1-INTRODUCTION

1.1 OVERVIEW

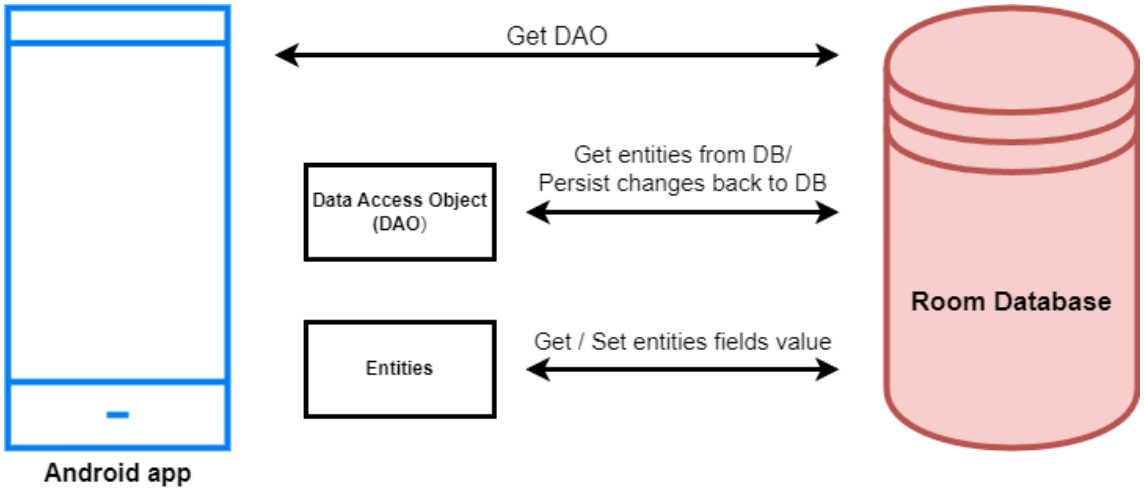
A Sleep Tracking App For A Better Night's Rest

Project Description:

A project that demonstrates the use of Android Jetpack Compose to build a UI for a sleep tracking app. The app allows users to track their sleep.With the “Sleep Tracker” app, you can assess the quality of sleep they have had in a day. It has been time and again proven that a good quality sleep is pretty essential for effective functioning of both mind and body.

“Sleep Tracker” application enables you to start the timer when they are in the bed and about to fall asleep. The timer will keep running in the background until it is stopped, whenever the user wakes up. Based on the sleep experience, you can rate your sleep quality. Finally , the app will display an analysis of the kind of sleep , you had the previous night.

Architecture



Learning Outcomes :

By end of this project:

- You’ll be able to work on Android studio and build an app.
- You’ll be able to integrate the database accordingly.

Project Workflow:

- Users register into the application.
- After registration , user logins into the application.
- User enters into the main page
- User can track the sleep timing and he record the time

Note:

To complete the project you need to finish up the tasks listed below:

Tasks:

1.Required initial steps
2.Creating a new project.
3.Adding required dependencies.
4.Creating the database classes.
5.Building application UI and connecting to database.
6.Using AndroidManifest.xml
7.Running the application.

1.2 PURPOSE

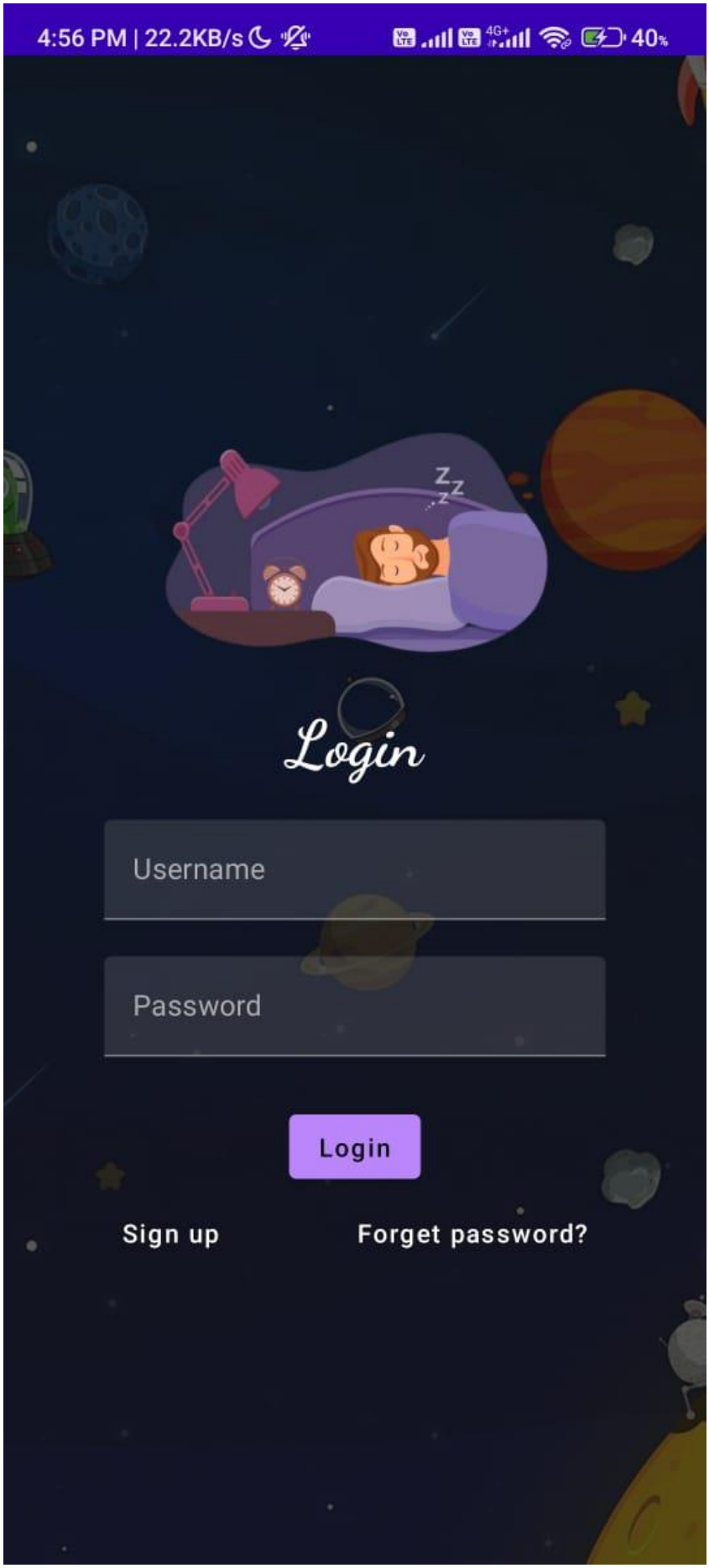
The purpose of a sleep tracking app is to help users improve the quality and quantity of their sleep. By tracking various metrics such as how long they sleep, how often they wake up during the night, and their overall sleep quality, users can gain insights into their sleep patterns and identify potential areas for improvement.

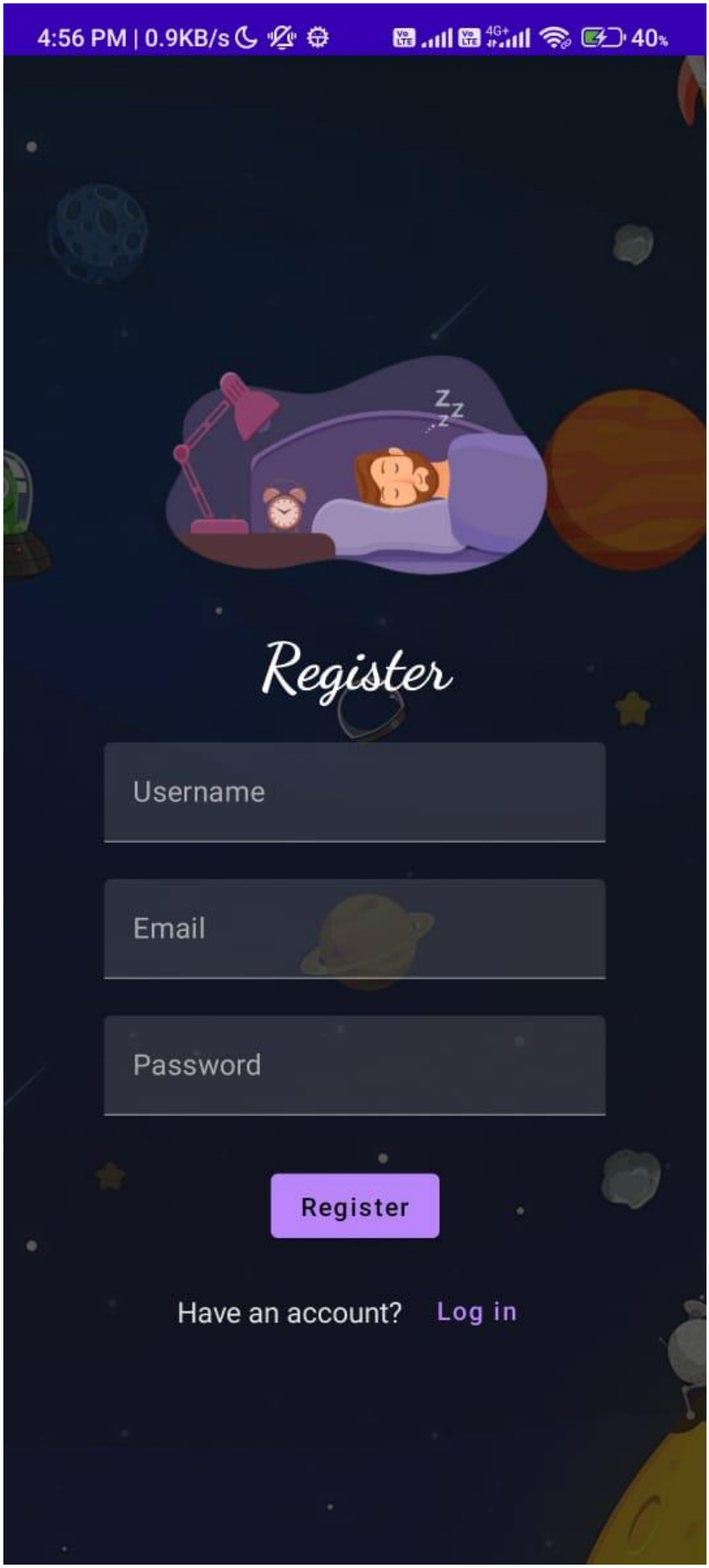
2-PROBLEM DEFINATION AND DESIGN THINKING:

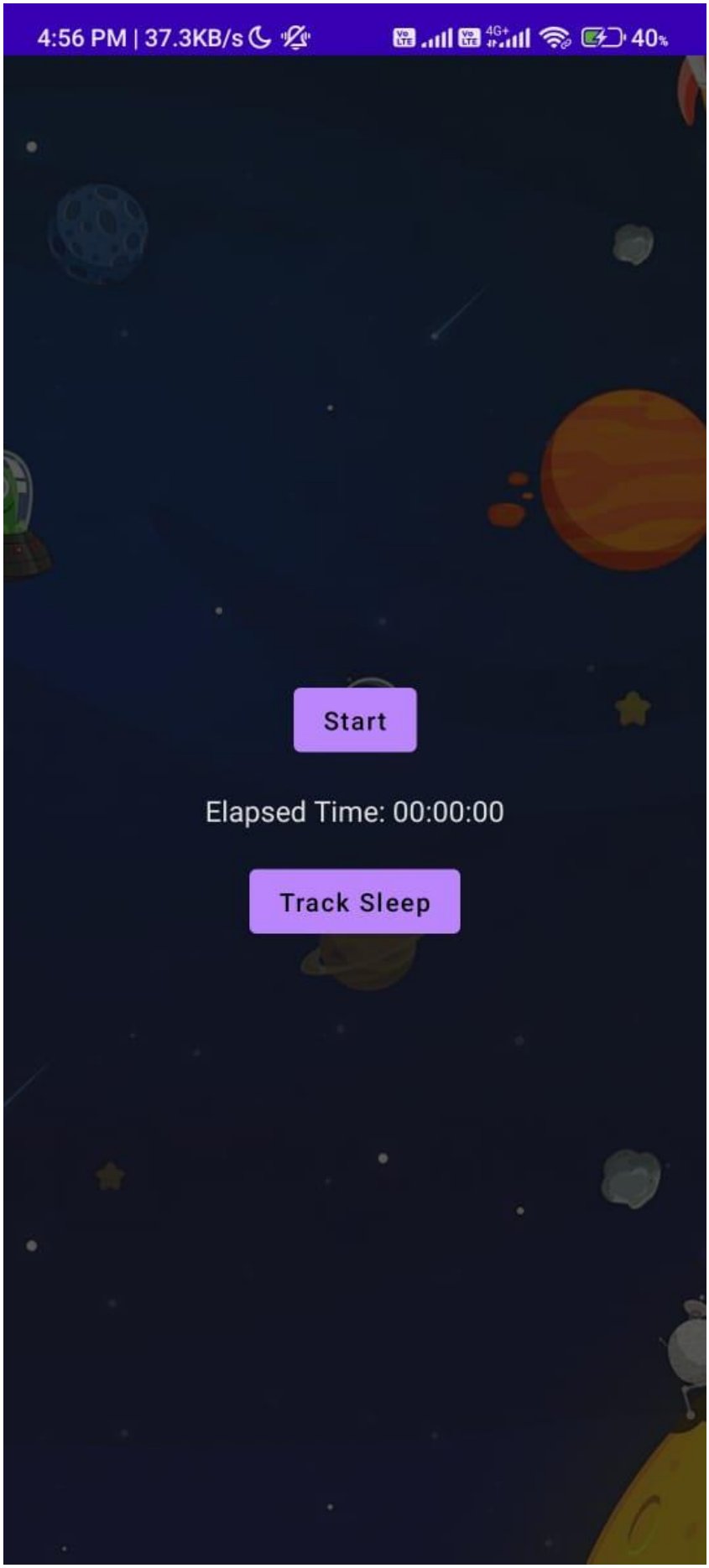
2.1 EMPATHY MAP

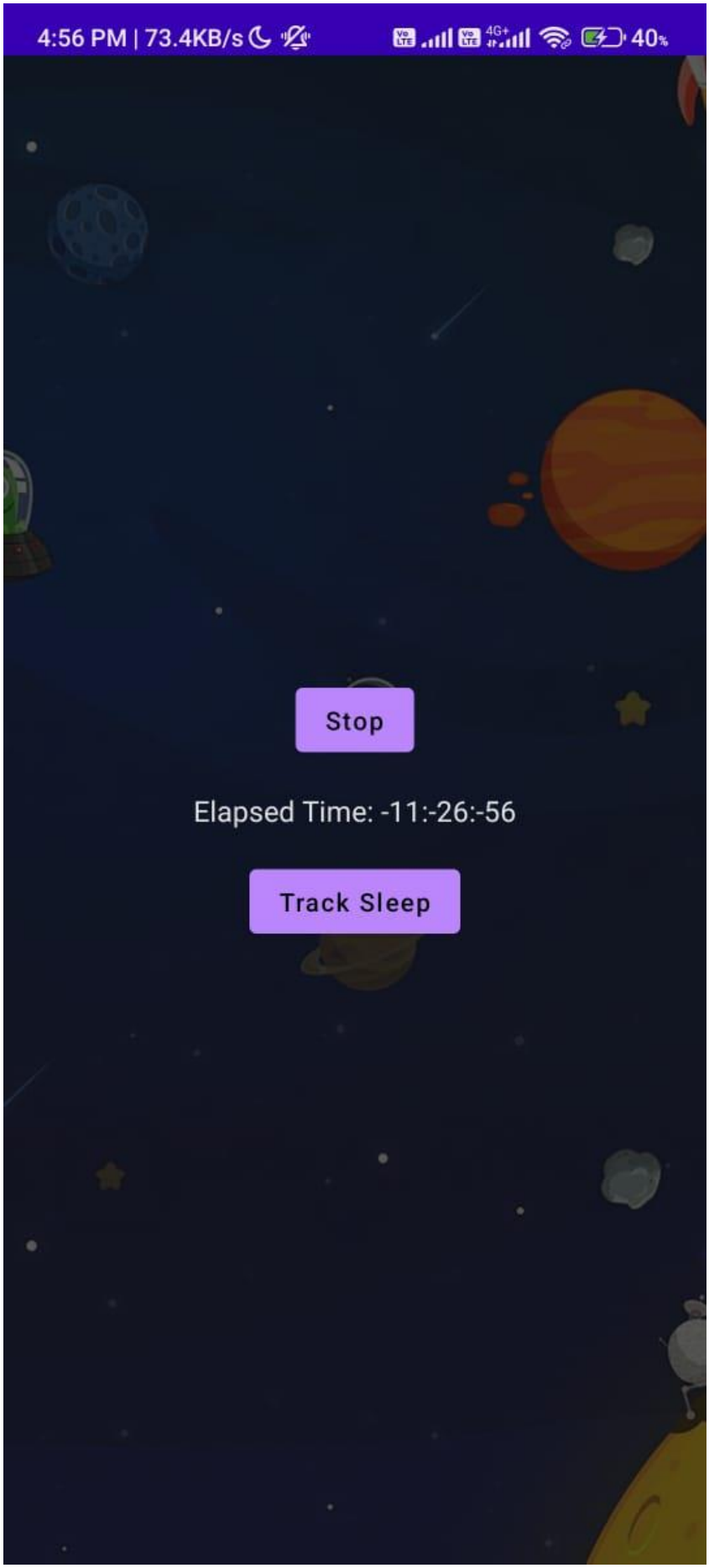
2.2 IDEATION AND BRAINSTORMING APP

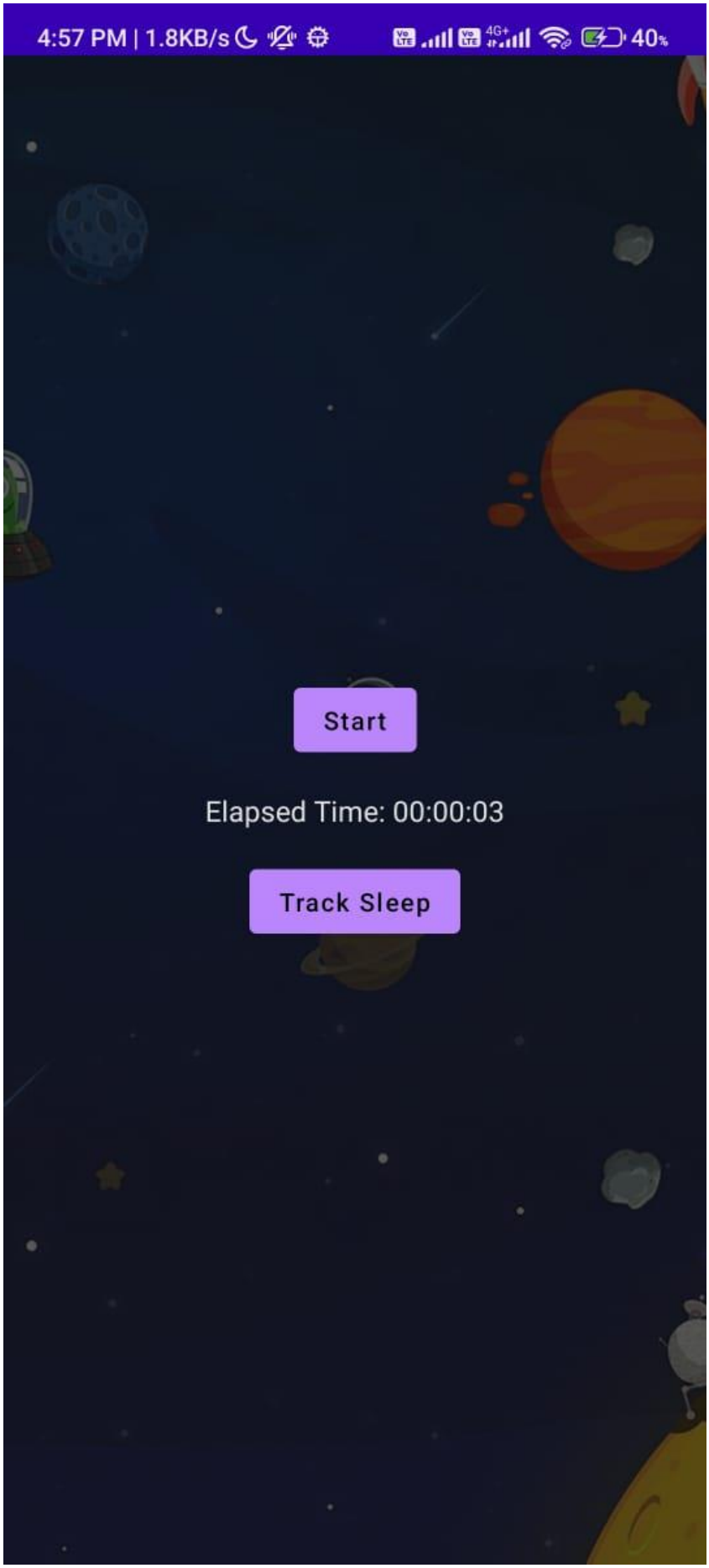
RESULT:

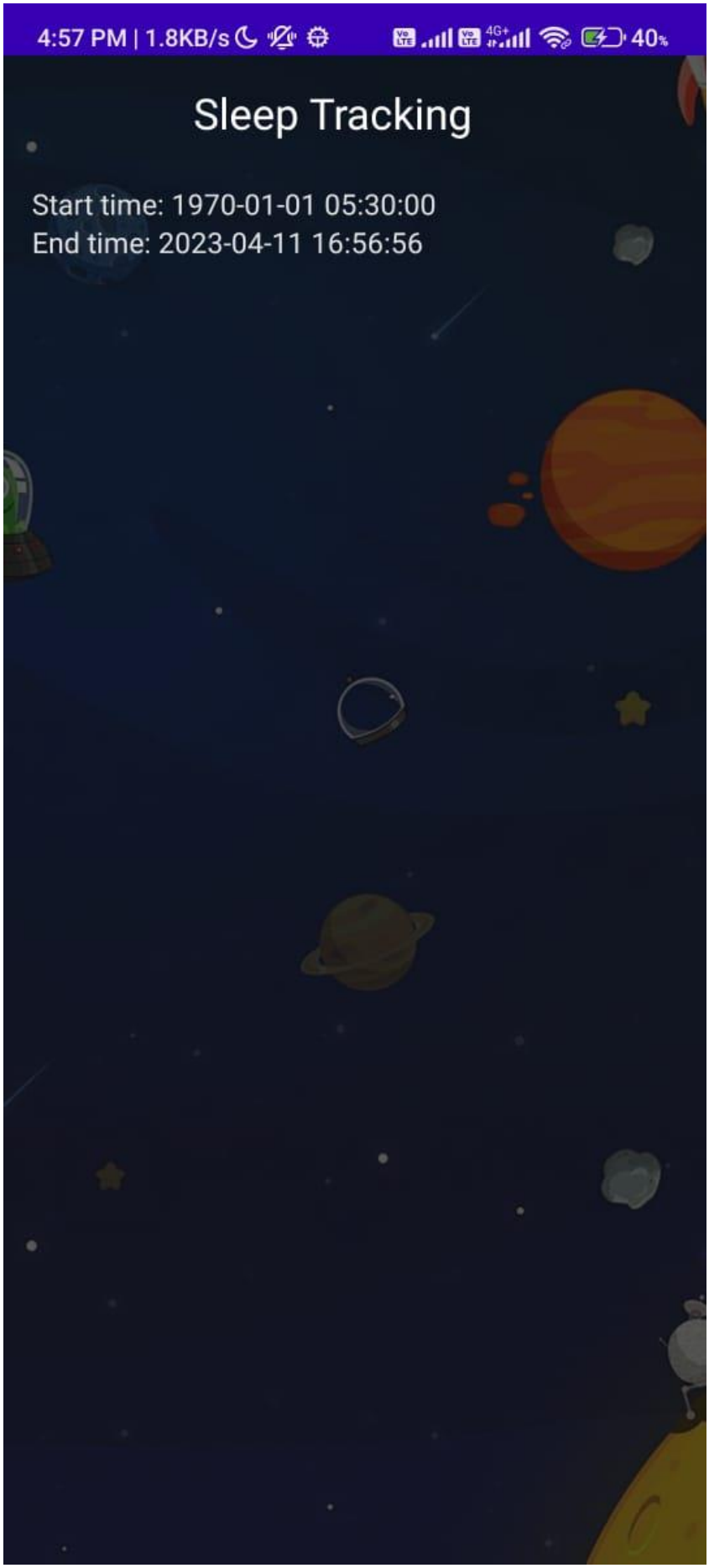

4:56 PM | 22.2KB/s
40%
Login
Username
Password
Login
Sign up
Forget password?


4:56 PM | 0.9KB/s
40%
Register
Username
Email
Password
Register
Have an account?
Log in


4:56 PM | 37.3KB/s
40%
Start
Elapsed Time: 00:00:00
Track Sleep


4:56 PM | 73.4KB/s
40%
Stop
Elapsed Time: -11:-26:-56
Track Sleep


4:57 PM | 1.8KB/s
40%
Start
Elapsed Time: 00:00:03
Track Sleep


4:57 PM | 1.8KB/s
40%
Sleep Tracking
Start time: 1970-01-01 05:30:00
End time: 2023-04-11 16:56:56

ADVANTAGE AND DISADVANTAGES:

Advantages of a Sleep Tracking App:

Awareness: One of the primary benefits of using a sleep tracking app is increased awareness of sleep patterns. Users can gain valuable insights into their sleep quality, duration, and efficiency, which can help them identify factors that may be affecting their sleep.

Personalization: Sleep tracking apps provide personalized recommendations based on an individual's sleep patterns. This can help users tailor their habits to achieve a better night's sleep.

Motivation: Sleep tracking apps can motivate users to adopt healthier sleep habits by setting goals, tracking progress, and providing positive feedback.

Improved sleep: By providing users with personalized recommendations and motivation to improve their sleep habits, sleep tracking apps can lead to better overall sleep quality and duration.

Disadvantages of a Sleep Tracking App:

Inaccuracy: Sleep tracking apps may not always be accurate. Factors such as movement, noise, and external disruptions can affect the readings provided by the app.

Anxiety: Some users may become anxious about their sleep patterns or become overly reliant on the app to track their sleep.

Technology reliance: Sleep tracking apps require a smartphone or wearable device to track sleep. Some users may become overly reliant on technology to monitor and improve their sleep, which can be problematic if they forget to charge their device or lose access to it.

Data privacy: Sleep tracking apps collect sensitive data such as sleep patterns, which could be at risk of being hacked or shared without the user's consent.

APPLICATIONS:

Health Management: Sleep tracking apps can help users manage their overall health and wellbeing by providing insights into their sleep quality. Good sleep is essential for maintaining physical and mental health, and tracking sleep patterns can help identify any underlying health issues that may be affecting sleep.

Performance Enhancement: Good sleep is also essential for optimal performance in sports, work, and other activities. Sleep tracking apps can help users identify factors that may be affecting their sleep, such as stress or caffeine consumption, and make adjustments to improve their sleep quality and overall performance.

6-CONCLUSION:

In conclusion, a sleep tracking app can be a useful tool for anyone looking to improve their sleep habits and achieve a better night's rest. By tracking various metrics such as sleep duration, efficiency, and quality, sleep tracking apps can provide valuable insights into an individual's sleep patterns and help identify potential areas for improvement.

While there are some disadvantages to using sleep tracking apps, such as inaccuracy and anxiety, the benefits of increased awareness, personalization, motivation, and improved sleep quality can outweigh the drawbacks. Sleep tracking apps can be used for a variety of applications, from health management to performance enhancement, and can help users achieve a better overall quality of life.

7-FUTURE SCOPE:

wearable devices like smartwatches to track their physical activity. In the future, sleep tracking apps could integrate with these devices to provide a more comprehensive picture of users' health and wellness. Wearable devices could also provide more accurate data about sleep quality, as they could track movement, heart rate, and other physiological indicators.

Artificial Intelligence (AI) algorithms: With the help of AI, sleep tracking apps could become more sophisticated in predicting and identifying sleep patterns. They could use machine learning algorithms

to analyze sleep data and provide personalized recommendations to improve sleep quality. For example, an app could suggest changes to a user's bedtime routine or offer tips to reduce stress and anxiety before bedtime.

8-APPENDIX:

A. SOURCE CODE

MainActivity.kt

```
package com.example.projectone

import android.content.Context
import android.content.Intent
import android.icu.text.SimpleDateFormat
import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.Image
import androidx.compose.foundation.layout.*
import androidx.compose.material.Button
import androidx.compose.material.MaterialTheme
import androidx.compose.material.Surface
import androidx.compose.material.Text
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.draw.alpha
import androidx.compose.ui.layout.ContentScale
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.unit.dp
import androidx.core.content.ContextCompat
import com.example.projectone.ui.theme.ProjectOneTheme
import java.util.*

class MainActivity : ComponentActivity() {

    private lateinit var databaseHelper: TimeLogDatabaseHelper

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        databaseHelper = TimeLogDatabaseHelper(this)
        databaseHelper.deleteAllData()
        setContent {
```

```
            ProjectOneTheme {
                // A surface container using the 'background' color from
the theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background
                ) {
                    MyScreen(this,databaseHelper)
                }
            }
        }
    }
}
@Composable
fun MyScreen(context: Context, databaseHelper: TimeLogDatabaseHelper) {
    var startTime by remember { mutableStateOf(0L) }
    var elapsedTime by remember { mutableStateOf(0L) }
    var isRunning by remember { mutableStateOf(false) }
    val imageModifier = Modifier
    Image(
        painterResource(id = R.drawable.sleeptracking),
        contentScale = ContentScale.FillHeight,
        contentDescription = "",
        modifier = imageModifier
            .alpha(0.3F),
    )

    Column(
        modifier = Modifier.fillMaxSize(),
        horizontalAlignment = Alignment.CenterHorizontally,
        verticalArrangement = Arrangement.Center
    ) {
        if (!isRunning) {
            Button(onClick = {
                startTime = System.currentTimeMillis()
                isRunning = true
            }) {
                Text("Start")
                //databaseHelper.addTimeLog(startTime)
            }
        } else {
            Button(onClick = {
                elapsedTime = System.currentTimeMillis()
                isRunning = false
            }) {
                Text("Stop")
                databaseHelper.addTimeLog(elapsedTime,startTime)
            }
        }
        Spacer(modifier = Modifier.height(16.dp))
        Text(text = "Elapsed Time: ${formatTime(elapsedTime - startTime)}")


        Spacer(modifier = Modifier.height(16.dp))
        Button(onClick = { context.startActivity(
            Intent(
                context,
                TrackActivity::class.java
            )
        ) }) {
            Text(text = "Track Sleep")
```

```
        }

    }

}

private fun startTrackActivity(context: Context) {
    val intent = Intent(context, TrackActivity::class.java)
    ContextCompat.startActivity(context, intent, null)
}
fun getCurrentDateTime(): String {
    val dateFormat = SimpleDateFormat("yyyy-MM-dd HH:mm:ss",
Locale.getDefault())
    val currentTime = System.currentTimeMillis()
    return dateFormat.format(Date(currentTime))
}

fun formatTime(timeInMillis: Long): String {
    val hours = (timeInMillis / (1000 * 60 * 60)) % 24
    val minutes = (timeInMillis / (1000 * 60)) % 60
    val seconds = (timeInMillis / 1000) % 60
    return String.format("%02d:%02d:%02d", hours, minutes, seconds)
}
```

LoginActivity.kt

```
package com.example.projectone

import android.content.Context
import android.content.Intent
import android.os.Bundle
import androidx.activity.ComponentActivity
import androidx.activity.compose.setContent
import androidx.compose.foundation.Image
import androidx.compose.foundation.layout.*
import androidx.compose.material.*
import androidx.compose.runtime.*
import androidx.compose.ui.Alignment
import androidx.compose.ui.Modifier
import androidx.compose.ui.draw.alpha
import androidx.compose.ui.graphics.Color
import androidx.compose.ui.layout.ContentScale
import androidx.compose.ui.res.painterResource
import androidx.compose.ui.text.font.FontFamily
import androidx.compose.ui.text.font.FontWeight
import androidx.compose.ui.unit.dp
import androidx.compose.ui.unit.sp
import androidx.core.content.ContextCompat
import com.example.projectone.ui.theme.ProjectOneTheme


class LoginActivity : ComponentActivity() {
    private lateinit var databaseHelper: UserDatabaseHelper
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        databaseHelper = UserDatabaseHelper(this)
```

```
        setContent {
            ProjectOneTheme {
                // A surface container using the 'background' color from
the theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background
                ) {
                    LoginScreen(this, databaseHelper)
                }
            }
        }
    }
}
@Composable
fun LoginScreen(context: Context, databaseHelper: UserDatabaseHelper) {
    var username by remember { mutableStateOf("") }
    var password by remember { mutableStateOf("") }
    var error by remember { mutableStateOf("") }
    val imageModifier = Modifier
    Image(
        painterResource(id = R.drawable.sleeptracking),
        contentScale = ContentScale.FillHeight,
        contentDescription = "",
        modifier = imageModifier
            .alpha(0.3F),
    )
    Column(
        modifier = Modifier.fillMaxSize(),
        horizontalAlignment = Alignment.CenterHorizontally,
        verticalArrangement = Arrangement.Center
    ) {

        Image(
            painter = painterResource(id = R.drawable.sleep),
            contentDescription = "",

            modifier = imageModifier
                .width(260.dp)
                .height(200.dp)
        )
        Text(
            fontSize = 36.sp,
            fontWeight = FontWeight.ExtraBold,
            fontFamily = FontFamily.Cursive,
            color = Color.White,
            text = "Login"
        )
        Spacer(modifier = Modifier.height(10.dp))

        TextField(
            value = username,
            onValueChange = { username = it },
            label = { Text("Username") },
            modifier = Modifier.padding(10.dp)
                .width(280.dp)
        )

        TextField(
            value = password,
            onValueChange = { password = it },
```

```
            label = { Text("Password") },
            modifier = Modifier.padding(10.dp)
                .width(280.dp)
        )

        if (error.isNotEmpty()) {
            Text(
                text = error,
                color = MaterialTheme.colors.error,
                modifier = Modifier.padding(vertical = 16.dp)
            )
        }

        Button(
            onClick = {
                if (username.isNotEmpty() && password.isNotEmpty()) {
                    val user = databaseHelper.getUserByUsername(username)
                    if (user != null && user.password == password) {
                        error = "Successfully log in"
                        context.startActivity(
                            Intent(
                                context,
                                MainActivity::class.java
                            )
                        )

                        //onLoginSuccess()
                    } else {
                        error = "Invalid username or password"
                    }
                } else {
                    error = "Please fill all fields"
                }
            },
            modifier = Modifier.padding(top = 16.dp)
        ) {
            Text(text = "Login")
        }
        Row {
            TextButton(onClick = {context.startActivity(
                Intent(
                    context,
                    MainActivity2::class.java
                )
            )}
            )
            { Text(color = Color.White,text = "Sign up") }
            TextButton(onClick = {
                /*startActivity(
                Intent(
                    applicationContext,
                    MainActivity2::class.java
                )
            )*/
            })

            {
                Spacer(modifier = Modifier.width(60.dp))
                Text(color = Color.White,text = "Forget password?")
            }
        }
```

```
    }
}
private fun startMainPage(context: Context) {
    val intent = Intent(context, MainActivity2::class.java)
    ContextCompat.startActivity(context, intent, null)
}
```