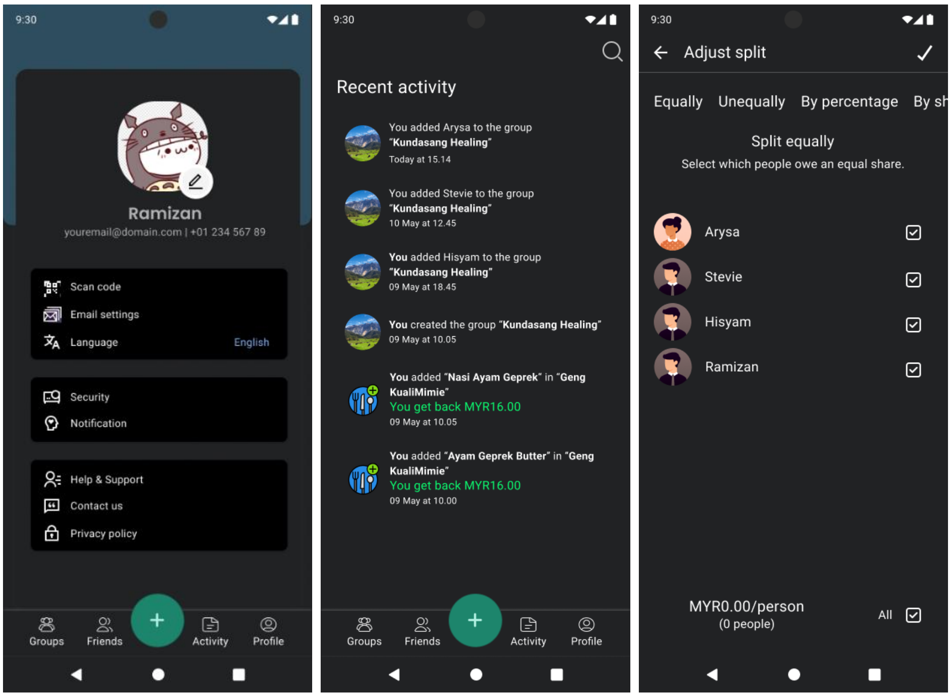
Create a group

Group name

New group...

Type

Figure 19: Group Expense Page and Group Creation Page



9:30

Ramizan

youremail@domain.com | +01 234 567 89

Scan code

Email settings

Language English

Security

Notification

Help & Support

Contact us

Privacy policy

Groups Friends + Activity Profile

9:30

Recent activity

You added Arysa to the group "**Kundasang Healing**"
Today at 15.14

You added Stevie to the group "**Kundasang Healing**"
10 May at 12.45

You added Hisyam to the group "**Kundasang Healing**"
09 May at 18.45

You created the group "**Kundasang Healing**"
09 May at 10.05

You added "**Nasi Ayam Geprek**" in "**Geng KualiMimie**"
You get back MYR16.00
09 May at 10.05

You added "**Ayam Geprek Butter**" in "**Geng KualiMimie**"
You get back MYR16.00
09 May at 10.00

Groups Friends + Activity Profile

9:30

← Adjust split ✓

Equally Unequally By percentage By sh

Split equally

Select which people owe an equal share.

Arysa ☑

Stevie ☑

Hisyam ☑

Ramizan ☑

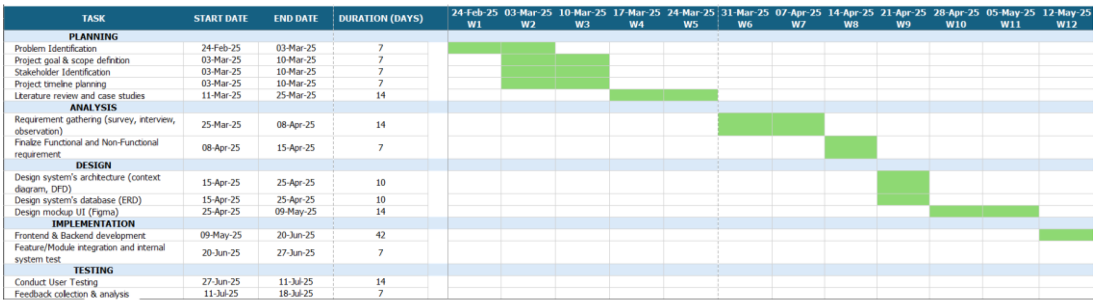
MYR0.00/person
(0 people)

All ☑

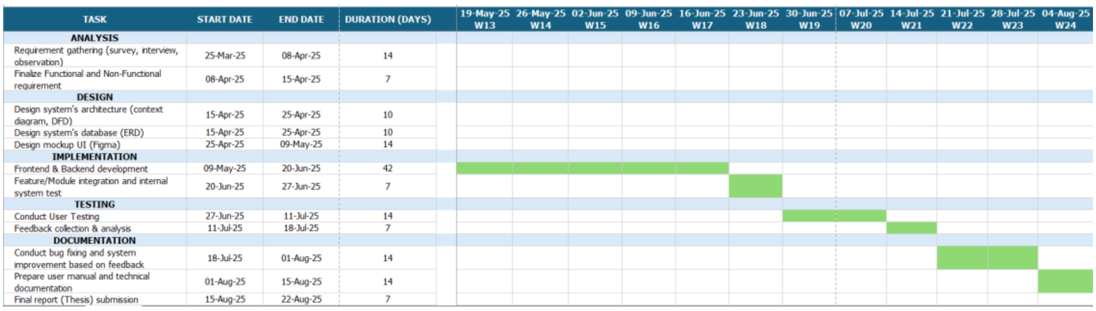
Figure 20: Profile Page, Activity Page, and Split Logic

APPENDICES

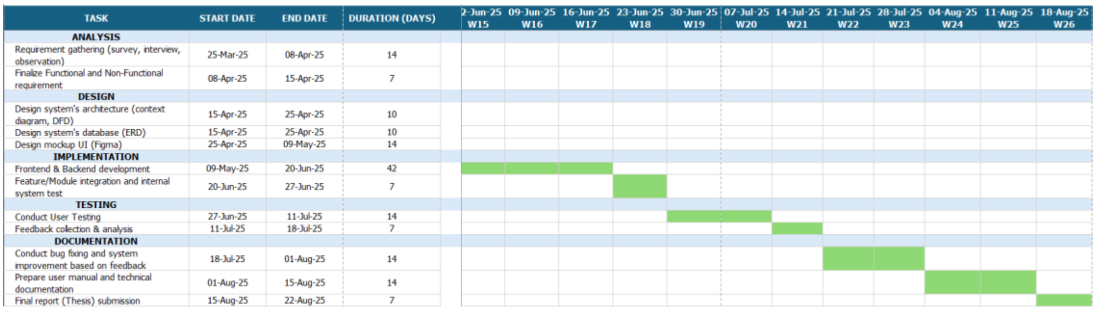
Appendix A: Gantt Chart



TASK	START DATE	END DATE	DURATION (DAYS)	24-Feb-25 W1	03-Mar-25 W2	10-Mar-25 W3	17-Mar-25 W4	24-Mar-25 W5	31-Mar-25 W6	07-Apr-25 W7	14-Apr-25 W8	21-Apr-25 W9	28-Apr-25 W10	05-May-25 W11	12-May-25 W12
PLANNING															
Problem Identification	24-Feb-25	03-Mar-25	7	🟩	🟩										
Project goal & scope definition	03-Mar-25	10-Mar-25	7		🟩	🟩									
Stakeholder Identification	03-Mar-25	10-Mar-25	7		🟩	🟩									
Project timeline planning	03-Mar-25	10-Mar-25	7		🟩	🟩									
Literature review and case studies	11-Mar-25	25-Mar-25	14				🟩	🟩							
ANALYSIS															
Requirement gathering (survey, interview, observation)	25-Mar-25	08-Apr-25	14						🟩	🟩					
Finalize Functional and Non-Functional requirement	08-Apr-25	15-Apr-25	7								🟩				
DESIGN															
Design system's architecture (context diagram, DFD)	15-Apr-25	25-Apr-25	10									🟩			
Design system's database (ERD)	15-Apr-25	25-Apr-25	10									🟩			
Design mockup UI (Figma)	25-Apr-25	09-May-25	14										🟩	🟩	
IMPLEMENTATION															
Frontend & Backend development	09-May-25	20-Jun-25	42												🟩
Feature/Module integration and internal system test	20-Jun-25	27-Jun-25	7												
TESTING															
Conduct User Testing	27-Jun-25	11-Jul-25	14												
Feedback collection & analysis	11-Jul-25	18-Jul-25	7												



TASK	START DATE	END DATE	DURATION (DAYS)	19-May-25 W13	26-May-25 W14	02-Jun-25 W15	09-Jun-25 W16	16-Jun-25 W17	23-Jun-25 W18	30-Jun-25 W19	07-Jul-25 W20	14-Jul-25 W21	21-Jul-25 W22	28-Jul-25 W23	04-Aug-25 W24
ANALYSIS															
Requirement gathering (survey, interview, observation)	25-Mar-25	08-Apr-25	14												
Finalize Functional and Non-Functional requirement	08-Apr-25	15-Apr-25	7												
DESIGN															
Design system's architecture (context diagram, DFD)	15-Apr-25	25-Apr-25	10												
Design system's database (ERD)	15-Apr-25	25-Apr-25	10												
Design mockup UI (Figma)	25-Apr-25	09-May-25	14												
IMPLEMENTATION															
Frontend & Backend development	09-May-25	20-Jun-25	42	🟩	🟩	🟩	🟩	🟩							
Feature/Module integration and internal system test	20-Jun-25	27-Jun-25	7						🟩						
TESTING															
Conduct User Testing	27-Jun-25	11-Jul-25	14							🟩	🟩				
Feedback collection & analysis	11-Jul-25	18-Jul-25	7									🟩			
DOCUMENTATION															
Conduct bug fixing and system improvement based on feedback	18-Jul-25	01-Aug-25	14										🟩	🟩	
Prepare user manual and technical documentation	01-Aug-25	15-Aug-25	14												🟩
Final report (Thesis) submission	15-Aug-25	22-Aug-25	7												



TASK	START DATE	END DATE	DURATION (DAYS)	2-Jun-25 W15	09-Jun-25 W16	16-Jun-25 W17	23-Jun-25 W18	30-Jun-25 W19	07-Jul-25 W20	14-Jul-25 W21	21-Jul-25 W22	28-Jul-25 W23	04-Aug-25 W24	11-Aug-25 W25	18-Aug-25 W26
ANALYSIS															
Requirement gathering (survey, interview, observation)	25-Mar-25	08-Apr-25	14												
Finalize Functional and Non-Functional requirement	08-Apr-25	15-Apr-25	7												
DESIGN															
Design system's architecture (context diagram, DFD)	15-Apr-25	25-Apr-25	10												
Design system's database (ERD)	15-Apr-25	25-Apr-25	10												
Design mockup UI (Figma)	25-Apr-25	09-May-25	14												
IMPLEMENTATION															
Frontend & Backend development	09-May-25	20-Jun-25	42	🟩	🟩	🟩									
Feature/Module integration and internal system test	20-Jun-25	27-Jun-25	7				🟩								
TESTING															
Conduct User Testing	27-Jun-25	11-Jul-25	14					🟩	🟩						
Feedback collection & analysis	11-Jul-25	18-Jul-25	7							🟩					
DOCUMENTATION															
Conduct bug fixing and system improvement based on feedback	18-Jul-25	01-Aug-25	14								🟩	🟩			
Prepare user manual and technical documentation	01-Aug-25	15-Aug-25	14										🟩	🟩	
Final report (Thesis) submission	15-Aug-25	22-Aug-25	7												🟩

Appendix B: Project Flowchart

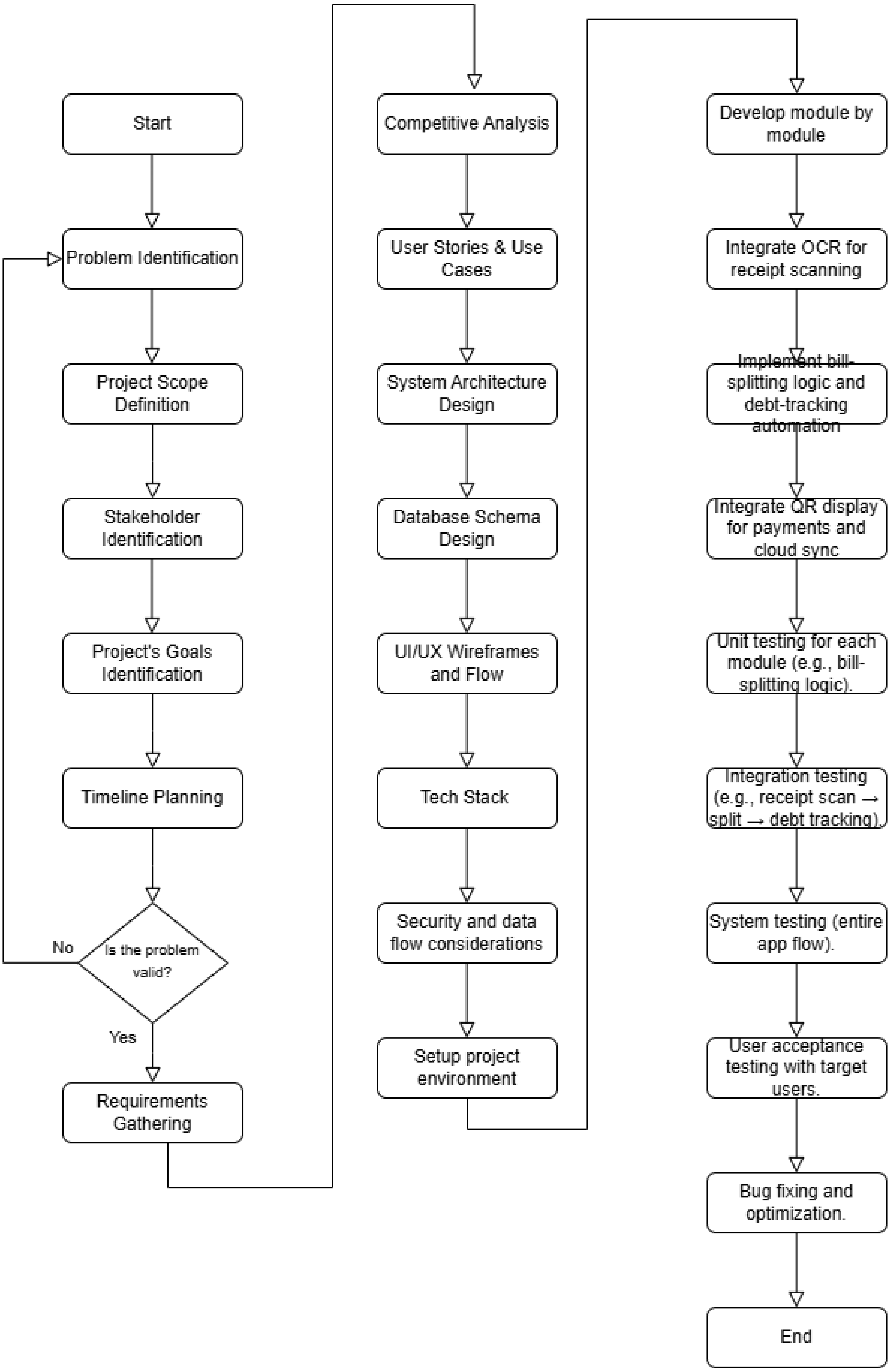
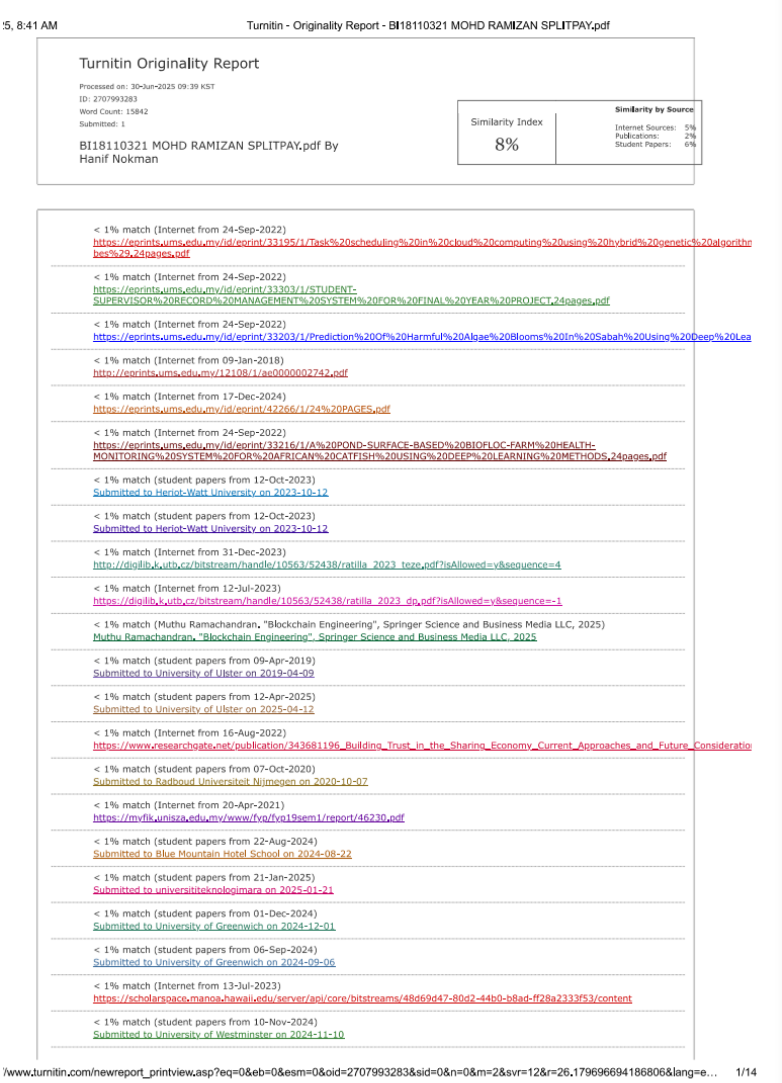

Start
Problem Identification
Project Scope Definition
Stakeholder Identification
Project's Goals Identification
Timeline Planning
Is the problem valid?
No
Yes
Requirements Gathering
Competitive Analysis
User Stories & Use Cases
System Architecture Design
Database Schema Design
UI/UX Wireframes and Flow
Tech Stack
Security and data flow considerations
Setup project environment
Develop module by module
Integrate OCR for receipt scanning
Implement bill-splitting logic and debt-tracking automation
Integrate QR display for payments and cloud sync
Unit testing for each module (e.g., bill-splitting logic).
Integration testing (e.g., receipt scan → split → debt tracking).
System testing (entire app flow).
User acceptance testing with target users.
Bug fixing and optimization.
End

Figure 21: Project Flowchart

Appendix C: Milestones & Date

Milestones	Date
Project Kickoff	24 February 2025
Project Goal & Scope Definition	10 March 2025
Requirement Gathering	26 May 2025
System Architecture & Database Design	16 June 2025
System UI/UX Design Mockup	30 September 2025
Frontend & Backend Development	9 December 2025
Conduct Testing & Feedback Analysis	15 December 2025
Documentation Completion	30 Decemeber 2025

Appendix D: Turnitin Plagiarism Report



:5, 8:41 AM Turnitin - Originality Report - BI18110321 MOHD RAMIZAN SPLITPAY.pdf

Turnitin Originality Report

Processed on: 30-Jun-2025 09:39 KST
ID: 2707993283
Word Count: 15842
Submitted: 1

BI18110321 MOHD RAMIZAN SPLITPAY.pdf By
Hanif Nokman

Similarity Index
8%

Similarity by Source
Internet Sources: 5%
Publications: 2%
Student Papers: 6%

< 1% match (Internet from 24-Sep-2022)
https://eprints.ums.edu.my/id/eprint/33195/1/Task%20scheduling%20in%20cloud%20computing%20using%20hybrid%20genetic%20algorithn
bes%29.24pages.pdf

< 1% match (Internet from 24-Sep-2022)
https://eprints.ums.edu.my/id/eprint/33303/1/STUDENT-SUPERVISOR%20RECORD%20MANAGEMENT%20SYSTEM%20FOR%20FINAL%20YEAR%20PROJECT.24pages.pdf

< 1% match (Internet from 24-Sep-2022)
https://eprints.ums.edu.my/id/eprint/33203/1/Prediction%20Of%20Harmful%20Algae%20Blooms%20In%20Sabah%20Using%20Deep%20Lea

< 1% match (Internet from 09-Jan-2018)
http://eprints.ums.edu.my/12108/1/ae0000002742.pdf

< 1% match (Internet from 17-Dec-2024)
https://eprints.ums.edu.my/id/eprint/42266/1/24%20PAGES.pdf

< 1% match (Internet from 24-Sep-2022)
https://eprints.ums.edu.my/id/eprint/33216/1/A%20POND-SURFACE-BASED%20BIOFLOC-FARM%20HEALTH-MONITORING%20SYSTEM%20FOR%20AFRICAN%20CATFISH%20USING%20DEEP%20LEARNING%20METHODS.24pages.pdf

< 1% match (student papers from 12-Oct-2023)
Submitted to Heriot-Watt University on 2023-10-12

< 1% match (student papers from 12-Oct-2023)
Submitted to Heriot-Watt University on 2023-10-12

< 1% match (Internet from 31-Dec-2023)
http://digilib.k.utb.cz/bitstream/handle/10563/52438/ratilla_2023_teze.pdf?isAllowed=y&sequence=4

< 1% match (Internet from 12-Jul-2023)
https://digilib.k.utb.cz/bitstream/handle/10563/52438/ratilla_2023_dp.pdf?isAllowed=y&sequence=-1

< 1% match (Muthu Ramachandran. "Blockchain Engineering", Springer Science and Business Media LLC, 2025)
Muthu Ramachandran. "Blockchain Engineering", Springer Science and Business Media LLC, 2025

< 1% match (student papers from 09-Apr-2019)
Submitted to University of Ulster on 2019-04-09

< 1% match (student papers from 12-Apr-2025)
Submitted to University of Ulster on 2025-04-12

< 1% match (Internet from 16-Aug-2022)
https://www.researchgate.net/publication/343681196_Building_Trust_in_the_Sharing_Economy_Current_Approaches_and_Future_Consideratio

< 1% match (student papers from 07-Oct-2020)
Submitted to Radboud Universiteit Nijmegen on 2020-10-07

< 1% match (Internet from 20-Apr-2021)
https://myfik.unisza.edu.my/www/fyp/fyp19sem1/report/46230.pdf

< 1% match (student papers from 22-Aug-2024)
Submitted to Blue Mountain Hotel School on 2024-08-22

< 1% match (student papers from 21-Jan-2025)
Submitted to universititeknologimara on 2025-01-21

< 1% match (student papers from 01-Dec-2024)
Submitted to University of Greenwich on 2024-12-01

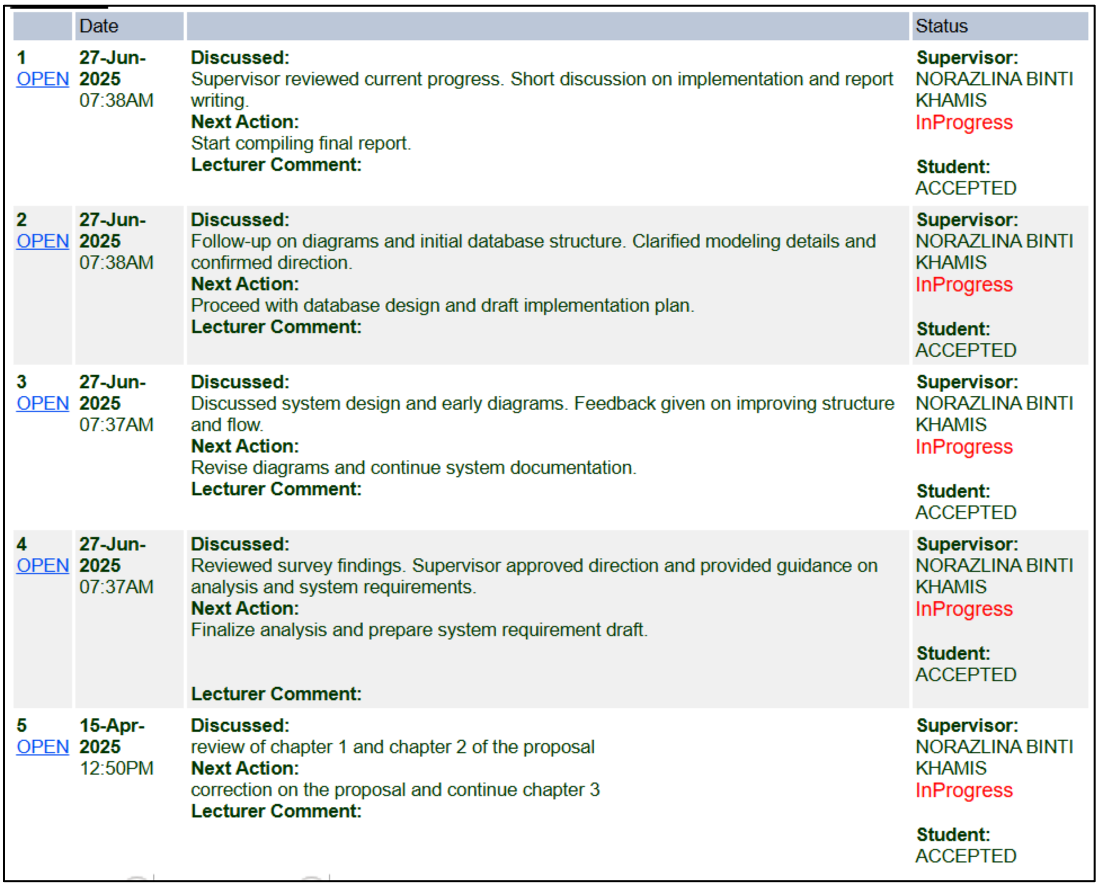
< 1% match (student papers from 06-Sep-2024)
Submitted to University of Greenwich on 2024-09-06

< 1% match (Internet from 13-Jul-2023)
https://scholarspace.manoa.hawaii.edu/server/api/core/bitstreams/48d69d47-80d2-44b0-b8ad-ff28a2333f53/content

< 1% match (student papers from 10-Nov-2024)
Submitted to University of Westminster on 2024-11-10

/www.turnitin.com/newreport_printview.asp?eq=0&eb=0&esm=0&oid=2707993283&sid=0&n=0&m=2&svr=12&r=26.179696694186806&lang=e... 1/14

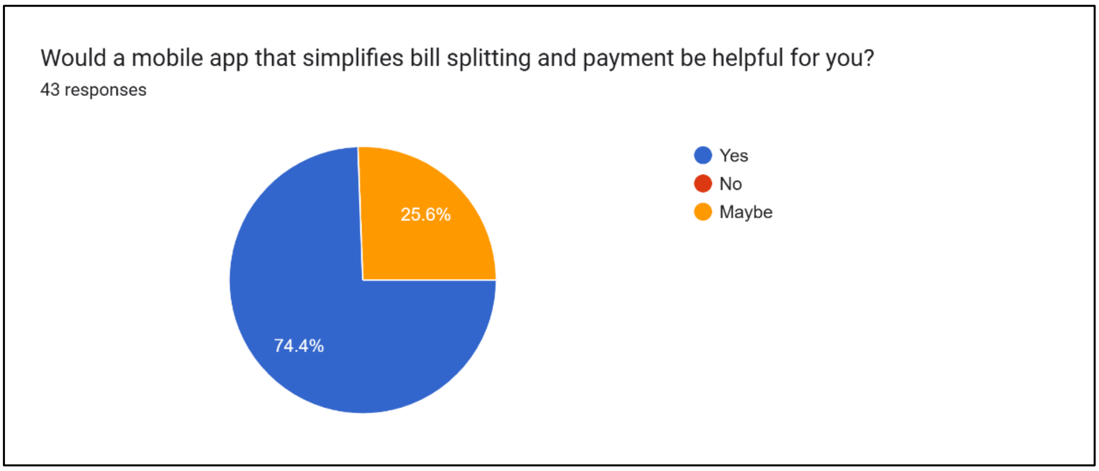
Appendix E: Meeting Log



	Date		Status
1 OPEN	**27-Jun-2025** 07:38AM	**Discussed:** Supervisor reviewed current progress. Short discussion on implementation and report writing. **Next Action:** Start compiling final report. **Lecturer Comment:**	**Supervisor:** NORAZLINA BINTI KHAMIS InProgress **Student:** ACCEPTED
2 OPEN	**27-Jun-2025** 07:38AM	**Discussed:** Follow-up on diagrams and initial database structure. Clarified modeling details and confirmed direction. **Next Action:** Proceed with database design and draft implementation plan. **Lecturer Comment:**	**Supervisor:** NORAZLINA BINTI KHAMIS InProgress **Student:** ACCEPTED
3 OPEN	**27-Jun-2025** 07:37AM	**Discussed:** Discussed system design and early diagrams. Feedback given on improving structure and flow. **Next Action:** Revise diagrams and continue system documentation. **Lecturer Comment:**	**Supervisor:** NORAZLINA BINTI KHAMIS InProgress **Student:** ACCEPTED
4 OPEN	**27-Jun-2025** 07:37AM	**Discussed:** Reviewed survey findings. Supervisor approved direction and provided guidance on analysis and system requirements. **Next Action:** Finalize analysis and prepare system requirement draft. **Lecturer Comment:**	**Supervisor:** NORAZLINA BINTI KHAMIS InProgress **Student:** ACCEPTED
5 OPEN	**15-Apr-2025** 12:50PM	**Discussed:** review of chapter 1 and chapter 2 of the proposal **Next Action:** correction on the proposal and continue chapter 3 **Lecturer Comment:**	**Supervisor:** NORAZLINA BINTI KHAMIS InProgress **Student:** ACCEPTED

Appendix F: Results from Survey

Section C: Expectations & Feature Needs



Would a mobile app that simplifies bill splitting and payment be helpful for you?

43 responses

Figure 22: User Interest

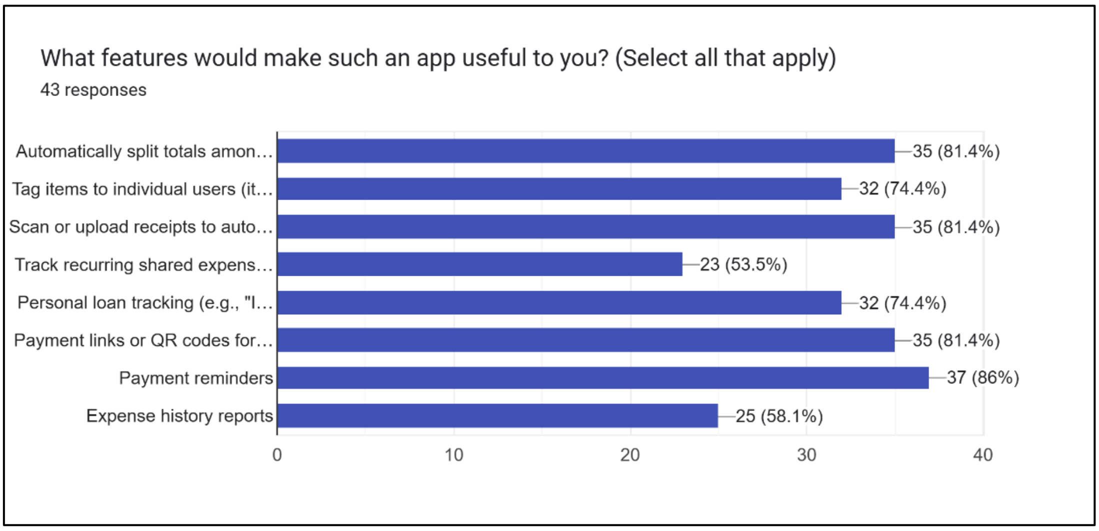


Figure 23: Desired Features for a Bill Splitting App

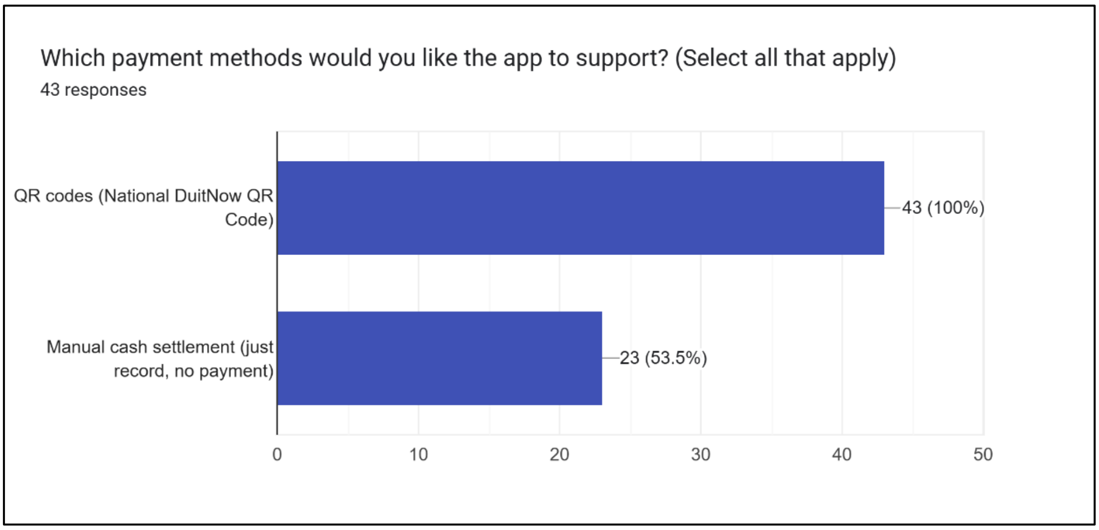


Figure 24: Preferred Payment Methods for Bill Splitting App

Section D: UI/UX Design Insights

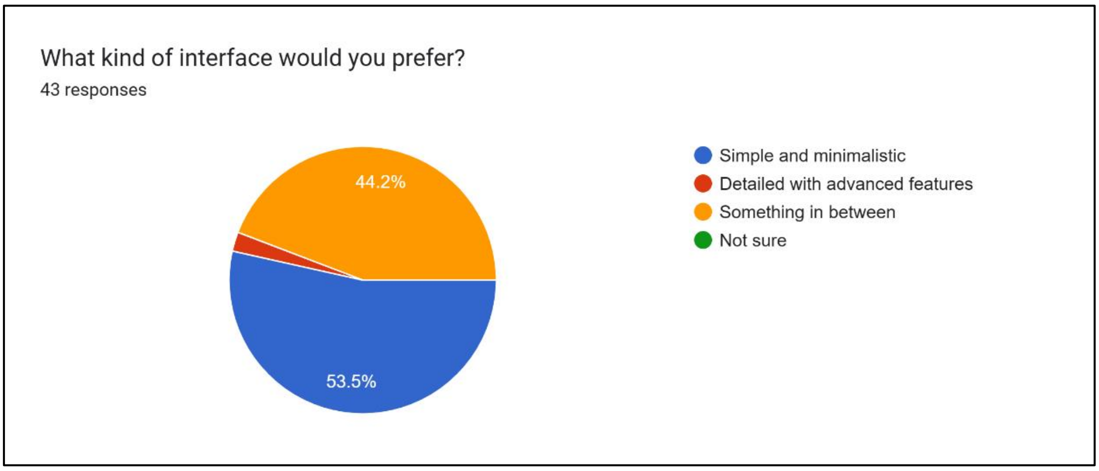


Figure 25: Preferred User Interface for an App

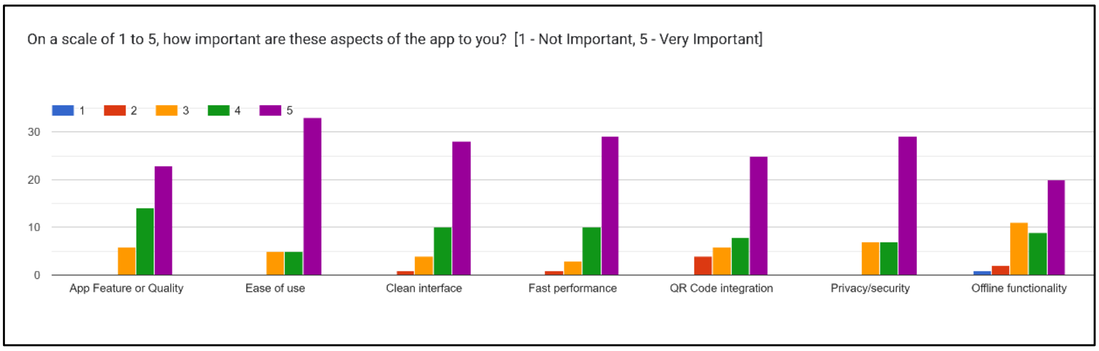


Figure 26: Importance of App Aspects

Section E: Open Comments

Have you ever used any expense-splitting apps before? If yes, what did you like/dislike about them?

If you could design your own bill-splitting app, what would be your top 3 must-have features?

Any additional comments, problems, or suggestions related to bill-splitting?

REFERENCES

Gneezy, U., Haruvy, E., & Yafe, H. (2004). The Inefficiency of Splitting the Bill. The Economic Journal, 114(495), 265–280. https://doi.org/10.1111/j.1468-0297.2004.00209.x

Split payment solutions: Innovations for hassle free bill splitting. (2025, 10 April) Faster Capital. https://fastercapital.com/content/Split-payment-solutions--Innovations-for-hassle-free-bill-splitting.html#The-challenges-of-traditional-bill-splitting-methods.html

European Central Bank. (2023, March). Study on digital wallet features [Annex to press release]. https://www.ecb.europa.eu/press/pr/date/2023/html/ecb.pr230424_1_annex~93abdb80da.en.pdf

Nidhibhat. (2023, December 27). Evaluating Splitwise's interface and its influence on group dynamics. Medium. https://medium.com/@nidhibhat.1098/evaluating-splitwises-interface-and-its-influence-on-group-dynamics-f6f1f3ef1355

Splitkaro Marketing Team. (2023, April 24). The Social Stigma Around Fair Bill Splitting. Medium. https://medium.com/splitkaro/the-social-stigma-around-fair-bill-splitting-6680368cb7f5Belk, R. (2014). You are what you can access: Sharing and collaborative consumption online. Journal of Business Research, 67(8), 1595–1600. https://doi.org/10.1016/j.jbusres.2013.10.001

Venmo. (2021). The ultimate guide to splitting bills with friends. https://venmo.com/about/ultimate-guide-to-splitting-bills/

Zhang, Y., Zhao, S., & Xu, H. (2017). Understanding user practices of mobile bill splitting apps.

Proceedings of the ACM on Human-Computer Interaction, 1(CSCW), 1–18. https://doi.org/10.1145/3134751

Zhou, M., & Rau, P.-L. P. (2020). Design and evaluation of a mobile application for fair cost splitting. International Journal of Human–Computer Interaction, 36(4), 359–374. https://doi.org/10.1080/10447318.2019.1655079

Gneezy, U., Haruvy, E., & Yafe, H. (2004). The inefficiency of splitting the bill. *The Economic Journal, 114*(495), 265–280. **https://doi.org/10.1111/j.1468-0297.2004.00209.x**

Roos, D., & Hahn, R. (2019). Understanding collaborative consumption: An extension of the theory of planned behavior with value-based personal norms. *Journal of Business Ethics, 158*(3), 679–697. **https://doi.org/10.1007/s10551-017-3675-3**

Anderson, C. A., Garcia, L. F., & Kim, S. Y. (2018). Evaluating the effectiveness of mobile applications for shared expenses in group travel. Journal of Travel & Tourism Marketing, 35(6), 750–765. https://doi.org/10.1080/10548408.2018.1467321

Belk, R. (2014). You are what you can access: Sharing and collaborative consumption online. Journal of Business Research, 67(8), 1595–1600. https://doi.org/10.1016/j.jbusres.2013.10.001

Böhringer, C., & Roth, K. (2020). The economics of cost-sharing in group consumption: Challenges and opportunities for digital tools. Journal of Economic Behavior & Organization, 173, 136–150. https://doi.org/10.1016/j.jebo.2020.02.003

Gibson, K. (2017). The social dynamics of shared expense management: How technology helps reduce interpersonal friction. Social Networks & Technology, 5(2), 112–130. https://doi.org/10.1080/23200065.2017.1205915

Karlan, D., Valdivia, M., & Boucher, S. (2016). The impact of mobile financial tools on group savings and consumption. World Development, 84, 81–92. https://doi.org/10.1016/j.worlddev.2016.03.008

Shen, Y. (2019). The evolving role of digital platforms in simplifying group financial management. Journal of Financial Technology, 2(3), 203–218. https://doi.org/10.1016/j.jfintec.2019.04.007