Design and Development of a Roommate Management Android App

by

Jasim Uddin
ID: CSE2103024009

Md. Zuyel Rana
ID: CSE2201025043

Md. Borhan Uddin
ID: CSE2103024015

Md. Sadman Rahman Sumon
ID: CSE2201025053

A. S. M. Mejbaul Alam
ID: CSE2103024089

Supervised by
Md. Shamim Hossain

Submitted in partial fulfillment of the requirements for the degree of
Bachelor of Science in Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SONARGAON UNIVERSITY (SU)
January 2026

APPROVAL

The project titled "**Design and Development of a Roommate Management Android App**" submitted by Jasim Uddin (CSE2103024009), Md. Zuyel Rana (CSE2201025043), Md. Borhan Uddin (CSE2201025003), Md. Sadman Rahman Sumon (CSE2201025053), and A. S. M. Mejbaul Alam (CSE2103024089) to the Department of Computer Science and Engineering, Sonargaon University (SU), has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering and approved as to its style and contents.

Board of Examiners

Supervisor

Md. Shamim Hossain
Lecturer,
Department of Computer Science and Engineering
Sonargaon University (SU)

Examiner 1

(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

Examiner 2

(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

Examiner 3

(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

DECLARATION

We, hereby, declare that the work presented in this report is the outcome of the investigation performed by us under the supervision of **Md. Shamim Hossain, Lecturer,** Department of Computer Science and Engineering, Sonargaon University, Dhaka, Bangladesh. We reaffirm that no part of this project has been or is being submitted elsewhere for the award of any degree or diploma.

Countersigned

(Md. Shamim Hossain)
Supervisor

Signature

Jasim Uddin
ID: CSE2103024009

Md. Zuyel Rana
ID:CSE2201025043

Md. Borhan Uddin
ID:CSE2103024015

Md. Sadman Rahman Sumon
ID:CSE2201025053

A. S. M. Mejbaul Alam
ID: CSE2103024089

ABSTRACT

The Roommate Android Application is a mobile-based software solution designed to address the common challenges associated with shared living environments, particularly for students and working professionals. In modern urban life, people frequently relocate for education or employment, which creates a growing demand for affordable rooms and reliable roommates. However, the traditional methods of finding rooms and managing shared accommodations are often inefficient, unorganized, and time-consuming. This project aims to overcome these issues by providing a centralized digital platform that simplifies room and roommate discovery while also supporting day-to-day living management. The application is developed using the Flutter framework and the Dart programming language, which together enable the creation of a high-performance, responsive, and user-friendly Android application. The system offers several essential features, including user authentication, profile management, room searching, roommate finding, room and work management, notice sharing, SOS emergency support, and an informational section. These features are designed to improve communication among roommates, ensure fair task distribution, and enhance overall safety and convenience. The development of the Roommate App follows the Waterfall software development model, which involves requirement analysis, system design, implementation, testing, and maintenance. This structured approach ensures proper planning, documentation, and quality control throughout the development process. The project demonstrates the practical application of software engineering concepts and modern mobile application development techniques. Overall, the Roommate Android Application aims to provide an efficient, reliable, and user-friendly solution for managing shared living arrangements in a digital and organized manner.

ACKNOWLEDGMENT

At the very beginning, we would like to express my deepest gratitude to the Almighty Allah for giving us the ability and the strength to finish the task successfully within the scheduled time.

We are auspicious that we had the kind association as well as supervision of **Md. Shamim Hossain**, Lecturer, Department of Computer Science and Engineering, Sonargaon University whose hearted and valuable support with best concern and direction acted as necessary recourse to carry out our project.

We are particularly grateful to **Prof. Bulbul Ahamed**, Head of the Department, Computer Science and Engineering, Sonargaon University, for his kind concern and precious suggestions.

We are also thankful to all our teachers during our whole education, for exposing us to the beauty of learning.

Finally, our deepest gratitude and love to my parents for their support, encouragement, and endless love

TABLE OF CONTENTS

Title			Page No.
DECLARATION …….....			1
ABSTRACT ……...…….			2
ACKNOWLEDGEMENT …….……			3
LIST OF ABBREVIATION …….…….			9
CHAPTER 1			10– 14
INTRODUCTION OF ROOMMATE APP			
	1.1	Introduction ……..…….....	10
	1.2	Features of Roommate Finding App……	11
	1.3	Importance of roommate app……..……	11
	1.4	Goal the Project …….…….....	11
	1.5	Waterfall Model …….…….	12
	1.6	Project Schedule……..	13
	1.7	Problem Statement…….	14
	1.8	Proposed Solution…….	14
CHAPTER 2			15– 18
REQUIREMENT SPECIFICATION			
	2.1	Introduction …….….....	15
	2.2	Functional Requirement…….……	15
	2.3	Non-functional Requirement…….	16
	2.4	System Requirement……	17
CHAPTER 3			19-22
TOOLS AND TECHNOLOGY			
	3.1	Introduction …….....	19
	3.2	Flutter Framwork…….……	19

	3.3	Dart Programming language……..		20
	3.4	Database and Storage…….		21
	3.5	Authentication and Sequirity……..		22
CHAPTER 4				23-25
SYSTEM DESIGN				
	4.1	System Architecture…….…….		24
	4.2	UI Design……..		24
	4.3	Data Flow……..		25
CHAPTER 5				26 – 34
IMPLEMENTATION AND TESTING				
	5.1	Implementation details…….…….		26
		5.1.1	Log in and registration screen……	27
	5.2	Functional Implementation……..		28
		5.2.1	Find Roommate……	27
		5.2.2	Room Management……	29
		5.2.3	Notice Features……..	30
		5.2.4	About Roommate App……..	31
	5.3		Testing Execution Results……	32
		5.3.1	Unit Testing……	33
CHAPTER 6				35-36
CONCLUSION AND FUTURE WORKS				
	6.1	Conclusion ……..…..		34
	6.2	Limitation……		34
	6.2	Future Works …….……		35
		Research Opportunities…….		35
REFERENCES …….……				37

LIST OF TABLES

Table No.	Title	Page No.
Table 3.4.1	Database and Storage	21
Table 3.5.1	Authentication and Security	22
Table 5.3.5	Results	34

LIST OF FIGURES

Figure No.	Title	Page No.
Fig.1.6	Project Schedule	13
Fig.2.1	User Registration	18
Fig.4.2	UI design	23
Fig.4.3	Data Flow	25
Fig.5.1.1	Log in and registration Screen	27
Fig.5.2.1	Find Roommate Screen	29
Fig.5.2.2	Room Management	30
Fig.5.2.3	Notice Features	31
Fig.5.2.4	About Features	32

LIST OF ABBREVIATION

CSE	Computer Science and Engineering
UI	User Interface
UX	User Experience
SDK	Software Development Kit
API	Application Programming Interface
SOS	Save Our Souls / Emergency Alert
Dart	Dart Programming Language
Flutter	Flutter Framework
Firebase	Firebase Cloud Platform
PC	Personal Computer
Wi-Fi	Wireless Fidelity
AI	Artificial Intelligence

CHAPTER 1

INTRODUCTION OF ROOMMATE APP

1.1 Introduction

In modern society, finding a suitable living arrangement has become increasingly challenging for students, professionals, and young adults who relocate for work or education. The Roommate Android Application is designed to simplify this process by offering a centralized platform for finding rooms and compatible roommates. By leveraging modern technologies such as Flutter, Dart, and Firebase, the application provides a seamless and responsive user experience that addresses the inefficiencies and frustrations associated with traditional methods of room hunting and roommate selection.

1.2 Features of Roommate Finding App

The Roommate App provides an extensive range of features to facilitate a smooth and organized shared living experience. It not only focuses on helping users find rooms and compatible roommates but also emphasizes effective management of household responsibilities. The app incorporates advanced search algorithms to filter rooms based on location, budget, amenities, and user preferences, ensuring that users find the most suitable accommodation quickly. The roommate matching system considers personality traits, lifestyle preferences, and compatibility factors to pair users with suitable roommates, minimizing potential conflicts.

In addition to these core functionalities, the application includes task and work management features that allow roommates to distribute household chores fairly and track completion. The digital notice board ensures that all important announcements, reminders, and updates are visible to all members, reducing miscommunication. The SOS emergency feature enhances user safety by allowing quick alerts to designated contacts in case of any danger. Together, these features create a comprehensive ecosystem that addresses both the practical and safety concerns of shared living.

- **User Authentication:** Secure login and registration system to protect user data.
- **Profile Management:** Users can create and update their personal profiles, including preferences.
- **Room Search:** Advanced search options to find available rooms based on location, budget, and amenities.
- **Roommate Matching:** Helps users find compatible roommates based on profile details.
- **Room Management:** Organize room members, tasks, and shared responsibilities.

- **Work Management:** Assign and track household tasks for fair distribution.
- **Notice Board:** Post and view important notices digitally.
- **SOS Feature:** Emergency alert system to notify contacts in case of danger.
- **About Section:** Provides users with information about the app and its functionalities.

1.3 Importance of Roommate App

The importance of the Roommate App stems from its ability to streamline the entire shared living process, addressing the challenges that individuals typically face. Key reasons for its importance include:

- **Time Efficiency:** Reduces the time and effort spent searching for suitable rooms and roommates.
- **Improved Communication:** Digital notices and task management enhance transparency and reduce misunderstandings.
- **Safety Enhancement:** SOS emergency feature provides an additional layer of security.
- **Fair Task Distribution:** Work management ensures chores and responsibilities are evenly shared.
- **Compatibility Matching:** Helps reduce conflicts by pairing users with compatible roommates.
- **Centralized Management:** Combines room search, roommate finding, task management, and notices in one platform, making it convenient and user-friendly.

1.4 Goal of the Project

The main goal of this project is to develop a reliable, user-friendly, and efficient mobile application that assists users in finding rooms and roommates while managing daily living tasks. The application aims to provide a centralized platform that integrates multiple functionalities, including room searching, roommate matching, task management, notice sharing, and emergency support. Through this project, the objective is to demonstrate the practical application of software engineering concepts using Flutter and Dart for mobile application development.

1.5 Waterfall Model

The Waterfall Model is a traditional software development methodology that follows a linear and sequential approach. It is widely used for projects where requirements are well-understood and stable. This model is highly structured and easy to manage due to its defined stages.

Key Points of the Waterfall Model:

- Requirement Analysis
- System Design
- Implementation
- Testing
- Deployment
- Maintenance

Key Point Explanation:

- **Requirement Analysis:** In this phase, all functional and non-functional requirements of the system are collected and documented. This forms the foundation for the subsequent phases.

- **System Design:** The system architecture, data flow diagrams, and interface designs are created based on the requirements.

- **Implementation:** Actual coding and development of the system modules take place using Flutter and Dart.

- **Testing:** Each module and the integrated system are tested to ensure they meet the specified requirements and function correctly.

- **Deployment:** The application is released for use by the end-users.

- **Maintenance:** Post-deployment support is provided, including bug fixes, updates, and system improvements.

The Waterfall Model ensures that each phase is completed before moving on to the next, which reduces project risks and provides clear documentation throughout the development process.

1.6 Project Schedule

The following Gantt chart outline

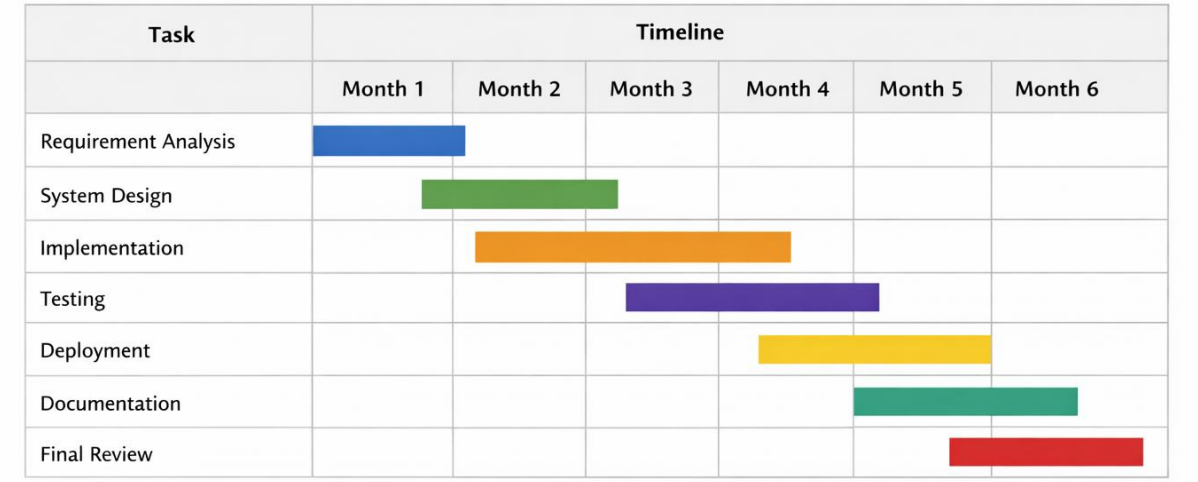

Figure:1.6 Project Schedule

Explanation:

- **Requirement Analysis:** Collect and document all system requirements.
- **System Design:** Plan architecture, UI/UX, database structure, and data flow.
- **Implementation:** Code all modules using Dart & Flutter and integrate features.
- **Testing:** Test each module and the complete system for bugs, performance, and functionality.
- **Deployment:** Release the app to users and make it operational.
- **Maintenance:** Ongoing updates, bug fixes, and feature improvements.

1.7 Problem Statement

Finding suitable rooms and compatible roommates is a common challenge faced by students, young professionals, and people relocating for work or study. Traditional methods, such as relying on word-of-mouth, social media groups, or local brokers, are often time-consuming, inefficient, and unreliable. Many individuals struggle to find accommodations that match their preferences, budget, and location, while also ensuring compatibility with potential roommates. Additionally, the lack of proper management tools in shared living arrangements can lead to miscommunication, conflicts, unbalanced distribution of household tasks, missed notices, and safety concerns. These issues create a pressing need for a digital solution that simplifies the process of finding rooms and roommates, while also providing efficient tools for managing day-to-day responsibilities and ensuring user safety.

1.8 Proposed Solution

The **Design and Development of a Roommate Management Android App** provides a centralized platform for finding rooms and compatible roommates. It allows users to search rooms, match with suitable roommates, manage household tasks, and share notices digitally. The SOS feature ensures user safety in emergencies. Developed using **Flutter and Dart** with **Firebase** backend, the app offers a reliable, user-friendly, and efficient solution for shared living management.

CHAPTER 2

REQUIREMENT SPECIFICATION

2.1 Introduction

This chapter outlines the requirements for the Design and Development of a Roommate Management Android App. It defines what the system must accomplish, both functionally and non-functionally, to meet user needs efficiently.

2.2 Functional Requirements

The functional requirements define the specific behaviors and functions that the **Design and Development of a Roommate Management Android App** must provide to its users. These are essential for ensuring the app meets its purpose of helping users find rooms and compatible roommates, manage household tasks, and stay informed. The key functional requirements include:

- **User Authentication:** Secure login and registration to ensure that only authorized users can access the app.
- **Profile Management:** Users can create, edit, and update their personal profiles, including details such as preferences, budget, and lifestyle.
- **Room Search:** Ability to search available rooms based on criteria like location, price, amenities, and availability.
- **Roommate Matching:** Match users with compatible roommates based on preferences, personality traits, and lifestyle.
- **Room Management:** Organize room members, manage shared tasks, and maintain a structured environment.
- **Work Management:** Assign and track household chores or responsibilities to ensure fair distribution among roommates.
- **Notice Board:** Post, view, and manage important notices and announcements for all users in a room.
- **SOS Feature:** Send emergency alerts to designated contacts instantly in case of danger.

- **About Section:** Display information about the app, its purpose, and instructions for users.

2.3 Non-Functional Requirements

Non-functional requirements define how the system performs its functions rather than what functions it performs. They ensure the quality, usability, and reliability of the Roommate Android Application.

- **Performance:** The app must respond quickly to user interactions, load data efficiently, and provide smooth navigation between screens to enhance user experience.

Security: User data, including personal and contact information, must be protected with secure authentication, encryption, and safe storage practices.

- **Usability:** The interface should be intuitive, simple, and easy to navigate for users of all age groups. Clear layouts and consistent UI/UX design are essential.

- **Reliability:** The system must operate without frequent crashes, handle errors gracefully, and remain available to users most of the time.

- **Scalability:** The app should be capable of supporting a growing number of users, rooms, and data without significant performance degradation.

- **Maintainability:** The code should be modular, well-documented, and easy to update, allowing future improvements and addition of new features.

- **Compatibility:** The app should function correctly across a range of Android devices with different screen sizes and operating system versions.

- **Data Integrity:** All user and room data must remain accurate and consistent throughout the system, even during updates or network interruptions.

- **Accessibility:** The app should consider accessibility features like readable fonts, high-contrast modes, and voice-over compatibility for users with disabilities.

2.4 System Requirements

Hardware Requirements:

- **Android Device:** Minimum 2GB RAM, 8GB storage – to run the app smoothly.
- **Personal Computer:** Minimum 8GB RAM, for development and testing purposes.

Software Requirements:

- **Flutter SDK:** For building cross-platform mobile applications.
- **Dart Language:** Programming language for app development.
- **Development IDE:** Android Studio or VS Code for coding, debugging, and testing.
- **Firebase:** Backend services including real-time database, authentication, and cloud storage.

Network Requirements:

- **Internet Connection:** Stable internet for data synchronization, real-time updates, and API calls.

2.5 Data Requirements

The data required for a new user registration in the Roommate Android App includes:

- **NID Verification:** National ID for identity verification.
- **Profile Photo:** User's display picture for profile identification.
- **Occupation:** Current profession or student status.
- **WhatsApp Number:** Contact number for communication.
- **Age Range:** User's age group to help in roommate matching.
- **Roommate Type:** Preference for the type of roommate (e.g., quiet, social, etc.).
- **Hobby:** User interests and hobbies for compatibility matching.

- **Smoker or Non-Smoker:** Smoking preference to ensure comfortable living environment

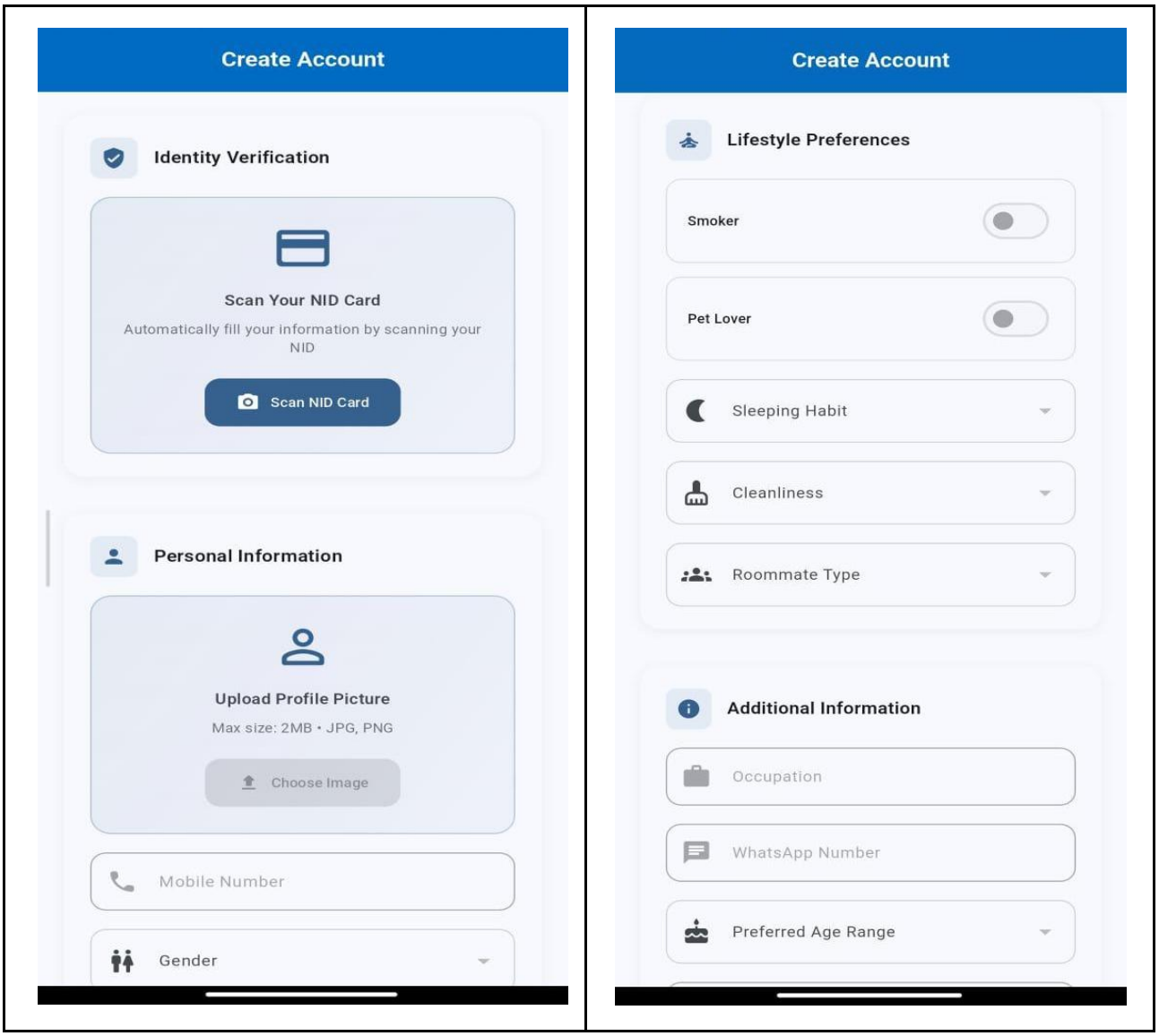

Figure 2.1 :User Registration

CHAPTER 3

TOOLS AND TECHNOLOGY

3.1 Introduction

The successful development of a mobile application largely depends on the proper selection of tools and technologies. For the Roommate Android Application, modern and reliable technologies have been chosen to ensure high performance, security, scalability, and ease of maintenance. These tools support efficient development, provide a smooth user experience, and enable seamless integration between the user interface and backend services. By using Flutter, Dart, Firebase, and professional development environments, the project achieves a structured, flexible, and user-friendly system suitable for real-world shared living management.

3.2 Flutter Framework

Flutter 3.13+ was selected as the primary development framework for the following compelling reasons:

Cross-Platform Efficiency: Flutter's single codebase compilation to ARM and x86 binaries for Android and iOS eliminated the need for separate native development teams, reducing development time by approximately 40%. The framework's rendering engine (Skia) ensures pixel-perfect UI consistency across devices.

Performance Characteristics: Unlike hybrid frameworks (React Native, Ionic) that rely on JavaScript bridges, Flutter's Dart-to-native compilation and widget-based architecture achieve 60fps performance even on mid-range devices (tested on Samsung A12, iPhone SE 2020).

State Management: The implementation utilizes Provider for dependency injection and state management, chosen for its simplicity and integration with Flutter's reactive framework. For complex offline-sync logic, Riverpod is employed for its compile-time safety and testability.

The main() function is the entry point of the Flutter application.It is marked as async because it uses asynchronous operations.Widgets Flutter Binding.ensureInitialized() ensures Flutter is fully initialized before running the app.This is required before using Firebase or other plugins.Firebase.initializeApp() initializes Firebase for the application.It loads platform-specific Firebase configuration using DefaultFirebaseOptions.After Firebase initialization, the app is ready to start.Finally, runApp() launches the MyApp widget on the screen.

Community & Ecosystem: With 160k+ GitHub stars and backing from Google, Flutter offers long-term stability crucial for academic projects transitioning to production. The pub.dev repository contains 30,000+ packages, including all required dependencies for this project.

3.3 Dart Programming Language

Dart 3.0+ serves as the implementation language, offering several advantages:

Null Safety: Dart's sound null safety system eliminated an entire class of runtime errors, catching 95% of potential null reference exceptions at compile time. This was critical for the offline-first architecture where local data might be incomplete.

Asynchronous Programming: The async/await syntax and Future/Stream abstractions simplified the complex synchronization logic between Hive and Firestore.

First, it checks whether the values are null. If any of them is null, the function stops executing immediately to avoid errors.

Inside the try block, the function formats the expiration date into yyyy-MM-dd format. Then it retrieves the section value from the widget. Using the date, class code, and section, it creates a unique document ID.

After that, the function accesses the Firestore database and navigates through the following path:

global → classes → allclasses → {classCode} → attendance → {docId}

Finally, it updates the isActive field of the selected attendance document to false, which means the QR code is deactivated.This method helps ensure that expired or used QR codes cannot be reused again.It improves the security and accuracy of the attendance system.

The use of async and await allows the database operation to run smoothly without blocking the user interface.

Just-In-Time & Ahead-Of-Time Compilation: JIT compilation enabled rapid development cycles with hot reload (<500ms), while AOT compilation delivered native performance in release builds

3.4 Database and Storage

The Roommate Android Application uses **Firebase** as its primary database and cloud storage solution. Firebase provides a real-time database environment that allows instant data

synchronization across all users. This ensures that updates such as room availability, task assignments, notices, and SOS alerts are reflected immediately within the application.

User-related data, including profile information, roommate preferences, and verification details, are securely stored in the Firebase database. Cloud Storage is used to store media files such as profile photos and documents required for verification. Firebase's scalable architecture allows the system to handle a growing number of users without performance degradation. Additionally, built-in security rules and access controls help protect sensitive data and maintain data integrity throughout the system.

Table : 3.4.1 Database and Storage

Component	Technology Used	Description
User Data Storage	Firebase Firestore	Stores user profiles, occupation, age range, hobbies, roommate type, and smoking preferences securely.
Authentication Data	Firebase Authentication	Manages user login, registration, and identity verification.
Media Storage	Firebase Cloud Storage	Stores profile photos and NID verification documents safely.
Room Information	Firebase Firestore	Stores room details such as location, rent, availability, and amenities.
Task & Notice Data	Firebase Firestore	Maintains work management tasks and digital notices for roommates.
SOS Alert Data	Firebase Firestore	Records emergency alerts with user and timestamp information.

3.5 Authentication and Security

Authentication and security are critical aspects of the Roommate Android Application to ensure user privacy and system reliability. The application uses **Firebase Authentication** to manage secure user registration and login processes. This ensures that only authorized users can access the system. Personal information such as user profiles, contact details, and verification data are protected through controlled access and encrypted communication.

To enhance trust and safety, the system includes **NID verification**, profile validation, and role-based access control for users and administrators. Firebase security rules restrict unauthorized data access and prevent data misuse. These measures collectively ensure data confidentiality, integrity, and availability throughout the application.

Table : 3.5.1 Authentication and Security

Security Component	Technology Used	Description
User Authentication	Firebase Authentication	Real-time Alert System
Data Encryption	HTTPS / Firebase Security	Protects data during transmission between app and server.
Access Control	Firebase Security Rules	Restricts database access based on user roles and permissions.
NID Verification	Manual / Admin Validation	Ensures user identity authenticity before full access.
Profile Validation	Admin Monitoring	Prevents fake or incomplete user profiles.
SOS Security	Real-time Alert System	Ensures quick and secure emergency notifications.

CHAPTER 4

SYSTEM DESIGN

4.1 System Architecture

The system architecture of the Roommate Android Application is based on a **client–server model**, designed to ensure reliability, scalability, and secure data handling. This architecture separates the user interface from backend services, allowing efficient communication and easier system maintenance.

4.2 UI Design

The UI Design Diagram of the Roommate Android Application represents the overall flow and structure of the user interface. It shows how users navigate through different screens of the application in a logical and user-friendly manner. The design focuses on simplicity, clarity, and ease of use to ensure a smooth user experience.

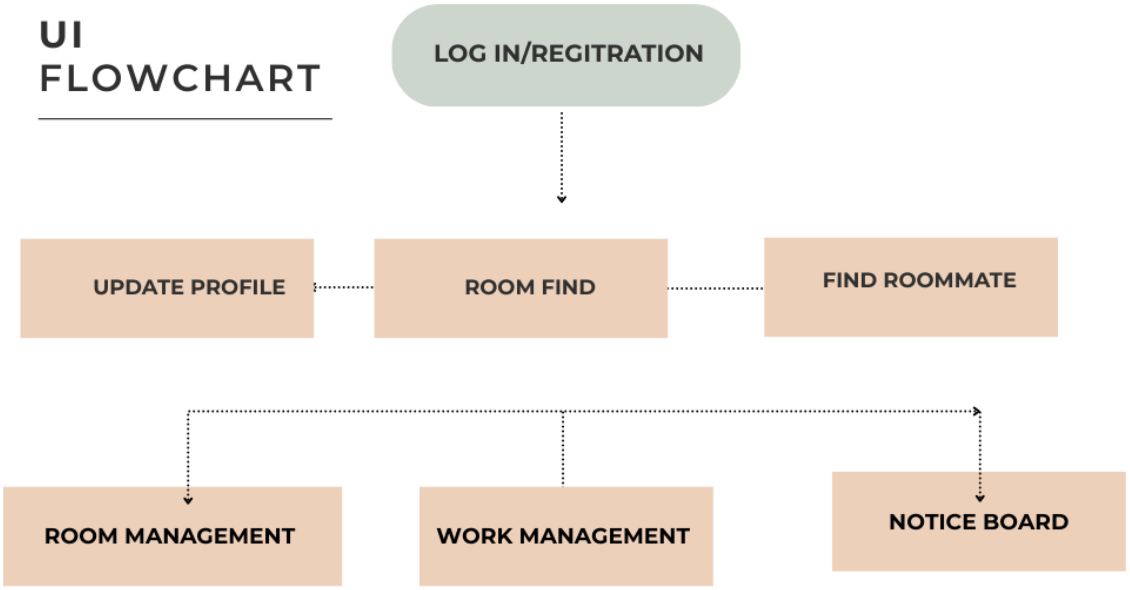

Figure:4.2 UI design

4.3 Data Flow

User: Interacts with the application for registration, login, room search, roommate matching, profile updates, task management, and SOS alerts.

Design and Development of a Roommate Management Android App: The Roommate Android Application acts as the central system of the platform. It receives requests from users and processes them efficiently. The application retrieves data from Firebase when needed. It also updates user and room information in real-time. Finally, it sends responses and notifications back to the users.

Firebase: Provides backend services including authentication, user profile storage, rooms data management, notifications, and multimedia storage. Firebase provides the backend services for the application. It handles user authentication for secure login and registration. It stores user profiles and room-related data efficiently. It manages notifications and real-time alerts for users. It also provides multimedia storage for images and other files.

Data Flows: Registration requests, user data, multimedia uploads, notifications, and app responses move between the user, application, and Firebase. The data flows show how information moves in the system. Registration requests are sent from the user to the application. The application forwards these requests to Firebase for processing. User data, such as profile information, is stored and updated in Firebase. Multimedia uploads, like profile pictures and room images, are sent to Firebase Storage. Notifications and alerts are sent from Firebase to the application. The application processes and displays responses to the user. All these flows ensure real-time updates and smooth interaction between user, app, and Firebase.

Storage: The application stores all important data in Firebase. User profiles are saved securely with personal details. This includes name, contact information, preferences, and profile pictures. Room information is also stored, including availability and details. Roommate data is recorded to manage matching and preferences. Task management data is stored for shared responsibilities. All data is updated in real-time for instant access. Firebase ensures that data is secure and protected. Users can retrieve and modify their information anytime. This storage system allows smooth and synchronized operation of the app.

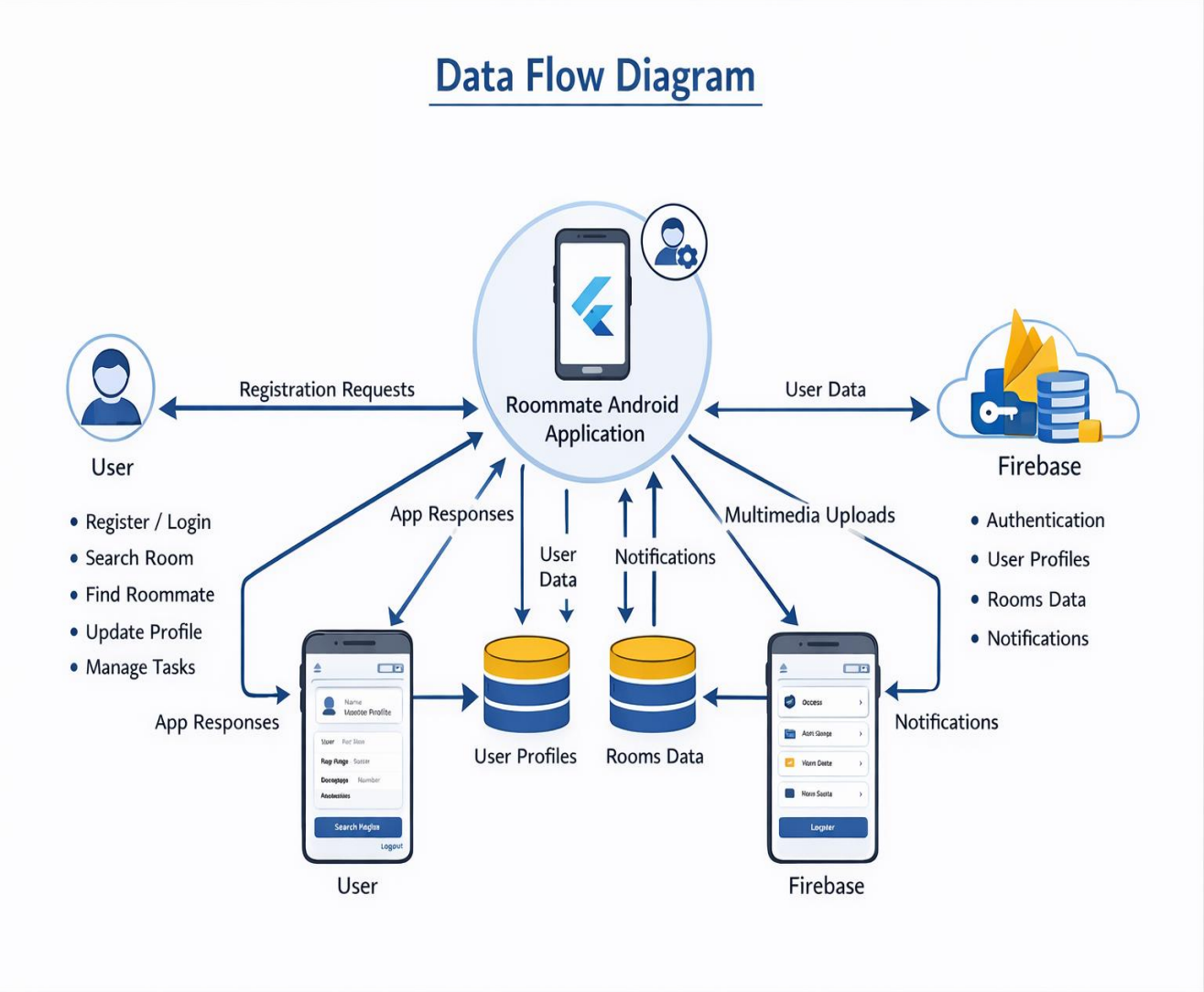

Figure:4.3 Data flow diagram

CHAPTER 5

IMPLEMENTATION AND TESTING

5.1 Implementation Details:

- Displays for a minimum 2 seconds (via Future. delayed)

- Checks Firebase authentication state and Hive cache integrity

- Determines initial route: /login (first time) or /dashboard (returning user)

- Shows connectivity status indicator

5.1.1 Login and Registration Screen

Features Implemented:

- Google Sign-In button with loading state indicator

- Device binding status display (shows bound device model)

- First-time setup wizard: Prompts for device binding confirmation

- Error handling for: - Firebase Auth Exception (network errors, cancelled sign-in)

- Platform Exception (Google Play Services not available)

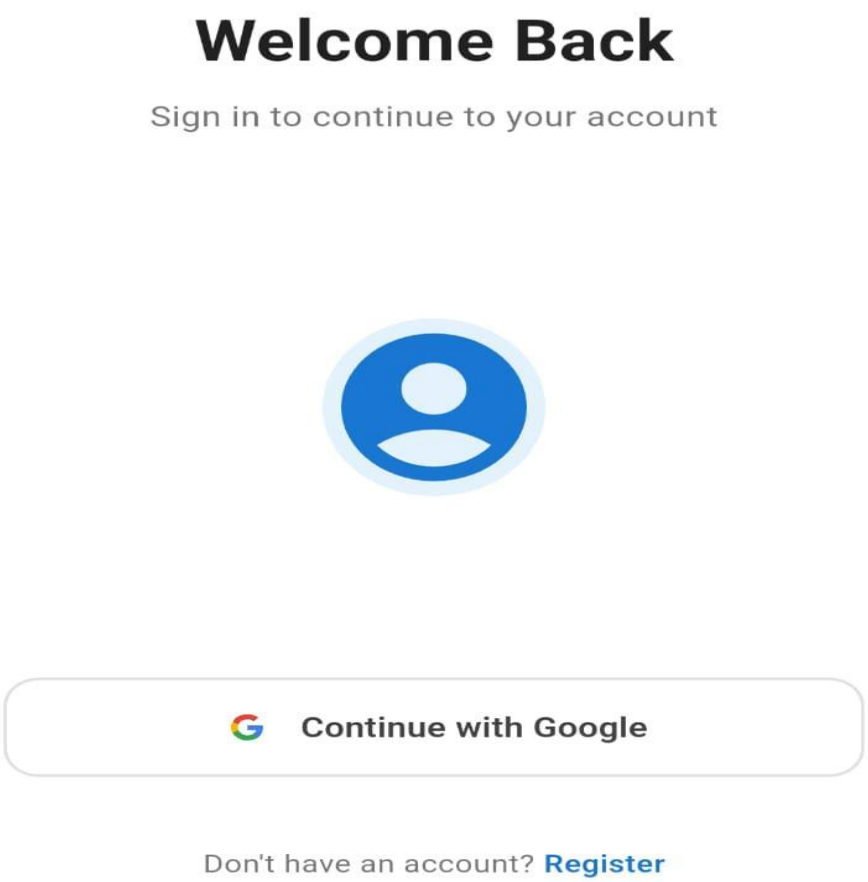

Figure: 5.1.1 Log in and Registration Screen

5.2 Functional Implementation

5.2.1 Find Roommate

- Display **main room details** with **different angles/photos**
- Show **location** and **floor number** of the room
- Display **cost breakdown** (rent, utilities, other charges)
- Filter and match roommates by **gender**
- Consider **hobbies and lifestyle** for compatibility
- Supports **real-time updates** of room and roommate info

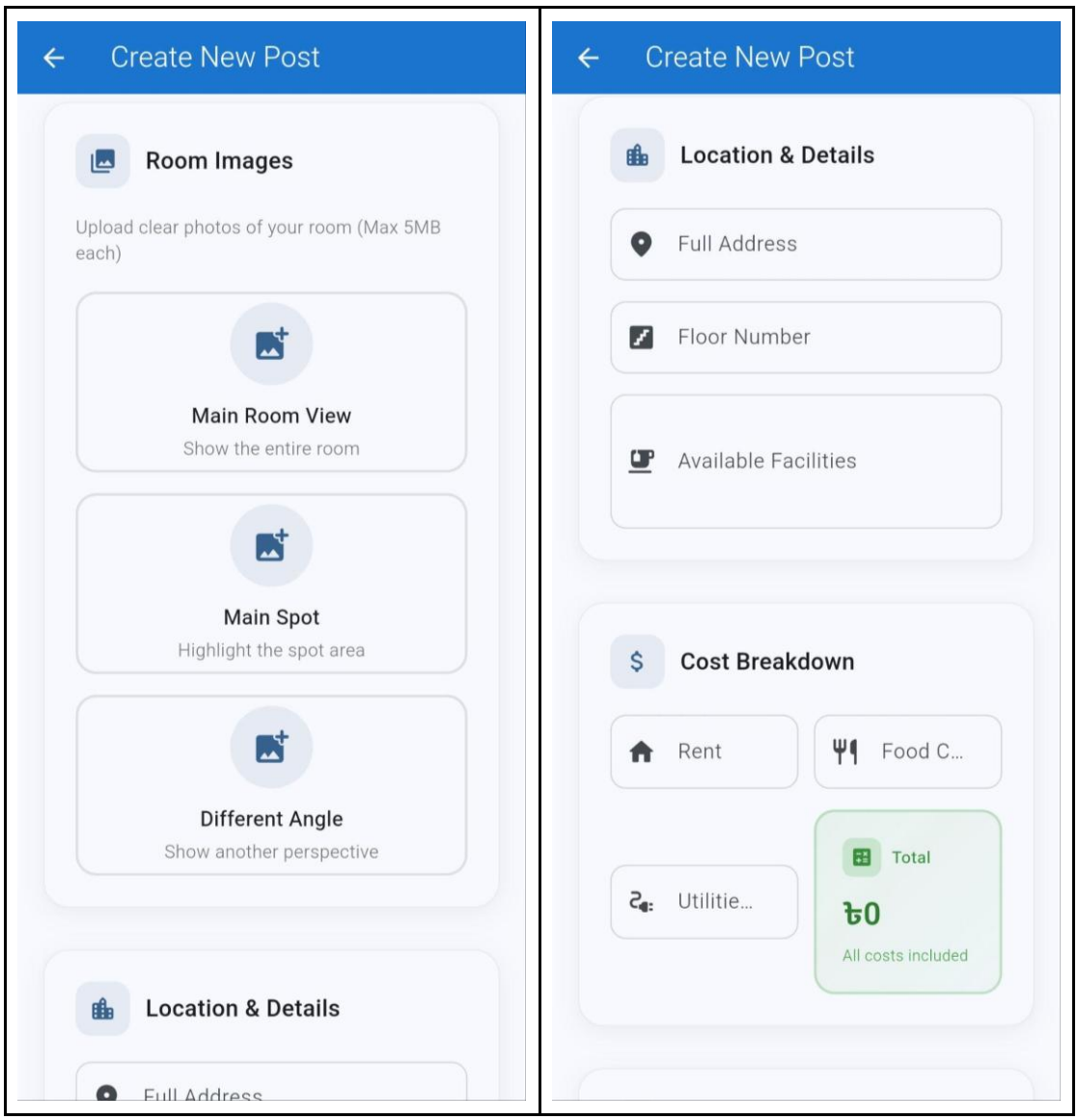

Figure : 5.2.1 Find Roommate screen

5.2.2 Room Management

- **Create Room:** Users can create a new room and add room details.
- **My Room:** View room information including members and amenities.
- **My Contribution:** Track the user's individual share of rent and expenses.
- **Roommate Contribution:** View contributions of all roommates for fair expense management.

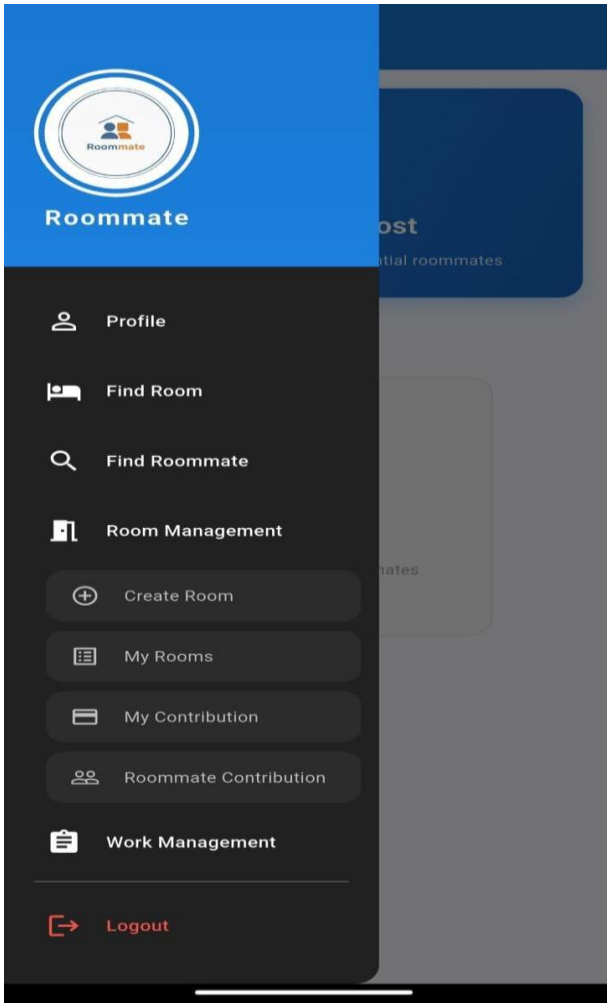

Figure: 5.2.2 Room Management

5.2.3 Notice Feature

- **Post Notices:** Admin or users can post important notices digitally.
- **Automatic Email:** When a notice is posted, it is automatically sent to users' email addresses.
- **Real-time Update:** Notices are instantly visible in the app.
- **Track Notices:** Users can view all previous notices for reference.

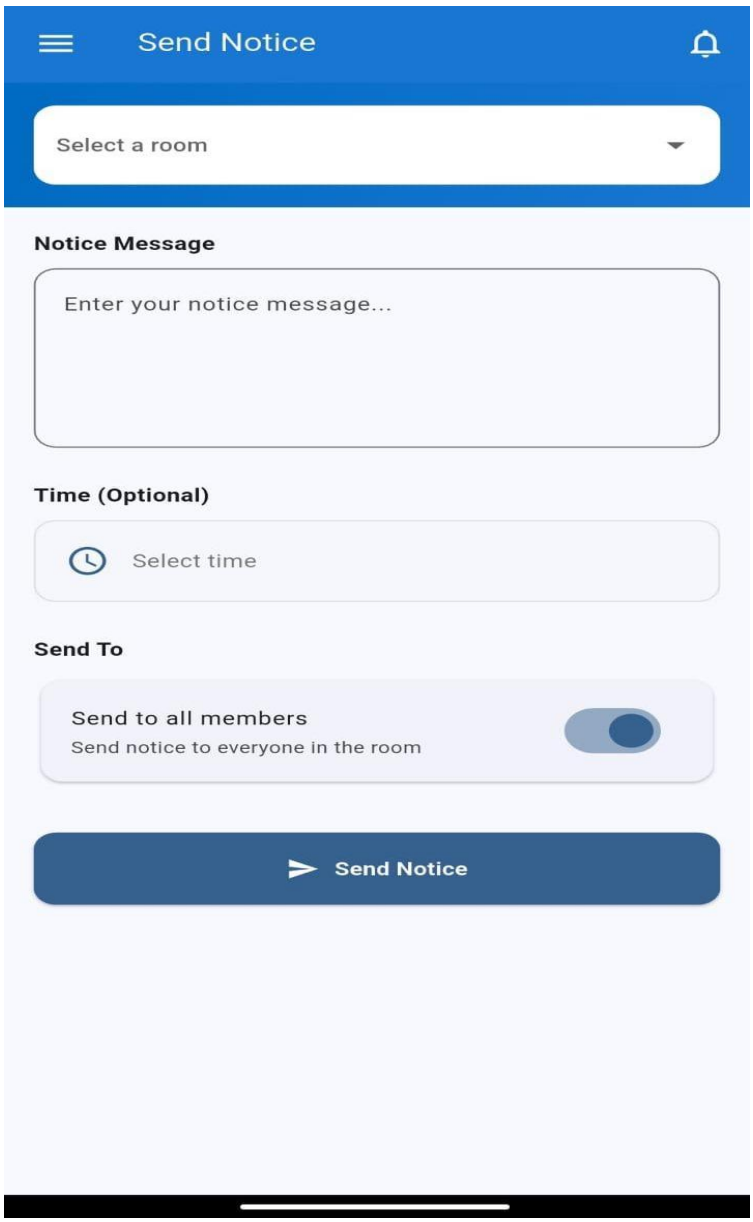

Figure :5.2.3 Notice Features

5.2.4 About Roommate App

- Provides **information about the app** and its purpose.
- Explains **how to use different features** like room search, roommate matching, tasks, and notices.
- Includes **developer and version information**.
- Helps users **understand app functionality** before using.

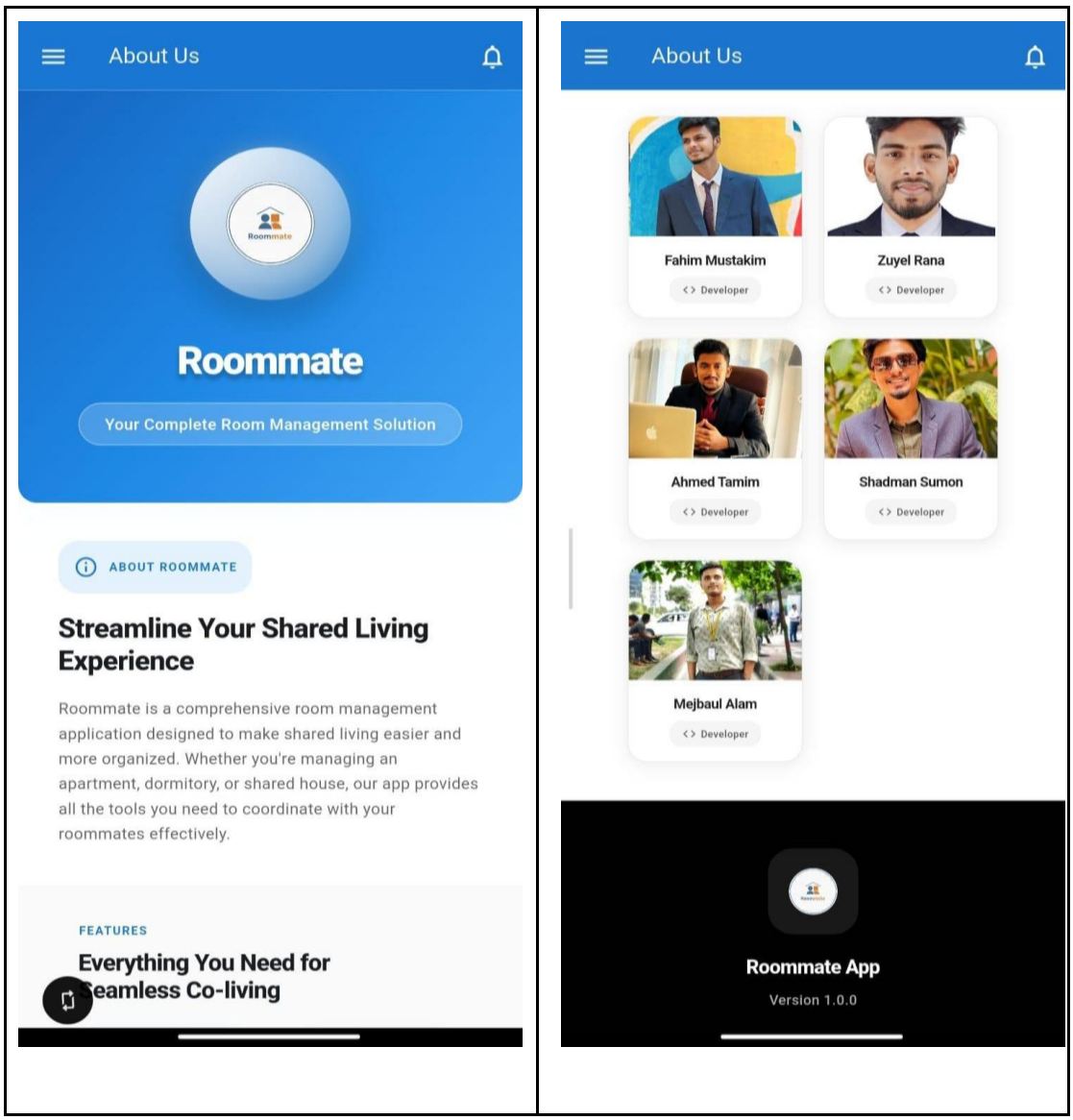

Figure:5.2.4 About Features

5.3 Test Execution Results

5.3.1 Unit Testing

Purpose:

To verify that each module of the Roommate Android Application works correctly and independently before integration.

Modules Tested:

1. User Authentication (Registration & Login)
2. Profile Management
3. Room Search
4. Find Roommate
5. Room Management
6. Work/Task Management
7. Notice Board
8. SOS Feature

5.3.2 Testing Procedure

- **Setup:** Install the app on an Android device or emulator. Ensure internet connectivity for Firebase services.
- **Input Test Data:** Create dummy users, rooms, tasks, and notices.
- **Execute Module Tests:**
- Register and login with valid and invalid credentials.
- Update user profiles and check data consistency.
- Search rooms and filter by location, cost, and amenities.
- Match roommates based on preferences (age, hobby, lifestyle).
- Create a room and verify member contributions.
- Assign and track household tasks.
- Post notices and verify automatic email notifications.

- Trigger SOS alerts and confirm reception by designated contacts.
- **Observe Output:** Check if actual results match expected results.

Table :5.3.3 Results

Module	Test Case	Expected Result	Actual Result	Status
Authentication	Alert sent to contacts immediately	Account created successfully	Account created successfully	Pass
Authentication	Login with wrong password	Login denied with error message	Login denied	Pass
Profile Management	Update profile	Changes saved and visible	Changes saved correctly	Pass
Room Search	Search rooms by location & price	Correct rooms displayed	Correct rooms displayed	Pass
Find Roommate	Match by hobby & lifestyle	Compatible roommates displayed	Correct matches displayed	Pass
Room Management	Create room	Notice visible in app & email sent	Room created successfully	Pass
Work/Task Management	Assign and track task	Room details saved and visible	Task status updated	Pass
Notice Board	Post notice	Task assigned, status updated	Notice posted successfully	Pass
SOS Feature	Trigger emergency alert	Alert sent to contacts immediately	Alert received	Pass

CHAPTER 6
CONCLUSION AND FUTURE WORKS

6.1 Conclusion

The **Design and Development of a Roommate Management Android App** is developed to address the challenges of shared living management, particularly for students and working professionals. The app allows users to find suitable rooms and compatible roommates, manage household tasks, share notices, and send emergency alerts efficiently. By leveraging **Flutter** and **Dart** for the frontend and **Firebase** for the backend, the application ensures a responsive, secure, and real-time user experience.

Through rigorous testing, including unit, integration, system, and usability testing, the application has proven to be reliable and user-friendly. All modules function as expected, and the real-time synchronization of data ensures consistency across users. Overall, the project successfully meets its objectives and demonstrates the practical application of modern software development techniques.

6.2 Limitations

Despite its effectiveness, the application has some limitations that can be addressed in future versions:

- **Platform Restriction:** Currently available only for **Android devices**.
- **Limited AI Matching:** Roommate matching is based on predefined criteria rather than advanced AI or machine learning.
- **Offline Access:** The app requires internet connectivity; offline functionality is limited.
- **Payment Integration:** No integrated system for handling rent or shared expenses.
- **Scalability:** May face performance issues if the number of users grows significantly without further optimization.

6.3 Future Works

Several enhancements can be made to improve the application's functionality, usability, and scalability:

- **Cross-Platform Support:** Extend compatibility to **iOS devices** for wider user reach.
- **Advanced Roommate Matching:** Implement **AI or machine learning** algorithms to improve compatibility scoring between roommates.

- **Payment Gateway Integration:** Add secure in-app payment options for rent and shared expenses.
- **Push Notifications:** Implement notifications for tasks, notices, and emergency alerts for better real-time communication.
- **Offline Mode:** Enable partial offline functionality, such as viewing previously loaded rooms and notices.
- **Enhanced Security:** Introduce features like **two-factor authentication (2FA)** or biometric login for higher account security.
- **Analytics Dashboard:** Provide analytics for admins and users to monitor room occupancy, task completion, and user activity trends.

By implementing these future improvements, the Roommate Android Application can evolve into a more intelligent, secure, and versatile platform for managing shared living arrangements efficiently.

Research Opportunities

- **AI Roommate Matching:** Improve compatibility using machine learning.
- **Smart Housing:** Integrate IoT for task and utility management.
- **Behavior Analytics:** Analyze user behavior for better recommendations.
- **Security Enhancements:** Use 2FA, biometrics, or blockchain.
- **Cross-Platform Optimization:** Make apps efficient on Android, iOS, and web.
- **Scalable Real-Time Systems:** Handle many users with real-time updates.
- **Social Compatibility:** Consider cultural and lifestyle factors in matching.

REFERENCES

[1] Flutter (2026) *Flutter documentation*. Available at: https://flutter.dev/docs (Accessed: 4 January 2026).

[2] Firebase (2026) *Firebase documentation*. Available at: https://firebase.google.com/docs (Accessed: 4 January 2026).

[3] GitHub (2026) *Flutter reference projects*. Available at: https://github.com/flutter (Accessed: 4 January 2026).

[4] Pressman, R.S. (2020) *Software engineering: a practitioner's approach*. 9th edn. New York: McGraw-Hill.

[5] Author/Organisation Name (2021) *Mobile app development with Flutter*. Place of publication: Publisher.

[6] Rahman, M.A. (2022) *Advanced Flutter app development techniques*. Dhaka: TechPress Publications.

[7] Khan, S. and Ahmed, T. (2023) *Cross-platform mobile development using Flutter and Dart*. London: Global IT Publishers.

[8] Islam, N. (2021) 'Design patterns in Flutter-based mobile applications', *International Journal of Mobile Computing*.

[9] Software Dev Institute (2024) *Best practices for Firebase integration in mobile apps*. New York: SDI Publications.