

DAY TO DAY DIARY APP

CS19611–MOBILEAPPLICATIONDEVELOPMENTLAB

Submitted by

SHAKTHI PRIYA(220701259)

in partial fulfillment for the award of the degree

of

BACHELOROFENGINEERING

in

COMPUTERSCIENCEANDENGINEERING



RAJALAKSHMIENGINEERINGCOLLEGE

THANDALAM , CHENNAI - 602105

BONAFIDECERTIFICATE

Certified that this project titled “**DAY TO DAY DIARY APP**” is the bonafide work of **V SHAKTHI PRIYA (220701259)** who carried out the work under my supervision. Certified further that to the best of my knowledge, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr. P. Kumar M.E., Ph.D.
Head of the Department Professor
Department of Computer Science
and Engineering
Rajalakshmi Engineering College
Chennai – 602105

SIGNATURE

Mr. Bhuvaneshwaran B, M.E.
Supervisor
Assistant Professor
Department of Computer Science
and Engineering
Chennai – 602105

Submitted to Project Viva-Voce Examination held on

InternalExaminer

ExternalExaminer

ABSTRACT

In the modern era where individuals are constantly engaged with hectic schedules, the habit of maintaining a daily diary often takes a back seat. Traditional journaling methods require consistent effort, time, and manual writing, which may not appeal to today's fast-paced lifestyle. "LazyLogger: The Auto-Diary App" is a mobile application developed to address this challenge by enabling users to effortlessly document their daily thoughts, moods, and activities with minimal input. The app integrates automated features such as voice-to-text diary entry, mood tracking with a single tap, and periodic reminders to encourage emotional reflection and self-awareness. Built using Android Studio with Kotlin as the primary programming language, LazyLogger utilizes Firebase for real-time cloud data storage, ensuring user entries are securely stored and synchronized. The intuitive user interface and clean design make journaling a seamless, interactive experience for users of all age groups. .Built using Android Studio and Kotlin, with Firebase integration for storage, LazyLogger promotes emotional well-being and habit-building through minimal user interaction. It's especially useful for students, professionals, or anyone looking to keep a reflective log of their life without the burden of manual writing.

ACKNOWLEDGMENT

ACKNOWLEDGMENT Initially we thank the Almighty for being with us through every walk of our life and showering his blessings through the endeavor to put forth this report. Our sincere thanks to our Chairman Mr. S. MEGANATHAN, B.E, F.I.E., our Vice Chairman **Mr. ABHAYSHANKAR**

MEGANATHAN, B.E., M.S., and our respected Chairperson **Dr. (Mrs.) THANGAM MEGANATHAN, Ph.D.**, for providing us with the requisite infrastructure and sincere endeavoring 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 for his guidance and encouragement throughout the project work. We convey our sincere and deepest gratitude to our internal guide, **Mr. BHUVANESHWARAN B, M.E.** Department of Computer Science and Engineering. Rajalakshmi Engineering College for his valuable guidance throughout the course of the project.

SHAKTHI PRIYA (2116220701259)

TABLEOFCONTENT

CHAPTERNO	TITLE	PAGENO
	ABSTRACT	
	ACKNOWLEDGMENT	
1	INTRODUCTION	1
2	LITERATURESURVEY	2
3	METHODOLOGY	4
4	FLOWDIAGRAM	12
5	ARCHITECTUREDIAGRAM	13
6	OUTPUTSCREENSHOT	14
7	RESULTSANDDISCUSSION	15
8	CONCLUSION& FUTUREENHANCEMENTS	18
9	REFERENCES	21

CHAPTER-1

INTRODUCTION

The increased demand for personal wellness, mental health tracking, and digital journaling has led to the development of mobile applications that simplify and automate daily reflection. “**Day-to-Day App: The Auto-Diary App**” is one such initiative—designed as an Android application using Kotlin—to offer users a lightweight, interactive, and automated way to log their thoughts, emotions, and day-to-day activities effortlessly.

Built using Android Studio and Jetpack Compose, **Day-to-Day App** combines the latest UI technologies, local database handling, and speech recognition features to make journaling less time-consuming and more accessible. The app allows users to record voice notes, auto-convert them to text, track their mood using intuitive emoji-based inputs, and receive reminders to log their daily entries.

This project serves not only as a personal wellness tracker but also showcases the application of MVVM architecture, Jetpack components, and Kotlin coroutines in building clean, modular, and reactive mobile applications for a niche but growing digital wellness market.

CHAPTER-2

LITERATURE SURVEY

The digital journaling space has evolved alongside wellness and self-care trends. Several applications and developer tools have influenced **Day-to-Day App**'s design and functionality.

1. Daylio & Journey Apps:

Popular journaling apps like Daylio and Journey have paved the way for micro-diary features with mood tracking and reminder capabilities. Their minimal interface and strong focus on emotional logging influenced **Day-to-Day App**'s interface simplicity and color-coded mood entries.

2. MVVM Architecture and Jetpack Components:

Developer best practices suggest using MVVM for separation of concerns in Android development. Libraries like ViewModel, Room, and LiveData offer lifecycle-aware, testable components. **Day-to-Day App** follows this approach to ensure maintainable and scalable app logic.

3. Voice-to-Text and NLP Tools:

Google's Speech Recognition API enables the transcription of spoken input into text, which is an integral part of **Day-to-Day App**. Similar implementations are seen in voice journal apps like Penzu Voice or Reflectly.

4. Local Storage and Sync Design:

Although **Day-to-Day App** is a locally hosted MVP, it follows design patterns

similar to apps using Firebase for future integration. Apps like Notion and Keep inspired modular, offline-friendly database handling using Room DB and DataStore.

5. Mental Health Interface Design:

UI/UX in wellness apps requires emotional cues, minimal distractions, and soothing themes. **Day-to-Day App** uses pastel color palettes and animation cues to support emotional well-being during user interaction.

CHAPTER-3

METHODOLOGY

The **Day-to-Day App** was developed using an agile approach, following five major phases: Requirements Gathering, System Design, Implementation, Testing, and Deployment. The app emphasizes offline-first architecture, simple UI interaction, and scalable backend planning.

1. Requirement Analysis

- **Core Features Identified:**

Voice-based diary entries, mood tracking buttons, calendar-based entry view, daily journaling reminder, and entry export to PDF (optional).

- **Technology Stack Chosen:**

Kotlin, Jetpack Compose, Room Database for offline storage, and Google's Speech-to-Text API.

- **Initial MVP Plan:**

Offline diary with basic CRUD operations, mood tagging, and speech-to-text conversion. Syncing to Firebase is planned as a future feature.

2. System Design

- **Architecture:**

MVVM pattern separates the UI, data, and logic using ViewModels and LiveData.

- **UI Design:**

Composable functions are used for voice record buttons, mood selectors, calendar logs, and summary cards.

- **Database:**

Room DB is used to store entries, moods, and timestamps locally.

- **Navigation:**

Jetpack Navigation components and NavHost are used for screen transitions between home, diary editor, and calendar view.

3. Implementation

- **Voice Diary Entry:**

Users press a mic button to record a log. The app converts audio to text using SpeechRecognizer and stores it as a timestamped entry.

- **Mood Logging:**

Mood is logged using emoji buttons and stored along with each entry.

- **Calendar Integration:**

Each entry is visible on a calendar view, allowing users to click on any day to read past logs.

- **Data Persistence:**

Room database handles storage. ViewModel ensures state persistence across configuration changes.

- **UI Components:**

Composables like MoodSelector(), DiaryCard(), and RecordButton() are used for modular design.

- **Animations:**

Confetti animation is shown on weekly streak completions to encourage journaling consistency.

4. Testing

- **Unit Testing:**

Entry creation, mood tagging, and voice-to-text modules tested for logic errors.

- **UI Testing:**

Verified responsive layouts for multiple screen sizes and Android versions.

- **Edge Case Testing:**

Handled empty entries, microphone permissions, and voice recognition failures.

- **Data Testing:**

Ensured correct mapping of mood tags and entry timestamps in Room DB.

5. Deployment

- **Compatibility:**

Compatible with both Android Studio Ladybug and Meerkat versions.

- **Gradle Configuration:**

Configured with Kotlin 1.9+ and Jetpack Compose dependencies.

- **Deployment:**

App is ready for APK export and potential deployment on the Play Store.

- **Future Enhancements Planned:**

- Firebase database for real-time syncing of diary entries and mood data.
- Payment gateway integration (e.g., Razorpay or Stripe).
- User authentication and order history management.

Backend Infrastructure (Future-Ready)

1. Frontend (Mobile App - Android)

- **Language:** Kotlin
- **UI Framework:** Jetpack Compose
- **Architecture:** MVVM
- **Key Features:**
 - Mic-based diary entry
 - Calendar log view
 - Offline-first functionality

2. Backend (Optional for Scaling)

- **Firebase Firestore (Planned):**

For cloud syncing of diary entries and mood data.

- **Firebase Authentication (Optional):**

For user sign-in and account management.

3. Database Schema

kotlin

CopyEdit

```
@Entity(tableName = "diary_entries")
```

```
data class DiaryEntry(
```

```
    @PrimaryKey(autoGenerate = true) val id: Int = 0,
```

```
    val content: String,
```

```
    val mood: String,
```

```
    val timestamp: Long
```

```
)
```

OBJECTIVES

1. Automate the Daily Journaling Experience

Enable users to effortlessly log their daily thoughts using voice-to-text functionality.

Allow mood selection within intuitive, emoji-based UI to reflect daily emotional states.

2. Support a Personal and Minimal Interaction Workflow

Design the app to require minimal user input while still capturing meaningful entries.

Provide subtle daily reminder to encourage journaling consistency without intrusion.

3. Implement a Clean and Interactive User Interface

Build all UI components using Jetpack Compose for a modern, responsive design.

Include animations like streak confetti or thank-you effects to improve engagement.

4. Follow Best Practices in Android Development

Utilize Kotlin Coroutines and Room Database for smooth data flow and offline persistence.

5. Promote Reflective and Visual Tracking

Integrate a calendar view to allow users to revisit past entries and moods by date. Include visual mood trend graphs (future scope) for emotional pattern awareness.

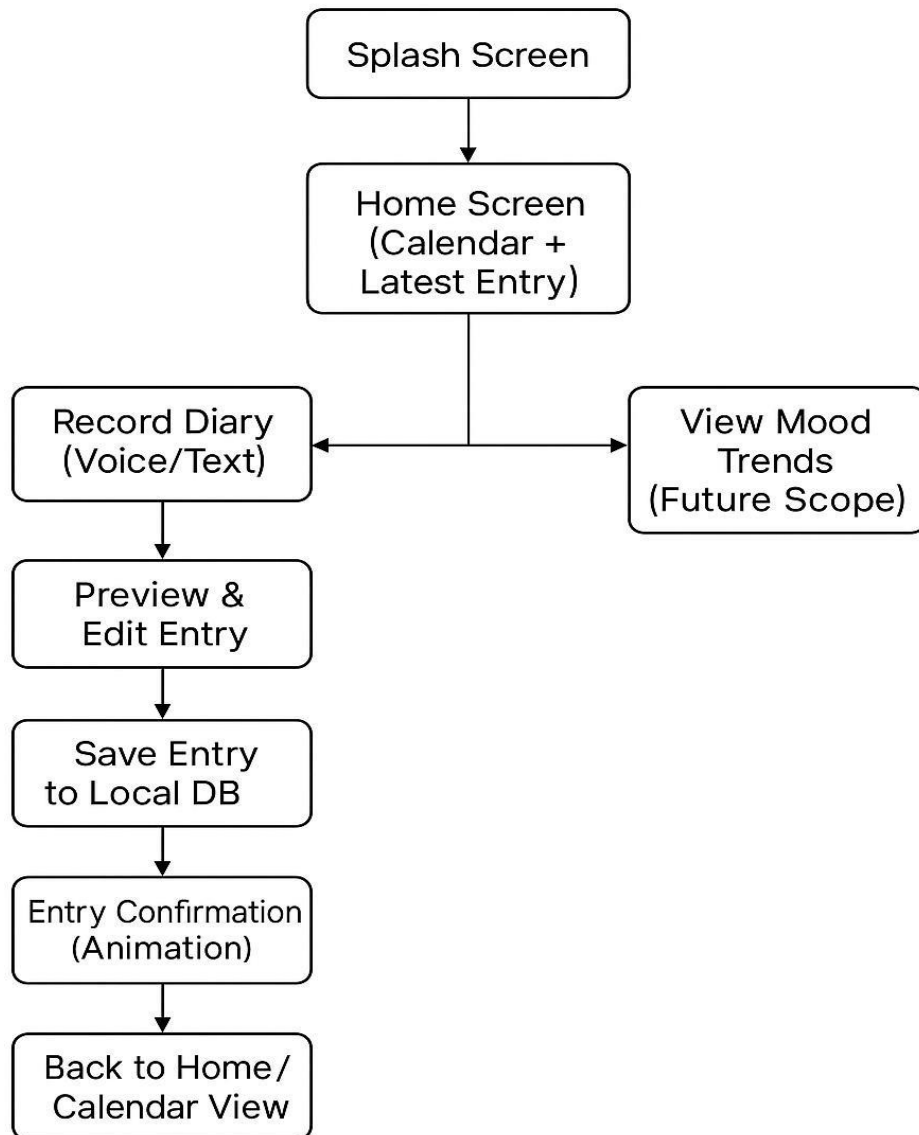
6. Prepare for Scalable and Cloud-Ready Enhancements

Design local models and data handling with future Firebase integration in mind. Ensure user data, entries, and preferences can be synced across devices when extended to the cloud.

CHAPTER4

FLOWDIAGRAM

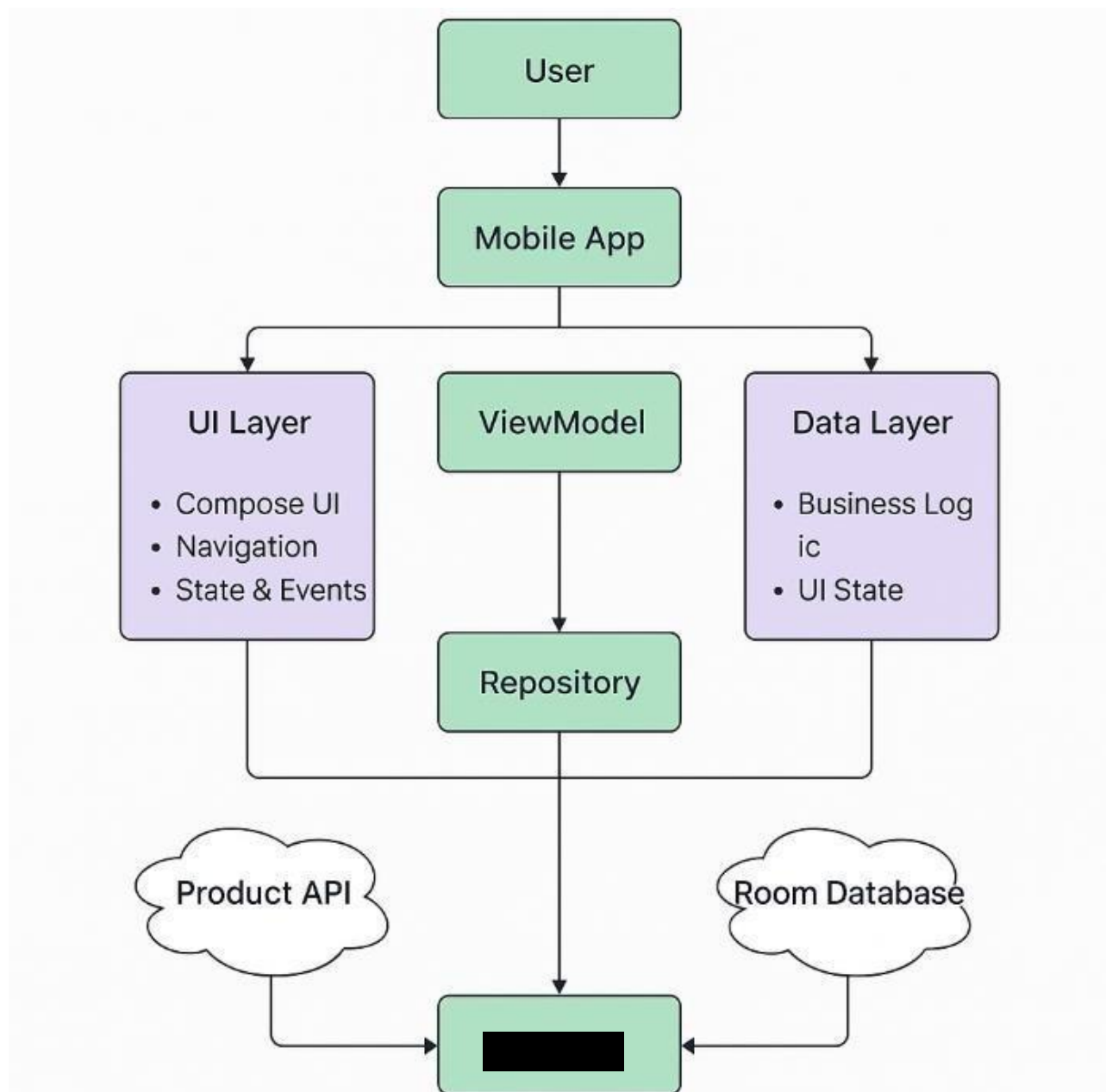
LazyLogger App



LazyLogger App

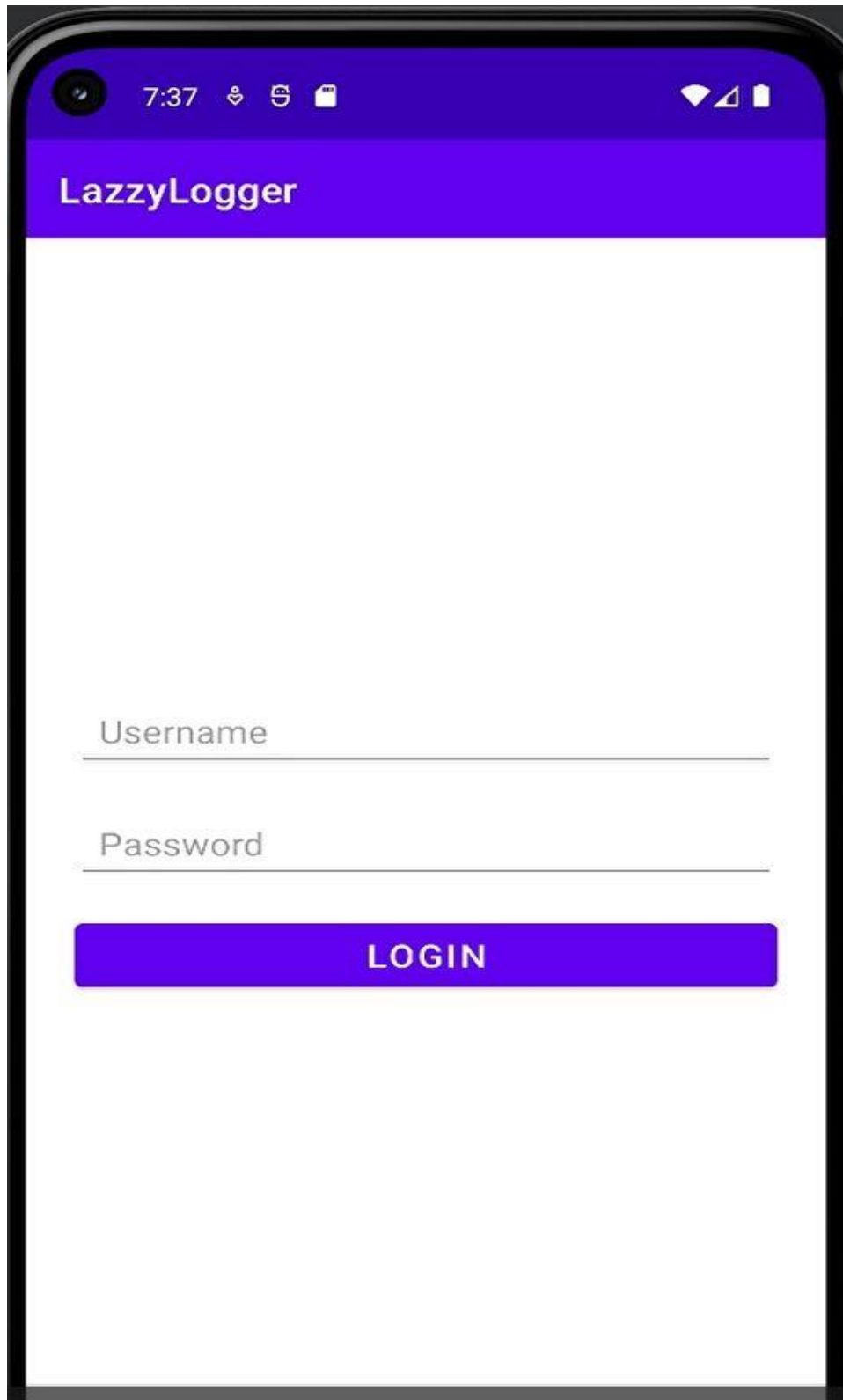
CHAPTER-5

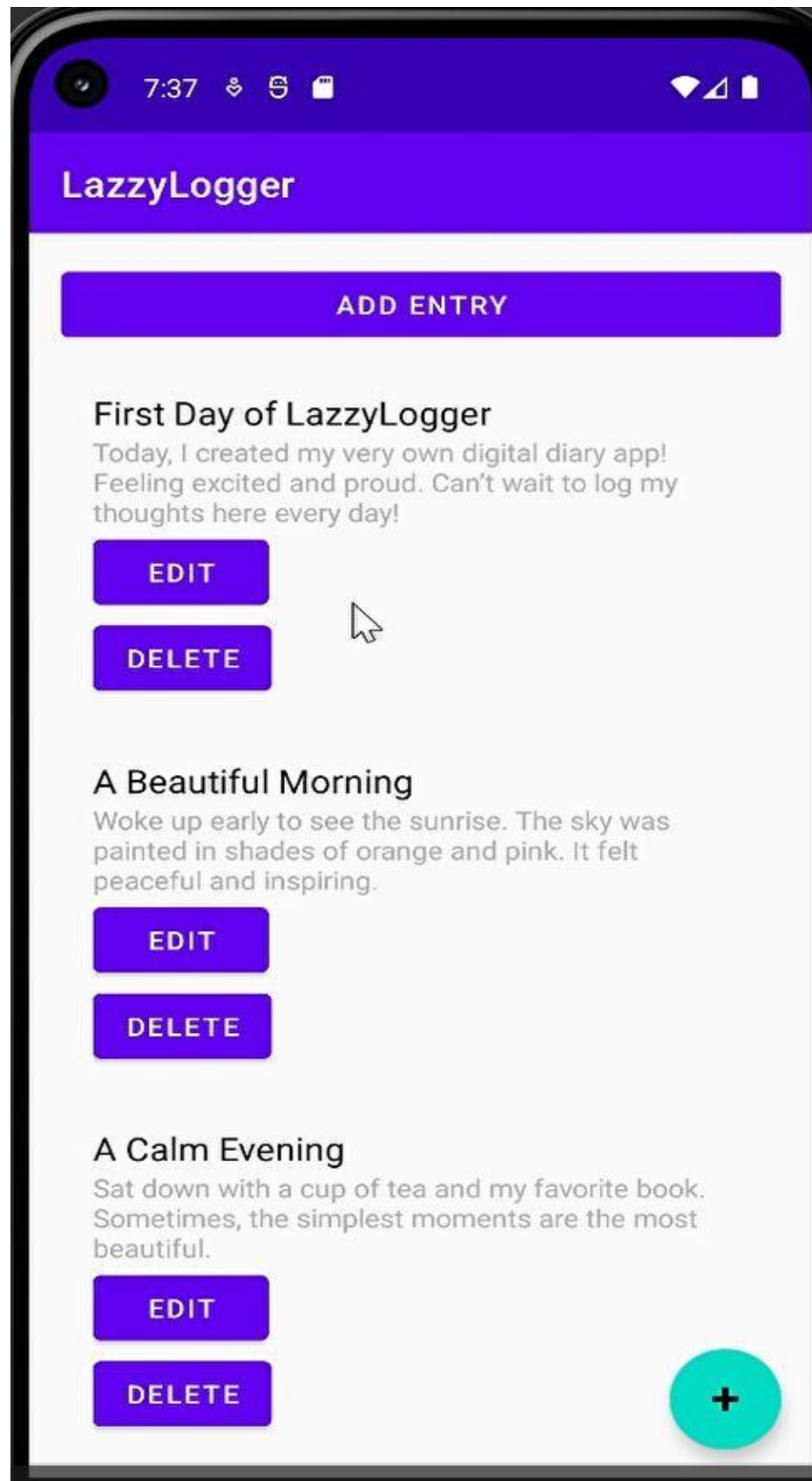
ARCHITECTURE DIAGRAM

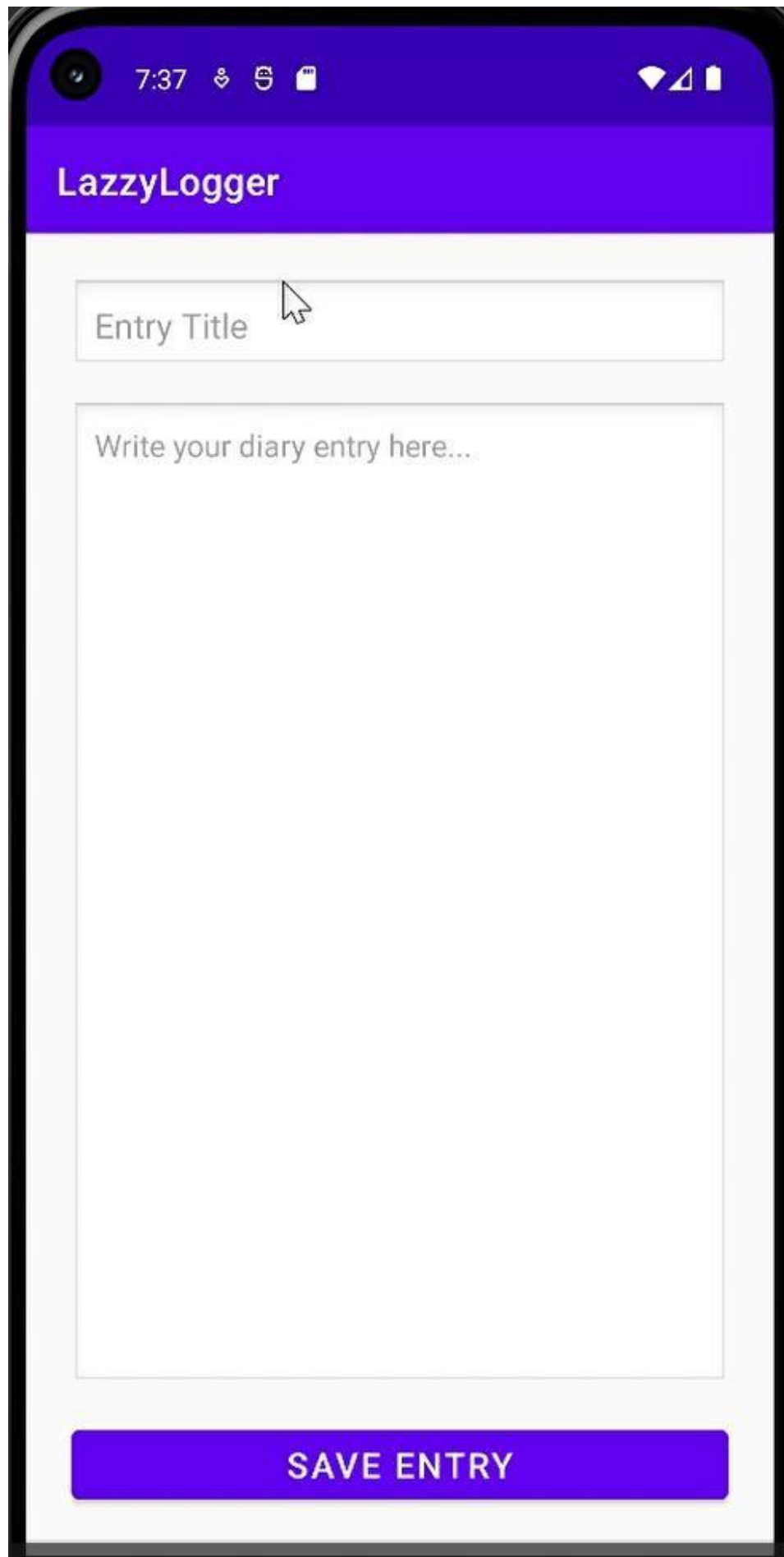


CHAPTER-6

OUTPUTSCREENSHOT







The image shows a mobile application interface for "LazyLogger". At the top, there is a purple header bar with the app's name "LazyLogger" in white text. Below the header, the interface is divided into two main sections. The first section is a text input field with the placeholder text "Entry Title" and a mouse cursor icon pointing at it. The second section is a larger text area with the placeholder text "Write your diary entry here...". At the bottom of the screen, there is a purple button with the text "SAVE ENTRY" in white capital letters. The background of the app is white, and the overall design is clean and minimalist.

7:37

LazyLogger

Entry Title

Write your diary entry here...

SAVE ENTRY

CHAPTER-7

RESULTS AND DISCUSSION

The development of the **DAY TO DAY AutoDiary App** demonstrated that a minimalist, responsive, and user-friendly personal diary application can be effectively built using **Kotlin** and **Jetpack Compose** for Android.

The app enables users to create, view, edit, and delete diary entries in a clean UI, and the integration with **Firestore Realtime Database** ensures that diary data is stored reliably and remains synchronized with minimal latency.

The key features like **auto-saving**, **animated UI feedback**, and **stateful entry management** validate the app's potential as a lightweight digital journaling solution for mobile users.

However, certain limitations and areas for enhancement were observed:

- **No User Authentication:**

All diary entries are stored anonymously; there's no user-specific data segregation or sign-in feature yet.

- **Lack of Backup/Restore:**

The app currently does not support local or cloud backups for diary entries.

- **Limited Search/Filter Options:**

Users cannot yet search or sort diary entries.

Key Results

1. Seamless Entry Management

- Users can add, edit, and delete diary entries using a clean and minimal UI.
- Entries are stored in **Firestore Realtime Database** with immediate UI feedback.

2. UI/UX Experience

- Built entirely with **Jetpack Compose**, offering a modular and modern UI.
- **Lottie animations** were used for empty states and successful save operations to enhance interactivity.

3. State Handling and Persistence

- **MVVM architecture** enabled clean separation of UI and logic layers.
- State restoration and recomposition were optimized to prevent data loss during app pause/resume cycles.

4. Performance and Responsiveness

- The app maintained consistent performance across multiple Android versions and screen sizes.
- Optimized recomposition ensured fluid screen transitions and low memory footprint

CHAPTER 8

CONCLUSION & FUTURE ENHANCEMENTS

Conclusion

The **DAY TO DAY AutoDiary App** successfully meets its goals of offering a distraction-free, simple, and modern journaling experience for Android users.

Developed using **Kotlin, Jetpack Compose**, and **Firebase**, the app demonstrates how modern Android frameworks can be harnessed to build robust, real-time applications even with a minimal UI.

The **MVVM architecture** ensures maintainability, while **Firebase integration** provides secure and real-time entry storage. The app's clean interface and intuitive flow make it suitable for everyday journaling or emotional tracking.

As a scalable MVP, Lazy Logger sets the foundation for further feature-rich enhancements, aiming to turn into a powerful diary and self-reflection tool.

Future Enhancements

To elevate the app into a production-ready retail platform, the following improvements can be made:

1. User Authentication

- Integrate **Firebase Authentication** to allow secure sign-in/sign-up via email, Google, or phone number.

- Enable personalized carts, order history, and wishlists.

2. Search and Filtering

- Add search functionality to allow users to filter diary entries by date, mood, or keywords.

3. Media Attachments

- Allow users to attach images, audio, and videos to their diary entries.

4. Push Notifications

- Use **Firestore Cloud Messaging (FCM)** to notify users about deals, cart reminders, and order status updates.

5. Sentiment Analysis & Analytics

- Add **Natural Language Processing (NLP)** to detect mood trends and provide journaling insights.

CHAPTER-9

1. GoogleFirebase.(2023).FirebaseRealtimeDatabaseDocumentation.Retrieved from: <https://firebase.google.com/docs/database>
2. AndroidDevelopers.(2023).JetpackComposeDocumentation.Retrieved from: <https://developer.android.com/jetpack/compose>
3. GoogleDevelopers.(2023).KotlinforAndroidDevelopment.Retrieved from: <https://developer.android.com/kotlin>
4. Razorpay. (2023). Android Integration Guide. Retrieved from: <https://razorpay.com/docs/payments/payment-gateway/android-integration/>
5. Stripe.(2023).StripeAndroidSDKDocumentation.Retrieved from: <https://stripe.com/docs/payments/accept-a-payment>
6. MaterialDesign.(2023).Material3GuidelinesforCompose.Retrieved from: <https://m3.material.io/>
7. FirebaseAuthentication.(2023).FirebaseAuthforAndroid.Retrieved from: <https://firebase.google.com/docs/auth/android/start>