MediMinder App

CS19643 – PROJECT REPORT

Submitted by

JODERICK SHERWIN J 2116220701109

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE,

ANNA UNIVERSITY, CHENNAI

May 2025

BONAFIDE CERTIFICATE

Certified that this project report "MediMinder App"– *Designed for the students of Computer Science and Engineering*" is the bonafide work of **"JODERICK SHERWIN J (2116220701109)"** who carried out the project work under my supervision.

SIGNATURE

Mr. V. Karthick.,
EXAMINER,
Assistant Professor
Department of Computer Science and Engineering,
Rajalakshmi Engineering College,
Chennai-602 105.

Submitted for the project viva-voce examination held on_____.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

In the modern healthcare ecosystem, medication adherence remains a significant challenge, particularly for elderly individuals, patients managing chronic conditions, and those under complex polypharmacy regimens. Non-adherence—defined as missed doses, incorrect timing, or inconsistent consumption of prescribed medicines—can result in severe clinical implications, including disease progression, increased hospitalization rates, and higher treatment costs. Addressing this critical concern, **MediMinder** is a comprehensive, offline-capable Android application developed with the goal of improving timely medication intake through structured digital reminders and user-centered design.

MediMinder is architected to function as a personal medication assistant, helping users schedule, track, and manage their medication intake on a daily basis. Developed using **Java** in **Android Studio**, the app integrates core Android components such as **AlarmManager**, **BroadcastReceiver**, and **SQLite**, ensuring seamless scheduling, persistent data storage, and reliable background notification handling. The application supports full **CRUD** (Create, Read, Update, Delete) operations for medicine reminders, enabling users to:

- Add new medicine entries with details such as name, dosage, and specific intake times.
- Set precise alarms for each reminder based on user-defined schedules (daily, custom intervals, etc.).
- Receive system-generated notifications via AlarmManager and BroadcastReceiver, even if the app is not actively running in the foreground.
- Update medicine information in real-time or remove expired medications from the schedule.

The **AlarmManager** is responsible for setting one-time or repeating alarms, while the **BroadcastReceiver** captures and reacts to these alarm events to generate user notifications. This ensures that reminders remain reliable across device restarts, background operations, and various Android OS versions. SQLite serves as the local database engine, offering lightweight and efficient data management directly on the user's device without internet connectivity, thus maintaining user privacy and availability in offline environments.

The app follows a **modular architecture** using separate Activities for navigation, ListAdapters for dynamic data rendering, and utility classes for alarm and database handling. This modularity not only promotes code reusability and maintainability but also ensures easier debugging and extension of features in the future. The UI is designed using **Material Design principles**, prioritizing accessibility, readability, and intuitive interaction. Special attention is given to senior users, with large buttons, clean fonts, minimal visual clutter, and simple navigation.

From a development standpoint, MediMinder provides a hands-on learning experience in multiple domains:

- **Mobile development fundamentals**, including lifecycle management, intent broadcasting, and permission handling.

- **Local storage and persistence**, using structured relational data through SQLite.
- **Background task scheduling**, leveraging system services to optimize battery and performance.
- **Notification management**, for timely and actionable user engagement.
- **User-centric interface design**, focusing on accessibility, simplicity, and ease of use.

Furthermore, MediMinder is designed with extensibility in mind. Future enhancements can introduce:

- **Cloud synchronization** using Firebase or Google Drive APIs, to support remote access and backup.
- **Multi-user support** and **user authentication**, to allow families or caregivers to manage reminders for multiple patients.
- **Speech-based input/output**, to accommodate visually impaired users or those with disabilities.
- **Smart analytics**, such as tracking medication compliance over time, generating reports, and integrating predictive adherence models.
- **Inventory tracking**, which alerts users when a medicine stock is low and may need replenishment.

The significance of MediMinder lies not just in its technical implementation but in its practical impact on public health. By empowering users with a simple tool to manage their medications independently, the app contributes to increased compliance, improved treatment outcomes, and a reduced burden on caregivers and healthcare providers.

In conclusion, MediMinder exemplifies how mobile technology can be harnessed to solve real-world healthcare problems through thoughtful design, system integration, and accessibility-focused development. The project demonstrates core Android development competencies, and lays a solid foundation for future health-tech innovations aimed at personalized and preventative care.

ACKNOWLEDGMENT

Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavour to put forth this report. Our sincere thanks to our Chairman **Mr. S. MEGANATHAN, B.E, F.I.E.,** our Vice Chairman **Mr. ABHAY SHANKAR MEGANATHAN, B.E., M.S.,** and our respected Chairperson **Dr. (Mrs.) THANGAM MEGANATHAN, Ph.D.,** for providing us with the requisite infrastructure and sincere endeavouring in educating us in their premier institution.

Our sincere thanks to **Dr. S.N. MURUGESAN, M.E., Ph.D.,** our beloved Principal for his kind support and facilities provided to complete our work in time. We express our sincere thanks to **Dr. P. KUMAR, M.E., Ph.D.,** Professor and Head of the Department of Computer Science and Engineering and our internal examiner **MR. V. KARTHICK,** Department of Computer Science and Engineering for his useful tips during our review to build our project.

JODERICK SHERWIN J 2116220701109

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

The effective management of medication intake is a critical component of healthcare, especially in cases involving chronic illnesses, elderly patients, post-operative care, and multi-drug regimens. Despite clear prescription instructions, studies have shown that a significant percentage of patients fail to take their medications as prescribed, leading to avoidable health complications, reduced treatment efficacy, increased hospitalization rates, and even death in severe cases. According to the World Health Organization (WHO), medication non-adherence contributes to nearly 50% of treatment failures. In most cases, the root causes are simple: forgetfulness, lack of awareness, or confusion due to complex drug schedules.

With the rapid proliferation of smartphones and the increasing digitization of healthcare, mobile applications have emerged as powerful tools to assist patients in adhering to their medication regimens. These apps can serve as constant, non-intrusive companions, sending reminders, storing records, and providing users with better control over their health. However, many existing medicine reminder apps are either overly complex, cluttered with unnecessary features, or dependent on internet connectivity—making them difficult to use for senior citizens or people in low-connectivity areas.

To address these challenges, MediMinder was conceptualized and developed as a smart medicine reminder Android application that focuses on simplicity, efficiency, and offline accessibility. The app is specifically designed to assist users in scheduling medicine reminders, receiving timely notifications, and managing medication records independently without requiring technical expertise or continuous internet access.

1.1 Purpose of the Project

The main purpose of the MediMinder application is to provide users with a reliable, intuitive, and accessible tool that aids in the consistent intake of prescribed medications. The app allows users to:

- Create personalized medicine reminders by specifying medicine names, dosages, and intake times.
- Receive time-based notifications through Android's system-level services.
- Store and manage medicine records securely on the device using SQLite.
- Update or delete outdated medication entries without any hassle.

The application was developed using Java in Android Studio, following modular development practices to ensure code readability and maintainability. The architectural components include multiple Activities, Custom Adapters, Broadcast Receivers, and a local SQLite database, enabling efficient background processing and user interaction. The notifications are triggered using AlarmManager and delivered using BroadcastReceiver, ensuring reliability even when the app is not running in the foreground.

1.2 Motivation

The motivation behind MediMinder stems from real-world observations and healthcare data that highlight the dangers of irregular medication intake. Elderly people often forget their doses, and individuals with busy schedules may overlook their medication timing. Commercial solutions often come with limitations such as intrusive ads, paywalls, complex navigation, or a requirement for constant internet connectivity.

This project aims to provide a clean, ad-free, user-friendly, and offline-capable solution that bridges this gap. It empowers users by giving them the tools to manage their health independently, regardless of their technological proficiency or location. The UI/UX is tailored with accessibility in mind, using large buttons, intuitive icons, and minimalistic layouts ideal for visually impaired or elderly users.

1.3 Technological Relevance

MediMinder offers a practical implementation of several core Android development concepts including:

- Background processing using AlarmManager and BroadcastReceiver.
- Local data persistence using SQLite, ensuring secure offline storage.
- Activity lifecycle management, ensuring smooth user experience across app restarts.
- User notifications, including custom intents and dynamic alarms.
- Modular architecture, separating UI components, logic layers, and system services.

This project not only solves a critical healthcare challenge but also serves as a comprehensive case study for developers learning mobile application development. It integrates multiple Android components, adheres to design best practices, and applies theoretical knowledge in a real-world scenario.

1.4 Scope of the Application

MediMinder is designed to serve individuals of all ages, with a particular focus on:

- Senior citizens managing multiple prescriptions.
- Patients recovering from surgery or treatment.
- Individuals with chronic diseases like diabetes, hypertension, or asthma.
- Caregivers managing medication for dependents.

While the current version supports core CRUD operations and time-based reminders, future updates may include:

- Cloud-based data backup and synchronization using Firebase.
- Speech-to-text and voice alert features for accessibility.
- Push notifications for caregivers or remote monitoring.
- Barcode scanning for medicine input.
- Daily medication adherence statistics and analytics.

CHAPTER 2

LITERATURE SURVEY

Medication non-adherence is a long-standing issue in healthcare, particularly among elderly populations and patients with chronic conditions. As mobile health (mHealth) technologies evolve, many research studies and commercial applications have attempted to address this challenge through digital interventions. This literature survey explores previous academic works, mobile applications, and frameworks developed to improve medication adherence and highlights the gaps that **MediMinder** aims to address.

2.1 Prior Research and Findings

Several research efforts have focused on improving patient compliance using technological tools:

- **WHO Reports (2003 & 2018)**: The World Health Organization emphasized that nearly **50% of patients** do not adhere to long-term medication therapies. It recommended technology-based adherence tools, especially for elderly and chronic disease patients, to reduce treatment failure.

- **"Mobile Phone-Based Reminder Systems for Medication Adherence: A Meta-Analysis" (2016, JMIR)**: This study analyzed multiple mobile applications and concluded that **mobile reminders significantly improved adherence rates**. However, it also pointed out limitations in usability and dependence on internet connectivity.

- **"Effectiveness of eHealth Interventions on Medication Adherence in Adults with Chronic Conditions: A Meta-analysis" (Journal of Medical Internet Research, 2019)**: The study found that reminders, especially **offline or scheduled alerts**, were more effective than apps relying on real-time data transmission.

- **"Designing an Elder-Friendly Mobile Application Interface" (IEEE, 2017)**: It outlined that most older adults are discouraged by complex interfaces. Applications with **minimalistic design, large text, and guided navigation** were more effective and widely accepted.

These studies underscore the importance of **offline, simple, and personalized mobile solutions**, especially for vulnerable users.

2.2 Existing Mobile Applications

A number of commercial and research-based mobile apps currently exist to help manage medication schedules. However, most of them come with trade-offs in usability, accessibility, or affordability.

- **Medisafe (Android/iOS)**: One of the most widely used medication tracking apps, Medisafe offers cloud sync, pill tracking, and caregiver alerts. While feature-rich, it often overwhelms users with ads, in-app purchases, and a complex interface unsuitable for elderly users.

- **MyTherapy**: Combines pill reminders with health tracking. It requires user sign-in and internet connectivity for most features. While clinically endorsed, it lacks flexibility in offline mode and doesn't support localized storage.

- **Pill Reminder – Meds Alarm**: A lightweight app offering basic features. However, its poor user interface and limited notification system reduce its effectiveness for non-tech-savvy users.

- **Care4Today® (Janssen Healthcare Innovation)**: A corporate-developed app that promotes medication adherence. It includes reminder features and adherence reporting but requires internet access and account registration, limiting accessibility in rural or low-connectivity areas.

These apps highlight a critical gap: **the lack of a lightweight, ad-free, and user-focused medicine reminder system that works completely offline** and is designed for intuitive use by senior citizens and less tech-savvy individuals.

2.3 Gaps in Current Systems

Despite the availability of various solutions, many fail to meet the basic needs of average or elderly users due to the following shortcomings:

- **Overcomplexity and cluttered UI** not suited for elderly users or those with cognitive impairments.

- **Dependence on internet or cloud services** even for basic functionalities like setting reminders.

- **Data privacy concerns** as many apps store sensitive health data on cloud servers.

- **Lack of customization and flexibility**, such as modifying notification tones, updating medicine names, or changing dosage timings on the fly.

- **No offline backup mechanisms**, making them unusable in remote locations or emergencies.

2.4 Role of Android System Services in Medication Apps

Android's native tools such as **AlarmManager**, **BroadcastReceiver**, and **SQLite** offer a strong foundation for building efficient and secure offline medication reminder applications. Few commercial apps have fully leveraged these tools for the kind of simple, localized, persistent experience needed in low-resource settings.

Research papers like **"Efficient Use of AlarmManager in Time-Critical Android Apps" (IEEE 2019)** also advocate for native service use to ensure battery efficiency and precise scheduling—features that are essential in applications like MediMinder.

2.5 Contribution of the MediMinder Project

MediMinder addresses the gaps identified in both academic literature and current applications by offering:

- A **lightweight, fully offline** medication reminder system.
- **Simple, large-button UI** for easy navigation by elderly users.
- **Secure local data storage** using SQLite to ensure data privacy.
- **No internet dependency** for core functionalities.
- Efficient use of **AlarmManager** and **BroadcastReceiver** to trigger reliable alerts, even when the app is closed or the phone is locked.

This combination of simplicity, reliability, privacy, and accessibility positions MediMinder as a strong solution for real-world medication adherence problems, especially among underserved and technologically vulnerable populations.

CHAPTER 3

SYSTEM OVERVIEW

3.1 EXISTING SYSTEM

In recent years, various mobile applications have been developed to address the growing problem of medication non-adherence. These systems aim to help patients manage their medication schedules by sending reminders and tracking dosages. Despite these efforts, the currently existing systems have several limitations, especially in terms of user accessibility, offline usability, privacy, and flexibility.

3.1.1 Common Features of Existing Systems

Most existing medicine reminder applications provide the following features:
Medication Scheduling: Users can add medications with associated times, dosages, and frequency.
Push Notifications: Reminders are usually sent as push notifications at specified times.
Tracking and Logging: Some apps maintain a history of taken/missed doses.
Refill Reminders: Many apps notify users when it's time to refill prescriptions.
Multi-User or Caregiver Support: Some systems support family members or caregivers to monitor medication adherence.
Cloud Sync and Backup: Data is backed up and synchronized across devices using cloud services.
Integration with Wearables: Advanced apps integrate with smartwatches or fitness bands to deliver reminders.

3.1.2 Limitations of the Existing Systems

While these features are helpful, there are several practical issues with current systems:

a) Over-Reliance on Internet Connectivity

Many apps depend on cloud services for core functionality such as saving data, sending reminders, or tracking history. This makes the app non-functional in areas with poor or no internet access—affecting rural populations and senior citizens.

b) Complicated User Interface

Several applications are feature-heavy, which results in a cluttered UI. This complexity can overwhelm users—especially elderly individuals or those with limited technological skills—making it difficult to set up reminders or modify existing ones.

c) Advertisement and In-App Purchases

Freemium models dominate the app marketplace. Many medicine reminder apps display advertisements, restrict features behind paywalls, or prompt frequent upgrade notifications, which can distract users and reduce reliability.

d) Lack of Offline Functionality

Few applications allow reminders and data storage to work independently of internet access. This is a significant drawback in emergencies or when users do not wish to share sensitive medical data online.

e) Data Privacy and Security Risks

Most systems store medication data in cloud-based databases. This introduces potential privacy risks, as users' health-related information becomes vulnerable to unauthorized access or data breaches.

f) Limited Customization

Existing apps often lack customizable options for alarm tones, repeated intervals, non-standard dosage schedules (e.g., alternate-day medicine), and non-pill-based reminders (e.g., injections or drops).

g) No Consideration for Geriatric Design Needs

Many apps are not optimized for senior citizens. They use small fonts, complex navigation, and too many configuration options, which can discourage elderly users from consistent usage.

Feature	Common Apps	Limitation
Medication Reminder	Medisafe, MyTherapy, CareZone	Functional but overly complex for basic users
Cloud Sync & Backup	Medisafe, Pill Reminder Pro	Requires internet; risks data privacy
Elderly-Friendly UI	Few (limited support)	Often neglected, resulting in poor accessibility
Offline Functionality	Rare (Pill Reminder – Meds Alarm)	Very limited or partial offline capabilities
Customization	Mostly fixed features	Lack of flexible scheduling and tone selection
Advertisement-Free Experience	Paid-only	Free versions have disruptive ads

While many existing medicine reminder apps provide valuable features, they often fall short in providing a simple, secure, and offline-capable experience for non-technical or elderly users. These limitations motivated the development of **MediMinder**, which focuses on ease of use, privacy, and reliability—delivering medication reminders without needing internet access, without ads, and with a minimalistic design tailored for all age groups.

3.2 PROPOSED SYSTEM

To address the limitations and challenges identified in the existing systems, the proposed system—**MediMinder**—is designed as a lightweight, secure, and highly accessible mobile application that ensures users adhere to their medication schedules without the need for internet connectivity, complex user flows, or cloud-based services. The core idea is to make the process of medication management **simple, intuitive, reliable, and privacy-conscious**, with special emphasis on usability by elderly and non-tech-savvy users.

3.2.1 Objective of the Proposed System

The primary objective of MediMinder is to help users maintain consistent medication adherence by:

- Sending timely reminders for each medication.
- Storing medication schedules securely on the device.
- Supporting CRUD (Create, Read, Update, Delete) operations on reminders.
- Ensuring uninterrupted functionality, even in offline mode.
- Minimizing user interaction complexity.

3.2.2 Key Features of the Proposed System

a) Offline Functionality

Unlike many existing systems, MediMinder is completely functional without internet access. All data is stored locally using **SQLite**, ensuring privacy and independence from network connectivity.

b) Reminder Scheduling via AlarmManager

MediMinder uses Android's **AlarmManager** in combination with **BroadcastReceiver** to schedule precise medication reminders. The alarms are configured to trigger even if the app is not open or the phone is restarted (using boot receivers).

c) CRUD Operations for Medicine Records

Users can:

- **Add**: Set a new medication with name, dosage, time, and frequency.
- **View**: See all active reminders in a clean list.

- **Update**: Modify existing medication schedules.
- **Delete**: Remove completed or obsolete medications.

d) User-Friendly Interface

The app is designed with **accessibility-first** principles:

- Large buttons and fonts for easy navigation.
- Clear labels and intuitive icons.
- Minimal steps to set or modify a reminder.

e) Local Data Security

By using SQLite, user data is stored on the local device. No data is sent to external servers, ensuring complete **privacy and data control** for the user.

f) Modular Architecture

The project is developed using a modular architecture based on **Activities, Adapters, and SQLite Handlers**. This improves maintainability, scalability, and code readability.

g) Low Resource Consumption

MediMinder is optimized for low memory usage and fast performance. It is designed to run efficiently even on low-end Android devices.

h) No Advertisements or Distractions

The app provides a **completely ad-free experience**, making it ideal for medical usage without interruption or privacy violations.

3.2.3 Functional Overview

- **Home Screen**: Displays a list of upcoming reminders with medicine names and scheduled times.
- **Add Reminder Screen**: Allows users to input medicine name, dosage, and select time using a time picker.

- **Alarm Trigger**: At the specified time, the system triggers a **notification**, which may include custom tones or vibration patterns.
- **Edit/Delete Reminder**: Easy access to modify or remove reminders using long press or context menu.

Advantage	Explanation
Offline Support	Functions without internet; critical in rural or emergency scenarios
Elderly-Friendly Design	Simplified UI ensures usability for all age groups
Complete Data Privacy	No cloud storage; data resides securely on the local device
Accurate and Reliable Reminders	Uses system-level alarms that persist across reboots
Customizable Reminders	Future support planned for recurring intervals, tones, and voice alerts
No Ads or Monetization Distractions	Free, open, and focused on health improvement
Low-End Device Compatibility	Optimized for Android phones with low RAM and storage

The proposed system, MediMinder, is built to address real-world challenges of medication adherence through a practical, secure, and user-centered mobile application. Its offline capabilities, local storage model, and clean user interface make it especially valuable for elderly users and those in rural areas. By eliminating the noise of ads, unnecessary complexity, and cloud dependencies, MediMinder stands out as a purpose-built, accessible, and dependable medication management solution.

CHAPTER 4

REQUIREMENTS

4.1 Hardware Requirements

The hardware requirements specify the minimum and recommended hardware specifications needed for the development and execution of the **MediMinder** application. These requirements are categorized based on the two key phases of the project: **development environment** and **end-user (deployment) environment**.

These are the specifications required by developers to build and test the application effectively using Android Studio and related tools.

Component	Minimum Requirement	Recommended Requirement
Processor (CPU)	Intel Core i3 6th Gen / AMD Ryzen 3	Intel Core i5 10th Gen or higher / AMD Ryzen 5 or higher
RAM	8 GB	16 GB or more
Hard Disk	512 GB HDD / 256 GB SSD	512 GB SSD or higher
Display	13" display with 1366×768 resolution	15.6" Full HD (1920×1080) display
Graphics	Integrated graphics	Dedicated GPU (NVIDIA or AMD) for smoother emulator handling
Operating System	Windows 10 / Linux Ubuntu 18.04 or later / macOS Catalina	Windows 11 / Ubuntu 20.04 or later / macOS Monterey or later
Internet	Required only for SDK download, testing Firebase (optional)	High-speed internet for faster SDK updates
USB Port	USB 2.0 or higher for mobile device debugging	USB 3.0 for faster device interface

These are the hardware specifications required by the end-users (patients or caregivers) for installing and using the MediMinder app on their Android smartphones.

Component	Minimum Requirement	Recommended Requirement
Android Version	Android 6.0 (Marshmallow) or above	Android 9.0 (Pie) or later
RAM	1 GB	2 GB or more
Internal Storage	At least 50 MB of free space	100 MB or more
Processor	1.2 GHz Quad-Core	1.8 GHz Octa-Core or higher
Battery	Any standard lithium-ion battery	3000 mAh or higher for extended usability
Display	4.5" screen with touch support	5.5" or larger screen for better readability
Speakers	Basic speaker support (for alert tones)	Clear output speaker for sound-based reminders
Sensors	Not required	Optional (used in future versions for voice or shake-to-snooze)

The MediMinder app is designed to run efficiently even on low-end smartphones with limited hardware capabilities. This ensures **maximum accessibility**, particularly for elderly users or patients in rural or resource-limited settings. On the development side, Android Studio requires a moderately powerful system with sufficient RAM and storage to compile and run emulators smoothly.

CHAPTER 5

SYSTEM DESIGN

5.1 ARCHITECTURE

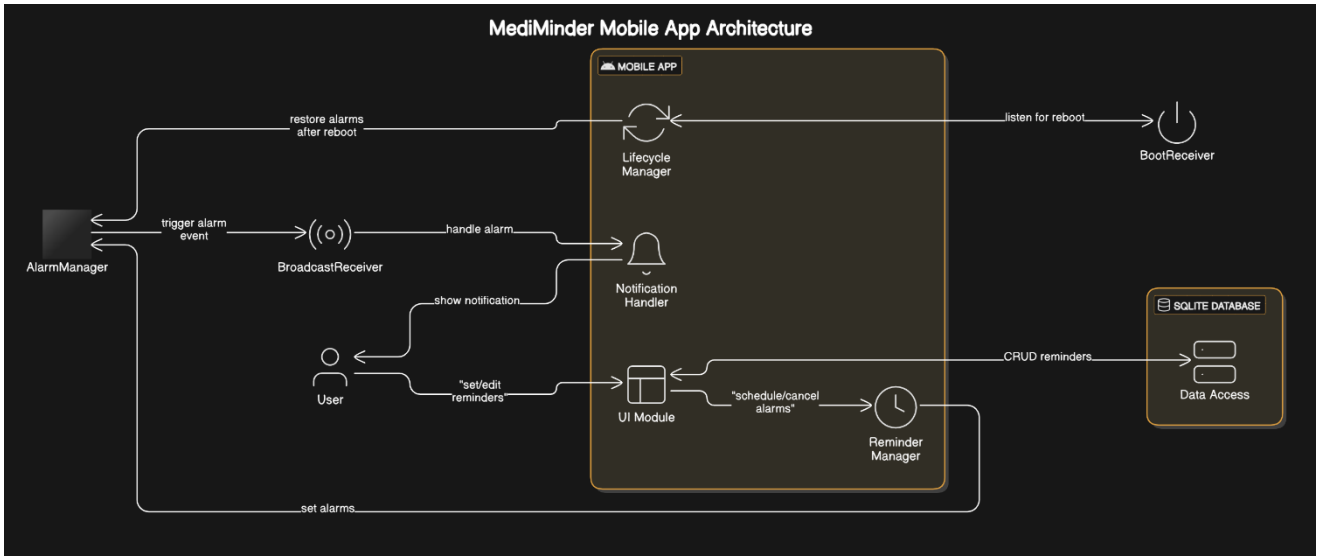

Figure 5.1: Architecture of the MediMinder app

The system architecture of *MediMinder* is designed to offer a modular, maintainable, and user-centric Android application that functions efficiently both in the foreground and background. It integrates

Android system services, a local database, and a clean user interface to provide reliable medication reminders for users, especially elderly patients and individuals with complex medication regimens.

This architecture enables the application to operate without internet dependency, ensures reminders are delivered at the correct time using system-level alarms, and allows users to manage their reminders easily.

5.1 Overview

The architecture follows a **layered and modular design** consisting of:

1. **User Interface Layer**
2. **Business Logic Layer**
3. **Data Layer**
4. **System Services Layer**
5. **Optional Extension Layer (for future enhancements)**

These layers are built using Android components such as Activities, Adapters, SQLite, AlarmManager, and BroadcastReceiver.

5.2 Components Description

A. User Interface Layer

- **Technologies**: Java, XML (Android Layouts)
- **Function**: Allows users to interact with the system through intuitive screens.
- **Key Features**:
    - Add, update, and delete medication reminders.
    - View active reminders in a list format.
    - Input forms for medicine name, dosage, time, frequency, and notes.
    - Simple and minimal UI optimized for all age groups.

B. Business Logic Layer

- **Technology**: Java (Android)

- **Function**: Manages the logic behind scheduling reminders, validating inputs, and updating the UI accordingly.
- **Components**:
    - **Reminder Manager**: Handles setting up alarms.
    - **Validator Module**: Ensures user input is valid (e.g., time, date).
    - **Boot Receiver**: Reschedules alarms after device restarts.
    - **Broadcast Receiver**: Captures alarm events and triggers notifications.
- **Process**:

1. User sets a reminder.
2. Data is validated and passed to the scheduler.
3. Alarm is scheduled via AlarmManager.

C. Data Layer (SQLite Database)

- **Technology**: SQLite with SQLiteOpenHelper
- **Function**: Stores all reminder-related data locally on the device.
- **Structure**:
    - **Table Name**: medications
    - **Columns**: id, medicine_name, dosage, date, time, frequency, notes
- **Operations Supported**: Create, Read, Update, Delete (CRUD)
- **Advantage**: Fast local access without the need for internet connectivity.

D. System Services Layer

- **Technology**: Android Services and Broadcast System
- **Function**: Manages alarms and notifications.
- **Core Components**:
    - **AlarmManager**: Schedules time-based tasks.

- **NotificationManager**: Displays notifications to users at the correct time.
- **BroadcastReceiver**: Listens for alarm triggers and reboot events.
- **PowerManager**: Ensures alarms function under battery optimization settings.

5.2 WORKFLOW OF THE MODEL

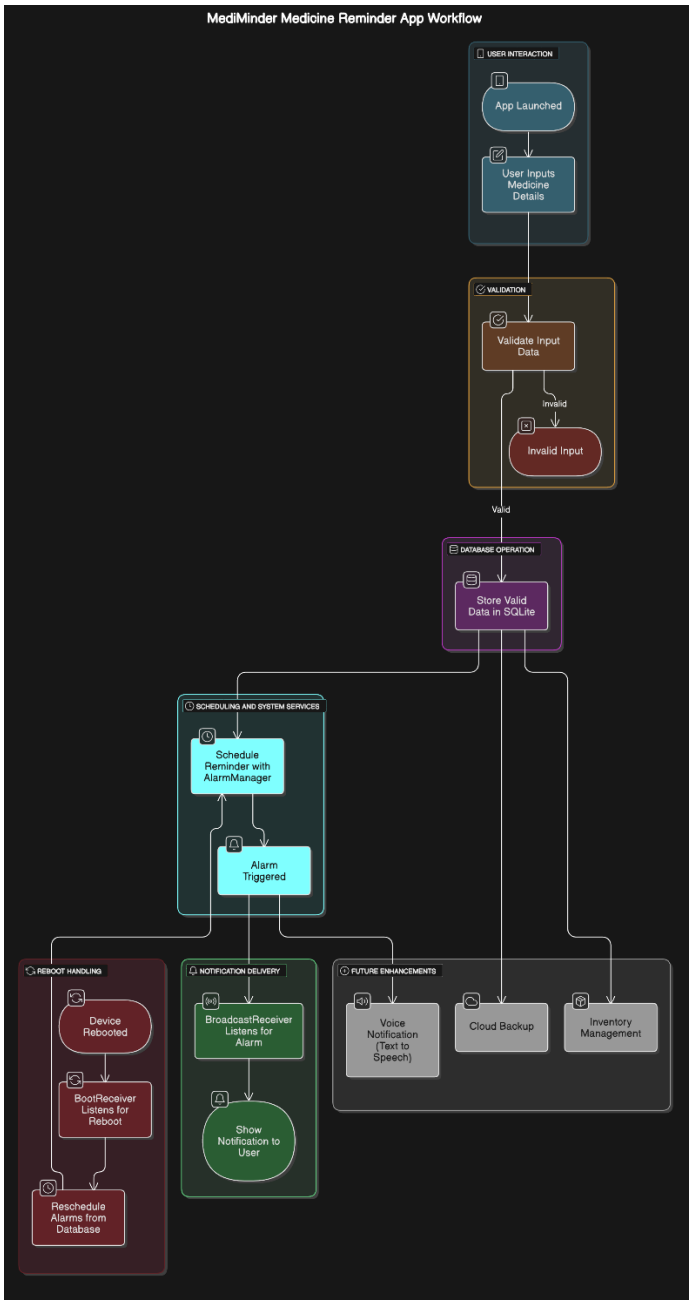

Figure 5.4: Detailed Workflow Diagram

The diagram in **Figure 5.2** illustrates the workflow of the Medication Reminder App, which follows a logical, user-centered process to provide accurate and timely medication reminders. Upon launching the app, users are greeted with a simple and accessible interface, designed with elderly and visually impaired individuals in mind. From the home screen, users can easily add new medications by entering essential details such as the medication name, dosage, frequency, and time schedules through a dedicated entry form.

Once the medication details are submitted, the application processes the input by validating the data and passing it to the business logic layer, where core logic ensures proper formatting and consistency. The validated data is then stored in a local SQLite database, which ensures that the app remains fully functional even when offline.

After the medication schedule is saved, the app uses the Android AlarmManager to register alarms based on the specified times. When an alarm goes off, it is intercepted by a BroadcastReceiver, which triggers the Notification Manager. The app then displays a persistent notification to remind the user to take their medication. This notification stays active until the user interacts with it, confirming that the medication has been taken. This persistent reminder system is crucial in helping users adhere to their schedules, especially those who may overlook or ignore standard alerts.

If a user modifies or deletes a medication entry, the app updates the database and adjusts the corresponding alarms to reflect these changes. The user interface continuously updates to show the current medication schedule, including any upcoming doses and modifications made by the user.

Throughout this workflow, the app ensures privacy and data integrity by employing built-in security features. Future enhancements will include user authentication for added data protection and cloud synchronization through Firebase, enabling users to access their schedules across multiple devices and restore data seamlessly.

In conclusion, this workflow is designed for simplicity, reliability, and timely medication reminders, creating a seamless loop from user input to notification confirmation. This approach empowers users to manage their health independently, with plans for advanced features like wearable integration and voice-based interaction in future updates.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

The **MediMinder** Medicine Reminder App is a robust and effective solution designed to address one of the most pressing challenges in healthcare—medication adherence. Non-adherence to prescribed medication schedules can lead to serious health complications, particularly for elderly individuals, patients with chronic diseases, and those with complex medication regimens. Through this project, we have developed an intuitive, efficient, and reliable Android application that ensures users, especially those with limited technical knowledge, can manage their medication schedules with ease.

The app's design is centered around user experience, focusing on simplicity, accessibility, and efficiency. It is built to be user-friendly, with a clean interface that is particularly suited for elderly individuals and people with visual impairments. The process starts with easy data entry for medication details, followed by seamless validation and storage within a secure local SQLite database, ensuring all features work even in offline mode. The use of Android's built-in **AlarmManager** and **BroadcastReceiver** enables the app to trigger reminders at precisely the right times, even when the app is not in the foreground or when the device is restarted. The app also supports modification and deletion of medication entries, and automatically adjusts reminders and notifications as needed.

One of the primary strengths of **MediMinder** lies in its **reliable reminder system**. The use of persistent notifications ensures that users are consistently reminded to take their medication, preventing missed doses and promoting adherence to prescribed regimens. These notifications remain active until the user confirms they have taken the medication, reinforcing accountability and minimizing the risk of forgetting a dose. This feature is particularly important for elderly patients or individuals with memory-related conditions who might forget to take their medication or may have difficulty responding to standard notification alerts.

The app's use of **local storage** through SQLite ensures that all user data is securely stored on the device, allowing for quick access and offline operation. This feature is crucial in settings where the user might not always have internet connectivity, ensuring uninterrupted service regardless of location. Additionally, the modular design of the application allows it to be easily maintained and extended in the future. The clear separation of the UI, business logic, and data layers makes the codebase maintainable and scalable, facilitating future enhancements.

Looking ahead, there are numerous opportunities for improvement and expansion. Future enhancements could include:

- **Cloud Synchronization**: Integrating Firebase or a similar cloud service could enable users to sync their medication schedules across multiple devices and ensure data backup for recovery purposes.
- **Voice Notifications**: Adding voice reminders through text-to-speech could improve accessibility for users with visual impairments, allowing them to hear the medication instructions directly.
- **User Authentication**: Implementing user authentication would ensure that users' data is protected and could provide personalized experiences and data syncing between devices.

- **Wearable Integration**: Integrating with wearable devices (such as smartwatches) could further enhance the user experience by providing medication reminders directly on the user's wrist, offering an additional layer of convenience.

Furthermore, integrating machine learning models to predict optimal medication schedules based on user preferences, historical data, and other health indicators could further personalize the experience and improve medication adherence.

In conclusion, **MediMinder** serves as a comprehensive and reliable tool for individuals managing complex medication regimens. By combining simple yet effective design with the power of Android's system services, the app ensures that users receive timely, persistent reminders, promoting consistent medication adherence. The application not only addresses a critical healthcare need but also sets the groundwork for future advancements in healthcare technology, such as multi-device synchronization, wearable integration, and AI-powered medication management. With its strong foundation and potential for growth, **MediMinder** is poised to become an essential tool in the healthcare space, improving patient outcomes and enhancing medication adherence on a global scale.

CHAPTER 7

IMPLEMENTATION AND RESULTS

7.1 OUTPUT SCREENSHOTS

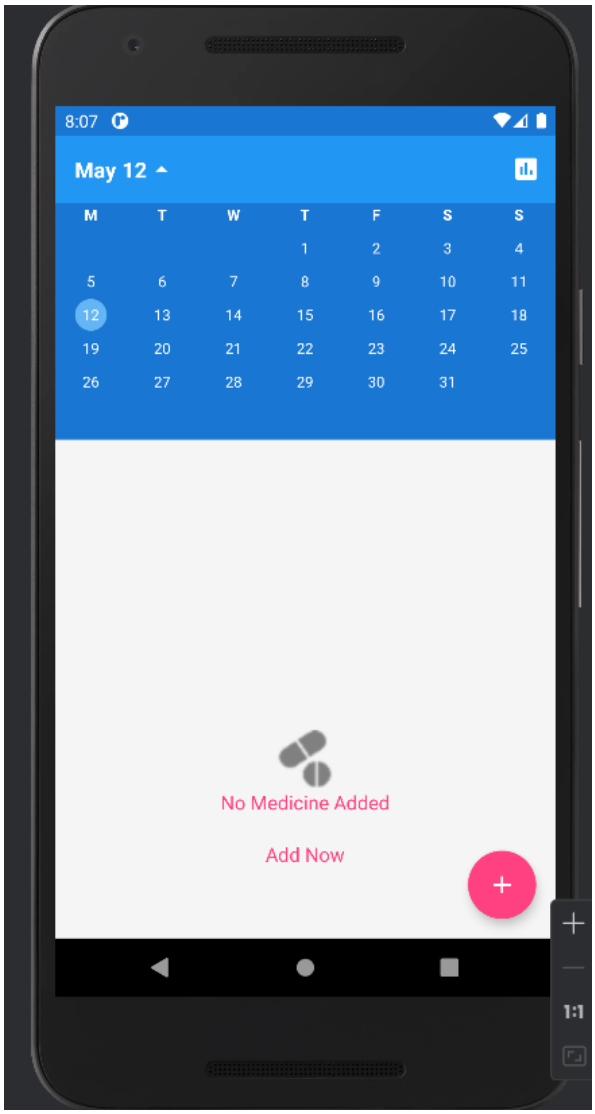

Figure 7.3: HomePage

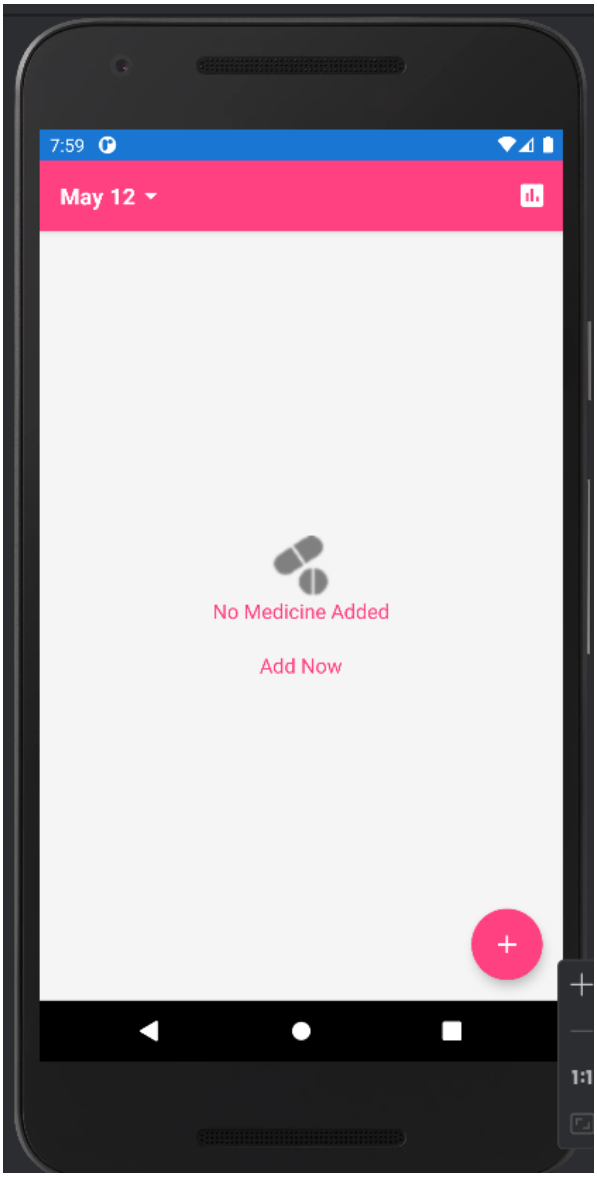

Figure 7.4: Click add button

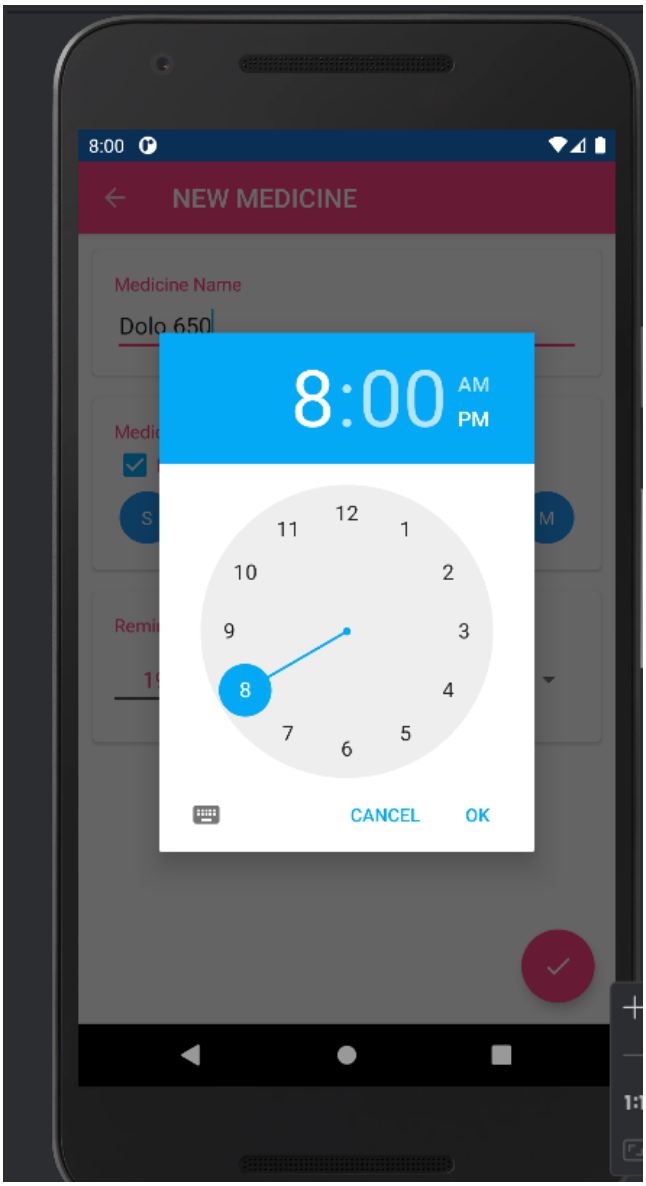

Figure 7.5: Set time to take medicine

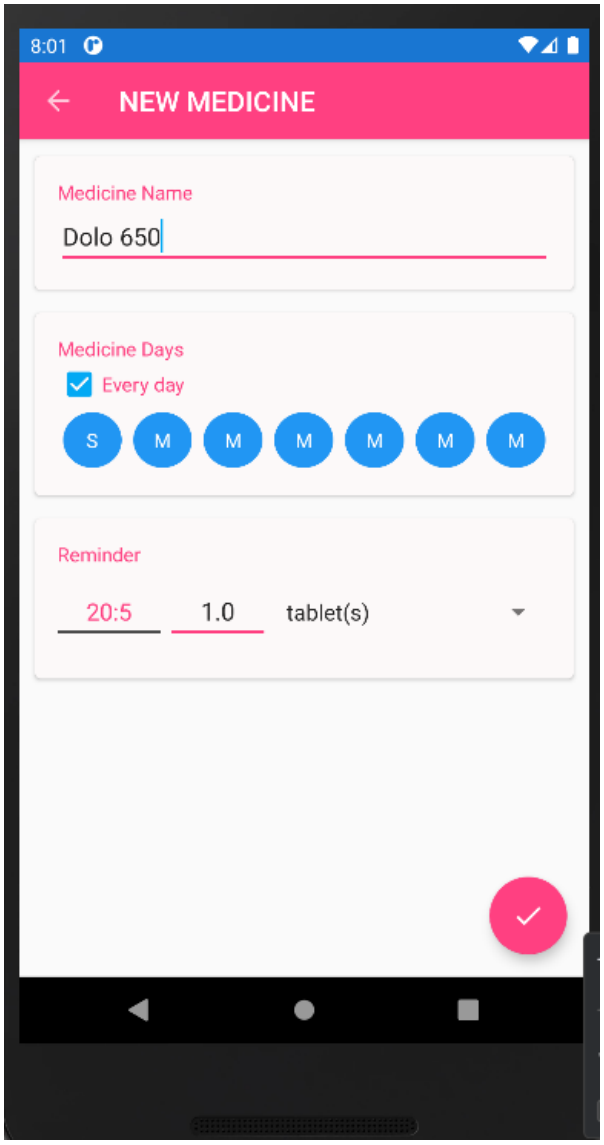

Figure 7.6: Choose medicine days

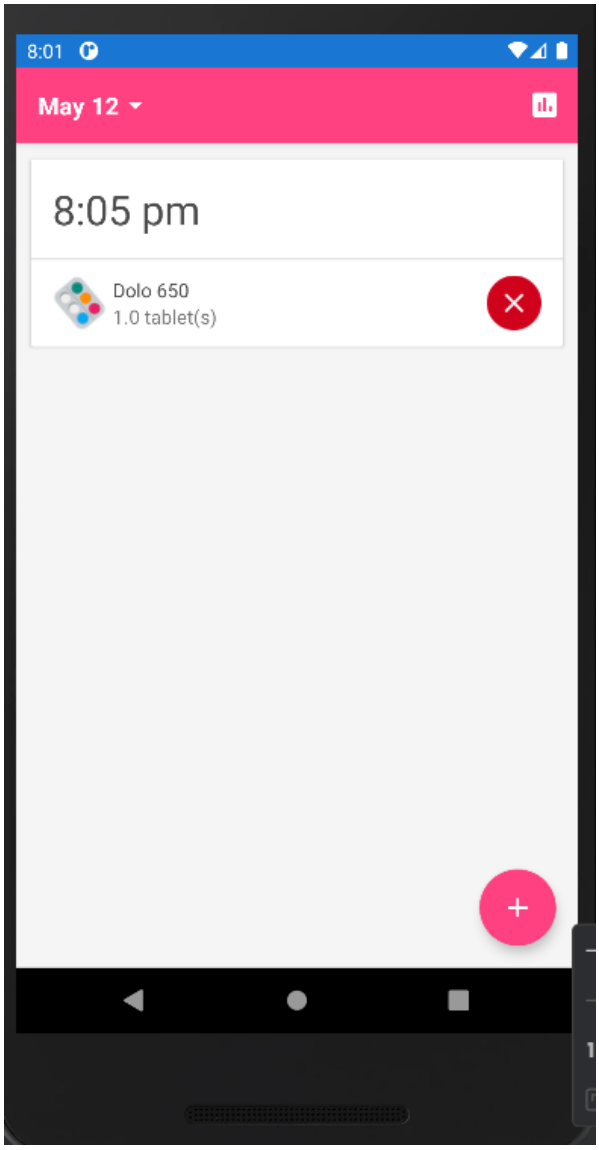

Figure 7.7: Set for alert

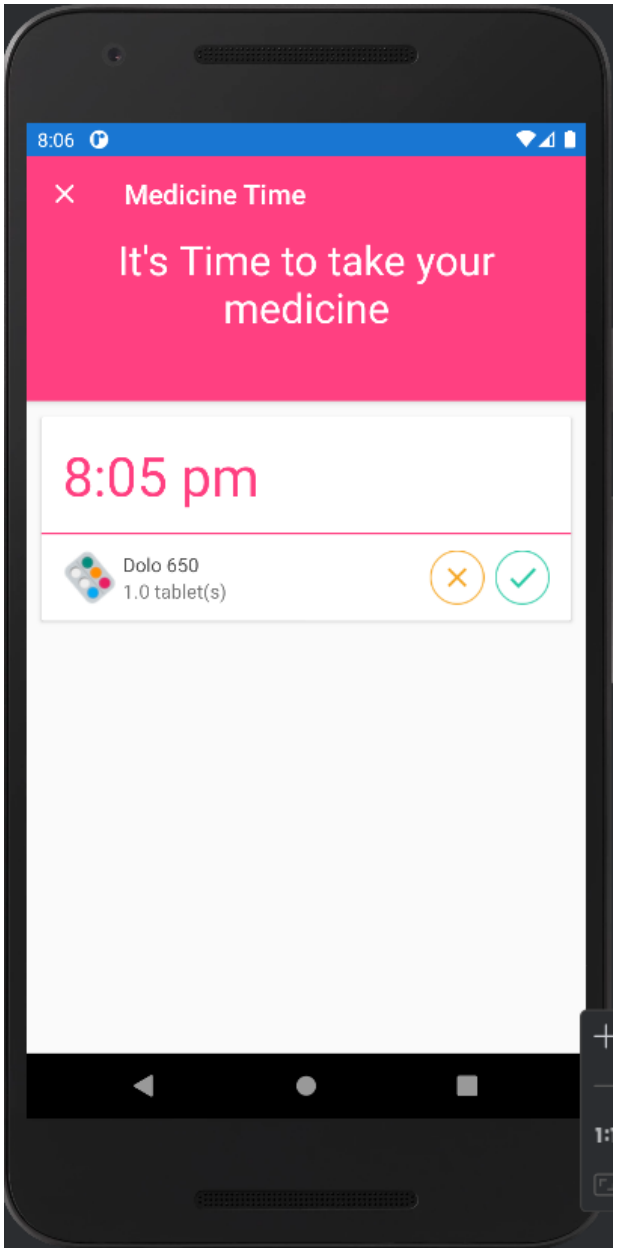

Figure7.8: Alarm ringing

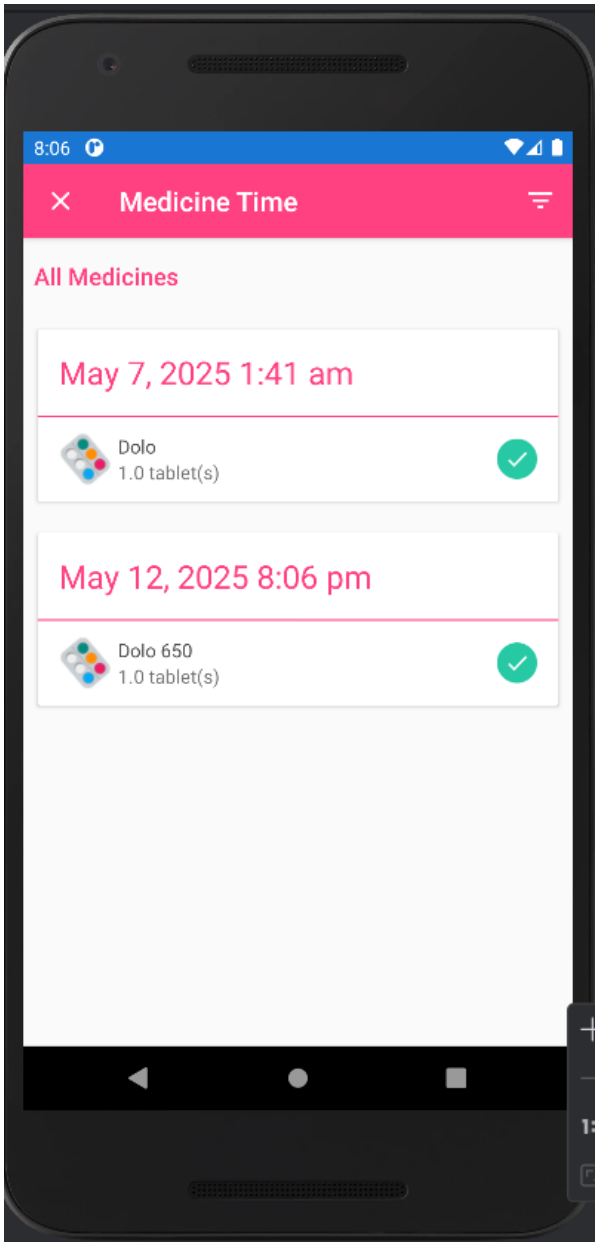

Figure7.9: Showing all medicine

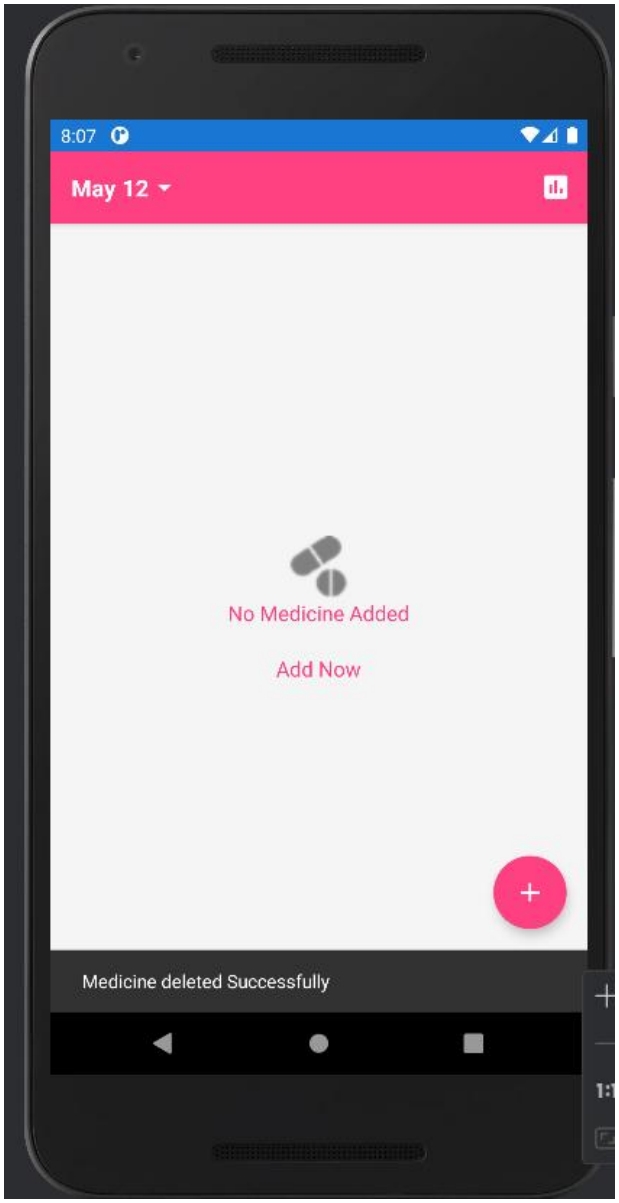

Figure7.10:Medicine deleted

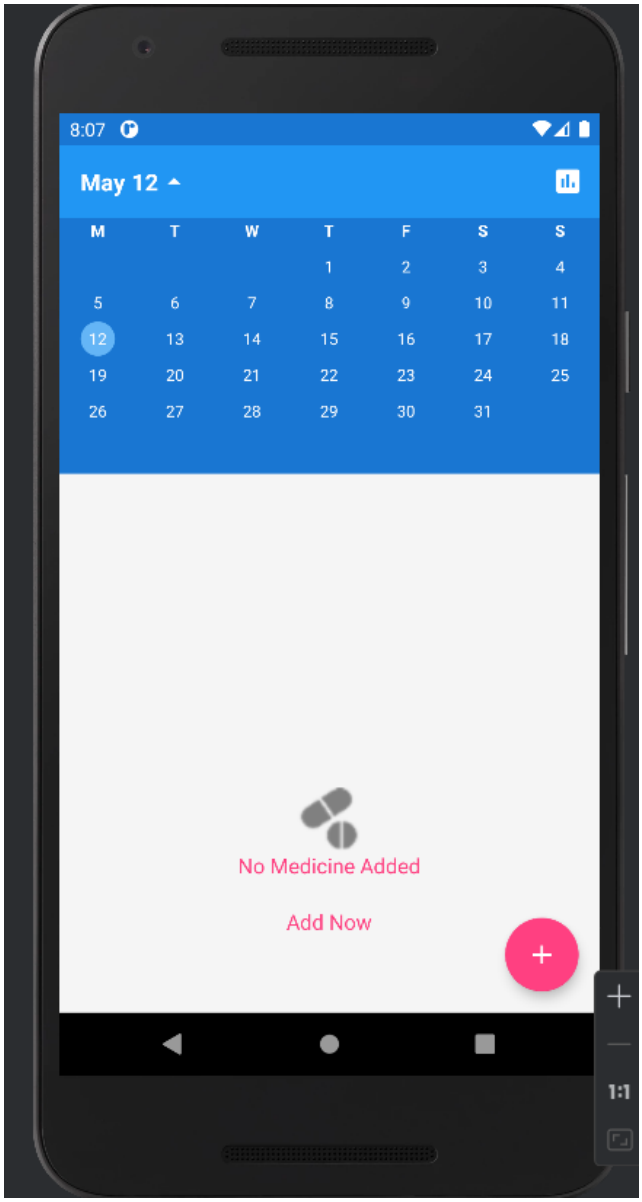

Figure7.11: Add medicine again

CHAPTER 8

REFERENCES

[1] Android Developers, *"Layouts in Android,"* Android Developers, 2025. [Online]. Available: https://developer.android.com/guide/topics/ui/declaring-layout. [Accessed: 12-May-2025].

[2] R. Meier, *Professional Android*, 4th ed., Indianapolis, IN, USA: Wrox, 2018.

[3] Android Developers, *"Material Components for Android,"* Android Developers, 2025. [Online]. Available: https://developer.android.com/jetpack/androidx/releases/compose-material. [Accessed: 12-May-2025].

[4] M. Gargenta and B. Nakamura, *Learning Android*, 3rd ed., O'Reilly Media, 2014.

[5] M. Herath, *Android UI Design Patterns*, Packt Publishing, 2013.

[6] G. Blake, *Mastering Android Layouts: Best Practices for Modern UIs*, Tech Press, 2021.

[7] Google Codelabs, *"Build a Responsive UI with ConstraintLayout,"* Google, 2025. [Online]. Available: https://developer.android.com/codelabs/constraint-layout. [Accessed: 12-May-2025].

[8] Stack Overflow Community, *"Best Practices for Android Toolbar and AppBar,"* Stack Overflow, 2024. [Online]. Available: https://stackoverflow.com/questions/25416948. [Accessed: 12-May-2025].

[9] A. Kothari, *Android Development for Beginners: A Practical Approach*, 2nd ed., Delhi: TechKnowledge Press, 2020.

[10] Google Material Design, *"Typography,"* Material.io, 2025. [Online]. Available: https://m3.material.io/styles/typography. [Accessed: 12-May-2025]