TRACK NIGHT'S : A SLEEP TRACKING APP FOR A BETTER NIGHT'S REST

PROJECT REPORT

Submitted by

ANTO STEPHIN C S (20203111506217)

ALTRIN F (20203111506216)

AKASH P S (20203111506215)

AJIN K K (20203111506214)

Submitted to Manonmaniam Sundaranar University. Tirunelveli

In partial fulfilment for the award of

BACHELOR OF SCIENCE IN COMPUTER SCIENCE

Under the Guidance of

Prof.D.H.KITTY SMAILIN M.Sc.,M.Phill

DEPARTMENT OF PG COMPUTER SCIENCE

NESAMONY MEMORIAL CHRISTIAN COLLEGE,

MARTHANDAM

KANYAKUMARI DISTRICT -629165

Re-accredited with 'A' grade by NAAC

MARCH 2023

DEPARTMENT OF PG COMPUTER SCIENCE

NESAMONY MEMORIAL CHRISTIAN COLLEGE

MARTHANDAM-629165, KANYAKUMARI DISTRICT

BONAFIDE CERTIFICATE

This is to certify that the project work entitled "**TRACK NIGHTS'S: A SLEEP TRACKING APP FOR A BETTER NIGHT'S"** is a bonafide record work done by **ANTO STEPHIN C S (20203111506217) , ALTRIN F (20203111506216) , AKASH P S (20203111506215) , AJIN K K (20203111506214)** during the academic year 2020-2023 in partial fulfilment of the requirements for the award of the degree in **BACHELOR OF SCIENCE IN COMPUTER SCIENCE** of Manonmaniam Sundaranar University, Tirunelveli.

Dr.D.Latha M.Sc., M.Phil., Ph.D., **Prof.D.H.KITTY SMAILIN M.Sc.,M.Phil**

Head of the Department ,
Department of PG computer science.

Project supervisor,
Department of PG computer science

DECLARATION

I hereby declare that the project work entitled "**TRACK NIGHT'S: A SLEEP TRACKING APP FOR A BETTER NIGHT'S REST"** is an original work done by me in partial fulfillment of the degree of **BACHELOR OF SCIENCE IN COMPUTER SCIENCE.** The study has been carried under the guidance of **Prof. D.H.KITTY SMAILIN M.Sc.,M.Phil Department of PG COMPUTER SCIENCE, Nesamony Memorial Christian College, Marthandam.** I declare that this work has not been submitted elsewhere for the award of any degree.

Place:Marthandam. Anto Stephin C S (202031115062117)

Altrin F (20203111506216)

Akash P S (20203111506215)

Ajin K K (20203111506214)

ACKNOWLEDGEMENT

First of all I offer my prayers to god almighty for blessing me to complete this internship programme successfully

I express my sincere thanks to **, Dr.K.Paul Raj M.Sc., M.Phil., M.Ed., M.Phil(Edu).,Ph.D.,** Principal Nesamony Memorial Christian college, for his official support to do my work.

I express my profound thanks to **Dr. Dr.D.Latha M.Sc., M.Phil., Ph.D.,** HOD, Department of PG Computer Science for his encouragement given to undertake this project.

I extend my deep sense of gratitude to **, Prof**. **D.H.KITTY SMAILIN**

M.Sc.,M.Phil Department of of PG Computer Science for guiding me in completing this project work successfully.

My special thanks to other Faculty members of Department of PG Computer Science for Guiding me in Completing this project.

I also wish to extend a special word of thanks to my parents, friends and all unseen hands that helped me for completing this project work successfully.

CONTENTS

1INTRODUCTION:

A sleep tracking app is a software application that can monitor and record your sleep patterns, allowing you to track your sleep quality and better understand your sleeping habits. A good sleep tracking app can provide valuable insights into your sleep behaviors, such as the amount of time you spend in different stages of sleep, the number of times you wake up during the night, and how long it takes you to fall asleep.The Better Nights app is a sleep tracking app that is designed to help users improve their sleep quality by providing them with

personalized insights and recommendations. The app uses advanced algorithms to analyze your sleep data and generate personalized sleep reports, which can help you identify factors that may be affecting your sleep and make adjustments to your sleep habits With the Better Nights app, you can set sleep goals, track your progress, and receive notifications to help you stick to your goals. The app also includes features such as relaxation exercises, sleep sounds, and guided meditations to help you wind down and fall asleep more easily.Overall, the Better Nights app can be a useful tool for anyone who wants to improve their sleep quality and gain a better understanding of their sleep habits. Whether you are struggling with insomnia or simply want to optimize your sleep for better health and well-being, the Better Nights app can help you achieve your goals.

Sleep tracking apps have become increasingly popular over the last few years, as more and more people are looking to improve the quality of their sleep. These apps use a variety of sensors and algorithms to track various aspects of sleep, including sleep duration, quality, and the amount of time spent in different stages of sleep.

Sleep tracking apps can be very useful for people who struggle with sleep, as they can provide valuable insights into their sleep patterns and help identify potential issues. For example, if someone is consistently waking up feeling tired, a sleep tracking app may be able to pinpoint specific times during the night when they are experiencing disrupted sleep.

One of the primary benefits of sleep tracking apps is that they allow users to gain a better understanding of their sleep patterns over time. By tracking sleep over the course of several nights or weeks, users can identify trends and patterns

in their sleep that may be contributing to poor sleep quality. For example, they may notice that they tend to sleep better on nights when they exercise or eat a certain type of food.

Another benefit of sleep tracking apps is that they can provide users with personalized recommendations for improving their sleep. For example, if the app detects that someone is not getting enough deep sleep, it may suggest changes to their bedtime routine or other lifestyle factors that can help promote deeper sleep.

ing apps come in many different forms, from simple apps that track sleep using the accelerometer in a smartphone to more advanced apps that use dedicated sensors or wearable devices to track sleep. Some of the most popular sleep tracking apps include Sleep Cycle, Pillow, and Fitbit.

Sleep Cycle is a popular sleep tracking app that uses the accelerometer in a smartphone to track sleep. The app works by placing the phone on the bed near the user's head, where it can detect movement and use this data to determine sleep cycles. Sleep Cycle also offers a variety of features to help users improve their sleep, including a smart alarm that wakes users up during their lightest sleep phase.

Pillow is another popular sleep tracking app that uses the accelerometer in a smartphone to track sleep. In addition to tracking sleep cycles, Pillow also tracks snoring and other sleep-related events, and provides users with detailed reports on their sleep quality. The app also offers a variety of tools and features to help users improve their sleep, including customized sleep goals and reminders to stick to a regular sleep schedule.

Fitbit is a well-known wearable device that includes a built-in

sleep tracking feature. The device uses a combination of sensors, including an accelerometer and a heart rate monitor, to track sleep quality and provide users with detailed reports on their sleep patterns. Fitbit also offers a variety of features to help users improve their sleep, including guided breathing exercises and personalized sleep recommendations based on their sleep data.While sleep tracking apps can be very useful for improving sleep quality, it is important to remember that they are not a substitute for professional

- **Overview:**

Technical Overview: Discuss the technical requirements for developing a real-time chat app using Android Studio.Explain the key technologies and tools involved, such as Android SDK, Firebase or other backend services for real-time messaging and user authentication, and XML for UI design.Provide an overview of the architecture of a real-time chat app, including the client-side and server-side components, and the flow of data and messages between them.Discuss the challenges and considerations in developing a real-time chat app, such as handling real-time updates, managing user authentication and authorization, handling network connectivity, and ensuring data security.

Android Studio is the official Integrated Development Environment (IDE) for developing Android applications. It is a powerful tool that allows developers to build high-quality applications for the Android platform. In this article, we will guide you through the process of setting up Android Studio on your computer. The steps on how to use Android Studio.

Step 1: Download Android Studio:

To set up Android Studio, you need to first download the IDE from

the official Android Studio download page. Choose the version that is compatible with your operating system, and download the installer. Android Studio is available for Windows, macOS, and Linux. Once the download is complete, run the installer and follow the instructions to install Android Studio on your computer.

Step 2: Install the Required Components:

During the installation process, Android Studio will prompt you to install the required components. These include the Android SDK, Android Virtual Device (AVD) Manager, and the Android Emulator. The Android SDK is a collection of libraries and tools that developers use to build Android applications. The AVD Manager is used to create and manage virtual devices for testing applications. The Android Emulator is a virtual device that allows developers to test their applications without having to use a physical device.

Step 3: Configure Android Studio:

After installing Android Studio, you need to configure it before you can start using it. When you launch Android Studio for the first time, you will be prompted to configure the IDE. Choose the "Standard" configuration and click on "Next". In the next screen, you can choose the theme of the IDE and click on "Next" again. You can also customize the settings based on your preferences.

Step 4: Create a New Project:

Once Android Studio is configured, you can start creating your first Android application. To create a new project, click on "Start a new Android Studio project" on the welcome screen, or select "New Project" from the "File" menu. You will be prompted to choose the project name, package name, and other project details. You can also choose the minimum SDK version, which determines the minimum version of Android that the application can run on.

Step 5: Build Your Application:

Once your project is created, you can start building your application using the various tools and features provided by Android Studio. You can use the visual layout editor to design the user interface, write code in Java or

Kotlin, and use the Android SDK to access device features such as the camera, sensors, and GPS. You can also use the built-in debugging tools to troubleshoot issues and optimize your application.

Step 6: Test Your Application:

Testing your application is an important step in the development process. Android Studio comes with an emulator that allows you to test your application on different virtual devices. You can also connect your Android device to your computer and test your application directly on the device. Use the "Run" button in Android Studio to launch your application and test it on the emulator or device. You can also use the built-in profiler to analyze the performance of your application and identify any bottlenecks or performance issues. In conclusion, setting up Android Studio is a crucial step in developing Android applications. By following these steps, you can easily set up Android Studio on your computer and start building high-quality Android applications. Android Studio provides a powerful set of tools and features that make the development process easier and more efficient.

Components in Android Studio:

1. Manifest File:

The AndroidManifest.xml file is a crucial component of any Android application. It provides essential information about the application to the Android operating system, including the application's package name, version, permissions, activities, services, and receivers. The manifest file is required for the Android system to launch the application and to determine its functionality. Here are some of the key uses of the manifest file in an Android application:

Declaring Application Components: The manifest file is used to declare the various components of an Android application, such as activities, services, and broadcast receivers. These components define the behavior and functionality of the application, and the Android system uses the manifest file to identify and launch them.

Specifying Permissions: Android applications require specific permissions to access certain features of the device, such as the camera, GPS, or storage. The manifest file is used to declare these permissions, which the Android system then checks when the application is installed. If

the user has not been granted the required permissions, the application may not be able to function correctly.

Defining App Configuration Details: The manifest file can also be used to define various configuration details of the application, such as the application's name, icon, version code and name, and supported screens. These details help the Android system to identify and manage the application properly.

Declaring App-level Restrictions: The manifest file can be used to declare certain restrictions at the app level, such as preventing the application from being installed on certain devices or specifying the orientation of the app on different screens.

In summary, the manifest file is an essential part of any Android application. It provides important information about the application to the Android system and enables the system to launch and manage the application correctly. Without a properly configured manifest file, an Android application may not be able to function correctly, or it may not be installed at all.

2. Build.gradle:

Gradle:

build.gradle is a configuration file used in Android Studio to define the build settings for an Android project. It is written in the Groovy programming language and is used to configure the build process for the project. Here are some of the key uses of the build.gradle file:

Defining Dependencies: One of the most important uses of the build.gradle file is to define dependencies for the project. Dependencies are external libraries or modules that are required by the project to function properly. The build.gradle file is used to specify which dependencies the project requires, and it will automatically download and include those dependencies in the project when it is built.

Setting Build Options: The build.gradle file can also be used to

configure various build options for the project, such as the version of the Android SDK to use, the target version of Android, and the signing configuration for the project.

Configuring Product Flavors: The build.gradle file can be used to configure product flavors for the project. Product flavors allow developers to create different versions of their application with different features or configurations. The build.gradle file is used to specify which product flavors should be built, and how they should be configured.

Customizing the Build Process: The build.gradle file can also be used to customize the build process for the project. Developers can use the build.gradle file to specify custom build tasks, define build types, or customize the build process in other ways.

Overall, the build.gradle file is a powerful tool for configuring the build process for an Android project. It allows developers to define dependencies, configure build options, customize the build process, and more. By understanding how to use the build.gradle file, developers can optimize the build process for their projects and ensure that their applications are built correctly and efficiently.

3. Git:

Git is a popular version control system that allows developers to track changes to their code and collaborate with other team members. Android Studio includes built-in support for Git, making it easy to manage code changes and collaborate with others on a project. Here are some of the key uses of Git in Android Studio:

Version Control: Git allows developers to track changes to their code over time. This means that they can easily roll back to a previous version of their code if needed, or review the changes made by other team members.

Collaboration: Git enables multiple developers to work on the same codebase simultaneously. Developers can work on different features or parts of the codebase without interfering with each other, and merge their changes together when they are ready.

Branching and Merging: Git allows developers to create branches of their codebase, which can be used to work on new features or bug fixes without

affecting the main codebase. When the changes are complete, the branch can be merged back into the main codebase.

Code Review: Git allows team members to review each other's code changes before they are merged into the main codebase. This can help ensure that the code is of high quality and meets the project's requirements.

Android Studio includes a built-in Git tool that allows developers to perform common Git tasks directly within the IDE. Developers can create new repositories, clone existing ones, and manage branches and commits. Android Studio also provides a visual diff tool that makes it easy to see the changes made to the codebase over time. To use Git in Android Studio, developers need to first initialize a Git repository for their project. Once the repository is set up, they can use the Git tool in Android Studio to manage changes to their code, collaborate with others, and review code changes.

In summary, Git is a powerful version control system that is essential for managing code changes and collaborating with other team members. Android Studio includes built-in support for Git, making it easy for developers to manage their code changes directly within the IDE.

4. Debug:

Debugging is an essential part of software development, and Android Studio provides a robust set of debugging tools to help developers identify and fix issues in their applications. Here are some of the key uses of debugging in Android Studio.

Identifying Issues: Debugging helps developers identify issues in their code by allowing them to inspect variables, evaluate expressions, and step through the code line by line. This allows developers to pinpoint exactly where a problem is occurring and fix it more quickly.

Optimizing Performance: Debugging can also be used to optimize the performance of an application by identifying bottlenecks or areas of inefficient code. By profiling an application while it is running, developers can identify areas of the code that are causing slow performance and make changes to improve performance.

Testing and Validation: Debugging is also useful for testing and validating

an application. By stepping through code and inspecting variables, developers can ensure that the application is behaving as expected and that it is producing the desired output.

Android Studio provides a comprehensive set of debugging tools, including breakpoints, watches, and the ability to evaluate expressions in real time. Developers can use these tools to inspect variables, step through code, and identify issues in their applications.

To use the debugging tools in Android Studio, developers need to first configure their project for debugging by adding breakpoints to their code. Breakpoints are markers that tell the debugger to pause execution at a certain point in the code. Once the breakpoints are set, developers can run their application in debug mode and step through the code line by line, inspecting variables and evaluating expressions as they go.

In summary, debugging is a critical part of software development, and Android Studio provides a robust set of debugging tools to help developers identify and fix issues in their applications. By using these tools, developers can optimize performance, test and validate their code, and improve the quality of their applications.

5. App Inspection:

Inspector:

App Inspection is a feature in Android Studio that allows developers to inspect and debug their Android applications. It provides a suite of tools for analyzing the performance of the application, identifying and fixing errors, and optimizing the code. Here are some of the key features and uses of App Inspection:

Performance Analysis: App Inspection provides tools for analyzing the performance of an Android application. Developers can use these tools to identify performance bottlenecks, such as slow database queries or inefficient network requests, and optimize the code to improve performance.

Error Detection and Debugging: App Inspection allows developers to detect and debug errors in their Android applications. It provides tools for tracking down errors and identifying the root cause of the issue, making it easier to fix bugs and improve the stability of the application.

Memory Management: App Inspection provides tools for managing the memory usage of an Android application. Developers can use these tools to identify memory leaks and optimize the code to reduce memory usage, which can improve the performance and stability of the application.

Network Profiling: App Inspection includes tools for profiling network traffic in an Android application. Developers can use these tools to monitor network requests, identify slow or inefficient requests, and optimize the code to improve network performance.

Overall, App Inspection is a valuable tool for Android developers. It provides a suite of tools for analyzing and debugging Android applications, identifying and fixing errors, and optimizing the code for improved performance and stability. By using App Inspection, developers can ensure that their Android applications are of the highest quality and provide the best possible user experience.

6. Build Variants:

Build variants in Android Studio are different versions of an Android app that can be built from the same source code. They are typically used to create multiple versions of an app that target different device configurations or use cases. Build variants are configured in the build.gradle file and can be built and installed separately from each other. Here are some examples of how build variants can be used.

Debug and Release Variants: The most common use of build variants is to create a debug variant and a release variant of an app. The debug variant is used for testing and debugging the app during development, while the release variant is used for production and is optimized for performance and stability.

Flavors: Build variants can also be used to create different flavors of an app, which can have different features or configurations. For example, an app might have a free version and a paid version, or a version that targets tablets and a version that targets phones.

Build Types: Build variants can also be used to create different build types, which can have different build options or signing configurations. For example, an app might have a debug build type and a release build type, each with its own set of build options.

Overall, build variants are a powerful tool for Android developers. They allow developers to create different versions of an app from the same source code, which can save time and improve the quality of the app. By using build variants, developers can easily target different device configurations or use cases, create different versions of the app with different features or configurations, and optimize the app for performance and stability.

1.2 Purpose:

Sleep tracking is the practice of monitoring and recording one's sleep patterns to gain insight into the quality and duration of their sleep. Sleep tracking has become increasingly popular in recent years, with the rise of wearable devices and smartphone apps that can monitor sleep metrics such as the amount of time spent in each stage of sleep, the number of times one wakes up during the night, and the overall duration of sleep.The purpose of sleep tracking is to help individuals understand their sleep patterns and identify any potential issues that may be affecting their sleep quality. By monitoring their sleep, individuals can gain insight into how various factors such as diet, exercise, stress levels, and environmental factors may be impacting their sleep.Sleep tracking can be particularly useful for individuals who have trouble sleeping or suffer from sleep disorders such as insomnia, sleep apnea, or restless leg syndrome. By tracking their sleep patterns, individuals can work with their healthcare provider to identify potential triggers for their sleep issues and develop strategies to improve their sleep quality.

In addition to identifying potential sleep issues, sleep tracking can also help individuals optimize their sleep habits. For example, by monitoring their sleep patterns, individuals can determine the optimal amount of sleep they need each night and adjust their sleep schedule accordingly. They can also identify habits that may be interfering with their sleep, such as consuming caffeine too late in the day or using electronic devices before bed, and make changes

to improve their sleep quality.Overall, sleep tracking can be a useful tool for individuals looking to improve their sleep quality and overall health. By gaining insight into their sleep patterns and making adjustments to their sleep habits, individuals can improve their mental and physical well-being, and enjoy the benefits of a good night's sleep.

A wide variety of sleep trackers have hit the market, with more being released all the time. Many are wearable trackers that you can strap to your wrist. Others clip on your pillow or sit on your bedside table.

Features of these devices vary, but some common capabilities include:

- Sleep duration: By tracking the time you're inactive, the devices can record when you fall asleep at night and when you stir in the morning.
- Sleep quality: Trackers can detect interrupted sleep, letting you know when you're tossing and turning or waking during the night.
- Sleep phases: Some tracking systems track the phases of your sleep and time your alarm to go off during a period when you're sleeping less deeply. In theory, that makes it easier for you to rouse.
- Environmental factors: Some devices record environmental factors like the amount of light or temperature in your bedroom.
- Lifestyle factors: Some trackers prompt you to enter information about activities that can affect sleep, such as how much caffeine you've had, when you've eaten or whether your

stress level is high.

2PROBLEM DEFINITION & DESIGN THINKING:

Problem Definition is the process of clearly defining and understanding a problem that needs to be solved. It involves identifying the root cause of the problem, its impact, and its scope. well understood before any solution is proposed. Design Thinking is a problem-solving approach that focuses on understanding the needs and perspectives of the user, and using that understanding to generate creative and effective solutions. It involves a series of iterative steps, including empathizing with the user, defining the problem, ideating potential solutions, prototyping, and testing. Design Thinking can be used to help with problem definition by encouraging a deep understanding of the problem and its impact on the user. By empathizing with the user and understanding their needs and perspective, designers can gain a better understanding of the problem and its scope. This understanding can then be used to generate more effective solutions that are more likely to meet the needs of the user.Overall, problem definition and Design Thinking are both important processes in the development of effective solutions. Problem definition ensures that the problem is well understood, while Design Thinking helps to generate creative and effective solutions that meet the needs of the user.

2.1 Empathy Map:

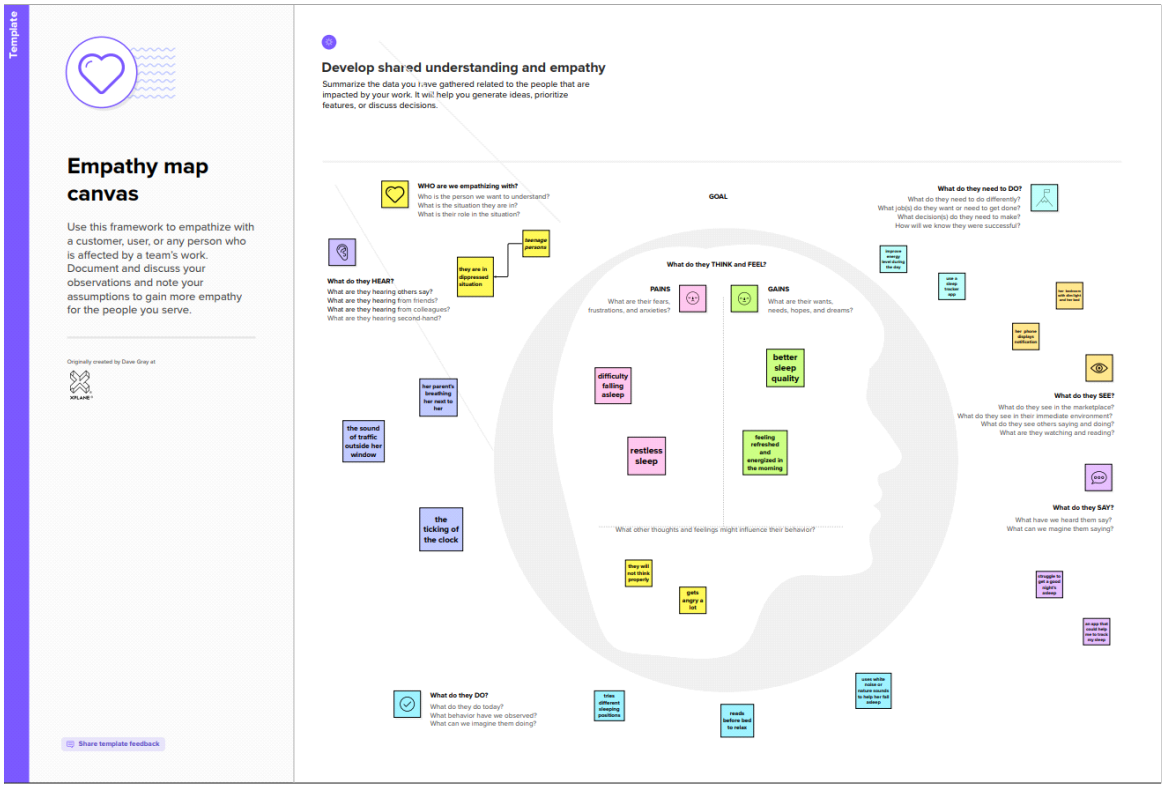
An empathy map is a tool that helps you gain a deeper understanding of your users by putting yourself in their shoes. It is a visual representation of what a user thinks, feels, sees, hears, and does in relation to a specific problem or situation. Let's say you are designing a chat app. To create an empathy map for this app, you would start by identifying your target users. For example, your target users could be young adults between the ages of 18-24 who use chat apps frequently to communicate with friends and family.

- Thinks: What are the user's thoughts and concerns when using a chat app? For example, they may worry about privacy and security or may be concerned about the app being too complicated to use.
- Feels: What are the user's emotions when using a chat app? For example, they may feel frustrated when the app crashes or may feel

happy when they receive a message from a friend.

- Sees: What does the user see when using a chat app? For example, they may see a list of contacts or a chat window with a friend.
- Hears: What does the user hear when using a chat app? For example, they may hear a notification sound when they receive a message or may hear the sound of typing when their friend is responding.
- Does: What does the user do when using a chat app? For example, they may send messages, create group chats, or share photos and videos.

Screenshot of Empathy Map:



2.2 Ideation & Brainstorming Map:

Ideation is the process of generating new ideas and solutions to a problem. It is an essential part of the design thinking process and involves a range of techniques and methods to help teams generate a large number of ideas quickly and effectively.

Brainstorming is a commonly used ideation technique that involves a group of people coming together to generate as many ideas as possible in a short amount of time. The goal of brainstorming is to generate a wide variety of ideas without judgment or evaluation. The focus is on quantity over quality. To conduct a successful brainstorming session, there are a few key principles to follow:Encourage participation: Everyone in the group should be encouraged to contribute their ideas, no matter how small or seemingly insignificant.Defer judgment: Criticism or evaluation of ideas should be deferred until after the brainstorming session is complete.

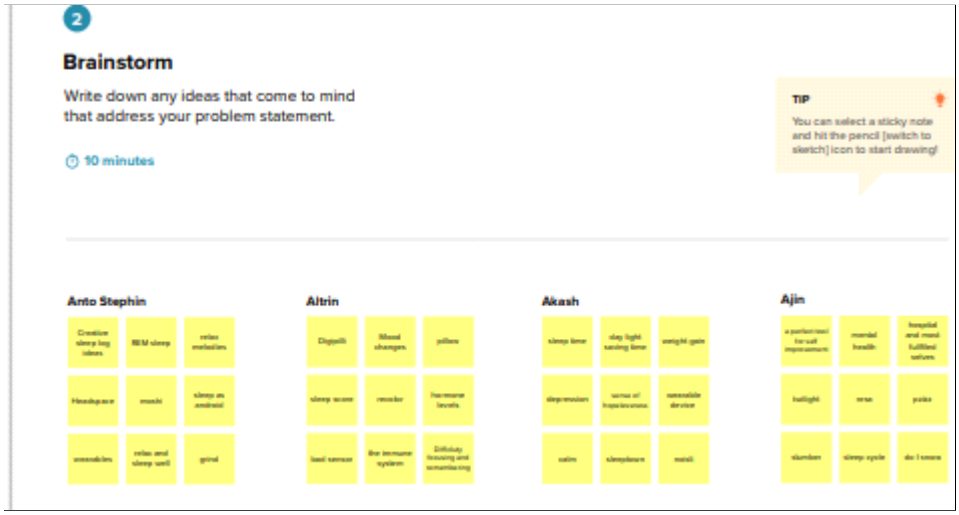
This creates a safe and supportive environment where participants feel comfortable sharing their ideas.Build on each other's ideas: Participants should be encouraged to build on the ideas of others, rather than simply coming up with new ideas.Stay focused on the topic:

The group should stay focused on the problem or challenge at hand, and all ideas should be related to the topic.Use visual aids: Using visual aids such as whiteboards, post-it notes, or mind maps can help to organize and stimulate the ideation process.

Once the brainstorming session is complete, the ideas generated can be evaluated and refined using other ideation techniques such as prioritization, clustering, or concept mapping. This helps to identify the most promising ideas and develop them further into viable solutions.

Overall, ideation and brainstorming are important processes in the design thinking approach, as they help to generate a wide range of ideas and solutions to a problem. By following the principles of brainstorming and using a variety of ideation techniques, teams can develop innovative and effective solutions to complex problems.

Screenshot of Ideation & Brainstorming Map:



3RESULT:

A data model for sleep tracking typically includes the following key entities:

User: A user represents an individual who uses the sleep tracking system. A user entity may include attributes such as name, age, gender, height, weight, and other personal information.

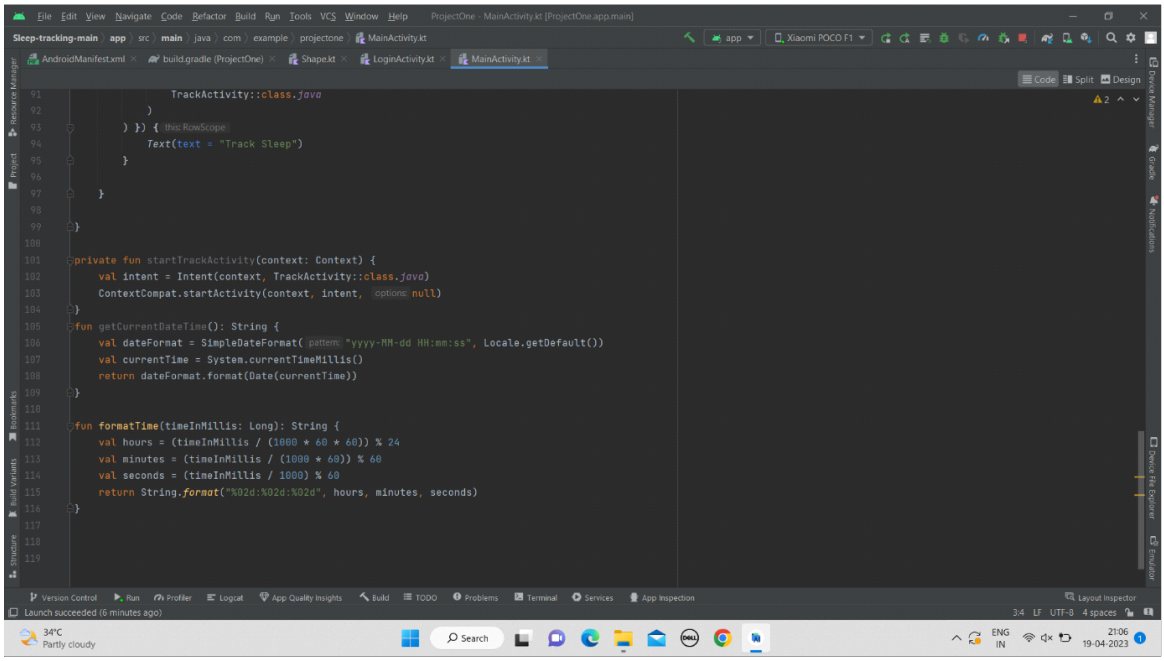
Sleep Session: A sleep session represents a specific period of time during which a user sleeps. A sleep session entity may include attributes such as start time, end time, duration, and quality of

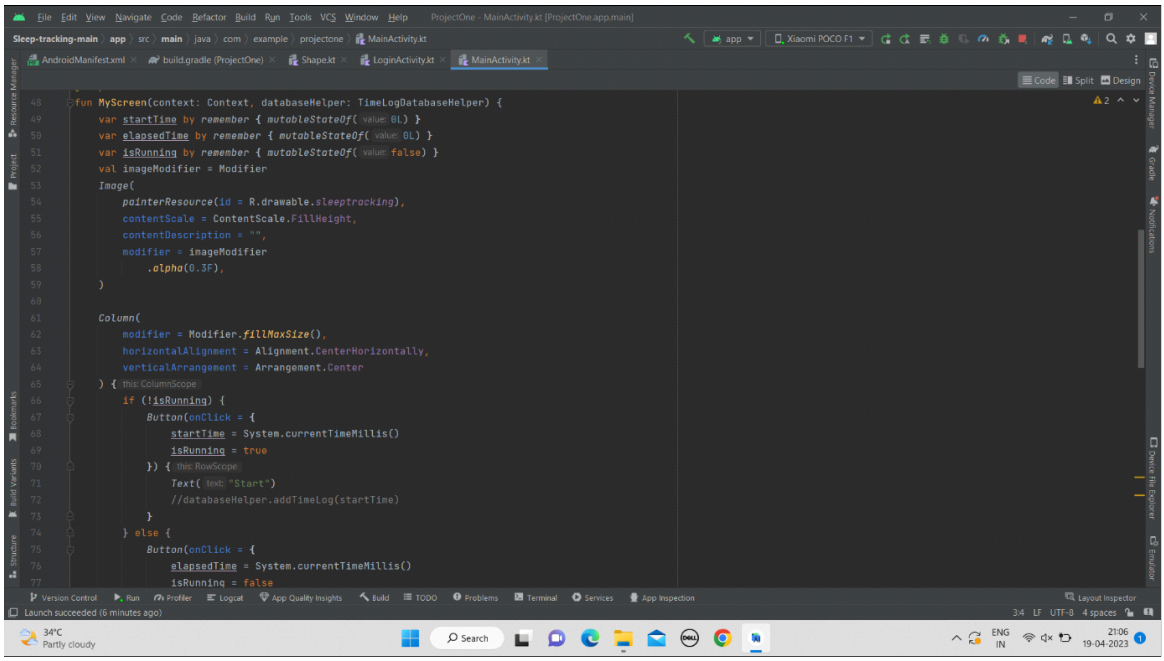
sleep.

Sleep Stage: A sleep stage represents the different stages of sleep that a user goes through during a sleep session. There are typically four stages of sleep: NREM 1, NREM 2, NREM 3, and REM. A sleep stage entity may include attributes such as start time, end time, and duration.

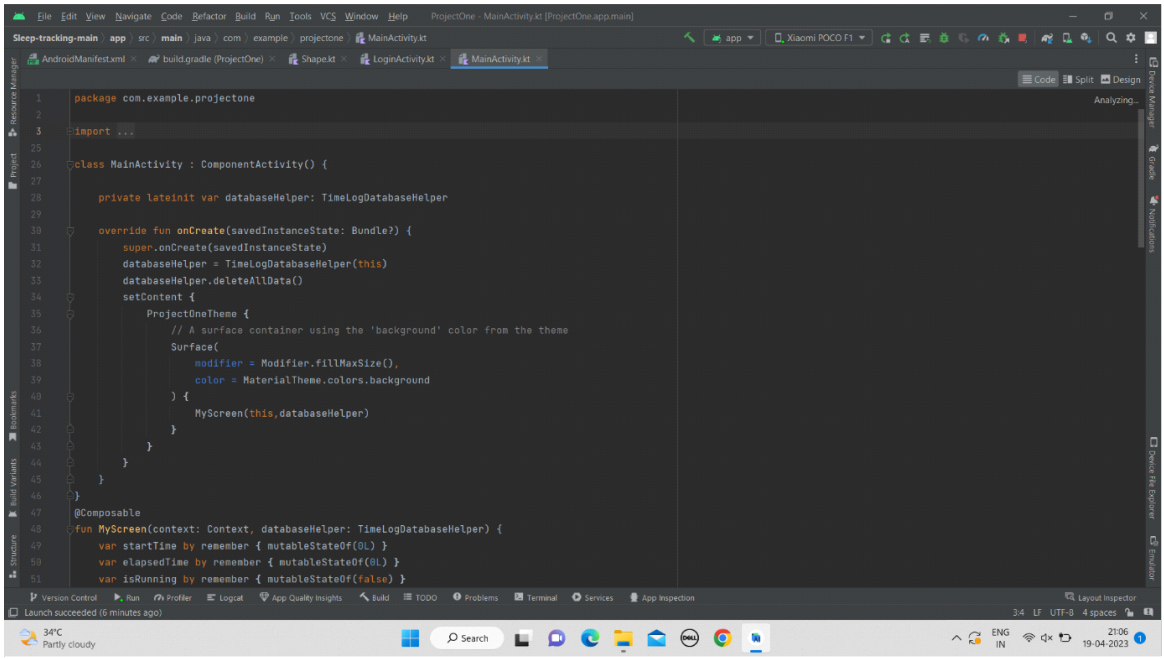
Sleep Metrics: Sleep metrics are measurements that are used to track a user's sleep quality and quantity. Sleep metrics can include things like heart rate, respiratory rate, movement, and snoring.

3.2 Activity & Screenshot

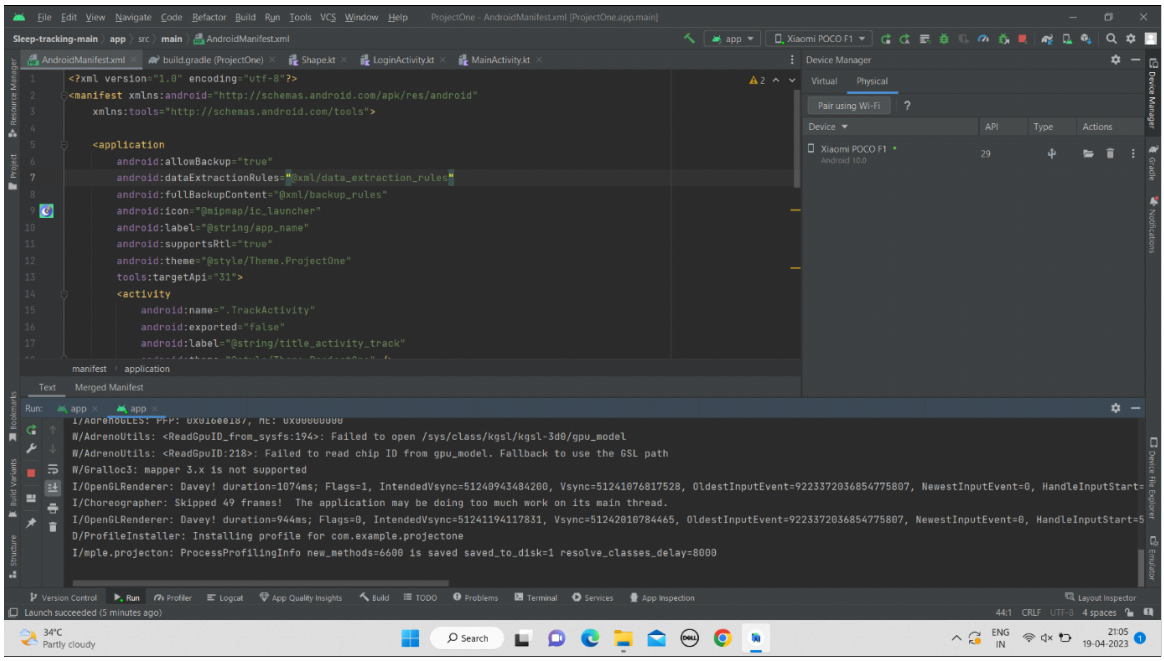


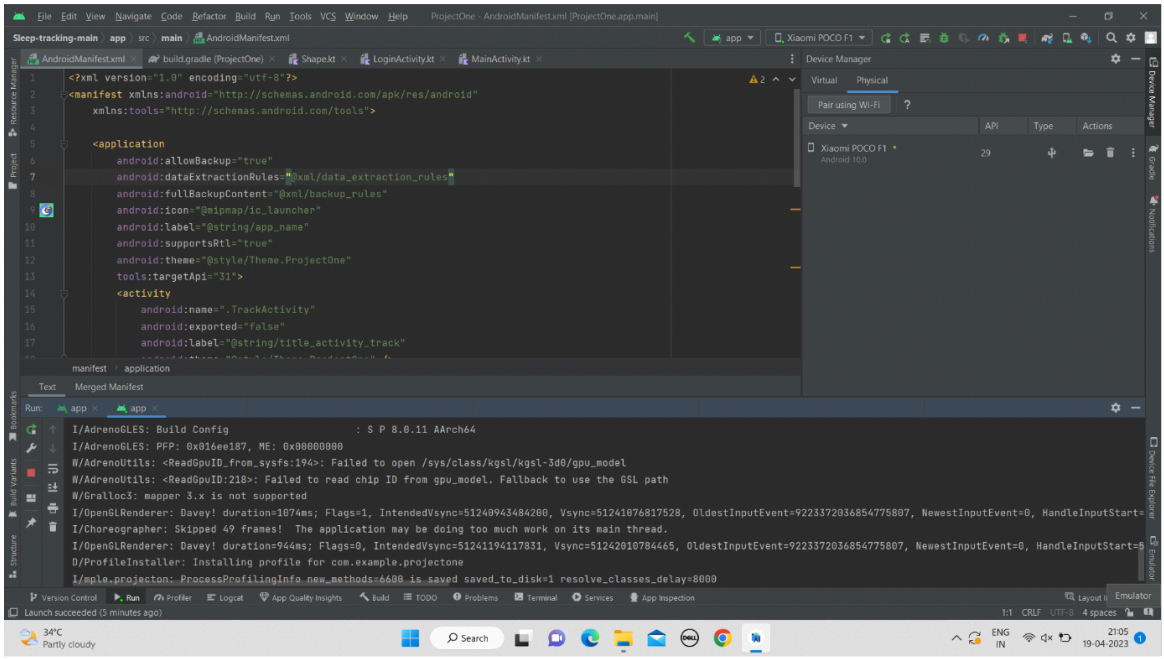


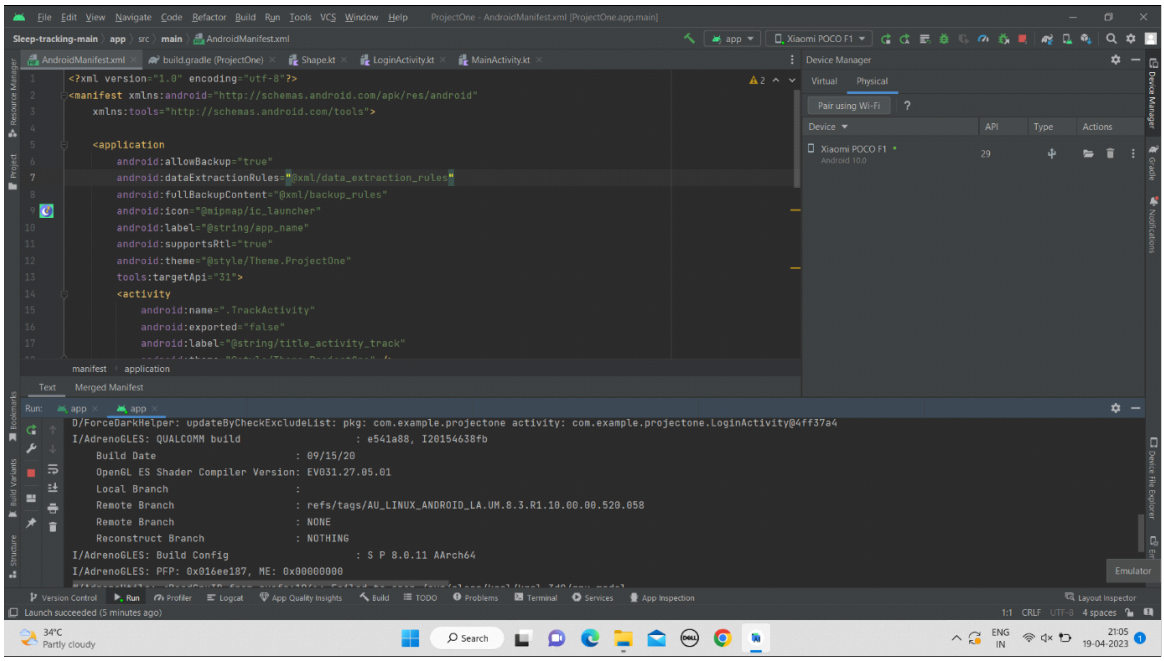
```
fun MyScreen(context: Context, databaseHelper: TimeLogDatabaseHelper) {
    var startTime by remember { mutableStateOf( value: 0L) }
    var elapsedTime by remember { mutableStateOf( value: 0L) }
    var isRunning by remember { mutableStateOf( value: false) }
    val imageModifier = Modifier
    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
    ) { this: ColumnScope
        if (!isRunning) {
            Button(onClick = {
                startTime = System.currentTimeMillis()
                isRunning = true
            }) { this: RowScope
                Text( text: "Start")
                //databaseHelper.addTimeLog(startTime)
            }
        } else {
            Button(onClick = {
                elapsedTime = System.currentTimeMillis()
                isRunning = false
```

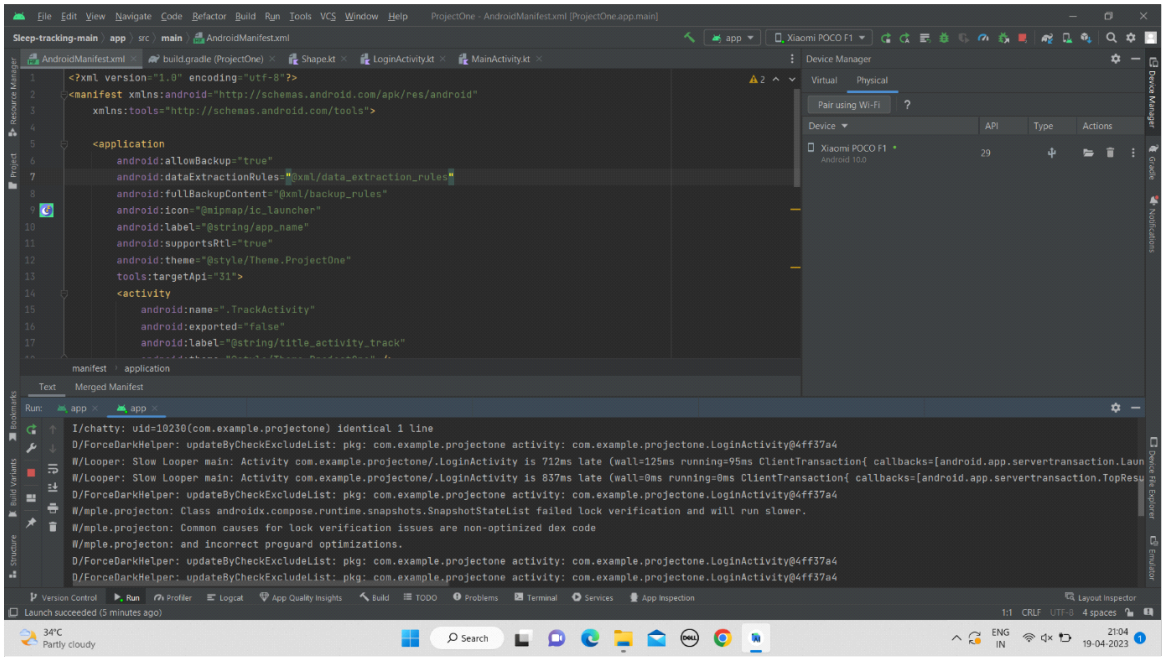


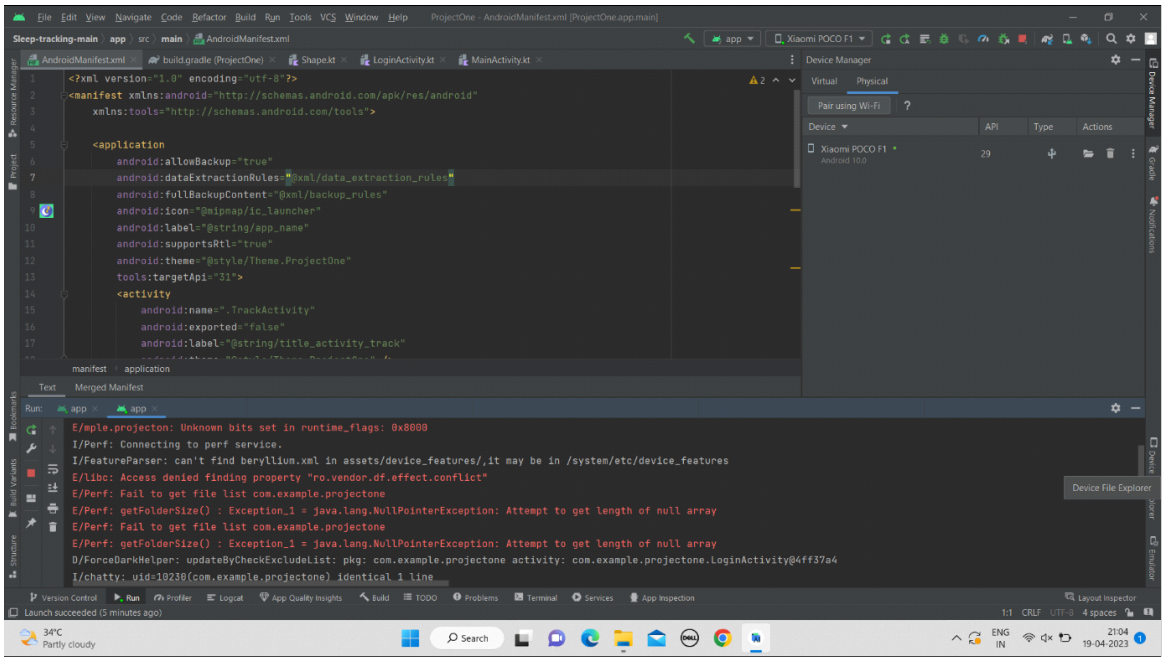
```
package com.example.projectone

import ...

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) }
```

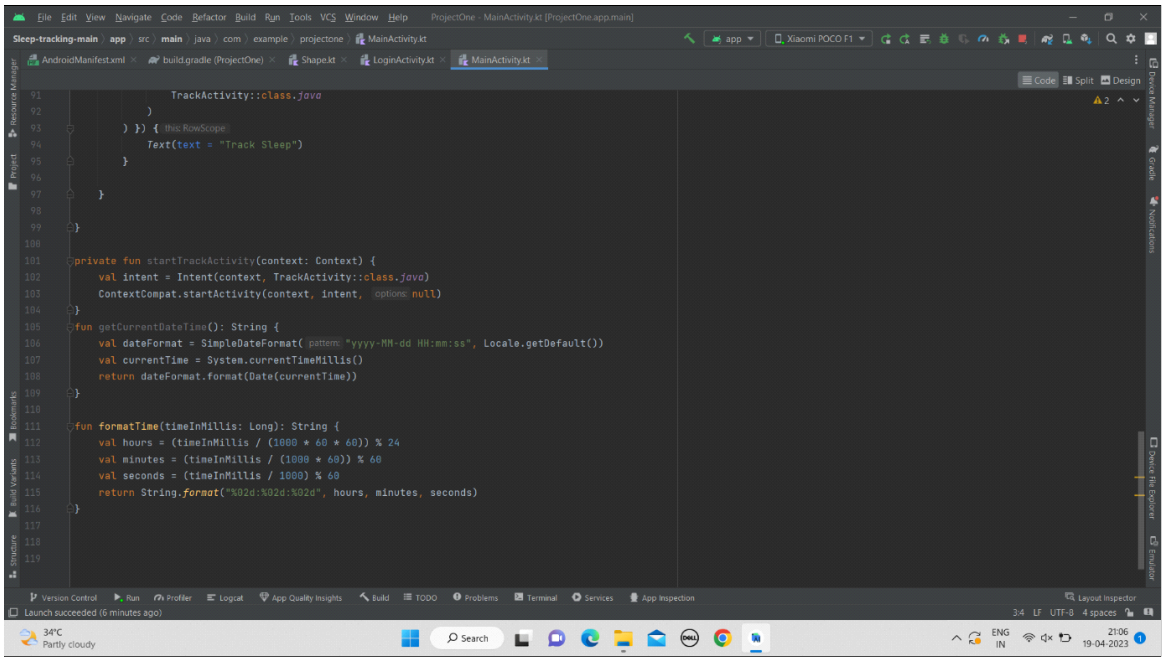

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - AndroidManifest.xml [ProjectOne.app.main]
Sleep-tracking-main app src main AndroidManifest.xml
app
Xiaomi POCO F1
AndroidManifest.xml
build.gradle (ProjectOne)
Shape.kt
LoginActivity.kt
MainActivity.kt
<?xml version="1.0" 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"
manifest application
Text
Merged Manifest
Device Manager
Virtual
Physical
Pair using Wi-Fi
Device
API
Type
Actions
Xiaomi POCO F1
Android 10.0
29
Run:
app
app
W/AdrenoUtils: <ReadGpuID_from_sysfs:194>: Failed to open /sys/class/kgsl/kgsl-3d0/gpu_model
W/AdrenoUtils: <ReadGpuID:218>: Failed to read chip ID from gpu_model. Fallback to use the GSL path
W/Gralloc3: mapper 3.x is not supported
I/OpenGLRenderer: Davey! duration=1074ms; Flags=1, IntendedVsync=51240943484200, Vsync=51241076817528, OldestInputEvent=9223372036854775807, NewestInputEvent=0, HandleInputStart=
I/Choreographer: Skipped 49 frames! The application may be doing too much work on its main thread.
I/OpenGLRenderer: Davey! duration=944ms; Flags=0, IntendedVsync=51241194117831, Vsync=51242010784465, OldestInputEvent=9223372036854775807, NewestInputEvent=0, HandleInputStart=5
D/ProfileInstaller: Installing profile for com.example.projectone
I/mple.projecton: ProcessProfilingInfo new_methods=6600 is saved saved_to_disk=1 resolve_classes_delay=8000
Version Control
Run
Profiler
Logcat
App Quality Insights
Build
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Launch succeeded (5 minutes ago)
44:1
CRLF
UTF-8
4 spaces
34°C
Partly cloudy
Search
ENG
IN
21:05
19-04-2023


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - AndroidManifest.xml [ProjectOne.app.main]
Sleep-tracking-main app src main AndroidManifest.xml
app
Xiaomi POCO F1
AndroidManifest.xml
build.gradle (ProjectOne)
Shape.kt
LoginActivity.kt
MainActivity.kt
<?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"
manifest application
Text
Merged Manifest
Device Manager
Virtual
Physical
Pair using Wi-Fi
Device
API
Type
Actions
Xiaomi POCO F1
Android 10.0
29
Run:
app
app
I/AdrenoGLES: Build Config : S P 8.0.11 AArch64
I/AdrenoGLES: PFP: 0x016ee187, ME: 0x00000000
W/AdrenoUtils: <ReadGpuID_from_sysfs:194>: Failed to open /sys/class/kgsl/kgsl-3d0/gpu_model
W/AdrenoUtils: <ReadGpuID:218>: Failed to read chip ID from gpu_model. Fallback to use the GSL path
W/Gralloc3: mapper 3.x is not supported
I/OpenGLRenderer: Davey! duration=1074ms; Flags=1, IntendedVsync=51240943484200, Vsync=51241076817528, OldestInputEvent=9223372036854775807, NewestInputEvent=0, HandleInputStart=
I/Choreographer: Skipped 49 frames! The application may be doing too much work on its main thread.
I/OpenGLRenderer: Davey! duration=944ms; Flags=0, IntendedVsync=51241194117831, Vsync=51242010784465, OldestInputEvent=9223372036854775807, NewestInputEvent=0, HandleInputStart=5
D/ProfileInstaller: Installing profile for com.example.projectone
I/mple.projecton: ProcessProfilingInfo new_methods=6600 is saved saved_to_disk=1 resolve_classes_delay=8000
Version Control
Run
Profiler
Logcat
App Quality Insights
Build
TODO
Problems
Terminal
Services
App Inspection
Emulator
Launch succeeded (5 minutes ago)
1:1
CRLF
UTF-8
4 spaces
34°C
Partly cloudy
Search
ENG
IN
21:05
19-04-2023

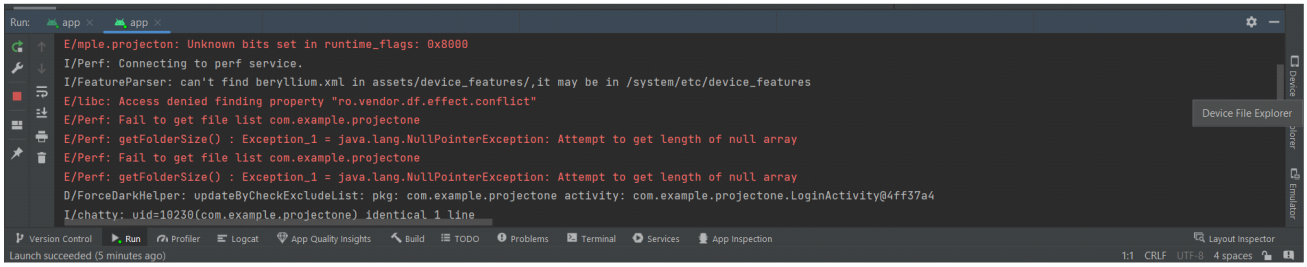

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - AndroidManifest.xml [ProjectOne.app.main]
Sleep-tracking-main app src main AndroidManifest.xml
app
Xiaomi POCO F1
AndroidManifest.xml
build.gradle (ProjectOne)
Shape.kt
LoginActivity.kt
MainActivity.kt
Device Manager
Virtual
Physical
Pair using Wi-Fi
Device
API
Type
Actions
Xiaomi POCO F1
Android 10.0
29
<?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"
manifest application
Text
Merged Manifest
Run:
app
app
D/ForceDarkHelper: updateByCheckExcludeList: pkg: com.example.projectone activity: com.example.projectone.LoginActivity@4ff37a4
I/AdrenoGLES: QUALCOMM build : e541a88, I20154638fb
Build Date : 09/15/20
OpenGL ES Shader Compiler Version: EV031.27.05.01
Local Branch :
Remote Branch : refs/tags/AU_LINUX_ANDROID_LA.UM.8.3.R1.10.00.00.520.058
Remote Branch : NONE
Reconstruct Branch : NOTHING
I/AdrenoGLES: Build Config : S P 8.0.11 AArch64
I/AdrenoGLES: PFP: 0x016ee187, ME: 0x00000000
Emulator
Version Control
Run
Profiler
Logcat
App Quality Insights
Build
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Launch succeeded (5 minutes ago)
1:1 CRLF UTF-8 4 spaces
34°C
Partly cloudy
Search
ENG
IN
21:05
19-04-2023


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - AndroidManifest.xml [ProjectOne.app.main]
Sleep-tracking-main app src main AndroidManifest.xml
app
Xiaomi POCO F1
AndroidManifest.xml
build.gradle (ProjectOne)
Shape.kt
LoginActivity.kt
MainActivity.kt
Device Manager
Virtual
Physical
Pair using Wi-Fi
Device
API
Type
Actions
Xiaomi POCO F1
Android 10.0
29
<?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"
manifest application
Text
Merged Manifest
Run:
app
app
I/chatty: uid=10230(com.example.projectone) identical 1 line
D/ForceDarkHelper: updateByCheckExcludeList: pkg: com.example.projectone activity: com.example.projectone.LoginActivity@4ff37a4
W/Looper: Slow Looper main: Activity com.example.projectone/.LoginActivity is 712ms late (wall=125ms running=95ms ClientTransaction{ callbacks=[android.app.servertransaction.Laun
W/Looper: Slow Looper main: Activity com.example.projectone/.LoginActivity is 837ms late (wall=0ms running=0ms ClientTransaction{ callbacks=[android.app.servertransaction.TopResu
D/ForceDarkHelper: updateByCheckExcludeList: pkg: com.example.projectone activity: com.example.projectone.LoginActivity@4ff37a4
W/mple.projecton: Class androidx.compose.runtime.snapshots.SnapshotStateList failed lock verification and will run slower.
W/mple.projecton: Common causes for lock verification issues are non-optimized dex code
W/mple.projecton: and incorrect proguard optimizations.
D/ForceDarkHelper: updateByCheckExcludeList: pkg: com.example.projectone activity: com.example.projectone.LoginActivity@4ff37a4
D/ForceDarkHelper: updateByCheckExcludeList: pkg: com.example.projectone activity: com.example.projectone.LoginActivity@4ff37a4
Emulator
Version Control
Run
Profiler
Logcat
App Quality Insights
Build
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Launch succeeded (5 minutes ago)
1:1 CRLF UTF-8 4 spaces
34°C
Partly cloudy
Search
ENG
IN
21:04
19-04-2023

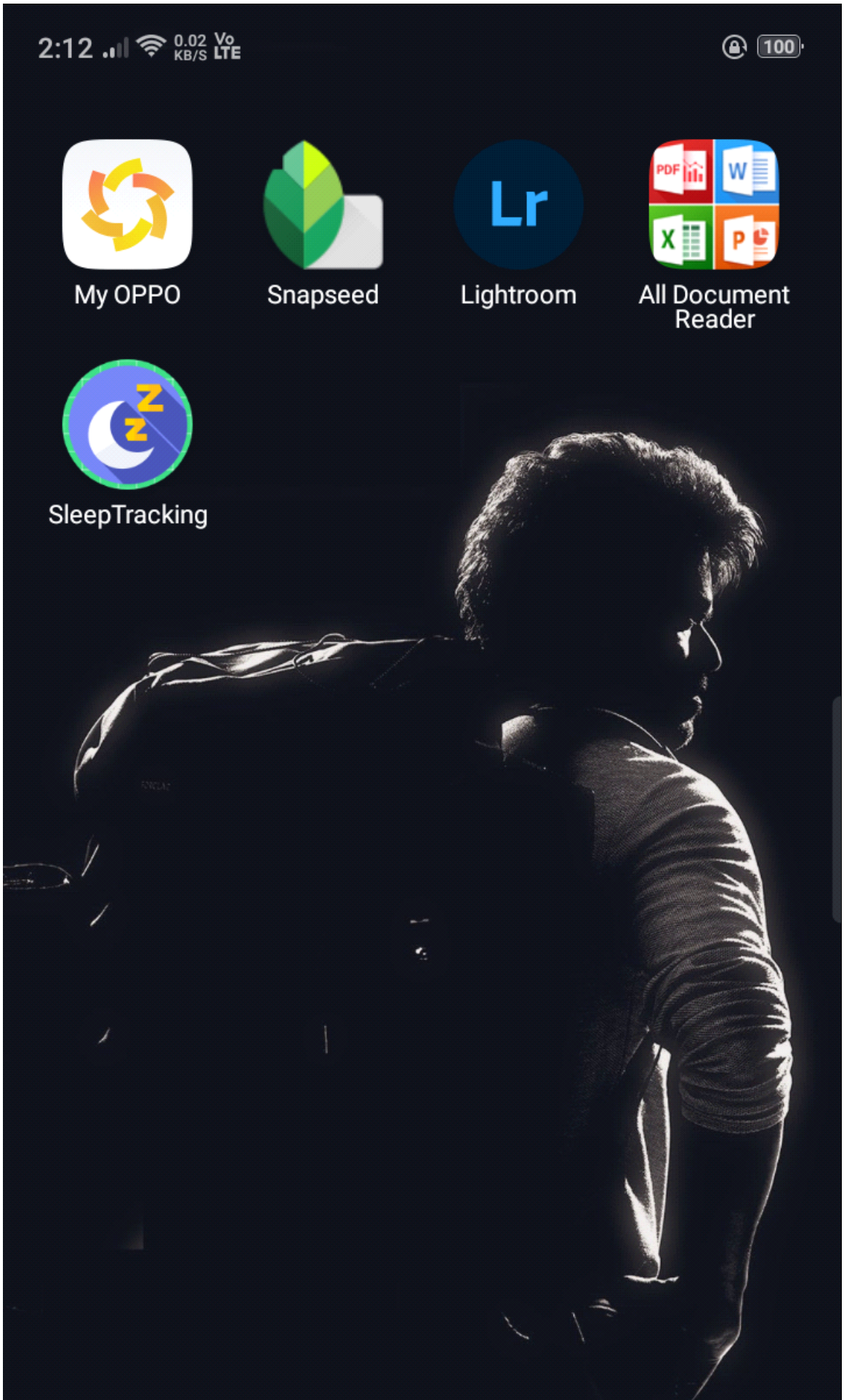

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - AndroidManifest.xml [ProjectOne.app.main]
Sleep-tracking-main app src main AndroidManifest.xml
app
Xiaomi POCO F1
AndroidManifest.xml
build.gradle (ProjectOne)
Shape.kt
LoginActivity.kt
MainActivity.kt
<?xml version="1.0" 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"
manifest application
Text
Merged Manifest
Device Manager
Virtual
Physical
Pair using Wi-Fi
Device
API
Type
Actions
Xiaomi POCO F1
Android 10.0
29
Run:
app
app
E/mple.projecton: Unknown bits set in runtime_flags: 0x8000
I/Perf: Connecting to perf service.
I/FeatureParser: can't find beryllium.xml in assets/device_features/,it may be in /system/etc/device_features
E/libc: Access denied finding property "ro.vendor.df.effect.conflict"
E/Perf: Fail to get file list com.example.projectone
E/Perf: getFolderSize() : Exception_1 = java.lang.NullPointerException: Attempt to get length of null array
E/Perf: Fail to get file list com.example.projectone
E/Perf: getFolderSize() : Exception_1 = java.lang.NullPointerException: Attempt to get length of null array
D/ForceDarkHelper: updateByCheckExcludeList: pkg: com.example.projectone activity: com.example.projectone.LoginActivity@4ff37a4
I/chatty: uid=10230(com.example.projectone) identical 1 line
Device File Explorer
Version Control
Run
Profiler
Logcat
App Quality Insights
Build
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Launch succeeded (5 minutes ago)
1:1
CRLF
UTF-8
4 spaces
34°C
Partly cloudy
Search
ENG
IN
21:04
19-04-2023



```
                    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)
}
```



```
E/mple.projecton: Unknown bits set in runtime_flags: 0x8000
I/Perf: Connecting to perf service.
I/FeatureParser: can't find beryllium.xml in assets/device_features/,it may be in /system/etc/device_features
E/libc: Access denied finding property "ro.vendor.df.effect.conflict"
E/Perf: Fail to get file list com.example.projectone
E/Perf: getFolderSize() : Exception_1 = java.lang.NullPointerException: Attempt to get length of null array
E/Perf: Fail to get file list com.example.projectone
E/Perf: getFolderSize() : Exception_1 = java.lang.NullPointerException: Attempt to get length of null array
D/ForceDarkHelper: updateByCheckExcludeList: pkg: com.example.projectone activity: com.example.projectone.LoginActivity@4ff37a4
I/chatty: uid=10230(com.example.projectone) identical 1 line
```


2:12
0.02 KB/S
Vo LTE
100
My OPPO
Snapseed
Lightroom
All Document Reader
SleepTracking

2:11 0.02 KB/S Vo LTE 100

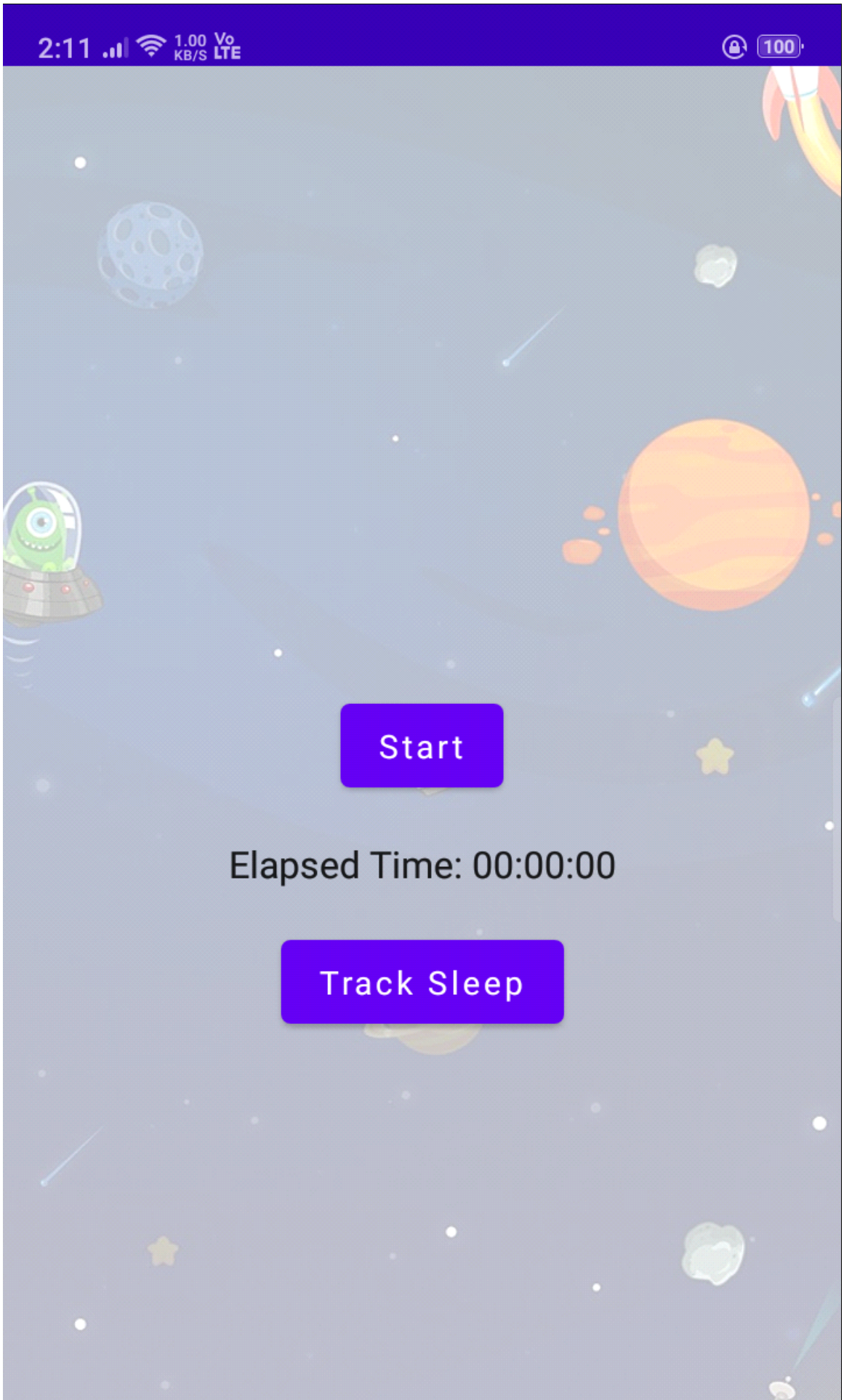
Register

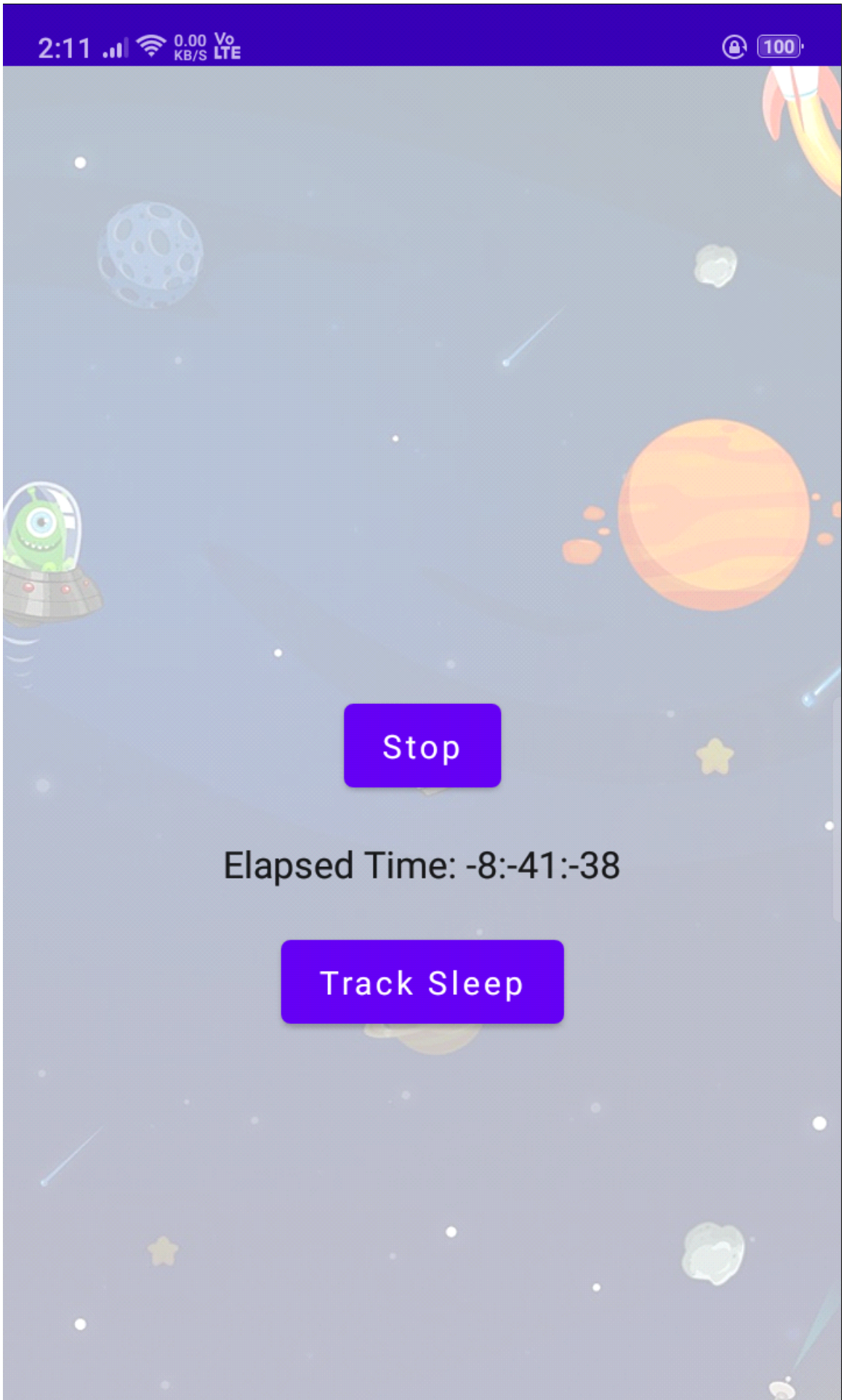
Username
Altrin

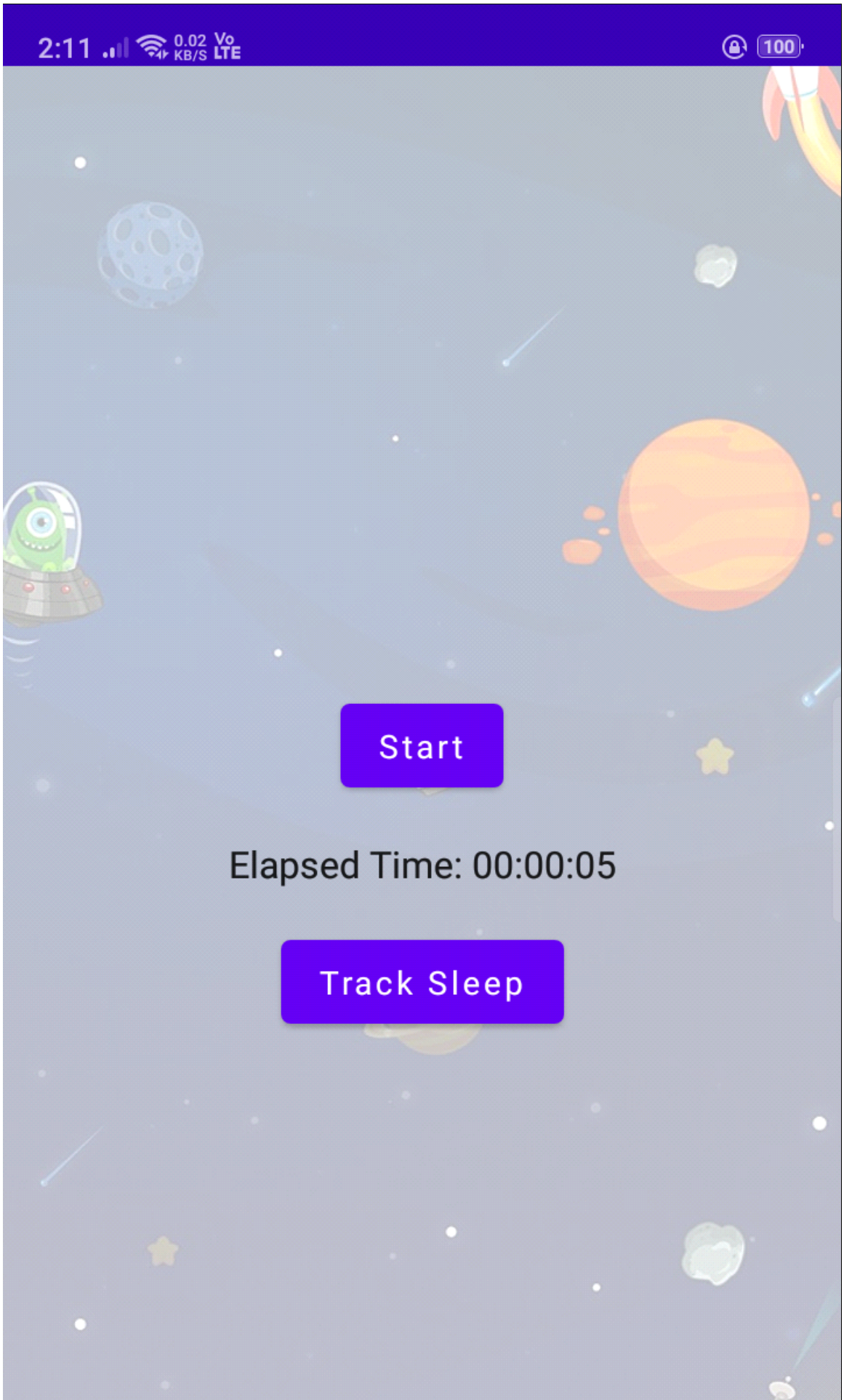
Email
altrinaltrin836@gmail.com

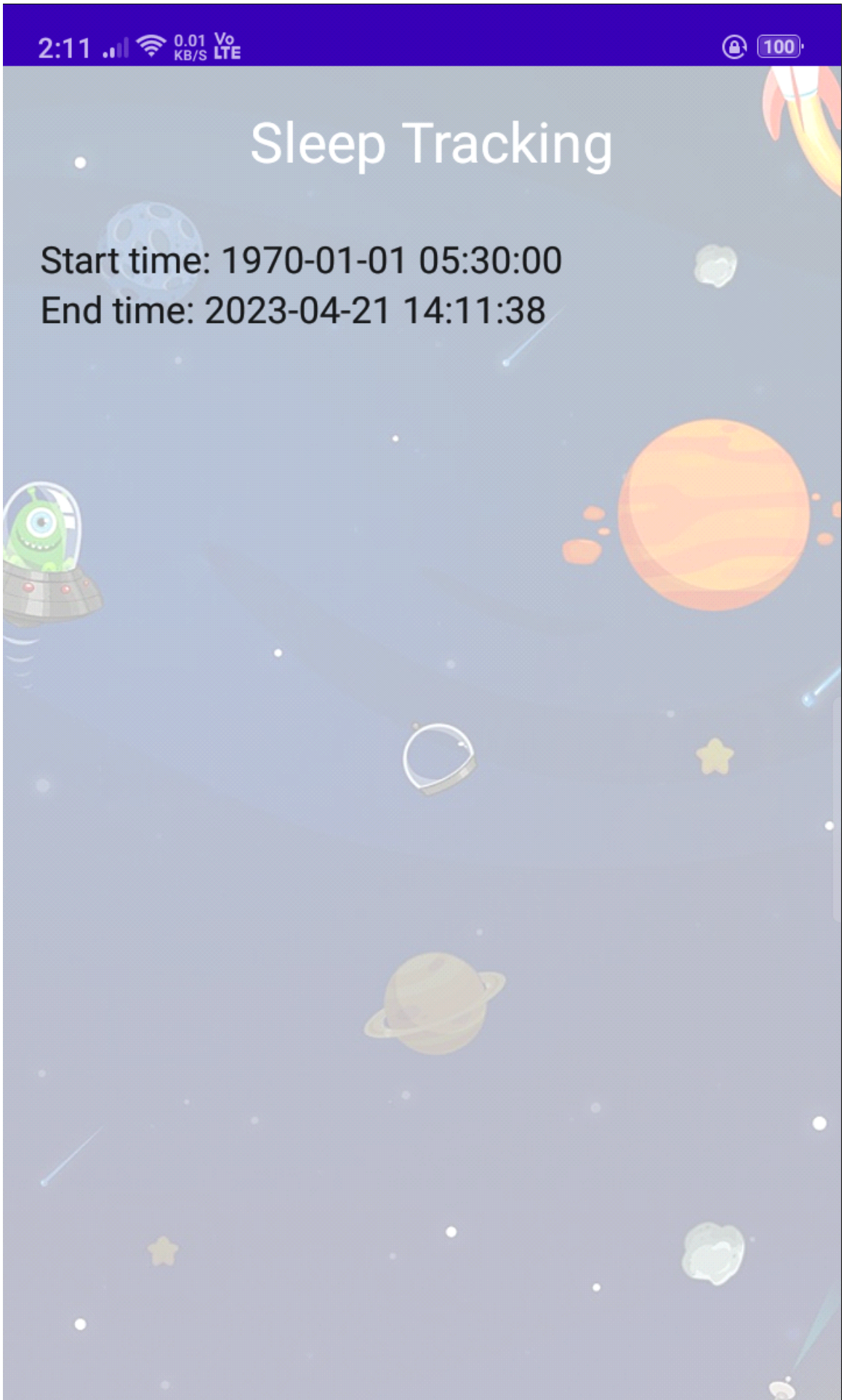
Password
12345678

Register


2:11
1.00 KB/S
Vo LTE
100
Start
Elapsed Time: 00:00:00
Track Sleep


2:11
0.00 KB/S
Vo LTE
100
Stop
Elapsed Time: -8:-41:-38
Track Sleep


2:11
0.02 KB/S
Vo LTE
100
Start
Elapsed Time: 00:00:05
Track Sleep


2:11
0.01 KB/S
Vo LTE
100
Sleep Tracking
Start time: 1970-01-01 05:30:00
End time: 2023-04-21 14:11:38

4ADVANTAGES & DISADVANTAGE:

4.1 Advantages:

Sleep tracking apps have become increasingly popular in recent years. These apps offer numerous advantages for those who use them, ranging from better sleep to improved overall health. In this passage, we will explore the advantages of using sleep tracking apps in more detail.

Better sleep quality: One of the primary advantages of using a sleep tracking app is that it can help improve the quality of your sleep. These apps use sensors to detect movement during the night, which can provide valuable insights into your sleep patterns. By tracking how often you wake up, how long it takes you to fall asleep, and how much time you spend in different stages of sleep, these apps can help you identify areas where you can improve your sleep hygiene.

Increased awareness of sleep habits: Another advantage of sleep tracking apps is that they can increase your awareness of your sleep habits. Many people are unaware of how much sleep they actually get each night, or how much time they spend in each stage of sleep. By tracking your sleep patterns, these apps can provide you with valuable insights into your sleep habits, allowing you to make more informed decisions about your sleep routine.

Better health outcomes: Studies have shown that there is a strong correlation between sleep and overall health. Poor sleep has been linked to a variety of health issues, including obesity, diabetes, and cardiovascular disease. By using a sleep tracking app to improve the quality of your sleep, you can potentially reduce your risk of these health problems.

More efficient use of time: For many people, time is a valuable commodity. Sleep tracking apps can help you make more efficient use of your time by identifying areas where you can improve your sleep hygiene. For example, if you are spending too much time on your phone before bed, a sleep tracking app can help you recognize this habit and make changes to improve your sleep.

Increased accountability: Finally, sleep tracking apps can help increase accountability for your sleep habits. By tracking your sleep patterns, these apps can help you identify areas where you may be falling short in your sleep routine. This can motivate you to make changes and take a more

proactive approach to your sleep hygiene.

4.2 Disadvantage:

The popularity of sleep tracking apps has increased over the years, with many people relying on them to monitor their sleep patterns and improve their sleep quality. However, there are some disadvantages to using these apps that people should be aware of before incorporating them into their daily routine.

Inaccuracy of Sleep Data:One of the biggest disadvantages of sleep tracking apps is the potential inaccuracy of the data they provide. These apps rely on sensors in the device, such as an accelerometer or gyroscope, to detect movement during sleep and determine sleep stages. However, this data may not always be reliable, as movements such as tossing and turning in bed or reading a book before sleep can be misinterpreted as wakefulness, leading to inaccurate results.

False Sense of Security:Another disadvantage of sleep tracking apps is the false sense of security they can provide. Users may feel that by monitoring their sleep, they are taking steps towards improving their health and well-being. However, the data provided by these apps may not be enough to detect serious sleep disorders or underlying health conditions, leading to delayed diagnosis and treatment.

Dependence on Technology:Using sleep tracking apps can also lead to a dependence on technology. Users may feel compelled to use the app every night to monitor their sleep, leading to anxiety or stress if they are unable to use the app for any reason. This dependence can also be a distraction from relaxing before bedtime, leading to further sleep problems.

Privacy Concerns:Sleep tracking apps often require users to grant access to their personal data, including sleep patterns, to third-party companies. This can raise privacy concerns, as the data collected may be used for advertising

or sold to other companies without the user's knowledge or consent.

Negative Impact on Sleep:Ironically, using sleep tracking apps can have a negative impact on sleep. Users may become obsessed with achieving a certain amount of sleep each night, leading to anxiety and stress that can make it more difficult to fall asleep. This can also lead to a decrease in sleep quality, as users may prioritize meeting sleep goals over relaxation and winding down before bed.

Cost:Some sleep tracking apps require users to pay a fee to access all of their features, while others may offer a limited version of the app for free with the option to upgrade. This cost can add up over time, especially if the user relies on the app to monitor their sleep on a regular basis.

Limited Access:Sleep tracking apps require a smartphone or other device to function, limiting access to those who may not have access to or use these devices. This can prevent people from utilizing the benefits of these apps, leaving them without access to important data about their sleep patterns.

Inaccurate data: Sleep tracking apps rely on sensors to detect movement during the night, and these sensors can be prone to errors. For example, if you share a bed with a partner, their movements could be mistakenly recorded as your own. Additionally, sleep tracking apps may not accurately identify the different stages of sleep, which can lead to inaccurate data.

Overreliance on technology: Some people may become overly reliant on sleep tracking apps, relying solely on the data provided by the app to dictate their sleep habits. This can lead to a reduction in self-awareness and a lack of understanding of one's own sleep patterns.

Sleep disturbances: The use of sleep tracking apps may lead to sleep disturbances, as individuals may become too focused on achieving a certain amount of sleep or meeting specific sleep goals. This can cause anxiety and

stress, which can negatively impact sleep quality.

Battery drain: Many sleep tracking apps require the use of sensors or other features that can drain the battery of your device. This can be inconvenient, especially if you rely on your phone as an alarm clock or for other purposes.

Privacy concerns: Some individuals may be uncomfortable with the amount of data that sleep tracking apps collect about their sleep patterns. This can lead to privacy concerns, as the data collected could potentially be shared with third-party advertisers or other entities.

5APPLICATIONS:

Sleep Trackers are ingenious smartphone applications as well as little devices that imperceptibly record several aspects of how well you sleep. Many people use them to zero in on ways to get more, high-quality, restful sleep. And yes, they definitely do work! Today, we're going to take a look at some of the reasons why sleep tracking apps and devices have become so popular.

Improve Your Sleep

Precisely why we sleep is still a bit of a mystery. That said, we can be absolutely sure poor quality sleep impacts almost every aspect of our physical health, mental health, and emotional health.

You don't need to take our word for it. Just think about what it's like to deal with someone who hasn't had enough sleep. Even according to the Harvard medical school, no one knows for sure exactly why we sleep.

Quality Of Sleep Might Be More Important Than Quantity

It's not that difficult to figure out how long you're sleeping. An approximation can be made from when you fall asleep and what time you wake up. Once you adjust for a trip or two to the bathroom, you can pretty much tell if you're getting eight hours.

Sleep quality is a different story. What we've learned about sleep quality comes from the many years of scientists running experiments in sleep labs. The evidence has been overwhelming in demonstrating the importance of sleep quality. However, up to now, tracking sleep quality involves lots of

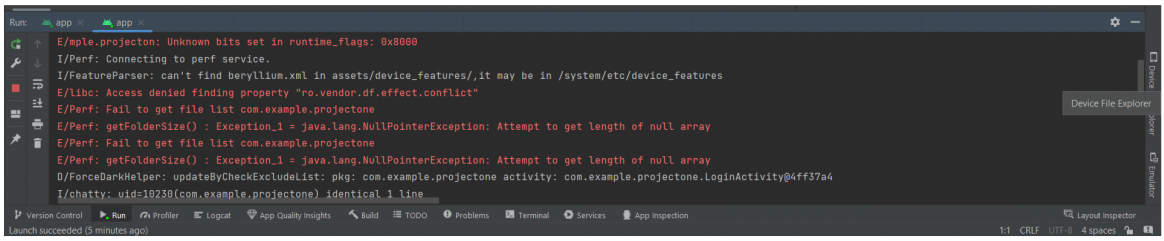
complicated sensors and equipment to measure things like heart rate, brain wave activity, blood oxygen level, R.E.M. or rapid eye movement, and other physical indicators or risk factors

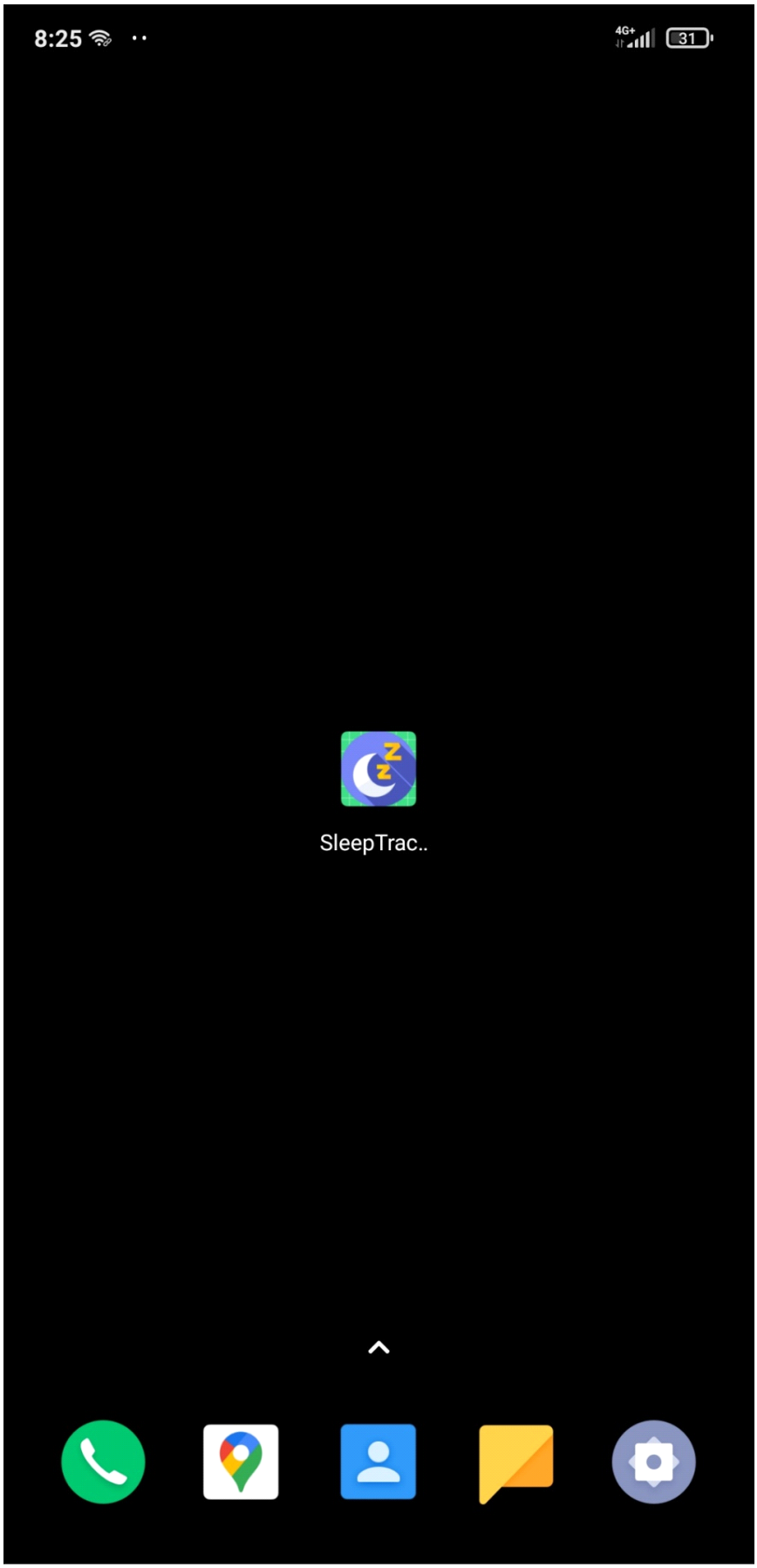
Quality Of Sleep Might Be More Important Than Quantity

It’s not that difficult to figure out how long you’re sleeping. An approximation can be made from when you fall asleep and what time you wake up. Once you adjust for a trip or two to the bathroom, you can pretty much tell if you’re getting eight hours.

Sleep quality is a different story. What we’ve learned about sleep quality comes from the many years of scientists running experiments in sleep labs. The evidence has been overwhelming in demonstrating the importance of sleep quality. However, up to now, tracking sleep quality involves lots of complicated sensors and equipment to measure things like heart rate, brain wave activity, blood oxygen level, R.E.M. or rapid eye movement, and other physical indicators or risk factors

THE AREAS WHERE THIS SOLUTION CAN BE APPLIED:




8:25
4G+
31
SleepTrac..

6 CONCLUSION:

A sleep tracking app is a valuable tool for people who want to monitor and improve their sleep patterns. By tracking factors such as sleep duration, quality, and patterns, users can gain insights into their sleep habits and identify areas for improvement. These apps can also provide personalized recommendations for improving sleep, such as adjusting sleep schedules, reducing caffeine intake, and practicing relaxation techniques.

However, it is important to note that sleep tracking apps are not a substitute for medical advice or treatment. If you are experiencing persistent sleep problems, it is recommended that you consult with a healthcare professional.

Overall, a sleep tracking app can be a helpful tool for promoting healthy sleep habits, but it should be used in conjunction with other self-care practices and medical advice as needed.

sleep tracking applications are powerful tools for monitoring and improving sleep quality. They can provide valuable insights into sleep patterns, help diagnose sleep disorders, and suggest personalized recommendations for healthier sleep habits. Whether you are an athlete, a fitness enthusiast, or simply looking to improve your overall health and well-being, a sleep tracking application can be a valuable tool to help you achieve your goals.

7FUTURE SCOPE

Sleep tracking apps are becoming increasingly popular as people become more aware of the importance of sleep for their health and well-being. Here are some potential future developments for sleep

tracking apps:Integration with other health and fitness apps: Sleep tracking apps could be integrated with other health and fitness apps to provide a more comprehensive overview of a user's overall health. For example, a sleep tracking app could be linked to a fitness tracker to show how physical activity affects sleep quality.

Advanced sleep tracking features: Sleep tracking apps could incorporate more advanced tracking features, such as monitoring sleep stages and detecting sleep disorders. This could be done through the use of wearable devices, such as smartwatches and sleep trackers.

Personalized sleep recommendations: Based on the data collected from a user's sleep tracking, these apps could provide personalized recommendations for improving sleep quality, such as adjusting sleep patterns or suggesting relaxation techniques.

Integration with smart home technology: Sleep tracking apps could be integrated with smart home technology, such as smart lighting and temperature control systems, to create a more conducive sleep environment.

Collaborative sleep tracking: Sleep tracking apps could allow for collaborative sleep tracking between partners or family members, which could be particularly useful for identifying patterns and behaviors that affect both people's sleep quality.

Overall, the future of sleep tracking apps looks bright, with new developments and innovations likely to improve their functionality and usefulness for users.

8 APPENDIX:

8.1. Source Code:

Manitests:

```
<?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" />

                <category
android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Color:

```
import androidx.compose.ui.graphics.Color


val Purple200 = Color(0xFFBB86FC)

val Purple500 = Color(0xFF6200EE)

val Purple700 = Color(0xFF3700B3)
```

SHAPE:

```
import androidx.compose.foundation.shape.RoundedCornerShape
import androidx.compose.material.Shapes
import androidx.compose.ui.unit.dp

val Shapes = Shapes(
    small = RoundedCornerShape(4.dp),
    medium = RoundedCornerShape(4.dp),
    large = RoundedCornerShape(0.dp)
)
```

THEME:

```
import androidx.compose.foundation.isSystemInDarkTheme
import androidx.compose.material.MaterialTheme
import androidx.compose.material.darkColors
import androidx.compose.material.lightColors
import androidx.compose.runtime.Composable

private val DarkColorPalette = darkColors(
    primary = Purple200,
    primaryVariant = Purple700,
    secondary = Teal200
)
```

```
private val LightColorPalette = lightColors(
    primary = Purple500,
    primaryVariant = Purple700,
    secondary = Teal200

    /* Other default colors to override
    background = Color.White,
    surface = Color.White,
    onPrimary = Color.White,
    onSecondary = Color.Black,
    onBackground = Color.Black,
    onSurface = Color.Black,
    */
)

@Composable
fun ProjectOneTheme(darkTheme: Boolean =
isSystemInDarkTheme(), content: @Composable () -> Unit) {
    val colors = if (darkTheme) {
        DarkColorPalette
    } else {
        LightColorPalette
    }
```

```
    MaterialTheme(
        colors = colors,
        typography = Typography,
        shapes = Shapes,
        content = content
    )
```

COLOR:

```
import androidx.compose.material.Typography
import androidx.compose.ui.text.TextStyle
import androidx.compose.ui.text.font.FontFamily
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.unit.sp

// Set of Material typography styles to start with
val Typography = Typography(
    body1 = TextStyle(
        fontFamily = FontFamily.Default,
        fontWeight = FontWeight.Normal,
        fontSize = 16.sp
    )
    /* Other default text styles to override
    button = TextStyle(
```

```
        fontFamily = FontFamily.Default,
        fontWeight = FontWeight.W500,
        fontSize = 14.sp
    ),
    caption = TextStyle(
        fontFamily = FontFamily.Default,
        fontWeight = FontWeight.Normal,
        fontSize = 12.sp
    )
```

APP DADABASE:

```
import androidx.room.RoomDatabase


@Database(entities = [TimeLog::class], version = 1, exportSchema
= false)
abstract class AppDatabase : RoomDatabase() {


    abstract fun timeLogDao(): TimeLogDao


    companion object {
        private var INSTANCE: AppDatabase? = null


        fun getDatabase(context: Context): AppDatabase {
            val tempInstance = INSTANCE
```

```
            if (tempInstance != null) {
                return tempInstance
            }
            synchronized(this) {
                val instance = Room.databaseBuilder(
                    context.applicationContext,
                    AppDatabase::class.java,
                    "app_database"
                ).build()
                INSTANCE = instance
                return instance
            }
        }
    }
}
```

LOGIN ACTIVITY:

```
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?")
        }
    }
}
```

MAIN ACTIVITY:

```
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")
        }

    }
```

REGISTER ACTIVITY:

```
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 MainActivity2 : 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
                ) {

                    RegistrationScreen(this,databaseHelper)
                }
            }
        }
    }
}



@Composable
fun RegistrationScreen(context: Context, databaseHelper:
UserDatabaseHelper) {
    var username by remember { mutableStateOf("") }
    var password by remember { mutableStateOf("") }
    var email 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 = "Register"
)

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 = email,
    onValueChange = { email = it },
    label = { Text("Email") },
    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()
&& email.isNotEmpty()) {
                val user = User(
                    id = null,
                    firstName = username,
                    lastName = null,
                    email = email,
                    password = password
                )
                databaseHelper.insertUser(user)
                error = "User registered successfully"
                // Start LoginActivity using the current context
                context.startActivity(
                    Intent(
                        context,
                        LoginActivity::class.java
                    )
                )

            } else {
                error = "Please fill all fields"
            }
        },
```

```
            modifier = Modifier.padding(top = 16.dp)
        ) {
            Text(text = "Register")
        }
        Spacer(modifier = Modifier.width(10.dp))
        Spacer(modifier = Modifier.height(10.dp))

        Row() {
            Text(
                modifier = Modifier.padding(top = 14.dp), text = "Have
an account?"
            )
            TextButton(onClick = {


            })


            {
                Spacer(modifier = Modifier.width(10.dp))
                Text(text = "Log in")
            }
        }
    }
```

TIME DATABASE:

```
import android.annotation.SuppressLint
import android.content.ContentValues
import android.content.Context
import android.database.Cursor
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper
import java.util.*

class TimeLogDatabaseHelper(context: Context) :
SQLiteOpenHelper(context, DATABASE_NAME, null,
DATABASE_VERSION) {
    companion object {
        private const val DATABASE_NAME = "timelog.db"
        private const val DATABASE_VERSION = 1
        const val TABLE_NAME = "time_logs"
        private const val COLUMN_ID = "id"
        const val COLUMN_START_TIME = "start_time"
        const val COLUMN_END_TIME = "end_time"

        // Database creation SQL statement
        private const val DATABASE_CREATE =
            "create table $TABLE_NAME ($COLUMN_ID integer
primary key autoincrement, " +
                    "$COLUMN_START_TIME integer not null,
```

```
$COLUMN_END_TIME integer);"
    }

    override fun onCreate(db: SQLiteDatabase?) {
        db?.execSQL(DATABASE_CREATE)
    }

    override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int,
newVersion: Int) {
        db?.execSQL("DROP TABLE IF EXISTS
$TABLE_NAME")
        onCreate(db)
    }

    // function to add a new time log to the database
    fun addTimeLog(startTime: Long, endTime: Long) {
        val values = ContentValues()
        values.put(COLUMN_START_TIME, startTime)
        values.put(COLUMN_END_TIME, endTime)
        writableDatabase.insert(TABLE_NAME, null, values)
    }

    // function to get all time logs from the database
    @SuppressLint("Range")
```

```
    fun getTimeLogs(): List<TimeLog> {
        val timeLogs = mutableListOf<TimeLog>()
        val cursor = readableDatabase.rawQuery("select * from
$TABLE_NAME", null)
        cursor.moveToFirst()
        while (!cursor.isAfterLast) {
            val id =
cursor.getInt(cursor.getColumnIndex(COLUMN_ID))
            val startTime =
cursor.getLong(cursor.getColumnIndex(COLUMN_START_TIME
))
            val endTime =
cursor.getLong(cursor.getColumnIndex(COLUMN_END_TIME))
            timeLogs.add(TimeLog(id, startTime, endTime))
            cursor.moveToNext()
        }
        cursor.close()
        return timeLogs
    }


    fun deleteAllData() {
        writableDatabase.execSQL("DELETE FROM
$TABLE_NAME")
    }
```

```
    fun getAllData(): Cursor? {
        val db = this.writableDatabase
        return db.rawQuery("select * from $TABLE_NAME", null)
    }

    data class TimeLog(val id: Int, val startTime: Long, val
endTime: Long?) {
        fun getFormattedStartTime(): String {
            return Date(startTime).toString()
        }

        fun getFormattedEndTime(): String {
            return endTime?.let { Date(it).toString() } ?: "not ended"
        }
    }
```

TIME LOG:

```
import androidx.room.Entity
import androidx.room.PrimaryKey
import java.sql.Date

@Entity(tableName = "TimeLog")
data class TimeLog(
    @PrimaryKey(autoGenerate = true)
```

```
    val id: Int = 0,
    val startTime: Date,
```

TRACK ACTIVITY:

```
import android.icu.text.SimpleDateFormat
import android.os.Bundle
import android.util.Log
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.Image
import androidx.compose.foundation.layout.*
import androidx.compose.foundation.lazy.LazyColumn
import androidx.compose.foundation.lazy.LazyRow
import androidx.compose.foundation.lazy.items
import androidx.compose.material.MaterialTheme
import androidx.compose.material.Surface
import androidx.compose.material.Text
import androidx.compose.runtime.Composable
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.unit.dp
import androidx.compose.ui.unit.sp
import com.example.projectone.ui.theme.ProjectOneTheme
import java.util.*

class TrackActivity : ComponentActivity() {

    private lateinit var databaseHelper: TimeLogDatabaseHelper

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        databaseHelper = TimeLogDatabaseHelper(this)
        setContent {
            ProjectOneTheme {
                // A surface container using the 'background' color from
the theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background
                ) {
                    //ListListScopeSample(timeLogs)
```

```
                val data=databaseHelper.getTimeLogs();
                Log.d("Sandeep" ,data.toString())
                val timeLogs = databaseHelper.getTimeLogs()
                ListListScopeSample(timeLogs)
            }
        }
    }
}


@Composable
fun ListListScopeSample(timeLogs:
List<TimeLogDatabaseHelper.TimeLog>) {
    val imageModifier = Modifier
    Image(
        painterResource(id = R.drawable.sleeptracking),
        contentScale = ContentScale.FillHeight,
        contentDescription = "",
        modifier = imageModifier
            .alpha(0.3F),
    )
```

```
    Text(text = "Sleep Tracking", modifier = Modifier.padding(top =
16.dp, start = 106.dp ), color = Color.White, fontSize = 24.sp)
    Spacer(modifier = Modifier.height(30.dp))
    LazyRow(
        modifier = Modifier
            .fillMaxSize()
            .padding(top = 56.dp),


        horizontalArrangement = Arrangement.SpaceBetween
    ){
        item {


            LazyColumn {
                items(timeLogs) { timeLog ->
                    Column(modifier = Modifier.padding(16.dp)) {
                        //Text("ID: ${timeLog.id}")
                        Text("Start time:
${formatDateTime(timeLog.startTime)}")
                        Text("End time: ${timeLog.endTime?.let {
formatDateTime(it) }}")
                    }
                }
            }
        }
```

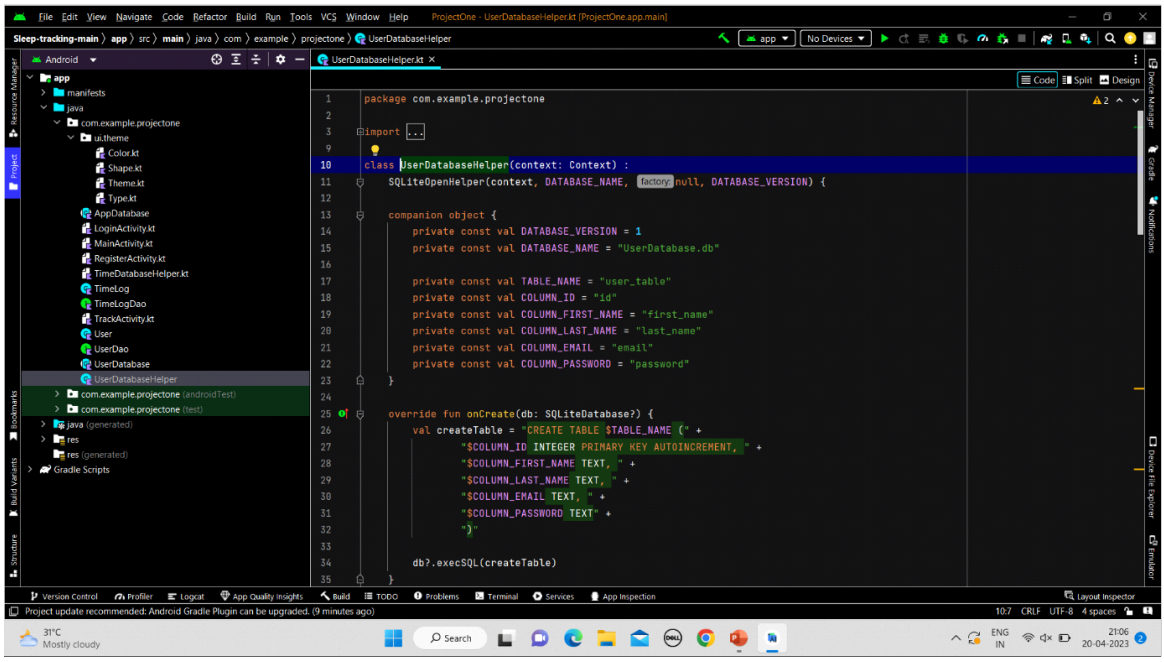
```
        }
    }
```

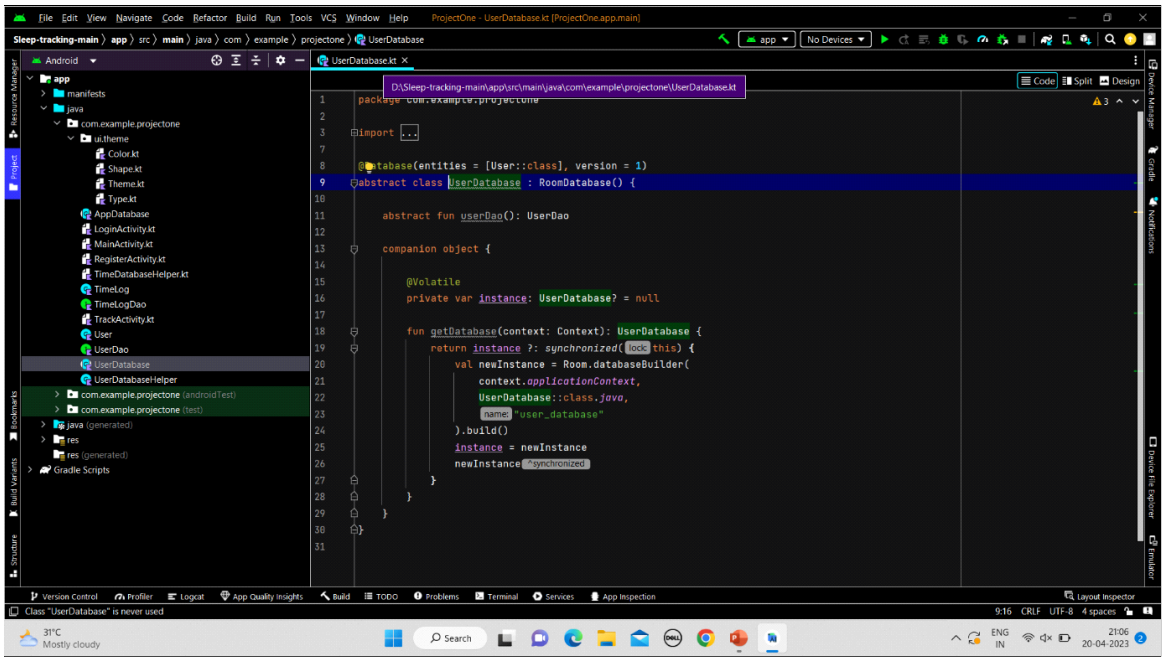
USER DATABSE:

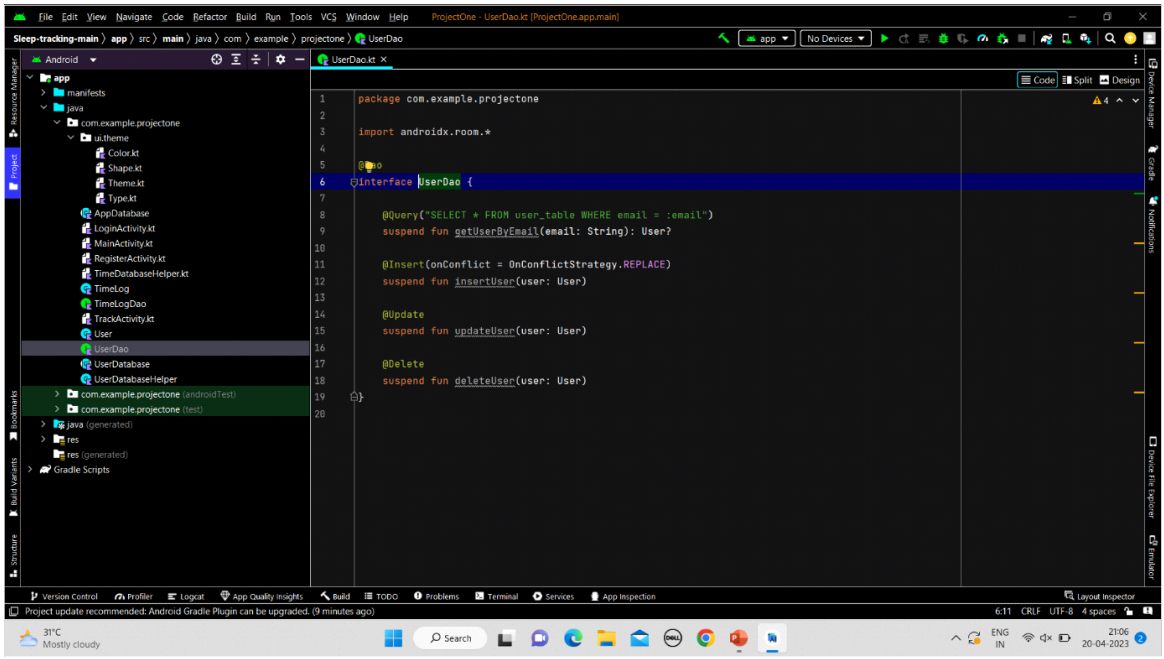
```
import android.content.Context
import androidx.room.Database
import androidx.room.Room
import androidx.room.RoomDatabase

@Database(entities = [User::class], version = 1)
abstract class UserDatabase : RoomDatabase() {

    abstract fun userDao(): UserDao

    companion object {

        @Volatile
        private var instance: UserDatabase? = null

        fun getDatabase(context: Context): UserDatabase {
            return instance ?: synchronized(this) {
                val newInstance = Room.databaseBuilder(
                    context.applicationContext,
```

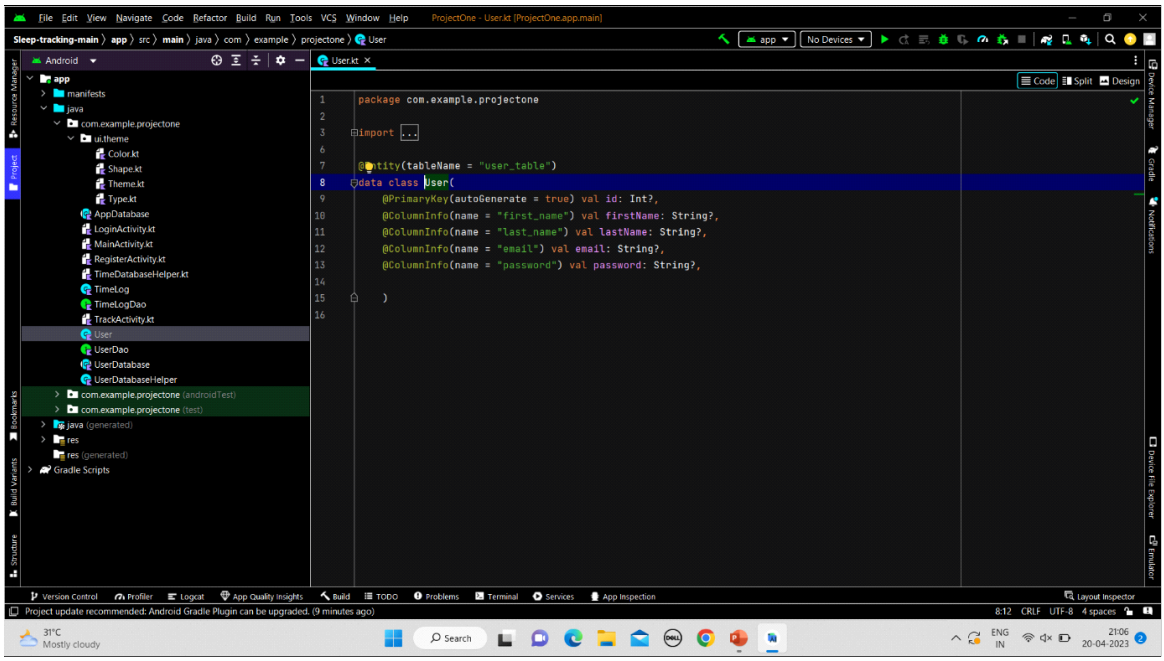
```
            UserDatabase::class.java,
            "user_database"
        ).build()
        instance = newInstance
        newInstance
    }
  }
}
```

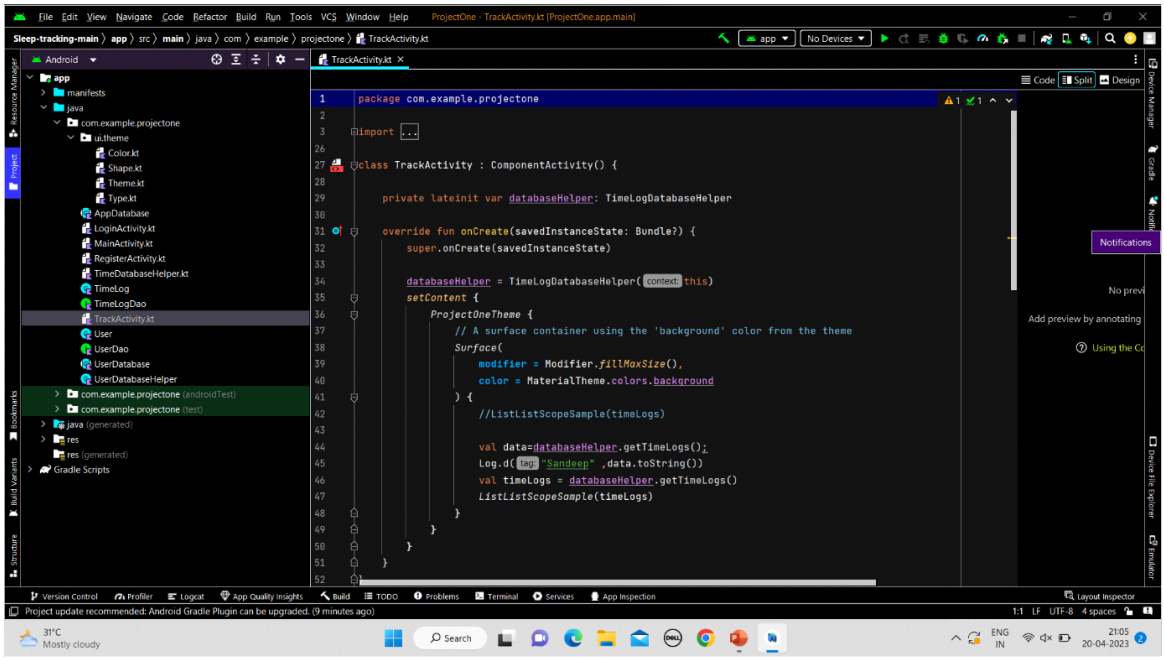
The Solution Built:

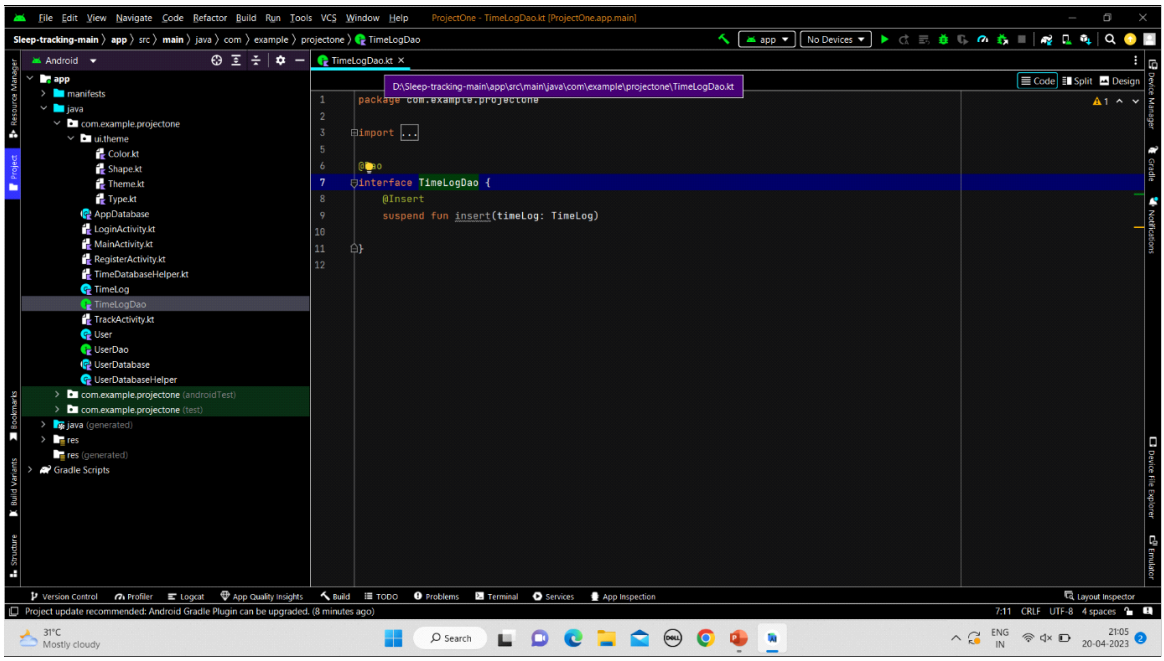


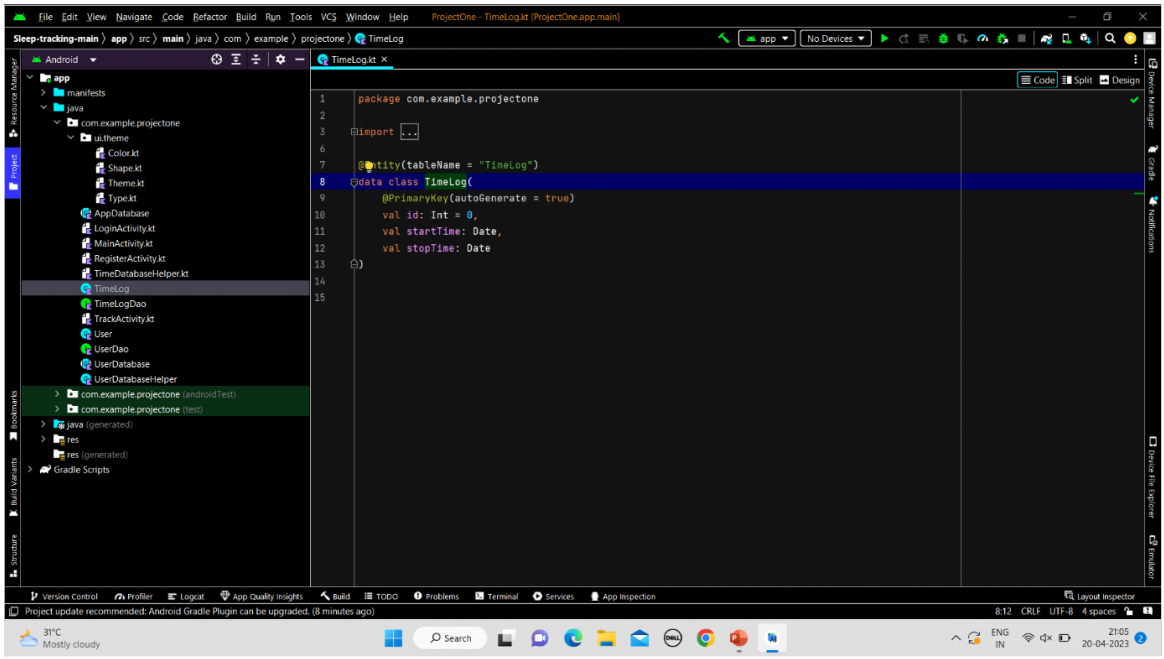

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - UserDatabase.kt [ProjectOne.app.main]
Sleep-tracking-main 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 projectone 〉 UserDatabase
app
No Devices
Android
UserDatabase.kt
D:\Sleep-tracking-main\app\src\main\java\com\example\projectone\UserDatabase.kt
Code
Split
Design
app
manifests
java
com.example.projectone
ui.theme
Color.kt
Shape.kt
Theme.kt
Type.kt
AppDatabase
LoginActivity.kt
MainActivity.kt
RegisterActivity.kt
TimeDatabaseHelper.kt
TimeLog
TimeLogDao
TrackActivity.kt
User
UserDao
UserDatabase
UserDatabaseHelper
com.example.projectone (androidTest)
com.example.projectone (test)
java (generated)
res
res (generated)
Gradle Scripts
package com.example.projectone
import ...
@Database(entities = [User::class], version = 1)
abstract class UserDatabase : RoomDatabase() {
abstract fun userDao(): UserDao
companion object {
@Volatile
private var instance: UserDatabase? = null
fun getDatabase(context: Context): UserDatabase {
return instance ?: synchronized(lock: this) {
val newInstance = Room.databaseBuilder(
context.applicationContext,
UserDatabase::class.java,
name: "user_database"
).build()
instance = newInstance
newInstance ^synchronized
}
}
}
}
Version Control
Profiler
Logcat
App Quality Insights
Build
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Class "UserDatabase" is never used
9:16
CRLF
UTF-8
4 spaces
31°C
Mostly cloudy
Search
ENG
IN
21:06
20-04-2023

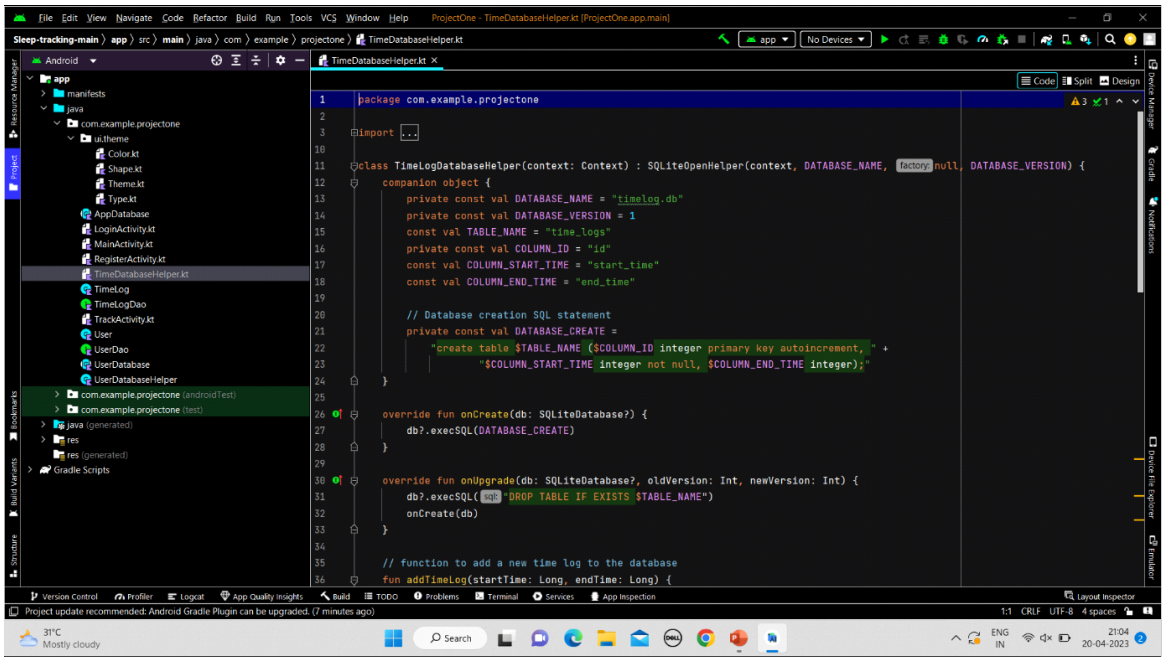

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - UserDao.kt [ProjectOne.app.main]
Sleep-tracking-main 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 projectone 〉 UserDao
app
No Devices
Android
UserDao.kt
Code
Split
Design
app
manifests
java
com.example.projectone
ui.theme
Color.kt
Shape.kt
Theme.kt
Type.kt
AppDatabase
LoginActivity.kt
MainActivity.kt
RegisterActivity.kt
TimeDatabaseHelper.kt
TimeLog
TimeLogDao
TrackActivity.kt
User
UserDao
UserDatabase
UserDatabaseHelper
com.example.projectone (androidTest)
com.example.projectone (test)
java (generated)
res
res (generated)
Gradle Scripts
package com.example.projectone
import androidx.room.*
@Dao
interface UserDao {
@Query("SELECT * FROM user_table WHERE email = :email")
suspend fun getUserByEmail(email: String): User?
@Insert(onConflict = OnConflictStrategy.REPLACE)
suspend fun insertUser(user: User)
@Update
suspend fun updateUser(user: User)
@Delete
suspend fun deleteUser(user: User)
}
Version Control
Profiler
Logcat
App Quality Insights
Build
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Project update recommended: Android Gradle Plugin can be upgraded. (9 minutes ago)
6:11
CRLF
UTF-8
4 spaces
31°C
Mostly cloudy
Search
ENG
IN
21:06
20-04-2023

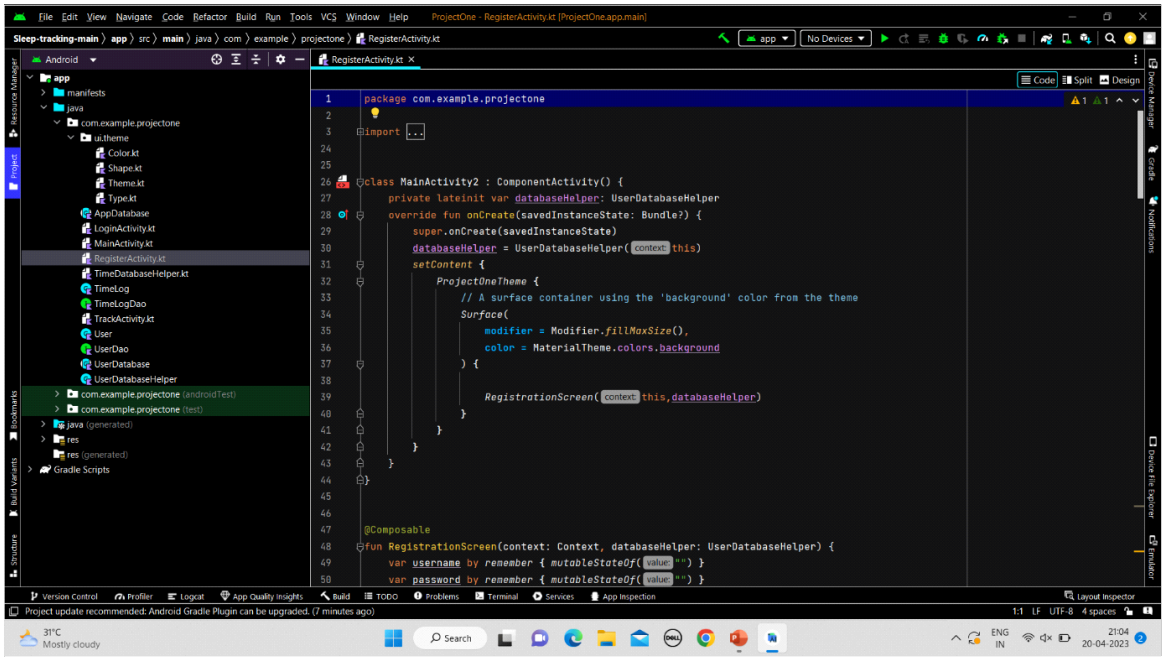

ProjectOne - User.kt [ProjectOne.app.main]
User.kt
package com.example.projectone
import ...
@Entity(tableName = "user_table")
data class User(
@PrimaryKey(autoGenerate = true) val id: Int?,
@ColumnInfo(name = "first_name") val firstName: String?,
@ColumnInfo(name = "last_name") val lastName: String?,
@ColumnInfo(name = "email") val email: String?,
@ColumnInfo(name = "password") val password: String?,
)

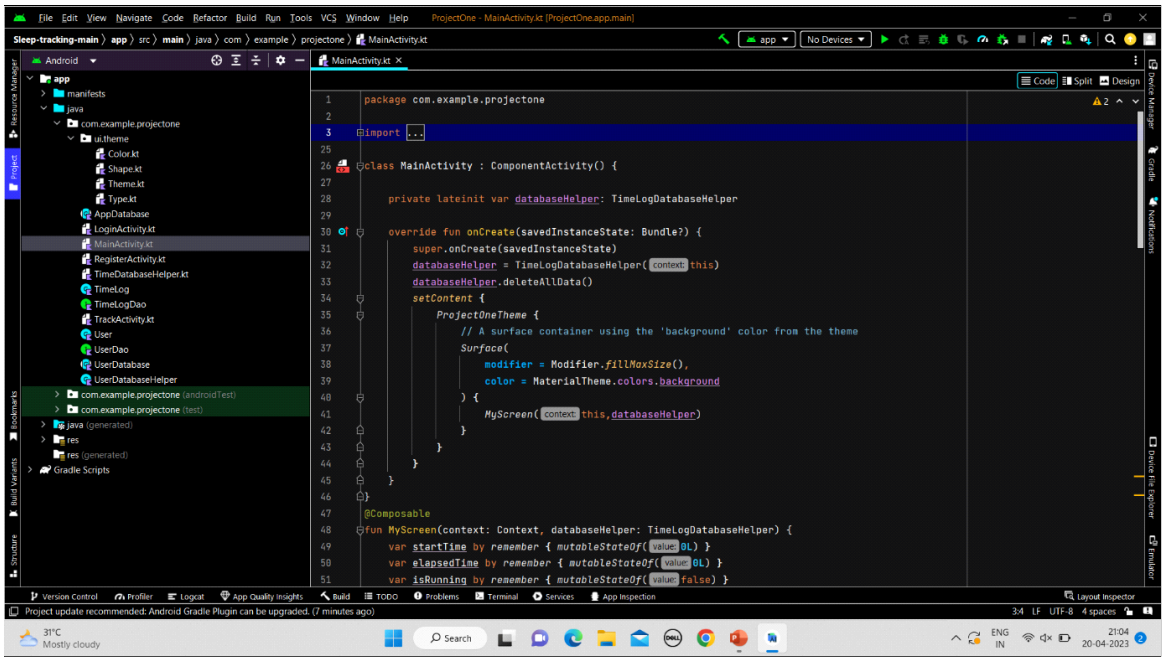

ProjectOne - TrackActivity.kt [ProjectOne.app.main]
TrackActivity.kt
package com.example.projectone
import ...
class TrackActivity : ComponentActivity() {
private lateinit var databaseHelper: TimeLogDatabaseHelper
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
databaseHelper = TimeLogDatabaseHelper(context: this)
setContent {
ProjectOneTheme {
// A surface container using the 'background' color from the theme
Surface(
modifier = Modifier.fillMaxSize(),
color = MaterialTheme.colors.background
) {
//ListListScopeSample(timeLogs)
val data=databaseHelper.getTimeLogs();
Log.d(tag: "Sandeep" ,data.toString())
val timeLogs = databaseHelper.getTimeLogs()
ListListScopeSample(timeLogs)
}
}
}
}
}
Notifications

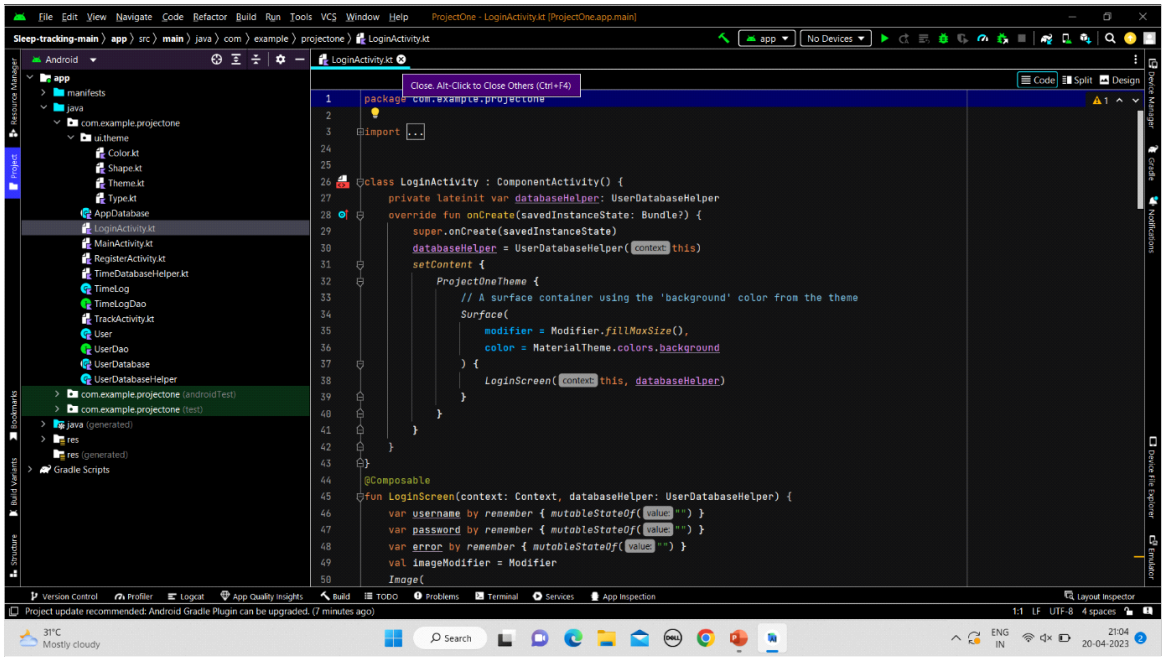

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - TimeLogDao.kt [ProjectOne.app.main]
Sleep-tracking-main 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 projectone 〉 TimeLogDao
app
No Devices
Android
TimeLogDao.kt
D:\Sleep-tracking-main\app\src\main\java\com\example\projectone\TimeLogDao.kt
Code
Split
Design
app
manifests
java
com.example.projectone
ui.theme
Color.kt
Shape.kt
Theme.kt
Type.kt
AppDatabase
LoginActivity.kt
MainActivity.kt
RegisterActivity.kt
TimeDatabaseHelper.kt
TimeLog
TimeLogDao
TrackActivity.kt
User
UserDao
UserDatabase
UserDatabaseHelper
com.example.projectone (androidTest)
com.example.projectone (test)
java (generated)
res
res (generated)
Gradle Scripts
1 package com.example.projectone
2
3 import ...
4
5
6 @Dao
7 interface TimeLogDao {
8 @Insert
9 suspend fun insert(timeLog: TimeLog)
10
11 }
12
Version Control
Profiler
Logcat
App Quality Insights
Build
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Project update recommended: Android Gradle Plugin can be upgraded. (8 minutes ago)
7:11
CRLF
UTF-8
4 spaces
31°C
Mostly cloudy
Search
ENG
IN
21:05
20-04-2023

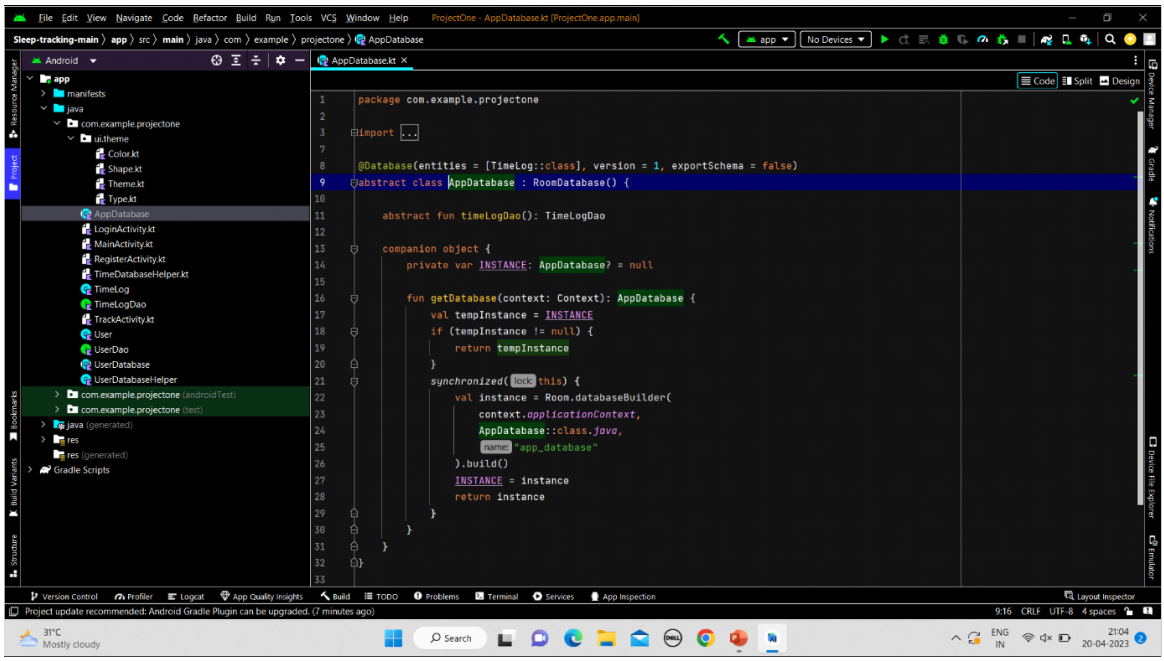

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - TimeLog.kt [ProjectOne.app.main]
Sleep-tracking-main 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 projectone 〉 TimeLog
app
No Devices
Android
TimeLog.kt
Code
Split
Design
app
manifests
java
com.example.projectone
ui.theme
Color.kt
Shape.kt
Theme.kt
Type.kt
AppDatabase
LoginActivity.kt
MainActivity.kt
RegisterActivity.kt
TimeDatabaseHelper.kt
TimeLog
TimeLogDao
TrackActivity.kt
User
UserDao
UserDatabase
UserDatabaseHelper
com.example.projectone (androidTest)
com.example.projectone (test)
java (generated)
res
res (generated)
Gradle Scripts
1 package com.example.projectone
2
3 import ...
4
5
6 @Entity(tableName = "TimeLog")
7 data class TimeLog(
8 @PrimaryKey(autoGenerate = true)
9 val id: Int = 0,
10 val startTime: Date,
11 val stopTime: Date
12)
13
14
15
Version Control
Profiler
Logcat
App Quality Insights
Build
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Project update recommended: Android Gradle Plugin can be upgraded. (8 minutes ago)
8:12
CRLF
UTF-8
4 spaces
31°C
Mostly cloudy
Search
ENG
IN
21:05
20-04-2023

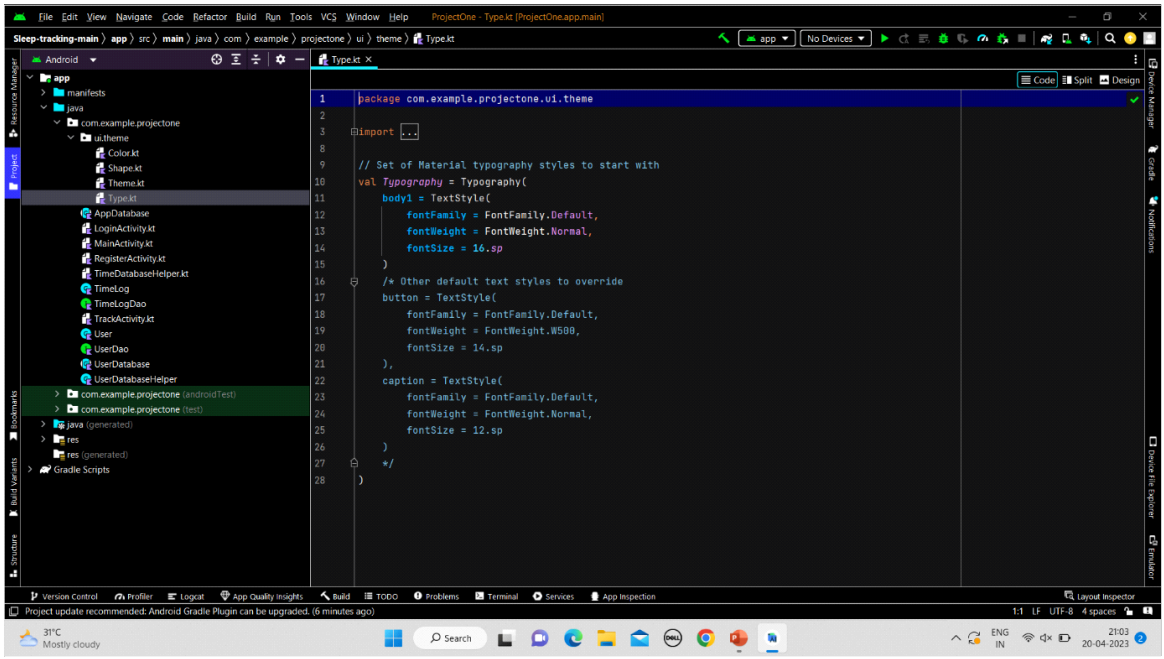

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - TimeDatabaseHelper.kt [ProjectOne.app.main]
Sleep-tracking-main 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 projectone 〉 TimeDatabaseHelper.kt
app
No Devices
Android
TimeDatabaseHelper.kt
Code
Split
Design
app
manifests
java
com.example.projectone
ui.theme
Color.kt
Shape.kt
Theme.kt
Type.kt
AppDatabase
LoginActivity.kt
MainActivity.kt
RegisterActivity.kt
TimeDatabaseHelper.kt
TimeLog
TimeLogDao
TrackActivity.kt
User
UserDao
UserDatabase
UserDatabaseHelper
com.example.projectone (androidTest)
com.example.projectone (test)
java (generated)
res
res (generated)
Gradle Scripts
package com.example.projectone
import ...
class TimeLogDatabaseHelper(context: Context) : SQLiteOpenHelper(context, DATABASE_NAME, factory: null, DATABASE_VERSION) {
companion object {
private const val DATABASE_NAME = "timelog.db"
private const val DATABASE_VERSION = 1
const val TABLE_NAME = "time_logs"
private const val COLUMN_ID = "id"
const val COLUMN_START_TIME = "start_time"
const val COLUMN_END_TIME = "end_time"
// Database creation SQL statement
private const val DATABASE_CREATE =
"create table $TABLE_NAME ($COLUMN_ID integer primary key autoincrement, " +
"$COLUMN_START_TIME integer not null, $COLUMN_END_TIME integer);"
}
override fun onCreate(db: SQLiteDatabase?) {
db?.execSQL(DATABASE_CREATE)
}
override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int, newVersion: Int) {
db?.execSQL(sql: "DROP TABLE IF EXISTS $TABLE_NAME")
onCreate(db)
}
// function to add a new time log to the database
fun addTimeLog(startTime: Long, endTime: Long) {
Version Control
Profiler
Logcat
App Quality Insights
Build
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Project update recommended: Android Gradle Plugin can be upgraded. (7 minutes ago)
1:1
CRLF
UTF-8
4 spaces
31°C
Mostly cloudy
Search
ENG
IN
21:04
20-04-2023

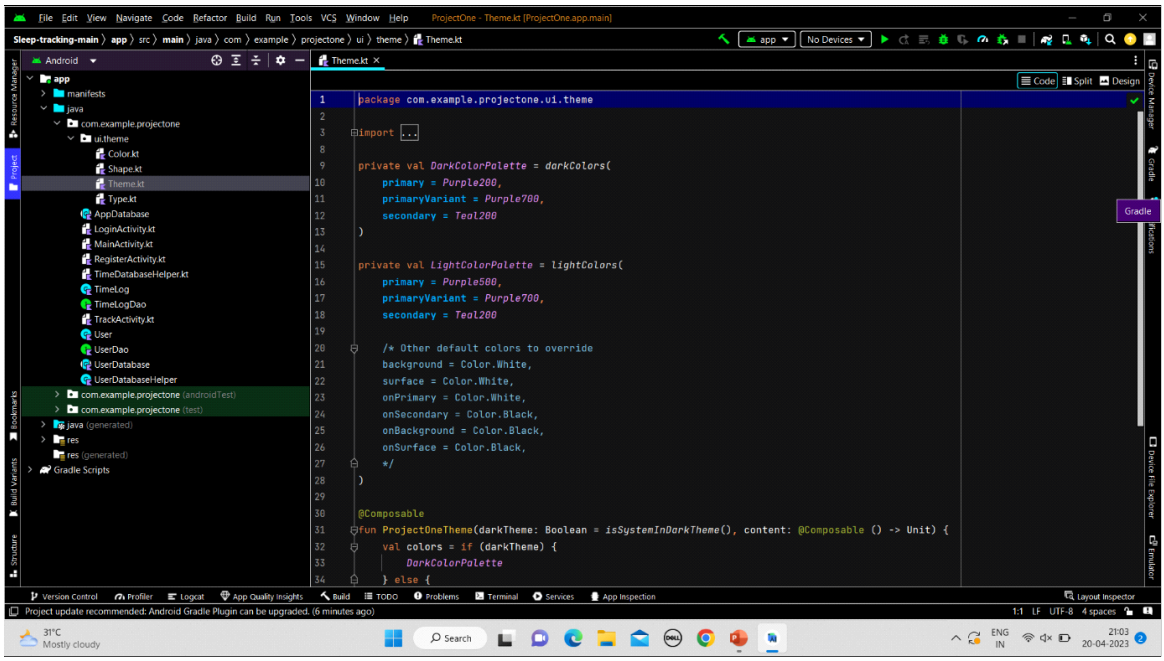

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - RegisterActivity.kt [ProjectOne.app.main]
Sleep-tracking-main 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 projectone 〉 RegisterActivity.kt
app
No Devices
Android
RegisterActivity.kt
Code
Split
Design
app
manifests
java
com.example.projectone
ui.theme
Color.kt
Shape.kt
Theme.kt
Type.kt
AppDatabase
LoginActivity.kt
MainActivity.kt
RegisterActivity.kt
TimeDatabaseHelper.kt
TimeLog
TimeLogDao
TrackActivity.kt
User
UserDao
UserDatabase
UserDatabaseHelper
com.example.projectone (androidTest)
com.example.projectone (test)
java (generated)
res
res (generated)
Gradle Scripts
package com.example.projectone
import ...
class MainActivity2 : ComponentActivity() {
private lateinit var databaseHelper: UserDatabaseHelper
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
databaseHelper = UserDatabaseHelper(context: this)
setContent {
ProjectOneTheme {
// A surface container using the 'background' color from the theme
Surface(
modifier = Modifier.fillMaxSize(),
color = MaterialTheme.colors.background
) {
RegistrationScreen(context: this,databaseHelper)
}
}
}
}
}
@Composable
fun RegistrationScreen(context: Context, databaseHelper: UserDatabaseHelper) {
var username by remember { mutableStateOf(value: "") }
var password by remember { mutableStateOf(value: "") }
Version Control
Profiler
Logcat
App Quality Insights
Build
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Project update recommended: Android Gradle Plugin can be upgraded. (7 minutes ago)
1:1
LF
UTF-8
4 spaces
31°C
Mostly cloudy
Search
ENG
IN
21:04
20-04-2023

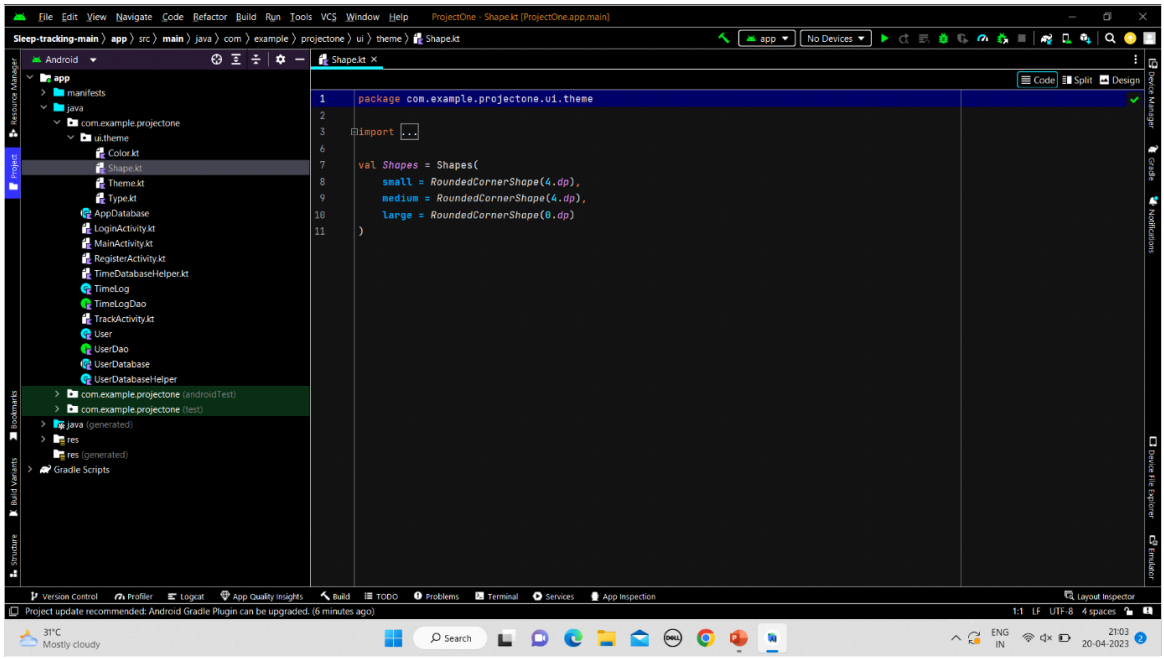

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - MainActivity.kt [ProjectOne.app.main]
Sleep-tracking-main 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 projectone 〉 MainActivity.kt
app
No Devices
Android
MainActivity.kt
Code Split Design
app
manifests
java
com.example.projectone
ui.theme
Color.kt
Shape.kt
Theme.kt
Type.kt
AppDatabase
LoginActivity.kt
MainActivity.kt
RegisterActivity.kt
TimeDatabaseHelper.kt
TimeLog
TimeLogDao
TrackActivity.kt
User
UserDao
UserDatabase
UserDatabaseHelper
com.example.projectone (androidTest)
com.example.projectone (test)
java (generated)
res
res (generated)
Gradle Scripts
package com.example.projectone
import ...
class MainActivity : ComponentActivity() {
private lateinit var databaseHelper: TimeLogDatabaseHelper
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
databaseHelper = TimeLogDatabaseHelper(context: this)
databaseHelper.deleteAllData()
setContent {
ProjectOneTheme {
// A surface container using the 'background' color from the theme
Surface(
modifier = Modifier.fillMaxSize(),
color = MaterialTheme.colors.background
) {
MyScreen(context: this,databaseHelper)
}
}
}
}
}
@Composable
fun MyScreen(context: Context, databaseHelper: TimeLogDatabaseHelper) {
var startTime by remember { mutableStateOf(value: 0L) }
var elapsedTime by remember { mutableStateOf(value: 0L) }
var isRunning by remember { mutableStateOf(value: false) }
Version Control Profiler Logcat App Quality Insights Build TODO Problems Terminal Services App Inspection Layout Inspector
Project update recommended: Android Gradle Plugin can be upgraded. (7 minutes ago)
3:4 LF UTF-8 4 spaces
31°C Mostly cloudy
Search
ENG IN
21:04 20-04-2023

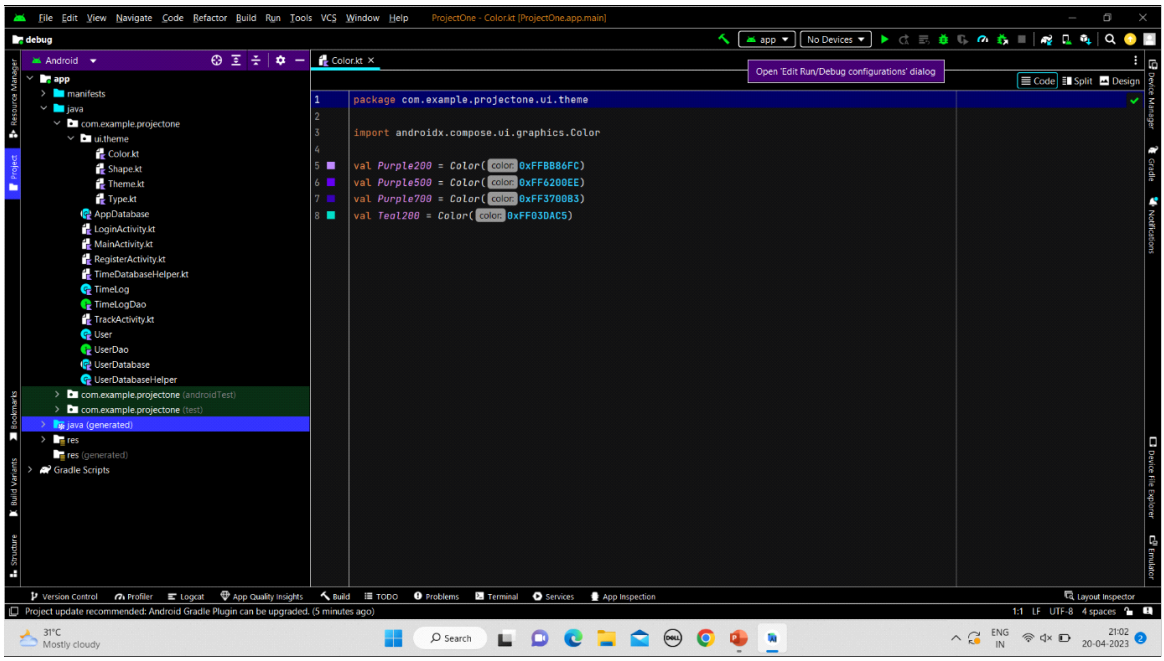

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - LoginActivity.kt [ProjectOne.app.main]
Sleep-tracking-main 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 projectone 〉 LoginActivity.kt
app
No Devices
Android
LoginActivity.kt
Close, Alt-Click to Close Others (Ctrl+F4)
Code Split Design
app
manifests
java
com.example.projectone
ui.theme
Color.kt
Shape.kt
Theme.kt
Type.kt
AppDatabase
LoginActivity.kt
MainActivity.kt
RegisterActivity.kt
TimeDatabaseHelper.kt
TimeLog
TimeLogDao
TrackActivity.kt
User
UserDao
UserDatabase
UserDatabaseHelper
com.example.projectone (androidTest)
com.example.projectone (test)
java (generated)
res
res (generated)
Gradle Scripts
package com.example.projectone
import ...
class LoginActivity : ComponentActivity() {
private lateinit var databaseHelper: UserDatabaseHelper
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
databaseHelper = UserDatabaseHelper(context: this)
setContent {
ProjectOneTheme {
// A surface container using the 'background' color from the theme
Surface(
modifier = Modifier.fillMaxSize(),
color = MaterialTheme.colors.background
) {
LoginScreen(context: this, databaseHelper)
}
}
}
}
}
@Composable
fun LoginScreen(context: Context, databaseHelper: UserDatabaseHelper) {
var username by remember { mutableStateOf(value: "") }
var password by remember { mutableStateOf(value: "") }
var error by remember { mutableStateOf(value: "") }
val imageModifier = Modifier
Image(
Version Control Profiler Logcat App Quality Insights Build TODO Problems Terminal Services App Inspection Layout Inspector
Project update recommended: Android Gradle Plugin can be upgraded. (7 minutes ago)
1:1 LF UTF-8 4 spaces
31°C Mostly cloudy
Search
ENG IN
21:04 20-04-2023


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - AppDatabase.kt [ProjectOne.app.main]
Sleep-tracking-main 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 projectone 〉 AppDatabase
app
No Devices
Android
AppDatabase.kt
Code Split Design
app
manifests
java
com.example.projectone
ui.theme
Color.kt
Shape.kt
Theme.kt
Type.kt
AppDatabase
LoginActivity.kt
MainActivity.kt
RegisterActivity.kt
TimeDatabaseHelper.kt
TimeLog
TimeLogDao
TrackActivity.kt
User
UserDao
UserDatabase
UserDatabaseHelper
com.example.projectone (androidTest)
com.example.projectone (test)
java (generated)
res
res (generated)
Gradle Scripts
package com.example.projectone
import ...
@Database(entities = [TimeLog::class], version = 1, exportSchema = false)
abstract class AppDatabase : RoomDatabase() {
abstract fun timeLogDao(): TimeLogDao
companion object {
private var INSTANCE: AppDatabase? = null
fun getDatabase(context: Context): AppDatabase {
val tempInstance = INSTANCE
if (tempInstance != null) {
return tempInstance
}
synchronized(lock: this) {
val instance = Room.databaseBuilder(
context.applicationContext,
AppDatabase::class.java,
name: "app_database"
).build()
INSTANCE = instance
return instance
}
}
}
}
Version Control Profiler Logcat App Quality Insights Build TODO Problems Terminal Services App Inspection Layout Inspector
Project update recommended: Android Gradle Plugin can be upgraded. (7 minutes ago)
9:16 CRLF UTF-8 4 spaces
31°C
Mostly cloudy
Search
ENG
IN
21:04
20-04-2023

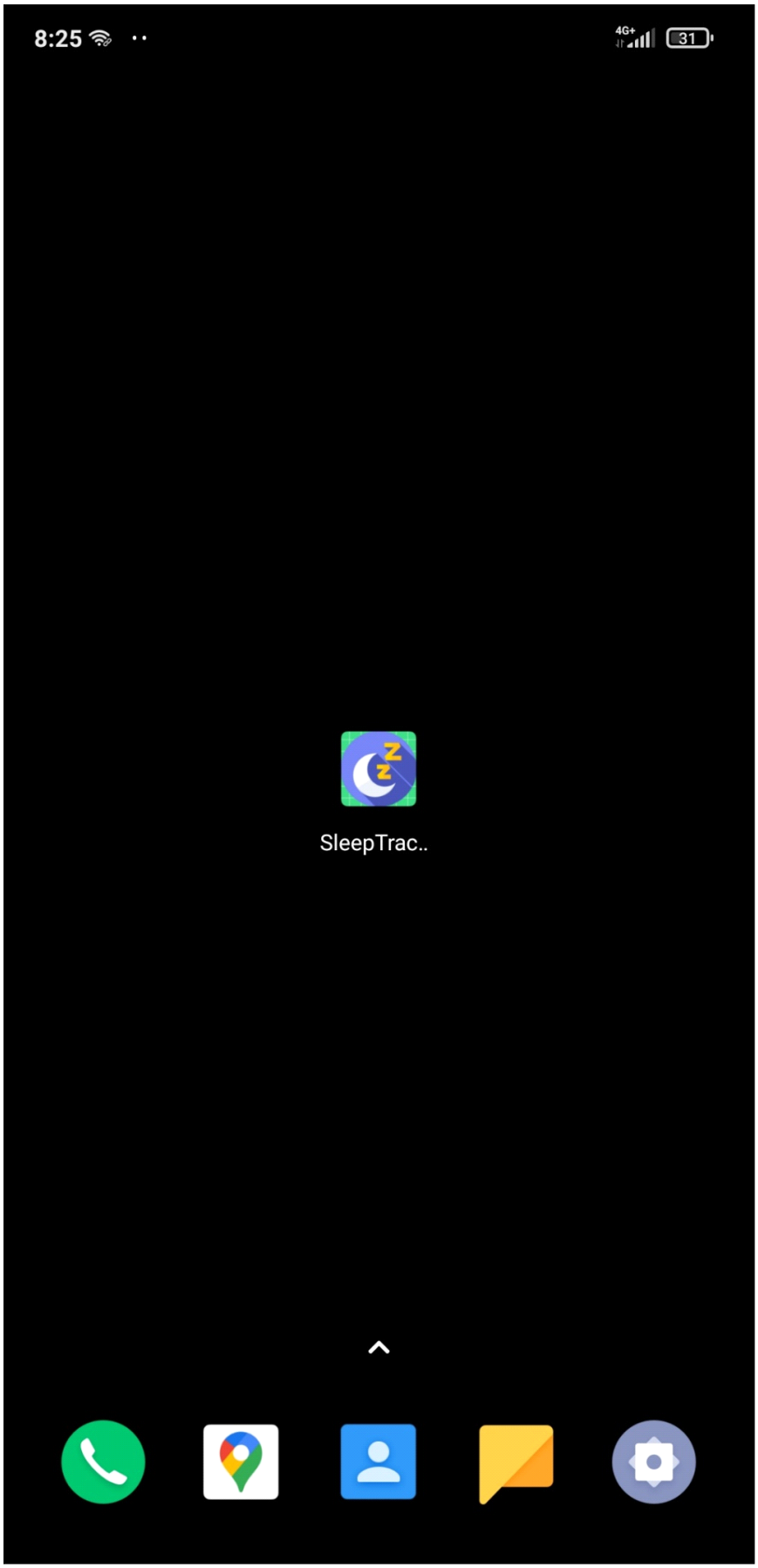

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - Type.kt [ProjectOne.app.main]
Sleep-tracking-main 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 projectone 〉 ui 〉 theme 〉 Type.kt
app
No Devices
Android
Type.kt
Code Split Design
app
manifests
java
com.example.projectone
ui.theme
Color.kt
Shape.kt
Theme.kt
Type.kt
AppDatabase
LoginActivity.kt
MainActivity.kt
RegisterActivity.kt
TimeDatabaseHelper.kt
TimeLog
TimeLogDao
TrackActivity.kt
User
UserDao
UserDatabase
UserDatabaseHelper
com.example.projectone (androidTest)
com.example.projectone (test)
java (generated)
res
res (generated)
Gradle Scripts
package com.example.projectone.ui.theme
import ...
// Set of Material typography styles to start with
val Typography = Typography(
body1 = TextStyle(
fontFamily = FontFamily.Default,
fontWeight = FontWeight.Normal,
fontSize = 16.sp
)
/* Other default text styles to override
button = TextStyle(
fontFamily = FontFamily.Default,
fontWeight = FontWeight.W500,
fontSize = 14.sp
),
caption = TextStyle(
fontFamily = FontFamily.Default,
fontWeight = FontWeight.Normal,
fontSize = 12.sp
)
*/
)
Version Control Profiler Logcat App Quality Insights Build TODO Problems Terminal Services App Inspection Layout Inspector
Project update recommended: Android Gradle Plugin can be upgraded. (6 minutes ago)
1:1 LF UTF-8 4 spaces
31°C
Mostly cloudy
Search
ENG
IN
21:03
20-04-2023

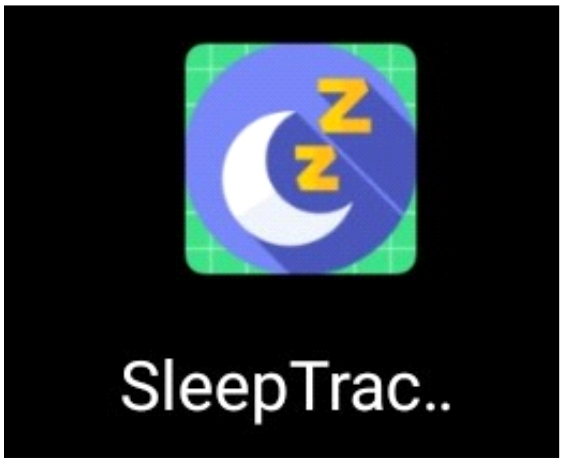

File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - Theme.kt [ProjectOne.app.main]
Sleep-tracking-main 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 projectone 〉 ui 〉 theme 〉 Theme.kt
app
No Devices
Android
Theme.kt
Code
Split
Design
app
manifests
java
com.example.projectone
ui.theme
Color.kt
Shape.kt
Theme.kt
Type.kt
AppDatabase
LoginActivity.kt
MainActivity.kt
RegisterActivity.kt
TimeDatabaseHelper.kt
TimeLog
TimeLogDao
TrackActivity.kt
User
UserDao
UserDatabase
UserDatabaseHelper
com.example.projectone (androidTest)
com.example.projectone (test)
java (generated)
res
res (generated)
Gradle Scripts
1 package com.example.projectone.ui.theme
3 import ...
9 private val DarkColorPalette = darkColors(
10 primary = Purple200,
11 primaryVariant = Purple700,
12 secondary = Teal200
13)
15 private val LightColorPalette = lightColors(
16 primary = Purple500,
17 primaryVariant = Purple700,
18 secondary = Teal200
20 /* Other default colors to override
21 background = Color.White,
22 surface = Color.White,
23 onPrimary = Color.White,
24 onSecondary = Color.Black,
25 onBackground = Color.Black,
26 onSurface = Color.Black,
27 */
28)
30 @Composable
31 fun ProjectOneTheme(darkTheme: Boolean = isSystemInDarkTheme(), content: @Composable () -> Unit) {
32 val colors = if (darkTheme) {
33 DarkColorPalette
34 } else {
Gradle
Version Control
Profiler
Logcat
App Quality Insights
Build
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Project update recommended: Android Gradle Plugin can be upgraded. (6 minutes ago)
1:1 LF UTF-8 4 spaces
31°C
Mostly cloudy
Search
ENG
IN
21:03
20-04-2023


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - Shape.kt [ProjectOne.app.main]
Sleep-tracking-main 〉 app 〉 src 〉 main 〉 java 〉 com 〉 example 〉 projectone 〉 ui 〉 theme 〉 Shape.kt
app
No Devices
Android
Shape.kt
Code
Split
Design
app
manifests
java
com.example.projectone
ui.theme
Color.kt
Shape.kt
Theme.kt
Type.kt
AppDatabase
LoginActivity.kt
MainActivity.kt
RegisterActivity.kt
TimeDatabaseHelper.kt
TimeLog
TimeLogDao
TrackActivity.kt
User
UserDao
UserDatabase
UserDatabaseHelper
com.example.projectone (androidTest)
com.example.projectone (test)
java (generated)
res
res (generated)
Gradle Scripts
1 package com.example.projectone.ui.theme
3 import ...
7 val Shapes = Shapes(
8 small = RoundedCornerShape(4.dp),
9 medium = RoundedCornerShape(4.dp),
10 large = RoundedCornerShape(0.dp)
11)
Version Control
Profiler
Logcat
App Quality Insights
Build
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Project update recommended: Android Gradle Plugin can be upgraded. (6 minutes ago)
1:1 LF UTF-8 4 spaces
31°C
Mostly cloudy
Search
ENG
IN
21:03
20-04-2023


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
ProjectOne - Color.kt [ProjectOne.app.main]
debug
app
No Devices
Android
app
manifests
java
com.example.projectone
ui.theme
Color.kt
Shape.kt
Theme.kt
Type.kt
AppDatabase
LoginActivity.kt
MainActivity.kt
RegisterActivity.kt
TimeDatabaseHelper.kt
TimeLog
TimeLogDao
TrackActivity.kt
User
UserDao
UserDatabase
UserDatabaseHelper
com.example.projectone (androidTest)
com.example.projectone (test)
java (generated)
res
res (generated)
Gradle Scripts
Color.kt
Open 'Edit Run/Debug configurations' dialog
Code
Split
Design
1 package com.example.projectone.ui.theme
3 import androidx.compose.ui.graphics.Color
5 val Purple200 = Color(color: 0xFFBB86FC)
6 val Purple500 = Color(color: 0xFF6200EE)
7 val Purple700 = Color(color: 0xFF3700B3)
8 val Teal200 = Color(color: 0xFF03DAC5)
Version Control
Profiler
Logcat
App Quality Insights
Build
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Project update recommended: Android Gradle Plugin can be upgraded. (5 minutes ago)
1:1 LF UTF-8 4 spaces
31°C
Mostly cloudy
Search
ENG
IN
21:02
20-04-2023


31°C
Haze
Search
ENG
IN
10:34
20-04-2023


8:25
4G+
31
SleepTrac..


SleepTrac..

2:11 0.02 KB/S Vo LTE 100

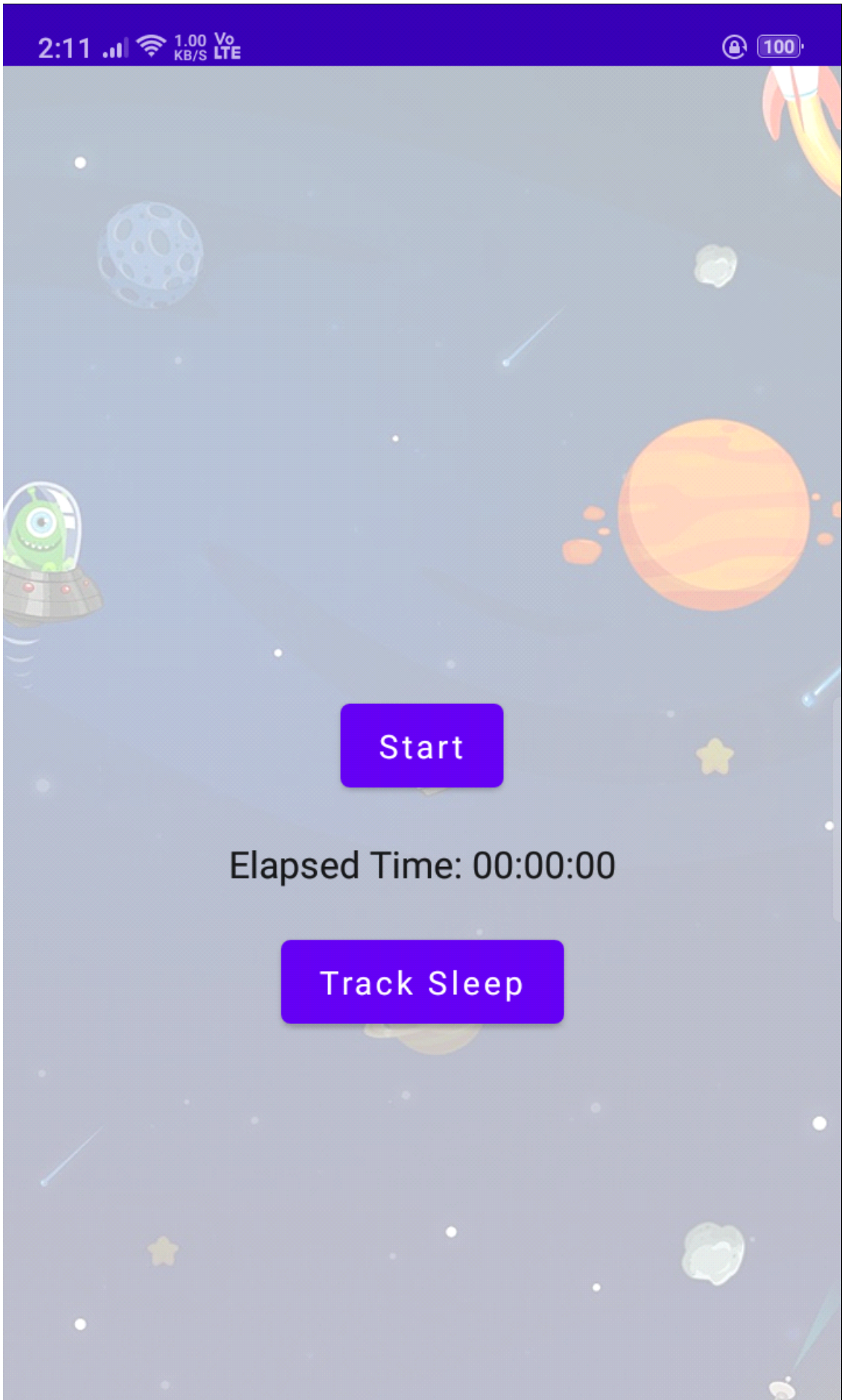
Register

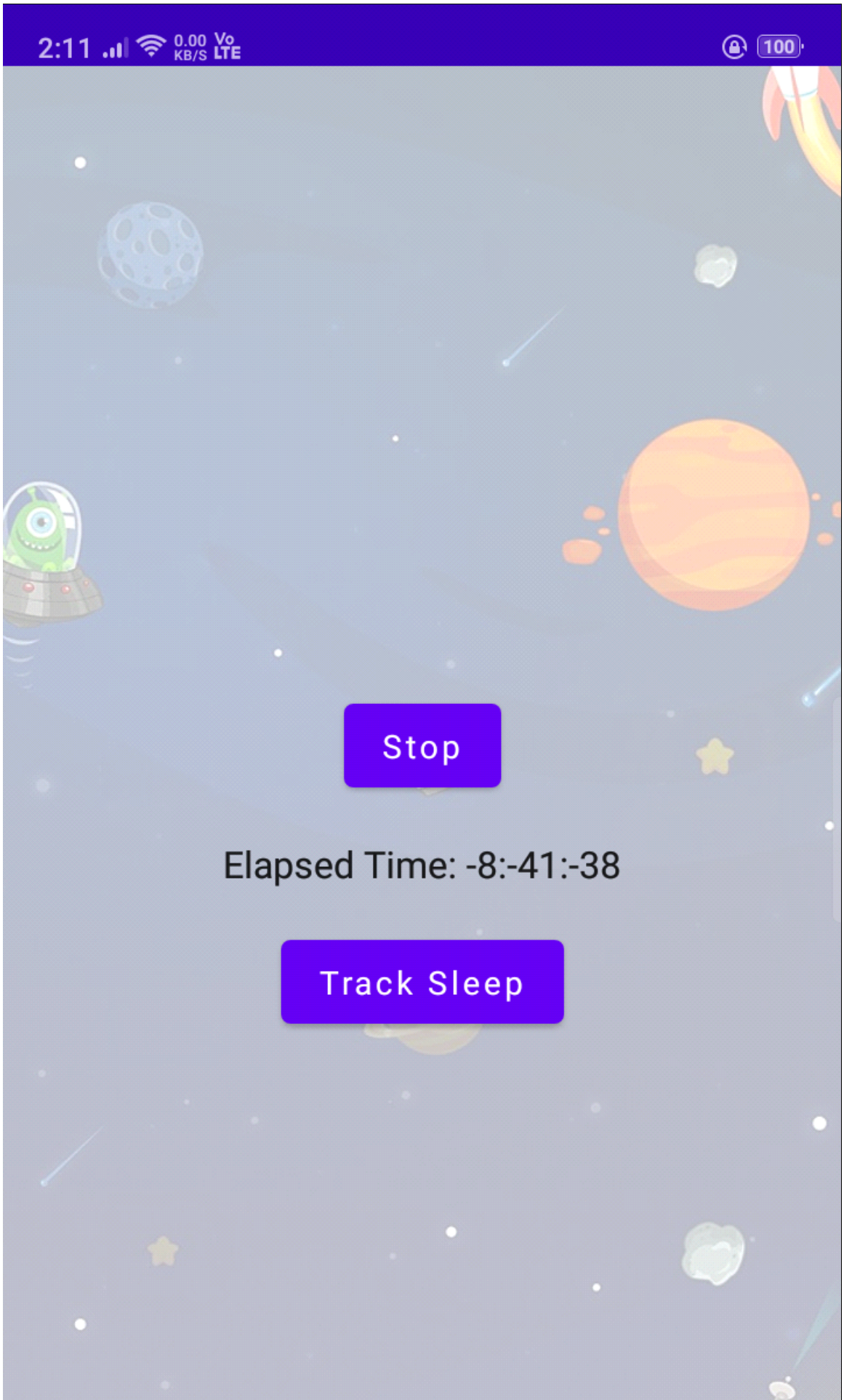
Username
Altrin

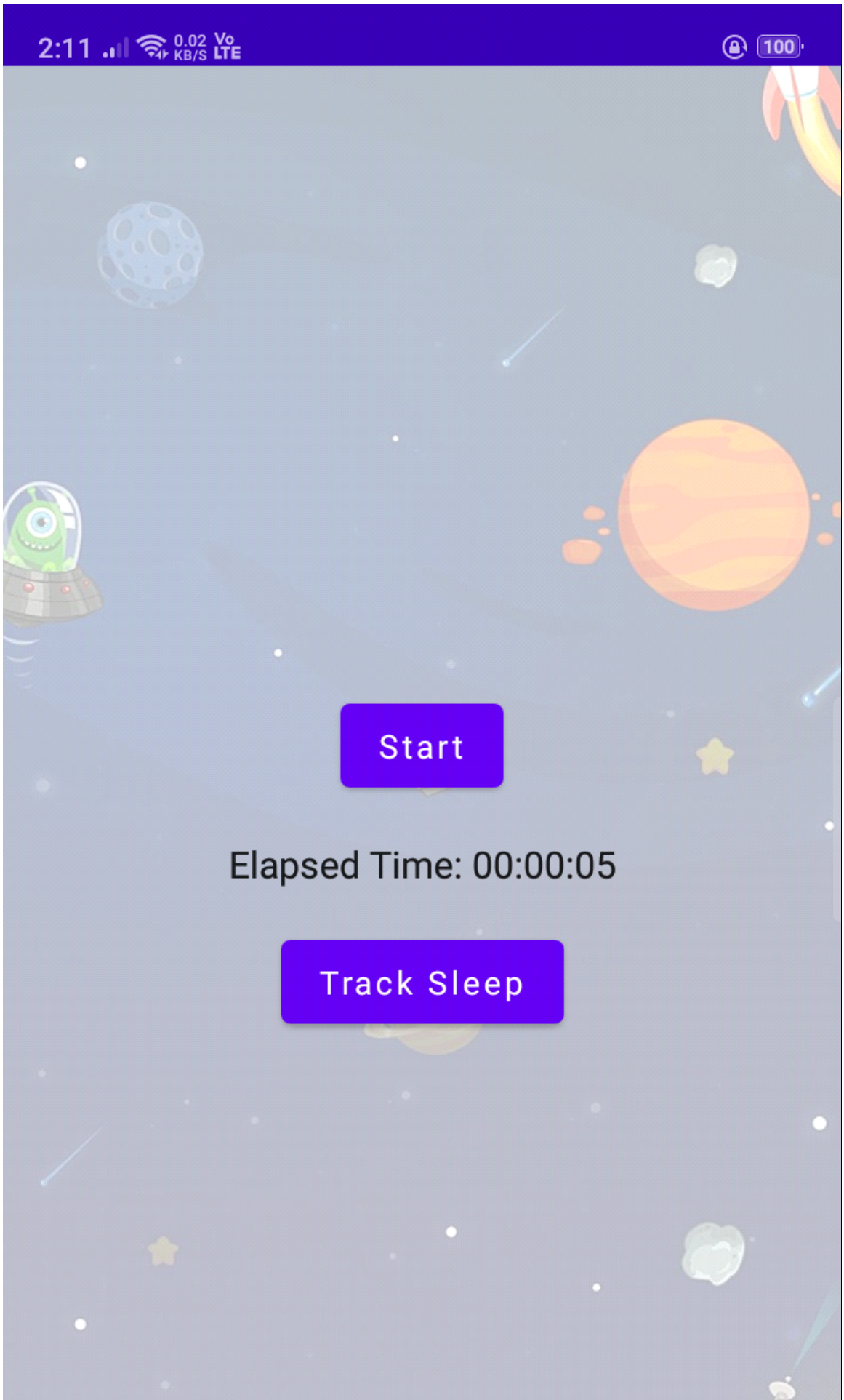
Email
altrinaltrin836@gmail.com

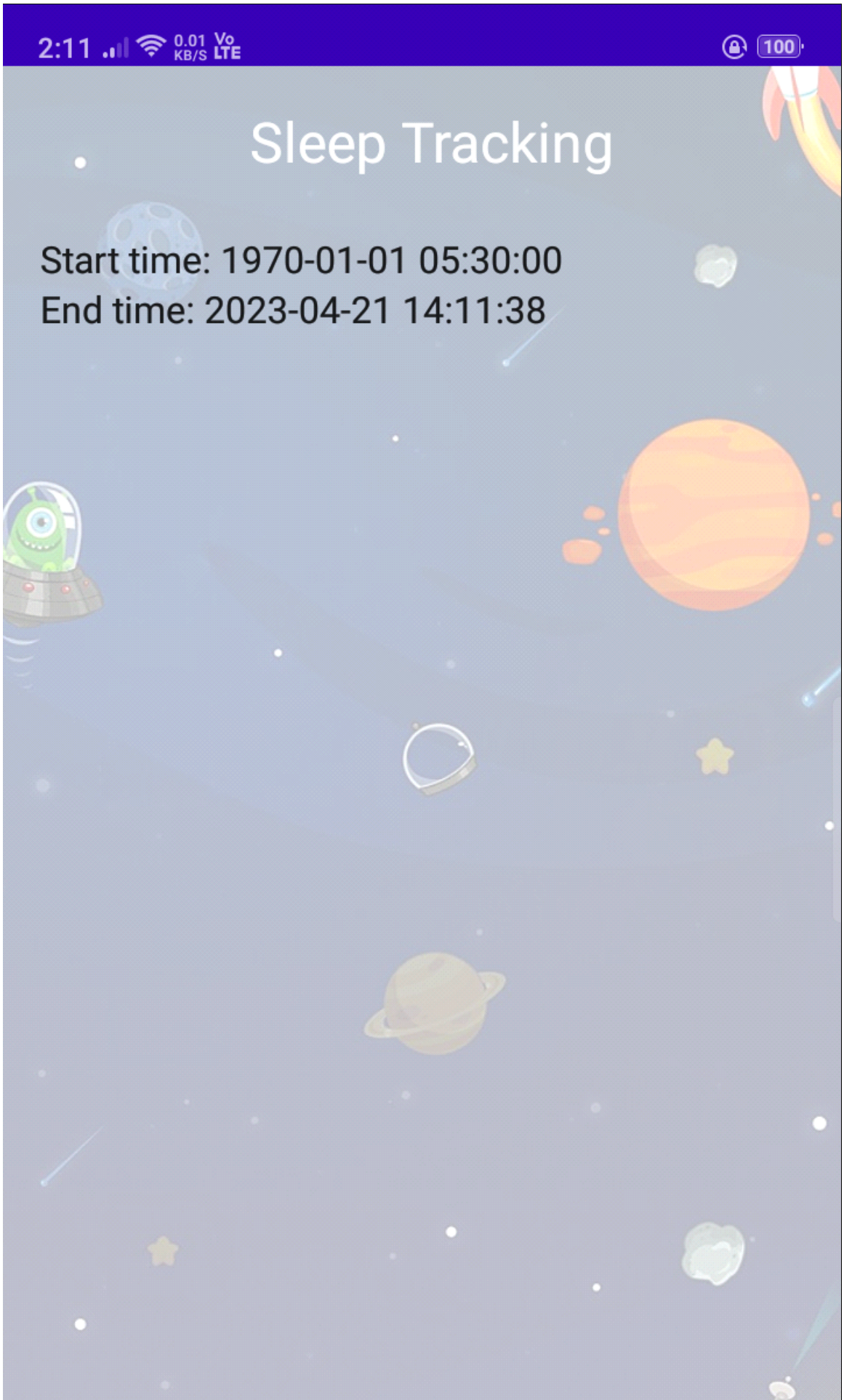
Password
12345678

Register


2:11
1.00 KB/S
Vo LTE
100
Start
Elapsed Time: 00:00:00
Track Sleep


2:11
0.00 KB/S
Vo LTE
100
Stop
Elapsed Time: -8:-41:-38
Track Sleep


2:11
0.02 KB/S
Vo LTE
100
Start
Elapsed Time: 00:00:05
Track Sleep


2:11
0.01 KB/S
Vo LTE
100
Sleep Tracking
Start time: 1970-01-01 05:30:00
End time: 2023-04-21 14:11:38