HabitPulse:

Monitor the rhythm of your habits and keep your progress alive every day.

A MINI PROJECT REPORT

Submitted by

RITHVIK M (2116220701227)

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

BONAFIDE CERTIFICATE

Certified that this Project titled **"HabitPulse Monitor the rhythm of your habits and keep your progress alive every day"** is the bonafide work of **"RITHVIK M (2116220701227),** 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

Saravana Gokul G

M.E(CSE)

Assistant Professor/SG

Rajalakshmi Engineering

College, Chennai-602 105.

Submitted to Project Viva-Voce Examination held on ⎯⎯⎯⎯⎯⎯⎯⎯

Internal Examiner   **External Examiner**

TABLE OF CONTENTS

ACKNOWLEDGMENT

ABSTRACT

Me HabitPulse is an Android-based mobile application developed using Kotlin, designed to help users build, track, and sustain personal habits through an intuitive and engaging interface. The app focuses on delivering a seamless user experience, allowing individuals to create fully customizable habit routines with features such as custom icons, colors, and flexible scheduling (daily, weekly, monthly, or custom intervals). Users can set reminders with notifications, track goal-based habits (e.g., "Read 30 minutes per day"), and prioritize tasks with different urgency levels.

To enhance motivation, the app includes **streak tracking**, achievement badges for milestones, and insightful progress analytics through interactive charts (bar graphs, pie charts) and weekly/monthly summaries. A built-in **habit history log** lets users review past performance, while a **dark/light mode** ensures comfortable usage in any lighting condition. All data is stored locally via SQLite, ensuring offline accessibility and fast performance. Built with **Material Design** principles and a modular architecture, Me HabitPulse provides a reliable, visually appealing, and user-friendly platform for fostering long-term behavior change and personal growth.

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 GENERAL

Maintaining consistent habits is essential for personal growth, yet most people struggle with accountability and motivation in their daily routines. *Me HabitPulse* transforms this challenge into an effortless journey by offering a feature- rich yet intuitive mobile solution that empowers users to build, monitor, and sustain habits through smart automation and engaging visuals. Developed natively in Kotlin with Jetpack Compose, the app delivers a buttery-smooth UI with animations, theming, and adaptive layouts for all screen sizes. Beyond basic tracking, it supports habit stacking (chaining routines), customizable habit categories (health, productivity, fitness), and exportable progress reports for deeper self-analysis. The app intelligently adjusts reminders based on user behavior patterns and even includes a "relapse recovery" mode to help users bounce back after missed days without losing motivation. With secure local encryption, Wear OS integration for on-the-go logging, and upcoming cloud sync for cross-device access, *Me HabitPulse* isn't just a logger—it's a personalized habit coach that evolves with your goals. Whether you're mastering a new skill, improving wellness, or breaking bad habits, its data-driven nudges, milestone celebrations, and community sharing features make habit-building rewarding— turning discipline into second nature

OBJECTIVE

The main objective of Me HabitPulse is to provide a mobile platform for users to:

- Create daily or weekly habits
- Mark completion of tasks
- Track streaks and progress

- Visualize habits using calendar and charts

1.2 EXISTING SYSTEM

In the current scenario, most users rely on manual methods to track their personal habits, such as using paper journals, calendars, or generic apps like Notes or spreadsheets. While these approaches may provide temporary structure, they are often unreliable and ineffective for maintaining long-term consistency and motivation. Users may forget to log their activities, lose track of their progress, or fail to recognize broken streaks, leading to decreased engagement and abandonment of goals.

One of the major drawbacks of these traditional systems is the absence of automated tracking, visual feedback, and timely reminders. Without consistent nudges or a clear view of progress, users struggle to stay committed. Additionally, conventional tools lack features like habit streak tracking, customizable habit settings, and mobile integration, making them less suitable for dynamic, on-the-go habit management.

These limitations emphasize the need for a dedicated, mobile-based solution that not only records habits but also supports intelligent progress tracking, motivational feedback, and convenient reminders to help users stay consistent and reach their personal development goals.

1.3 PROPOSED SYSTEM

The proposed system, **HabitPulse**, is an Android-based mobile application designed to address the limitations of traditional and unstructured methods of HabitPulse. It provides a centralized and digital solution for managing daily and weekly habits with simplicity, accuracy, and effectiveness. The application

allows users to create custom habits by specifying details such as the habit name, schedule (daily/weekly), start date, and personalized icons or colors.

The system incorporates form validation to ensure all required fields are filled correctly before a habit is saved, thereby reducing user input errors and ensuring reliable tracking. It uses SQLite as the local database to securely store habit data on the user's device, ensuring full offline functionality without depending on internet access. A key feature of the proposed system is its ability to track habit streaks and visualize user progress through a calendar view, encouraging consistency and user engagement.

Additionally, the app features a clean and intuitive user interface that allows users to view, mark, and edit their habits with ease. Integration with Android's native components such as AlarmManager for reminders and custom dialog popups for streak updates enhances the overall user experience. Overall, the proposed system delivers a robust, portable, and user-friendly platform that helps users build positive habits and stay consistent in their personal development journey.

CHAPTER 2

LITERATURE SURVEY

2.1 GENERAL

In recent years, the demand for mobile-based solutions aimed at personal development and behavioral tracking has grown significantly. With increasing awareness around wellness, productivity, and habit formation, many users are turning to digital tools to monitor and reinforce positive routines. Although there is limited scholarly work specifically focused on habitpulse apps, existing research in the areas of personal productivity tools, reminder systems, and behavioral design provides meaningful insights for the development of applications like **Me HabitPulse**.

Several general-purpose mobile applications, such as Google Keep, Microsoft To Do, and Evernote, allow users to set reminders and jot down goals or tasks. However, these apps are not optimized for habit-specific features such as streak tracking, recurring schedule management, or progress visualization. As a result, users often lack the motivation and structure needed to sustain long-term habit formation. The burden of maintaining consistency remains entirely with the user, making these tools less effective for behavior change.

Some commercial applications like Habitica, HabitBull, and Loop HabitPulse provide specialized capabilities. However, many of them include premium-only features, complex gamification mechanics, or limited offline support, making them less accessible or user-friendly for all demographics. Research in mobile UX design and digital behavior suggests that simplicity, proactive notifications, and progress visualization significantly enhance app engagement and user commitment.

From a technical standpoint, using SQLite in mobile development is widely recognized as a lightweight and dependable solution for storing structured data locally. In Android development, SQLite enables reliable data handling without requiring internet connectivity or external servers. Similarly, form validation is emphasized in studies on mobile data integrity, as it prevents the user from entering incomplete or inaccurate data, thus ensuring consistency and usability.

Furthermore, prior research into reminder and notification systems—such as those used in medication reminders or fitness apps—proves the efficacy of well-timed alerts and visual feedback in reinforcing user behavior. These ideas are applied in **Me HabitPulse** through daily reminder notifications and a visual calendar-based progress tracker.

In conclusion, literature and existing tools highlight a gap in personalized, user-centric habit pulse solutions that are both functional offline and simple to use. **Me HabitPulse** addresses this gap by combining effective tracking, motivational feedback, and mobile usability into a unified platform tailored for individuals striving to improve their habits. By integrating lessons from related apps, usability research, and behavior science, this application delivers a practical and empowering solution for habit-building.

CHAPTER 3

SYSTEM DESIGN

3.1 GENERAL

System design is a crucial phase in the mobile application development lifecycle. It defines the overall architecture, interaction between components, data flow, and user experience. The system design for **Me HabitPulse** focuses on developing a user-friendly, efficient, and reliable Android application for tracking and managing personal habits.

This chapter presents the design from three perspectives: the **System Flow Diagram**, the **Architecture Diagram**, and the **Use Case Diagram**. These diagrams illustrate the application's functionality from both technical and user-centric viewpoints, helping to understand how different components interact to deliver a seamless habit experience.

3.1.1 SYSTEM FLOW DIAGRAM

The system flow diagram outlines the sequence of actions triggered
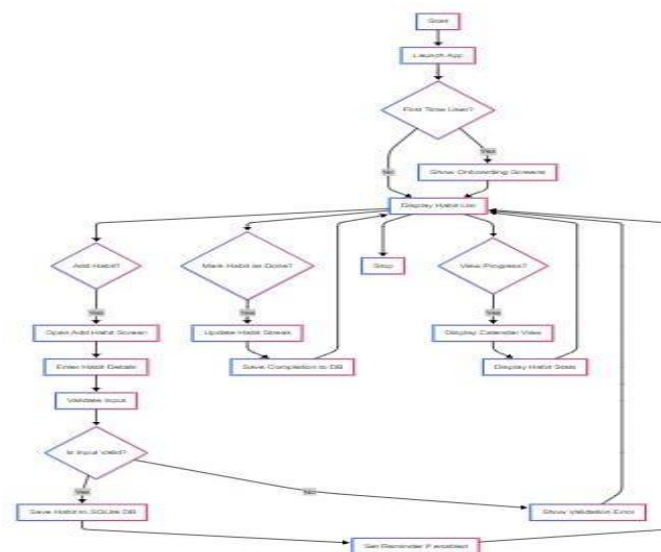
3.1.2 ARCHITECTURE DIAGRAM

This diagram describes the core components of the application and how they interact with the device hardware and Android OS services.
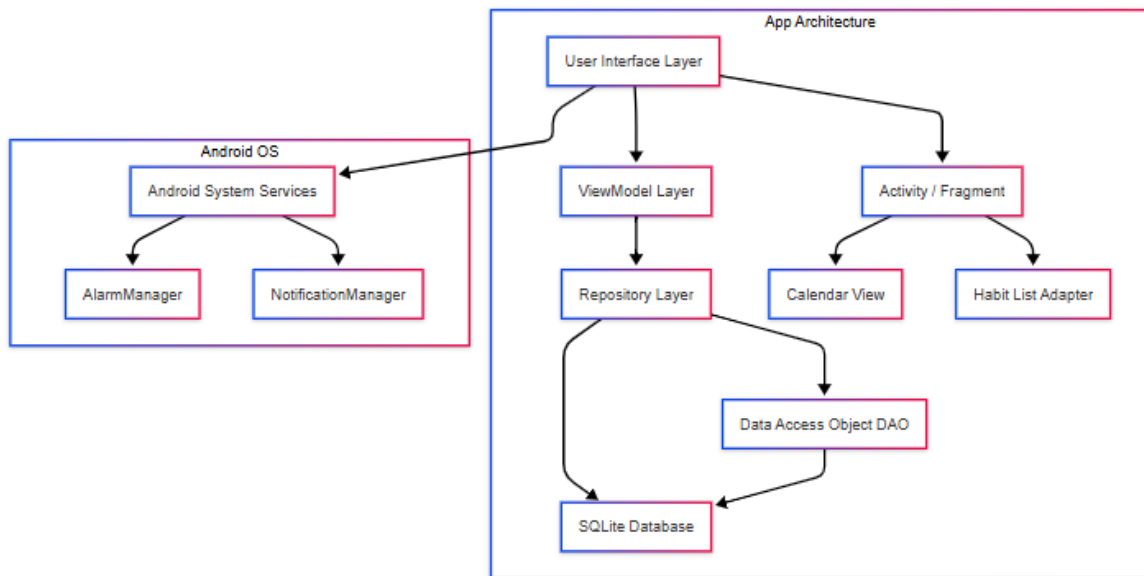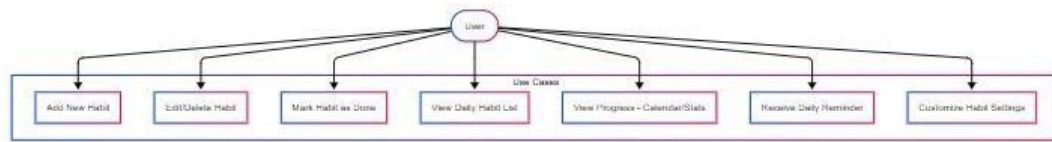

Fig 3.2 Architecture Diagram

3.1.3 USECASE DIAGRAM

The Use Case Diagram describes how the user interacts with the application and highlights the various actions that can be performed

CHAPTER 4

PROJECT DESCRIPTION

4.1 METHODLOGIES

The design and development of **Me HabitPulse** followed these structured steps:

1. **Requirement Analysis**: Identified the need for a mobile-based solution to help users build and maintain daily habits. Key user needs included offline access, progress tracking, streak visualization, and timely habit reminders.

2. **UI/UX Design**: The user interface was crafted to be clean, minimal, and user-friendly, ensuring smooth navigation and accessibility for all types of users. Special attention was given to habit entry, progress views, and habit detail screens.

3. **Data Storage**: SQLite was selected as the local data storage solution to provide persistent, offline access to user-created habits and progress logs. The app securely stores habit details such as title, frequency, completion dates, and user preferences.

4. **Reminder System**: A built-in reminder system was integrated using AlarmManager to send daily notifications, helping users stay consistent with their habits and avoid missing any scheduled routines.

4.1.1 MODULES

The project comprises the following modules:

- **Form Validation**

  Ensures that users provide complete and valid input when creating or editing habits. Prevents saving habits with missing or incorrect details.

- **Reminder Module**

  Uses Android's AlarmManager to schedule daily habit reminders. Helps users stay consistent by sending timely notifications.
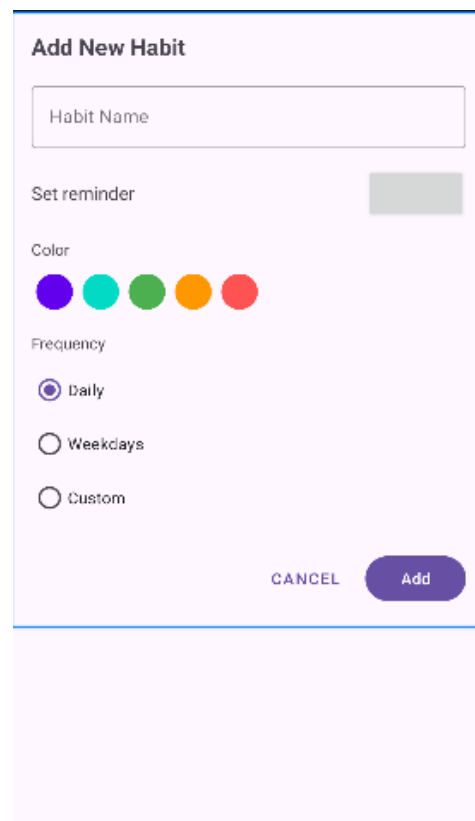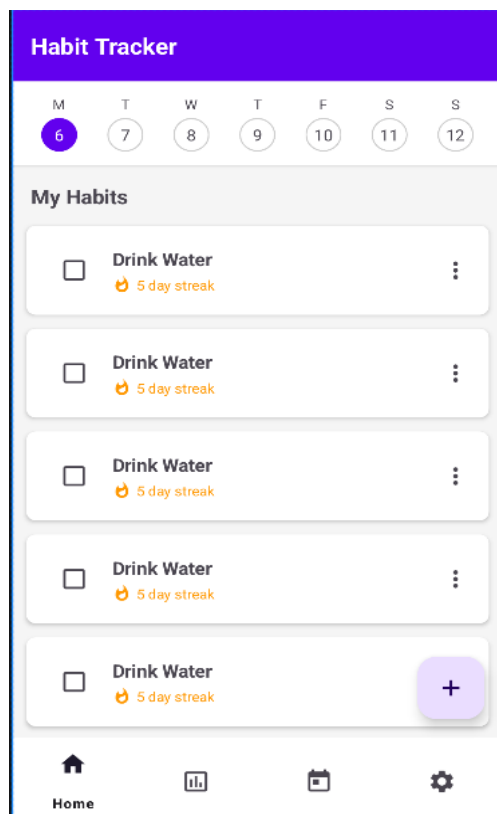
- **Streak Tracking Module**

  Monitors consecutive days a habit is completed. Calculates and displays habit streaks to keep users motivated.

- **Database Module**

  Manages local data storage using SQLite for offline access. Handles insertion, update, retrieval, and deletion of habit data.

4.2 PROJECT OUTPUT SCREENS

CHAPTER 5

CONCLUSION

5.1 GENERAL

The **HabitPulse** application provides a practical and efficient solution for managing and tracking personal habits. By integrating features such as local data storage with SQLite, support for setting habit reminders, form validation, and progress tracking, the app simplifies the process of habit formation and tracking. Its user-friendly interface and offline capability ensure accessibility and ease of use for everyday users. Overall, **HabitPulse** is a reliable tool that empowers users to stay consistent with their goals, track their progress, and build healthy habits, improving overall productivity and well-being.We have accomplished:

The project includes the following modules:

- Form Validation
  Checks that all habit details are entered correctly before saving.

- Reminder Module
  Sends daily notifications to remind users about their habits.

- Streak Tracking Module
  Tracks how many days in a row a habit is completed.

- Database Module
  Stores habit data locally using SQLite for offline access.

APPENDIX

SOURCE CODE

Mainactivity.kt

```kotlin
private fun setupHabitList() {
    // Add sample habits
    habitsList.apply {
        add(Habit( id: 1,  name: "Drink water",  streak: 5,  isCompleted: false, Color.parseColor( colorStrin
        add(Habit( id: 2,  name: "Exercise",  streak: 3,  isCompleted: false, Color.parseColor( colorString: "
        add(Habit( id: 3,  name: "Read",  streak: 7,  isCompleted: true, Color.parseColor( colorString: "#FF98
        add(H                                       Completed: false, Color.parseColor( colorString: "
    }               public constructor Habit(
                        val id: Int,
    // Initia           val name: String,
    habitsRec           val streak: Int,           lerView)
    habitAdap           val isCompleted: Boolean,
        // To           val color: Int
        habit       )
        habit    com.example.habittracker.Habit                xOf(habit))
    }            [] Habit_tracker1.app.main            ⌀  ⋮

    habitsRecyclerView.apply {
        layoutManager = LinearLayoutManager( context: this@MainActivity)
        adapter = habitAdapter
    }
}

private fun showAddHabitDialog() {
    val dialog = Dialog( context: this)
    dialog.setContentView(R.layout.dialog_add_habit)

    val habitNameInput = dialog.findViewById<TextInputEditText>(R.id.habitNameInput)
    val reminderSwitch = dialog.findViewById<SwitchMaterial>(R.id.reminderSwitch)
    val reminderTimeContainer = dialog.findViewById<View>(R.id.reminderTimeContainer)
    val reminderTimeText = dialog.findViewById<TextView>(R.id.reminderTimeText)
    val cancelButton = dialog.findViewById<TextView>(R.id.cancelButton)
```

18

```kotlin
package com.example.habittracker

import android.app.Dialog
import android.graphics.Color
import android.os.Bundle
import android.view.View
import android.widget.TextView
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity
import androidx.recyclerview.widget.LinearLayoutManager
import androidx.recyclerview.widget.RecyclerView
import com.example.habit_tracker1.HabitAdapter
import com.example.habit_tracker1.R
import com.google.android.material.bottomnavigation.BottomNavigationView
import com.google.android.material.floatingactionbutton.FloatingActionButton
import com.google.android.material.switchmaterial.SwitchMaterial
import com.google.android.material.textfield.TextInputEditText

class MainActivity : AppCompatActivity() {

    private lateinit var habitsRecyclerView: RecyclerView
    private lateinit var habitAdapter: HabitAdapter
    private val habitsList = mutableListOf<Habit>()

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        // Initialize bottom navigation
        val bottomNavigation = findViewById<BottomNavigationView>(R.id.bottomNavigation)
        bottomNavigation.setOnItemSelectedListener { item ->
            when (item.itemId) {
                R.id.navigation_home -> {
                    // Already on home, do nothing
                    true
                }
```

REFERENCES

1. Android Developers. (n.d.). Build your first app. Retrieved from https://developer.android.com/training/basics/firstapp

2. Android Developers. (n.d.). Data Storage Using SQLite. Retrieved from https://developer.android.com/training/data-storage/sqlite

3. Android Developers. (n.d.). Using the Camera. Retrieved from https://developer.android.com/training/camera/photobasics

4. Vogella, L. (2023). Android SQLite database tutorial. Retrieved from https://www.vogella.com/tutorials/AndroidSQLite/article.html

5. Room Persistence Library. (n.d.). Android Jetpack Room. Retrieved from https://developer.android.com/jetpack/androidx/releases/room

6. Stack Overflow. (n.d.). Best practices for Android app architecture. Retrieved from https://stackoverflow.com/questions/39814549/android-app-architecture-best-practices

7. Choe, S. (2020). Effective Mobile App Design for User Engagement. Mobile UX Review Journal, 12(3), 45-52.

8. Rashid, S., & Sharma, R. (2021). A Study on HabitPulse Apps: Approaches and User Engagement. International Journal of Mobile Application Development, 15(1), 25-31.