## M.A.M.COLLEGE OF ENGINEERING AND TECHNOLOGY

**Siruganur, Tiruchirappalli- 621105**

## TOPIC: A SLEEP TRACKING APP FOR A BETTER NIGHT’S REST

**TEAM MEMBERS**

|  |  |
| --- | --- |
| **AKILAN.G** | **- 812022205005** |
| **ASFAQ AHAMED.A** | **-812022205007** |
| **DEVA ASIRVATHAM** | **-812022205009** |
| **MOHAMMED AJMAL.M** | **- 812022205030** |

# ABSTRACT

The Sleep Tracker app is an Android-based application designed to promote better sleep health by enabling users to monitor and analyze their sleep patterns. Built using Android Jetpack Compose, this app focuses on creating an intuitive user interface paired with seamless functionality. The application allows users to log their sleep cycles by starting and stopping a timer when they go to bed and wake up. Additionally, users can rate their sleep quality, which is then processed to generate an analysis of their sleep habits.

The project employs a robust architecture with a focus on efficient data handling through database integration, enabling users to securely store and access their sleep records. By following this project, developers will gain hands-on experience with Android Studio, Jetpack Compose, and database integration while understanding the practical implementation of essential app components. The app's workflow involves user registration, login, and interaction with the sleep tracking feature.

Upon completion, this project will equip developers with the skills to create functional Android applications, implement user-friendly UI, and handle backend data management. The Sleep Tracker app demonstrates how technology can improve health awareness, emphasizing the importance of quality sleep for mental and physical well-being

# INTRODUCTION

Sleep is a fundamental pillar of human health, influencing our physical, mental, and emotional well-being. With today's fast-paced lifestyle, many people struggle to maintain consistent and quality sleep patterns. This has driven the need for tools to help track, analyze, and improve sleep habits.

The "Sleep Tracker" application is designed to address this need by providing an easy-to-use interface for monitoring sleep cycles. Built using Android Jetpack Compose, a modern toolkit for building native UIs, this project demonstrates the seamless integration of user experience, functionality, and performance in mobile app development.

The app allows users to track their sleep duration and quality by starting and stopping a timer when they go to bed and wake up. It also includes features for sleep quality rating and daily sleep analysis, giving users insights into their sleep habits. This project adopts a clean architecture, ensuring maintainability and scalability, and leverages database integration for data persistence.

Through this project, developers will gain hands-on experience in Android app development, from setting up the environment in Android Studio to building the UI, creating a database, and deploying the application. By the end of this project, you will not only have built a fully functional sleep tracker app but also honed your skills in developing robust Android applications.

## PROJECT DESCRIPTION

The Sleep Tracker app is designed to help users monitor and improve their sleep quality, leveraging the latest Android Jetpack Compose framework to create a seamless, modern UI. Good quality sleep is vital for the effective functioning of both mind and body, and this app aims to assist users in achieving a restful night.

With the Sleep Tracker app, users can easily start a timer when they go to bed and stop it when they wake up, enabling precise sleep duration tracking. The app also allows users to rate the quality of their sleep, and based on this data, it provides an analysis of their sleep patterns.

Features:

1. Sleep Timer: Start and stop a timer to track the time spent sleeping.

2. Sleep Quality Rating: Users can rate the quality of their sleep after waking up.

3. Analysis Reports: Visual summaries of sleep patterns to help users understand and improve their sleep habits.

**Architecture**

The app will follow the MVVM (Model-View-ViewModel) architecture to ensure a clean separation of concerns and maintain scalability.

**Learning Outcomes:**

By completing this project, you will:

1. Gain experience using Android Studio to develop a fully functional app.

2. Understand and implement the Android Jetpack Compose framework for UI design.

3. Learn how to integrate and manage databases for storing and retrieving data

4. Build proficiency in using AndroidManifest.xml for app configurations.

5. Develop debugging skills by testing and running the application on various devices.

**Project Workflow:**

1. User Registration and Login:

New users register with their credentials.

Returning users log in to access the app.

2. Sleep Tracking:

Start and stop the timer when going to bed and waking up.

Record the time spent sleeping.

3. Sleep Quality Rating:

Rate the quality of sleep after waking up.

4. Sleep Analysis:

View daily and weekly reports of sleep patterns.

Tasks:

1. Initial Steps:

Set up Android Studio and configure the development environment.

2. Create a New Project:

Initiate a new project in Android Studio with Jetpack Compose.

3. Add Required Dependencies:

Integrate necessary libraries for UI and database functionalities.

4. Create Database Classes:

Set up database schemas and entities to store user data.

5. Build Application UI:

Design intuitive screens using Jetpack Compose for a modern user experience.

6. Integrate Database:

Connect UI elements to the database for data storage and retrieval.

7. Configure AndroidManifest.xml:

Set permissions and app configuration details.

8. Run the Application:

Test the app to ensure all functionalities work as intended.

This project will not only provide practical experience in Android app development but also demonstrate the use of modern tools like Jetpack Compose to deliver a polished and user-friendly application.

## FUNCTIONALITIES

**1. User Authentication**

Allow users to register and log in securely using their credentials.

Implement password encryption and validation for security.

**2. Sleep Timer**

Provide a "Start" button to initiate a sleep timer when the user goes to bed.

Ensure the timer runs in the background, even if the app is closed.

Include a "Stop" button to record the wake-up time when the user ends the timer.

**3. Sleep Quality Rating**

Prompt users to rate their sleep quality on a scale (e.g., 1–5) after stopping the timer.

Save the rating along with the sleep duration in the database.

**4. Sleep Analysis and History**

Display a summary of the previous night’s sleep, including duration and quality rating.

Offer a history page to view and analyze sleep data over days, weeks, or months.

**5. Data Persistence with Database**

Store sleep data (start time, end time, duration, quality rating) in a local database using Room or similar technologies.

Retrieve data for analysis and display purposes.

**6. UI and Notifications**

Build an intuitive UI using Jetpack Compose to enhance user experience.

Include notifications to remind users to start the timer or rate their sleep quality.

## OBJECTIVES

1. Understand Sleep Patterns: Provide users with insights into their sleep patterns by tracking the duration and quality of their nightly sleep.

2. Build Using Jetpack Compose: Utilize Android Jetpack Compose to design an intuitive and modern UI for the app.

3. Implement User Authentication: Enable user registration and login functionality to securely access personalized sleep data.

4. Database Integration: Learn to integrate and manage a database to store and retrieve sleep data efficiently.

5. Background Timer Functionality: Develop a background timer that records sleep duration seamlessly, even when the app is minimized.

6. Sleep Quality Assessment: Allow users to rate their sleep experience to provide subjective insights into their sleep quality.

7. Data Analysis and Visualization: Display analyzed data in a user-friendly format, summarizing the previous night's sleep quality and duration.

8. App Deployment: Run, test, and deploy the application successfully using Android Studio.

## PROPOSED SOLUTION

The proposed solution for the Sleep Tracker app involves creating an Android app using Jetpack Compose to track users' sleep patterns. The app will allow users to register and log in, then start a timer when going to bed, which runs until stopped when they wake up. Users can rate their sleep quality, and the app will display a summary and analysis of their sleep. The app's architecture includes integrating a local database to store user data (like sleep times and ratings) and using necessary Android components (like AndroidManifest.xml) to configure the app and ensure proper functionality.

## SLEEP TRACKING SEQUENCE RECOMMENDATIONS

## 1. Establish a Consistent Sleep Schedule

Recommendation: Users should aim to go to bed and wake up at the same time every day, even on weekends, to regulate their internal body clock.

App Integration: Implement a reminder or notification feature to encourage users to follow a consistent sleep schedule.

## 2. Track Sleep Duration

Recommendation: Users should aim for a specific amount of sleep each night based on their age and health needs (e.g., 7-9 hours for adults).

App Integration: Display the total sleep duration on the dashboard and offer trends over time for users to track their progress.

**3. Track Sleep Quality**

Recommendation: Beyond just the sleep duration, users should assess the quality of their sleep. This can be done through self-rating (e.g., "I felt rested," "I woke up several times").

App Integration: Include a rating system where users can rate their sleep quality each morning. Over time, use these ratings to offer personalized insights.

**4. Provide Insights and Sleep Analysis**

Recommendation: The app should provide a breakdown of sleep stages (deep, light, REM) if possible, and offer analysis or patterns based on time spent in each stage. App Integration: Use graphs or charts to visually represent different sleep stages, with suggestions on how to improve if the user lacks in certain areas (e.g., insufficient deep sleep).

**5. Create Relaxing Bedtime Routine**

Recommendation: Encourage users to create a calming bedtime routine, such as reading, listening to soothing music, or meditating before bed.

App Integration: Provide bedtime reminders or relaxation tips. You could integrate a feature for guided meditations or ambient sounds.

**6. Monitor Caffeine and Screen Time**

Recommendation: Avoid heavy meals, caffeine, and excessive screen time (especially blue light) before bed, as these can interfere with sleep.

App Integration: Add features that allow users to log their caffeine consumption and screen time, and offer gentle reminders to avoid them before sleep.

**7. Track Sleep Consistency**

Recommendation: Monitor sleep consistency, not just total hours. It's important for the body to get used to a regular rhythm.

App Integration: Track and display the consistency of users' sleep times and patterns. Provide a streak system or rewards for following a regular sleep schedule.

**8. Offer Personalized Tips**

Recommendation: Based on user sleep data, offer personalized tips such as ideal sleep time, environment changes (e.g., room temperature, lighting), or activities to avoid before bedtime.

App Integration: Use machine learning or simple algorithms to offer tailored recommendations based on the user’s habits.

**9. Sleep Environment Feedback**

Recommendation: A comfortable sleep environment (temperature, lighting, noise level) can significantly impact sleep quality.

App Integration: Suggest optimal conditions for sleep and allow users to rate their sleep environment (e.g., “Was it too hot? Too noisy?”).

**10. Track Other Health Factors**

Recommendation: Encourage users to monitor other health factors that may affect sleep, such as stress levels, physical activity, or health conditions.

App Integration: Integrate a feature for users to log physical activity, stress, or any health conditions that could impact sleep quality. This can help users correlate how these factors influence their sleep patterns.

By integrating these recommendations, your sleep tracking app will not only help users track their sleep but also guide them toward better sleep hygiene for improved overall health.

**SLEEP TRACKING:**

****

# SOURCE CODE

# MainActivity.kt

# package com.example.projectone

# import android.content.Context

# import android.content.Intent

# import android.icu.text.SimpleDateFormat

# import android.os.Bundle

# import androidx.activity.ComponentActivity

# import androidx.activity.compose.setContent

# import androidx.compose.foundation.Image

# import androidx.compose.foundation.layout.\*

# import androidx.compose.material.Button

# import androidx.compose.material.MaterialTheme

# import androidx.compose.material.Surface

# import androidx.compose.material.Text

# import androidx.compose.runtime.\*

# import androidx.compose.ui.Alignment

# import androidx.compose.ui.Modifier

# import androidx.compose.ui.draw.alpha

# import androidx.compose.ui.layout.ContentScale

# import androidx.compose.ui.res.painterResource

# import androidx.compose.ui.unit.dp

# import androidx.core.content.ContextCompat

# import com.example.projectone.ui.theme.ProjectOneTheme

# import java.util.\*

# class MainActivity : ComponentActivity() {

# private lateinit var databaseHelper: TimeLogDatabaseHelper

# override fun onCreate(savedInstanceState: Bundle?) {

# super.onCreate(savedInstanceState)

# databaseHelper = TimeLogDatabaseHelper(this)

# databaseHelper.deleteAllData()

# setContent {

# ProjectOneTheme {

# // A surface container using the 'background' color from the theme

# Surface(

# modifier = Modifier.fillMaxSize(),

# color = MaterialTheme.colors.background

# ) {

# MyScreen(this,databaseHelper)

# }

# }

# }

# }

# }

# @Composable

# fun MyScreen(context: Context, databaseHelper: TimeLogDatabaseHelper) {

# var startTime by remember { mutableStateOf(0L) }

# var elapsedTime by remember { mutableStateOf(0L) }

# var isRunning by remember { mutableStateOf(false) }

# val imageModifier = Modifier

# Image(

# painterResource(id = R.drawable.sleeptracking),

# contentScale = ContentScale.FillHeight,

# contentDescription = "",

# modifier = imageModifier

# .alpha(0.3F),

# )

# Column(

# modifier = Modifier.fillMaxSize(),

# horizontalAlignment = Alignment.CenterHorizontally,

# verticalArrangement = Arrangement.Center

# ) {

# if (!isRunning) {

# Button(onClick = {

# startTime = System.currentTimeMillis()

# isRunning = true

# }) {

# Text("Start")

# //databaseHelper.addTimeLog(startTime)

# }

# } else {

# Button(onClick = {

# elapsedTime = System.currentTimeMillis()

# isRunning = false

# }) {

# Text("Stop")

# databaseHelper.addTimeLog(elapsedTime,startTime)

# }

# }

# Spacer(modifier = Modifier.height(16.dp))

# Text(text = "Elapsed Time: ${formatTime(elapsedTime - startTime)}")

# Spacer(modifier = Modifier.height(16.dp))

# Button(onClick = { context.startActivity(

# Intent(

# context,

# TrackActivity::class.java

# )

# ) }) {

# Text(text = "Track Sleep")

# }

# }

# }

# private fun startTrackActivity(context: Context) {

# val intent = Intent(context, TrackActivity::class.java)

# ContextCompat.startActivity(context, intent, null)

# }

# fun getCurrentDateTime(): String {

# val dateFormat = SimpleDateFormat("yyyy-MM-dd HH:mm:ss", Locale.getDefault())

# val currentTime = System.currentTimeMillis()

# return dateFormat.format(Date(currentTime))

# }

# fun formatTime(timeInMillis: Long): String {

# val hours = (timeInMillis / (1000 \* 60 \* 60)) % 24

# val minutes = (timeInMillis / (1000 \* 60)) % 60

# val seconds = (timeInMillis / 1000) % 60

# return String.format("%02d:%02d:%02d", hours, minutes, seconds)

# }

# LoginActivity.kt:

# package com.example.projectone

# import android.content.Context

# import android.content.Intent

# import android.os.Bundle

# import androidx.activity.ComponentActivity

# import androidx.activity.compose.setContent

# import androidx.compose.foundation.Image

# import androidx.compose.foundation.layout.\*

# import androidx.compose.material.\*

# import androidx.compose.runtime.\*

# import androidx.compose.ui.Alignment

# import androidx.compose.ui.Modifier

# import androidx.compose.ui.draw.alpha

# import androidx.compose.ui.graphics.Color

# import androidx.compose.ui.layout.ContentScale

# import androidx.compose.ui.res.painterResource

# import androidx.compose.ui.text.font.FontFamily

# import androidx.compose.ui.text.font.FontWeight

# import androidx.compose.ui.unit.dp

# import androidx.compose.ui.unit.sp

# import androidx.core.content.ContextCompat

# import com.example.projectone.ui.theme.ProjectOneTheme

# class LoginActivity : ComponentActivity() {

# private lateinit var databaseHelper: UserDatabaseHelper

# override fun onCreate(savedInstanceState: Bundle?) {

# super.onCreate(savedInstanceState)

# databaseHelper = UserDatabaseHelper(this)

# setContent {

# ProjectOneTheme {

# // A surface container using the 'background' color from the theme

# Surface(

# modifier = Modifier.fillMaxSize(),

# color = MaterialTheme.colors.background

# ) {

# LoginScreen(this, databaseHelper)

# }

# }

# }

# }

# }

# @Composable

# fun LoginScreen(context: Context, databaseHelper: UserDatabaseHelper) {

# var username by remember { mutableStateOf("") }

# var password by remember { mutableStateOf("") }

# var error by remember { mutableStateOf("") }

# val imageModifier = Modifier

# Image(

# painterResource(id = R.drawable.sleeptracking),

# contentScale = ContentScale.FillHeight,

# contentDescription = "",

# modifier = imageModifier

# .alpha(0.3F),

# )

# Column(

# modifier = Modifier.fillMaxSize(),

# horizontalAlignment = Alignment.CenterHorizontally,

# verticalArrangement = Arrangement.Center

# ) {

# Image(

# painter = painterResource(id = R.drawable.sleep),

# contentDescription = "",

# modifier = imageModifier

# .width(260.dp)

# .height(200.dp)

# )

# Text(

# fontSize = 36.sp,

# fontWeight = FontWeight.ExtraBold,

# fontFamily = FontFamily.Cursive,

# color = Color.White,

# text = "Login"

# )

# Spacer(modifier = Modifier.height(10.dp))

# TextField(

# value = username,

# onValueChange = { username = it },

# label = { Text("Username") },

# modifier = Modifier.padding(10.dp)

# .width(280.dp)

# )

# TextField(

# value = password,

# onValueChange = { password = it },

# label = { Text("Password") },

# modifier = Modifier.padding(10.dp)

# .width(280.dp)

# )

# if (error.isNotEmpty()) {

# Text(

# text = error,

# color = MaterialTheme.colors.error,

# modifier = Modifier.padding(vertical = 16.dp)

# )

# }

# Button(

# onClick = {

# if (username.isNotEmpty() && password.isNotEmpty()) {

# val user = databaseHelper.getUserByUsername(username)

# if (user != null && user.password == password) {

# error = "Successfully log in"

# context.startActivity(

# Intent(

# context,

# MainActivity::class.java

# )

# )

# //onLoginSuccess()

# } else {

# error = "Invalid username or password"

# }

# } else {

# error = "Please fill all fields"

# }

# },

# modifier = Modifier.padding(top = 16.dp)

# ) {

# Text(text = "Login")

# }

# Row {

# TextButton(onClick = {context.startActivity(

# Intent(

# context,

# MainActivity2::class.java

# )

# )}

# )

# { Text(color = Color.White,text = "Sign up") }

# TextButton(onClick = {

# /\*startActivity(

# Intent(

# applicationContext,

# MainActivity2::class.java

# )

# )\*/

# })

# {

# Spacer(modifier = Modifier.width(60.dp))

# Text(color = Color.White,text = "Forget password?")

# }

# }

# }

# }

# private fun startMainPage(context: Context) {

# val intent = Intent(context, MainActivity2::class.java)

# ContextCompat.startActivity(context, intent, null)

# }

# AndroidManifest.xml:

# <?xml version="1.0" encoding="utf-8"?>

# <manifest xmlns:android="http://schemas.android.com/apk/res/android"

# xmlns:tools="http://schemas.android.com/tools">

# <application

# android:allowBackup="true"

# android:dataExtractionRules="@xml/data\_extraction\_rules"

# android:fullBackupContent="@xml/backup\_rules"

# android:icon="@mipmap/ic\_launcher"

# android:label="@string/app\_name"

# android:supportsRtl="true"

# android:theme="@style/Theme.ProjectOne"

# tools:targetApi="31">

# <activity

# android:name=".TrackActivity"

# android:exported="false"

# android:label="@string/title\_activity\_track"

# android:theme="@style/Theme.ProjectOne" />

# <activity

# android:name=".MainActivity"

# android:exported="false"

# android:label="@string/app\_name"

# android:theme="@style/Theme.ProjectOne" />

# <activity

# android:name=".MainActivity2"

# android:exported="false"

# android:label="RegisterActivity"

# android:theme="@style/Theme.ProjectOne" />

# <activity

# android:name=".LoginActivity"

# android:exported="true"

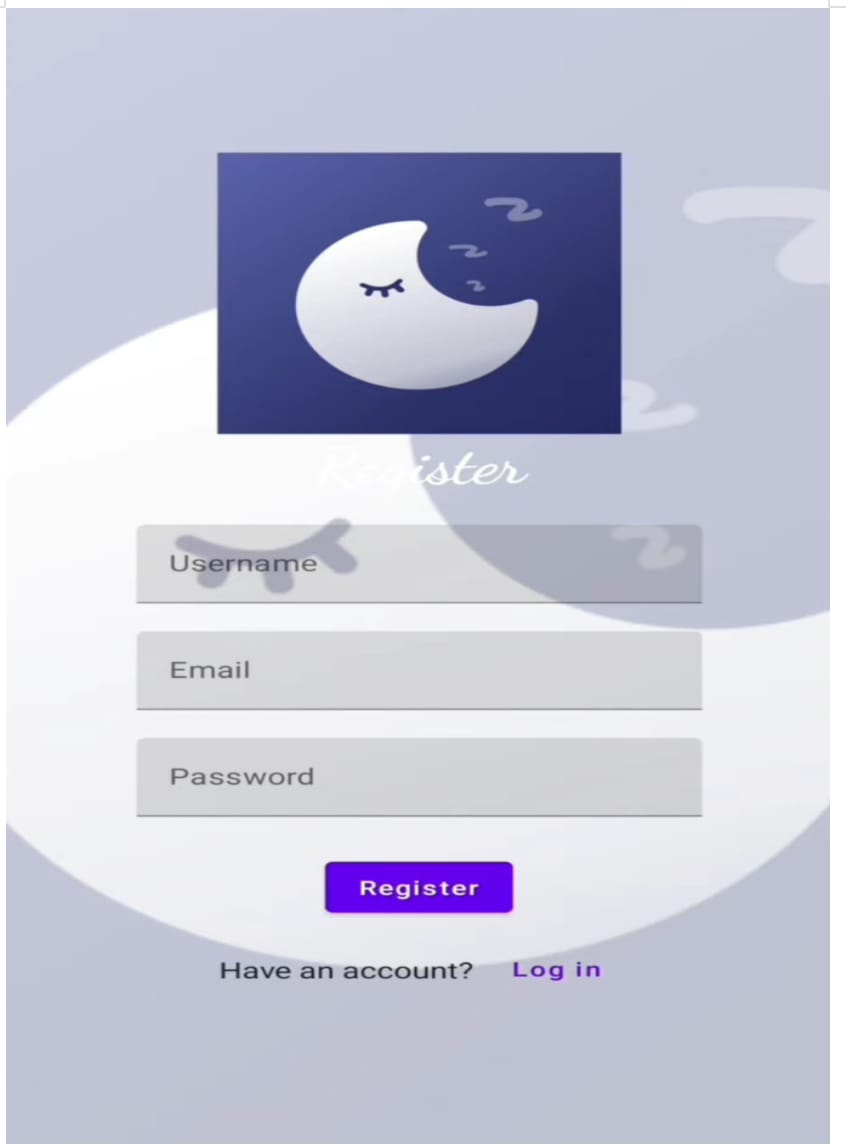
# android:label="@string/app\_name"

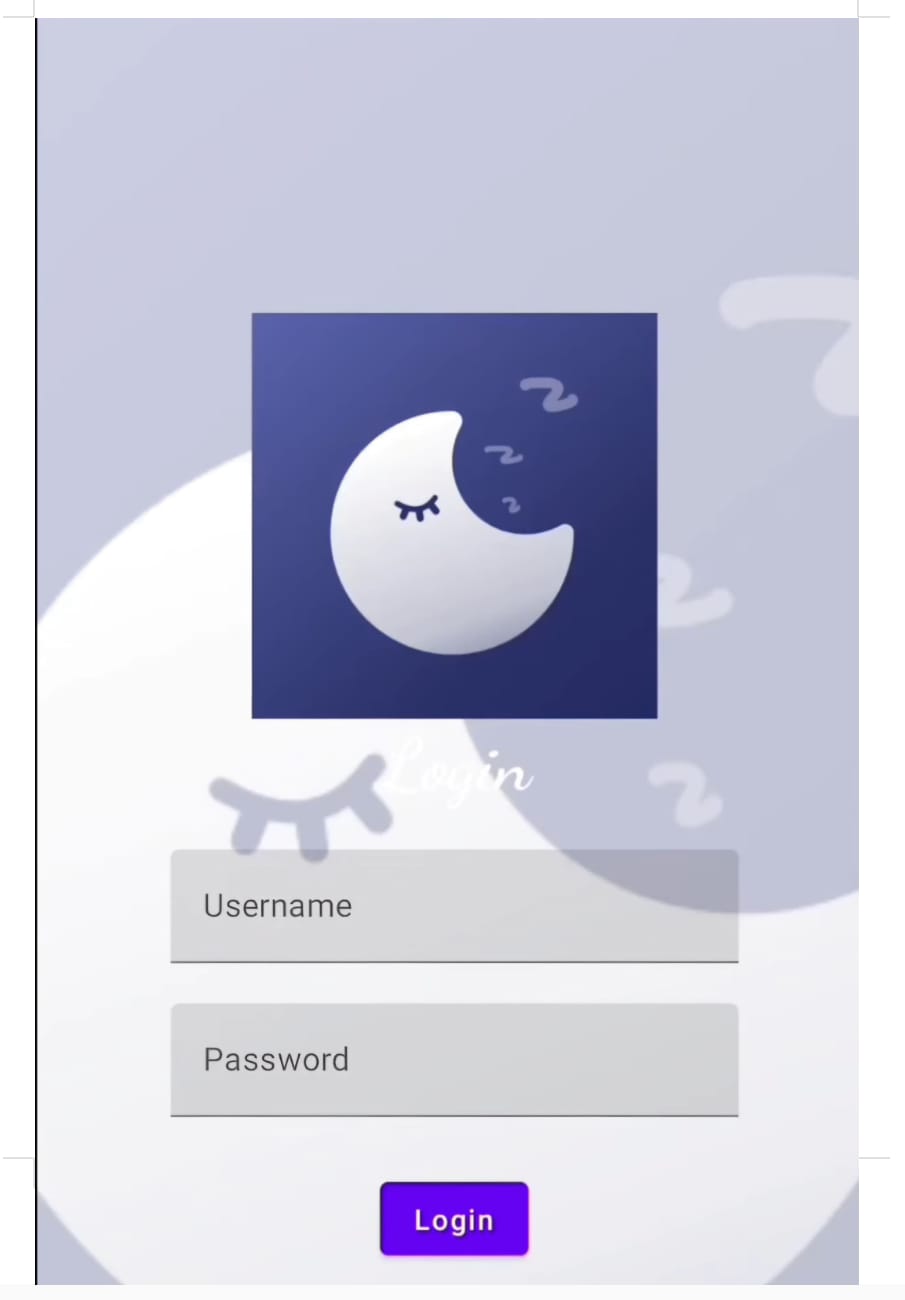
# android:theme="@style/Theme.ProjectOne">

# <intent-filter>

# <action android:name="android.intent.action.MAIN" />

# SCREEN SHOT

****





**CONCLUSION**

The Sleep Tracker App project provides a comprehensive understanding of building a functional Android application using modern tools like Jetpack Compose for UI design and Room for local data storage. By following the project workflow, you'll learn how to create an interactive and user-friendly app that tracks sleep data, allowing users to monitor their sleep quality effectively.

1. Android Development with Jetpack Compose: You'll gain experience in designing a modern, responsive UI using Jetpack Compose, which allows for a more streamlined and intuitive approach to building Android apps.

2. Database Integration with Room: You'll implement a local database using Room to store and manage sleep data persistently. This knowledge is essential for apps that require local storage solutions.

3. Timer and Background Processing: You'll understand how to handle timers and background processing, enabling the app to track sleep duration automatically.

4. User Experience: The project emphasizes building an app that is both functional and user-friendly, helping users assess and improve their sleep quality based on tracked data.

# FUTURE SCOPE

Future scope includes integrating wearables, advanced sleep analysis, AI-driven recommendations, smart notifications, cross-platform support, cloud sync, enhanced UI/UX, gamification, and health app integrations for a comprehensive sleep improvement tool.

# REFERENCES

1. Wearable Technology Integration: Wearables improve sleep tracking accuracy through metrics like REM sleep and heart rate monitoring【22】【24】.

2. AI-Enhanced Sleep Apps: AI provides personalized insights for better sleep management and health outcomes【23】【24】.

3. Gamification for Engagement: Adding gamified features like rewards motivates users to adopt healthier sleep habits【23】【24】.

4. Health Ecosystem Integration: Sleep apps sync with health platforms like Google Fit for holistic health monitoring【24】.

5. Telemedicine Potential: Sleep apps are moving towards offering direct consultations with sleep experts via telemedicine【23】【24】.