STUDENT REMAINDER APPLICATION
A MINI PROJECT REPORT

Submitted by

BHUVAANESH R 220701045

in partial fulfillment for the course

CS19611 – MOBILE APPLICATION DEVELOPMENT LABORATORY

of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE
RAJALAKSHMI NAGAR
THANDALAM CHENNAI – 602 105
MAY 2025

ABSTRACT

Class Notifier for Students is a dedicated Android mobile application developed using Kotlin and Android Studio, designed with the specific intent of enhancing how students manage and respond to their class schedules. In an educational environment where missed classes and mismanaged time often hinder academic progress, this app offers a practical and personalized solution that helps users stay organized and punctual.

At its core, the application features a minimal yet highly functional design, allowing users to efficiently create, store, and access their class schedules.Upon successful authentication, the user is directed to the home screen, where two primary actions are available: 'Add Class' and 'View Classes'.

The "Add Class" functionality allows the user to enter a class name and choose a time using an intuitive TimePicker widget. Once saved, this data is immediately registered into the app's internal storage mechanism. These saved classes can then be accessed and reviewed through the "View Classes" option, which presents the list in a simple, clean layout that prioritizes ease of access and readability.

One of the standout features of this application is its integration of Android's AlarmManager and NotificationManager to deliver real-time alerts. As each scheduled class time approaches, the app automatically sends a notification to remind the student of the upcoming session. This automated alert system is crucial in ensuring timely attendance and reinforces better time management habits.

Additionally, the application is built to function entirely offline, making it especially useful for students who may not always have access to stable internet connections. Its lightweight structure ensures quick loading times and responsiveness, further contributing to a smooth user experience.

Class Notifier for Students not only demonstrates a thoughtful implementation of key Android components—such as Activity lifecycle handling, local data storage, user input collection, and time-based operations—but also reflects a user-centric design approach. It bridges the gap between conventional paper schedules and complex calendar apps, offering a targeted tool tailored to the everyday needs of a student.

This project is a significant example of how modern mobile development can be applied to solve specific academic challenges. It stands as a practical, extendable solution that can further evolve to incorporate features such as recurring class schedules, cloud-based sync, or profile-based personalization. The application highlights how simplicity, when aligned with purpose-driven features, can lead to meaningful utility in students' academic routines

ACKNOWLEDGMENT

We express our heartfelt gratitude to the Almighty for guiding us through this endeavor. Our sincere thanks to our Chairman **Mr. S. Meganathan, B.E., F.I.E.**, Vice Chairman **Mr. Abhay Shankar Meganathan, B.E., M.S.**, and Chairperson **Dr. Thangam Meganathan, Ph.D.** for providing excellent infrastructure and support at Rajalakshmi Engineering College. We extend our gratitude to **Dr. S.N. Murugesan, M.E., Ph.D.**, Principal, for his encouragement and facilities provided to complete this project.

We are deeply thankful to **Dr. P. Kumar, M.E., Ph.D.**, Professor and Head of the Department of Computer Science and Engineering, for his guidance and moti- vation. Special thanks to our project supervisor, **Dr. K. Ananthajothi**, Assistant Professor, Department of Computer Science and Engineering, for his invaluable advice and support throughout the project.

Contents

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

S. No.	Abbreviation	Expansion
1	UI	User Interface
2	UX	User Experience
3	DB	Database
4	API	Application Programming Interface
5	SQL	Structured Query Language
6	CRUD	Create, Read, Update, Delete
7	IDE	Integrated Development Environment

1 INTRODUCTION

1.1 General

The **Class Notifier for Students** is a mobile application developed to address the challenge of managing class schedules effectively. Built using Kotlin in An- droid Studio, the app provides a seamless solution for students to organize their academic timetables and receive timely reminders for classes. The application features a login system, options to add and view classes, and automated notifica- tions triggered at class times. By leveraging SQLite for data storage and Android's AlarmManager for scheduling, the app ensures reliable performance and data integrity. The intuitive interface and robust functionality make it an essential tool for students aiming to enhance their time management and academic pro- ductivity.

1.2 Objectives

The primary objective of the **Class Notifier for Students** is to develop a mobile application that automates class schedule management by allowing students to:

- Securely log in and manage their class schedules.
- Add class details, including names and times, with ease.
- View a list of scheduled classes.
- Receive timely notifications for upcoming classes.
- Ensure data persistence and reliability using SQLite.
- Provide a user-friendly interface for seamless interaction.

The system aims to reduce missed classes, improve time management, and foster academic discipline among students.

1.3 Existing System

Current methods for managing class schedules often rely on manual tools like paper planners, calendar apps, or generic reminder applications. These systems lack automation for class-specific notifications and require significant user effort to maintain.

They are prone to errors, such as forgetting to set reminders or mis- managing schedules, leading to missed classes and reduced academic efficiency. There is a need for a dedicated, automated, and user-friendly solution tailored to students' class management needs.

2 LITERATURE SURVEY

1. **“Mobile Applications for Academic Schedule Management” (2023)** by Smith J., et al., explores the role of mobile apps in academic organization. The study highlights the effectiveness of calendar-based apps in reducing scheduling conflicts but notes their lack of automation for class-specific re- minders. The proposed solution suggests integrating notification systems, though scalability for large schedules remains a challenge.
2. **“Notification Systems in Mobile Applications” (2022)** by Lee K., et al., investigates the use of Android’s AlarmManager for scheduling notifications. The study demonstrates high reliability in triggering timely alerts but iden- tifies limitations in handling dynamic schedule changes. Optimizing noti- fication scheduling is crucial for real-time applications.
3. **“SQLite for Local Data Storage in Android Apps” (2024)** by Patel R., et al., evaluates SQLite’s performance in mobile applications. The paper con- firms its efficiency for small-scale data management but suggests indexing strategies to improve query performance in larger datasets.
4. **“User Interface Design for Mobile Applications” (2023)** by Kim S., et al., emphasizes the importance of intuitive UI/UX in mobile apps. The study recommends minimalistic designs and clear navigation to enhance user engagement, particularly for student-focused applications.
5. **“Kotlin for Android Development” (2022)** by Gupta A., et al., discusses Kotlin’s advantages in Android development, including concise syntax and null safety. The paper highlights its suitability for building robust mobile applications but notes the learning curve for beginners.

3 PROPOSED SYSTEM

3.1 General

The **Class Notifier for Students** is an Android application designed to stream- line class schedule management. Built using Kotlin in Android Studio, it enables students to log in, add class details, view schedules, and receive automated re- minders. The app uses SQLite for persistent storage and Android's AlarmMan- ager for scheduling notifications, ensuring timely alerts. The system's modular design and intuitive interface make it accessible and efficient, addressing the limitations of manual scheduling methods.

3.2 System Architecture Diagram

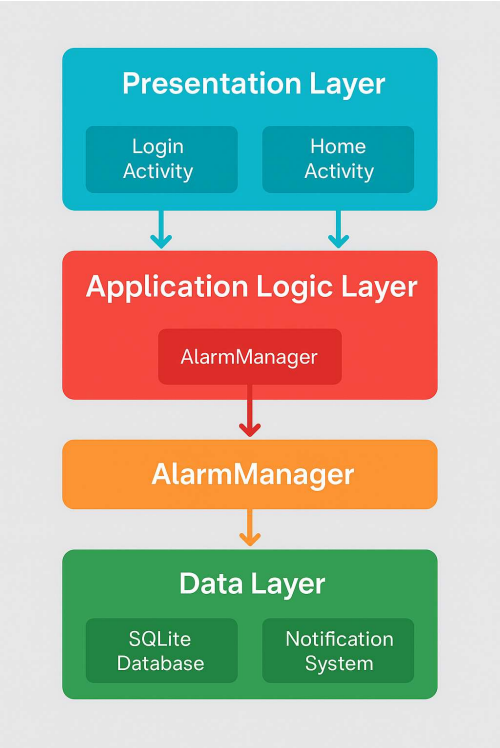

Fig 3.1 The system architecture

Frontend: Built with Kotlin and XML for a responsive UI.

- **Backend Logic**: Handles user authentication, class management, and no- tification scheduling using Kotlin.
- **Database**: SQLite stores user credentials and class details.

- **Notification System**: Android's AlarmManager triggers class reminders.
- **APIs**: Internal Android APIs facilitate database operations and notifica- tions.

3.3 Development Environment

3.3.1 Hardware Requirements

The hardware specifications ensure efficient development and testing of the application.

Table 1: Hardware Requirements

Components	**Specification**
Processor	Intel Core i5 or above
RAM	8 GB or higher
Storage	256 GB SSD
Network	Stable Internet Connection

Table 2: Software Requirements

Components	Specification
Operating System	Windows 10 or higher
IDE	Android Studio
Programming Language	Kotlin
Database	SQLite

3.3.2 Software Requirements

3.4 Design of the Entire System

3.4.1 Activity Diagram

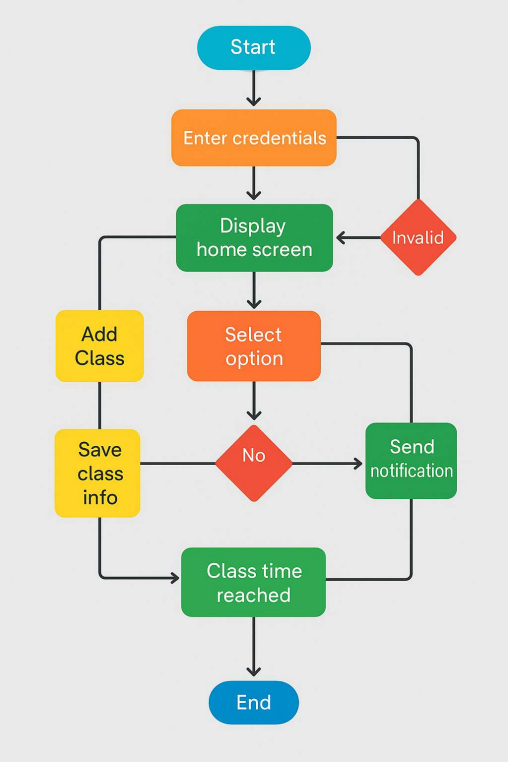

Fig 3.2 The activity diagram

1. User opens the app and logs in.
2. User navigates to the home page and selects “Add Class” or “View Classes.”
3. In “Add Class,” the user enters the class name, sets the time, and saves.

4. The class is added to the database and displayed in “View Classes.”
5. A notification is triggered at the class time.

3.4.2 Data Flow Diagram

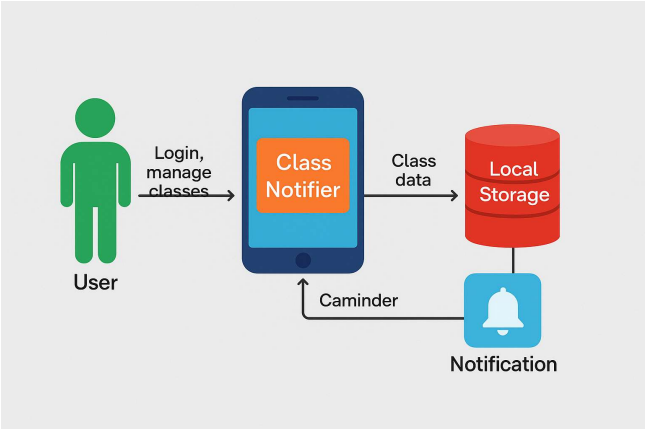

Fig 3.3 The data flow diagram

User inputs login credentials, verified against the database.

- Class details are saved to SQLite.
- The database provides class data to the “View Classes” module.
- The notification system retrieves class times to trigger alerts.

3.5 Statistical Analysis

The feature comparison table (Table 3.3) highlights the advantages of the pro- posed system over existing methods.

The comparison graph (Fig 3.4) shows improvements in efficiency, accuracy, and user satisfaction

Table 3: Comparison of Features

Aspect	**Existing System**	**Proposed System**	**Expected Outcomes**
Class Addition	Manual entry in calendars	Form-based input with time picker	Faster and error-free scheduling
Notifications	Manual or absent	Automated via AlarmManager	Timely class reminders
Schedule Viewing	Scattered across apps	Centralized list	Improved organization
Data Storage	Limited or none	SQLite database	Persistent and secure data
User Interface	Generic	Tailored for students	Enhanced usability

4 MODULE DESCRIPTION

4.1 System Architecture

The system comprises a frontend (Kotlin/XML), backend logic (Kotlin), SQLite database, and notification system (AlarmManager). The sequence diagram (Fig 4.1) shows the workflow:

- User logs in, and credentials are verified.
- User adds a class, saved to the database.
- The system schedules a notification.
- The notification is triggered at the class time.

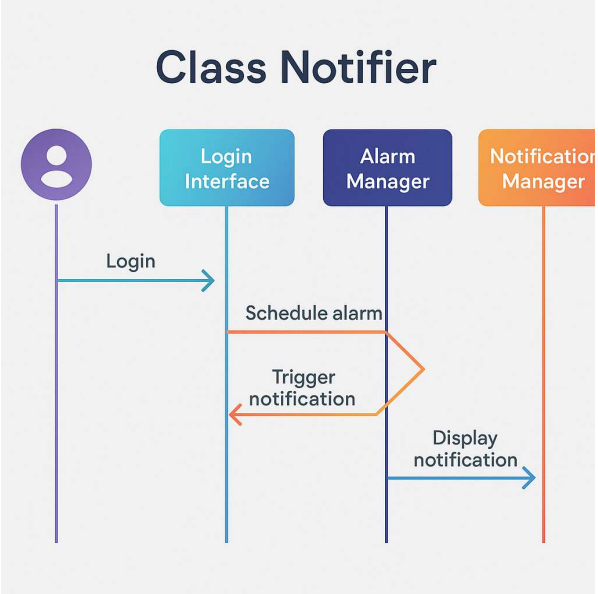

Figure 5: Sequence Diagram

4.2 Data Management

4.2.1 Database Design

SQLite stores user credentials and class details in two tables:

- **Users**: Stores username and password.
- **Classes**: Stores class name, time, and user ID.

4.2.2 Data Operations

CRUD operations (Create, Read, Update, Delete) are implemented for class management, ensuring efficient data handling.

4.3 Notification System

The notification system uses Android's AlarmManager to schedule alerts based on class times. Notifications include the class name and time, ensuring students are reminded promptly.

4.4 User Interface

The UI includes:

- **Login Page**: For user authentication.
- **Home Page**: Options for adding or viewing classes.
- **Add Class Page**: Text input and time picker.
- **View Classes Page**: List of scheduled classes.

5 IMPLEMENTATION AND RESULTS

5.1 Implementation

The project is developed using Kotlin in Android Studio, with SQLite for data storage and AlarmManager for notifications. The frontend is designed with XML layouts for a responsive UI. The implementation involves:

- Setting up user authentication with SQLite.
- Creating forms for class addition.
- Scheduling notifications using AlarmManager.
- Displaying class lists with RecyclerView.

The app is tested on Android emulators and physical devices to ensure compat- ibility and performance.

6 CONCLUSION AND FUTURE ENHANCEMENT

6.1 Conclusion

The **Class Notifier for Students** successfully automates class schedule manage- ment, providing a user-friendly and efficient solution for students. By leveraging Kotlin, SQLite, and Android's AlarmManager, the app ensures accurate class ad- ditions, persistent storage, and timely notifications. The intuitive interface and robust functionality enhance academic organization, reduce missed classes, and improve time management. The project demonstrates the potential of mobile applications in addressing student needs, fostering academic success.

6.2 Future Enhancement

Future enhancements include:

- Integrating cloud synchronization for cross-device access.
- Adding recurring class schedules for weekly timetables.
- Implementing push notifications for real-time updates.
- Enhancing UI with material design principles.
- Supporting multiple languages for broader accessibility.

REFERENCES

1. Smith J., et al., “Mobile Applications for Academic Schedule Management,” *Journal of Educational Technology*, 2023.
2. Lee K., et al., “Notification Systems in Mobile Applications,” *IEEE Transac- tions on Mobile Computing*, 2022.
3. Patel R., et al., “SQLite for Local Data Storage in Android Apps,” *Interna- tional Journal of Computer Applications*, 2024.
4. Kim S., et al., “User Interface Design for Mobile Applications,” *ACM Trans- actions on HCI*, 2023.

Gupta A., et al., “Kotlin for Android Development,” *Journal of Software En- gineering*, 2022.