**Project Report**

1. **INTRODUCTION**
   1. **Project Overview**

Sleep tracking apps can be a helpful tool for improving your sleep quality. By tracking your sleep patterns over time, you can identify trends and patterns. This information can help you make changes to your sleep habits, such as going to bed and waking up at the same time each day, creating a relaxing bedtime routine, and avoiding caffeine and alcohol before bed.

* 1. **Project Overview**

Sleep tracking apps are designed to help you track your sleep patterns and identify areas where you can improve your sleep quality. They can help you:

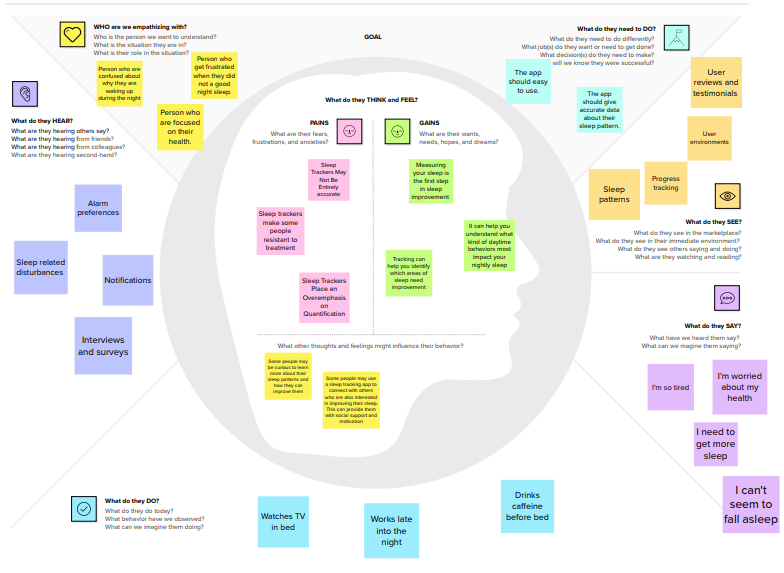
* Understand your sleep patterns
* Identify potential sleep problems
* Set and track sleep goals
* Improve your sleep habits

1. **LITERATURE SURVEY**
   1. **Existing Problem**

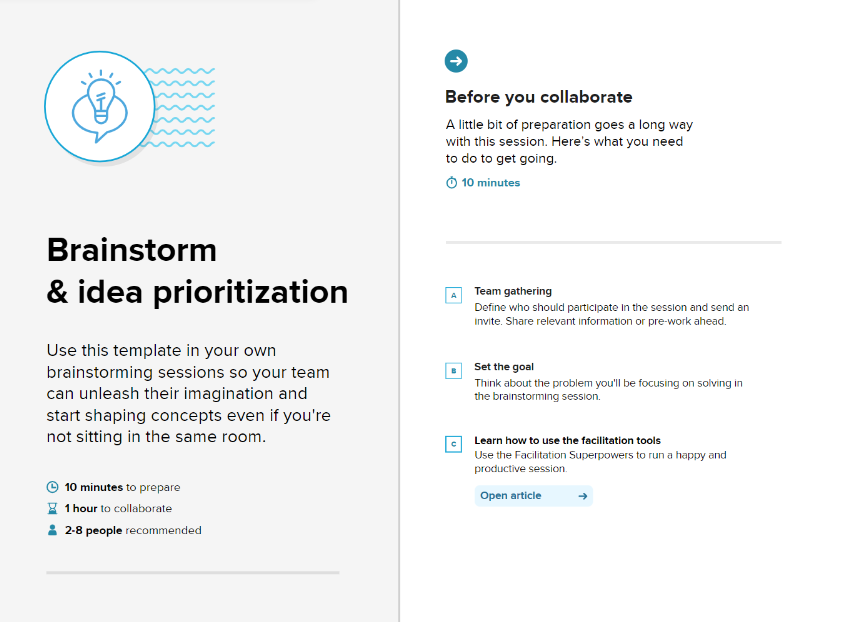
The existing problem with sleep tracking apps is that these apps rely on various methods like smartphone sensors, wearable watches, sound analysis, etc. to track sleep parameters. However, these methods can be susceptible to noise and interference, leading to inaccurate sleep data.

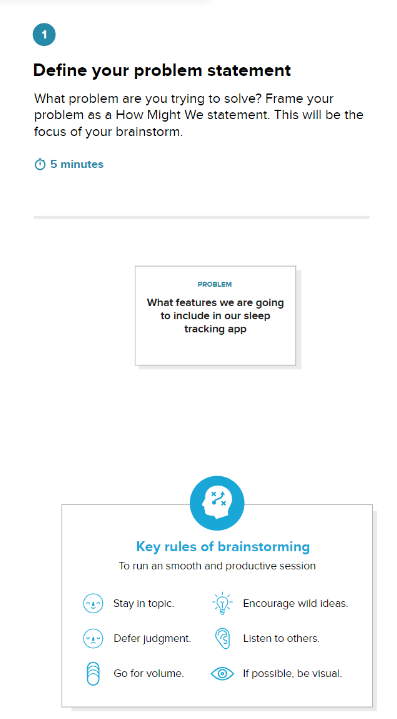
1. **IDEATION & PROPOSED SOLUTION**

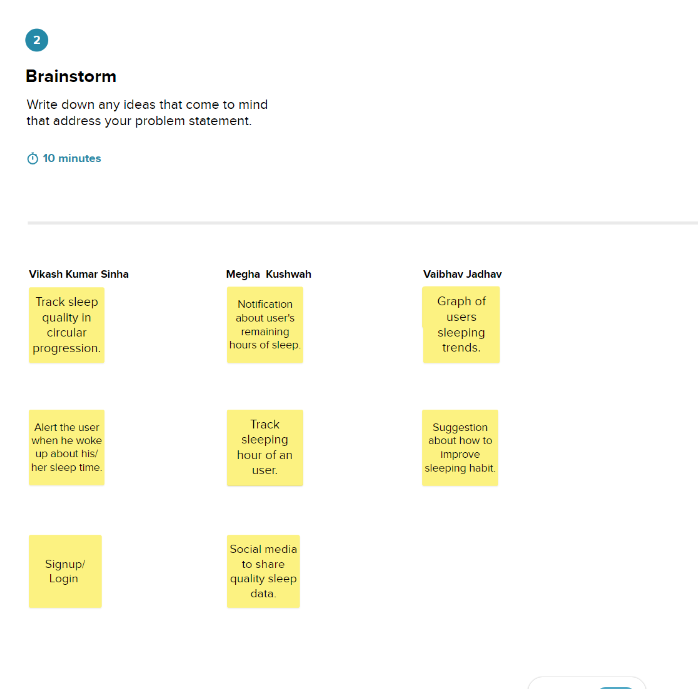
**3.1 Empathy map canvas**

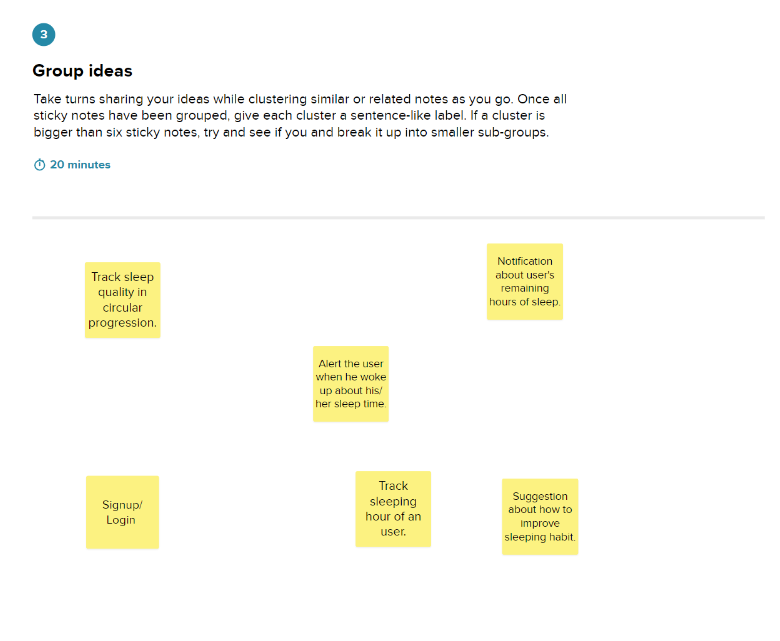


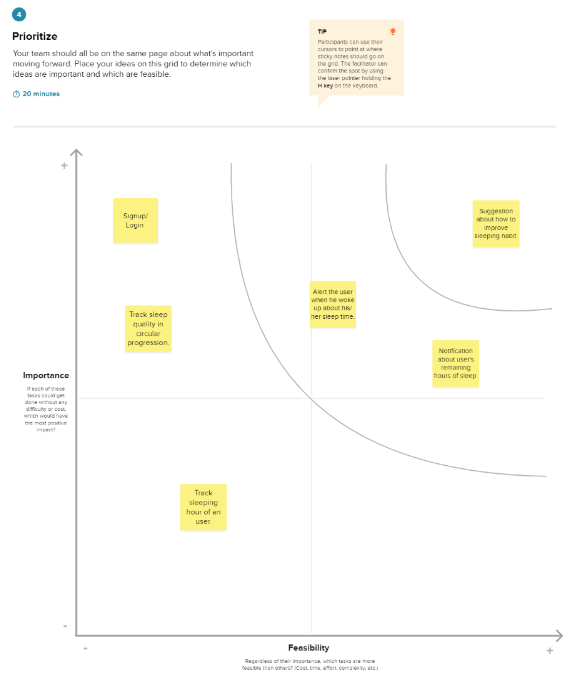
**3.2 Ideation & Brainstorm**











1. **REQUIREMENT ANALYSIS**
   1. **Functional Requirements**

Functional requirements for a sleep tracking app should encompass a range of features that enable users to effectively monitor and improve their sleep patterns.

* Track sleep time
* Sleep quality checking
* Sleep analysis
  1. **Non-Functional Requirements**

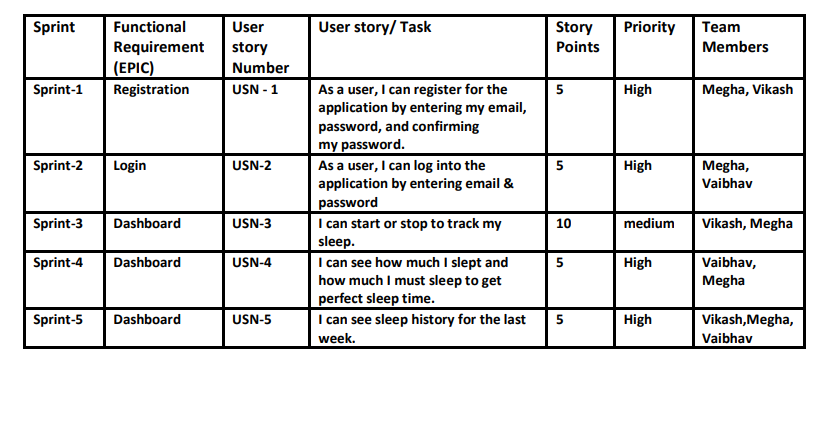
Non-functional requirements for a sleep tracking app outline the performance, reliability, and usability aspects of the app, ensuring it meets user expectations and functions effectively.

* Ease of Use
* Help and support
* Error Handling

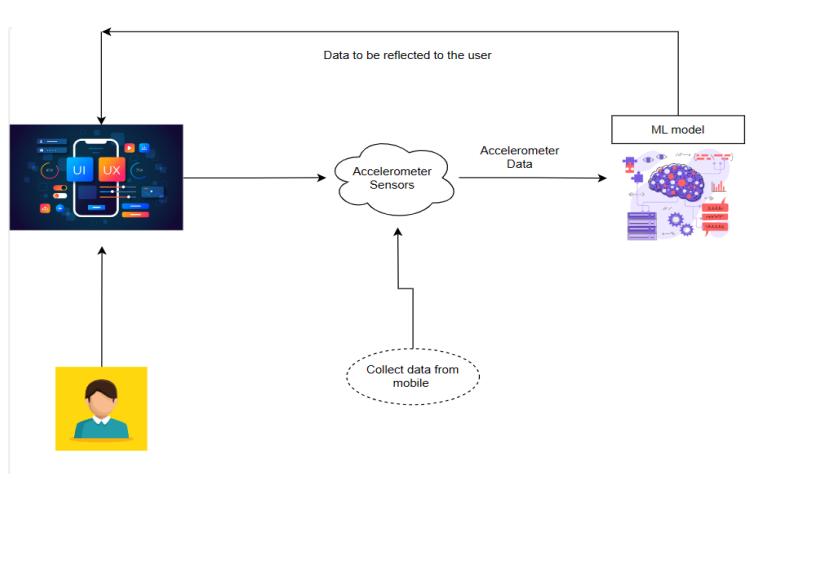
1. **PROJECT DESIGN**

**5.1 Data flow diagram and user stories**



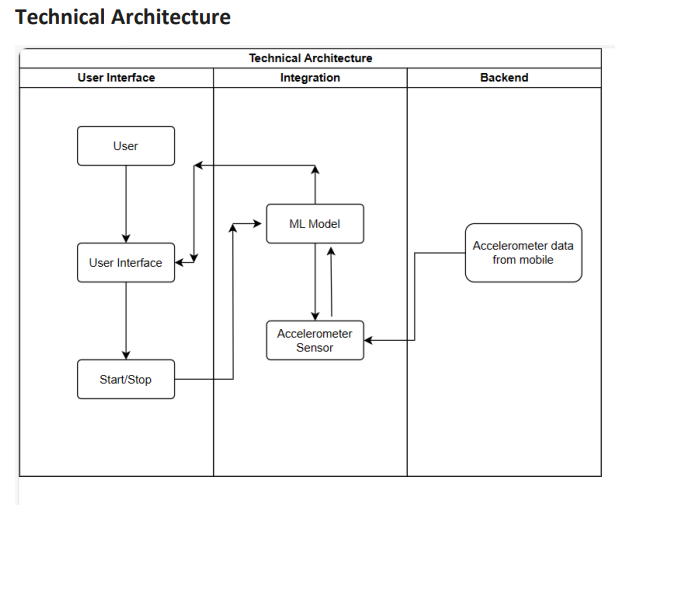


**5.2 Solution Architecture**

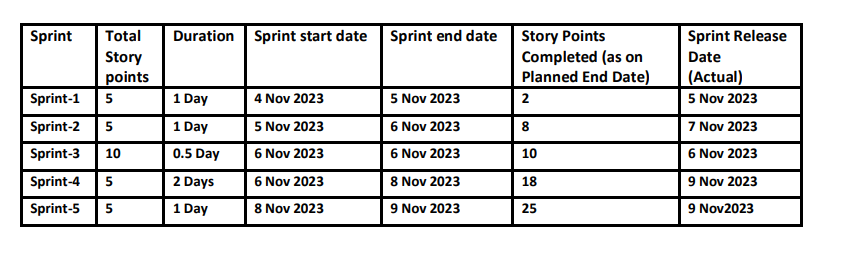


1. **PROJECT PLANNING AND SCHEDULING**

**6.1 Technical Architecture**



* 1. **Sprint planning and estimation**



1. **CODING & SOLUTIONING**

**7.1 Feature 1: User Registration**

The Sleep Tracking App features a user registration process implemented in the RegistrationActivity. Users can provide their name, email, and password for registration. The app ensures all fields are filled in before allowing registration. Upon successful registration, users are redirected to the main activity (MainActivity).

**7.2 Feature 1: User Login**

The app includes a user login feature in the LoginActivity, allowing users to log in by entering their username and password. The app validates input, requiring non-empty fields for login. Successful login redirects users to the registration activity (RegistrationActivity), which may need adjustment for typical behavior. The app also provides a "Forget Password" feature for initiating password recovery.

**7.3 Feature 3: Forget Password**

This feature allows users to recover their password. Clicking on the "Forget Password" TextView navigates users to the ForgetPasswordActivity for further recovery steps.

**7.4 Feature 4: Track Sleep Session**

The TrackSleepActivity enables users to track their sleep sessions. Sleep session details, including start and end times, are passed through intents. The activity formats and displays these times in the startTimeTextView and endTimeTextView respectively.

**7.5 Feature 5: Average Sleep Duration Calculation**

The app could benefit from a feature to calculate the average sleep duration over multiple sessions. This would provide users with insights into their overall sleep patterns.

1. **PERFORMANCE TESTING**

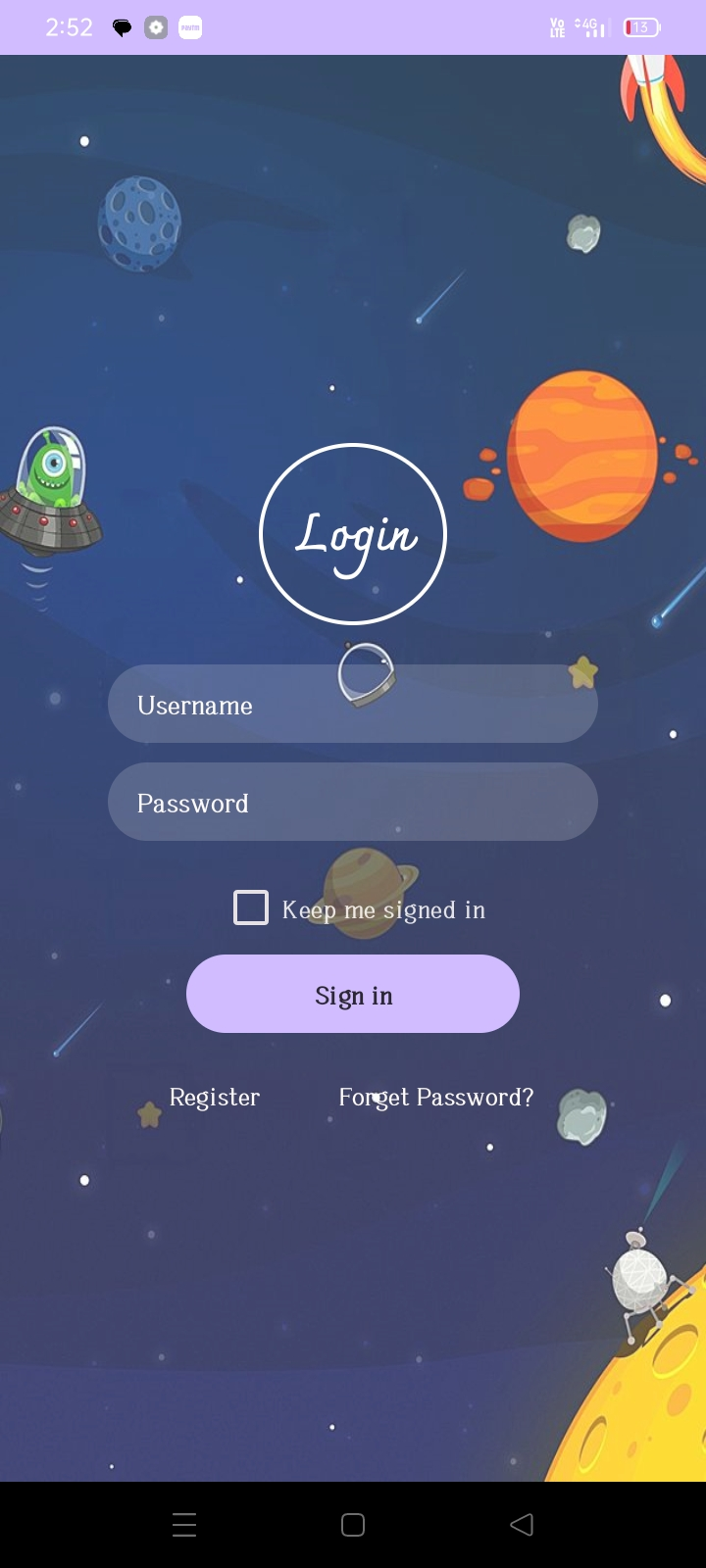
**8.1 Performance Metrics**

1. **RESULTS**

**9.1 Output Screenshots**

**Final Output of the Application :**

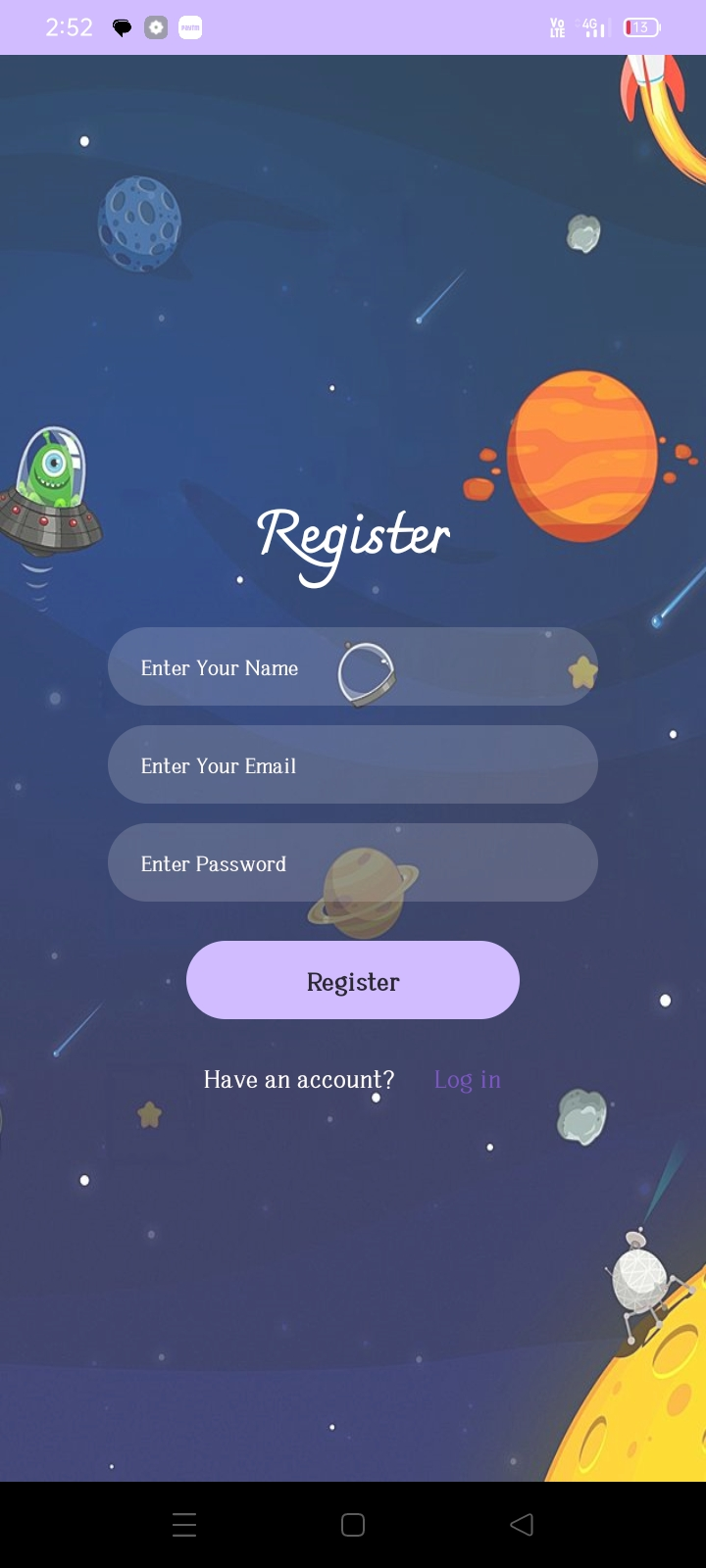
**Login Page :**



**Reset Password Page:**



**Registration Page :**



**Main Page:**



**Track Sleep Page:**



1. **ADVANTAGES AND DISADVANTAGES**

**10.1 ADVANTAGES:**

* Access to Sleep Information
* Figure Out Underlying Problems
* Learn How Interrupted Your Sleep Is
* Tracking can help you identify which areas of sleep need improvement
* Monitor Heart Rate
* Help Waking up

**10.2 DISADVANTAGES**

* Time Expenditure
* Overthinking
* Limited Feedback and Reliability
* The temptation to Check During the Night

1. **CONCLUSION**

The Sleep Tracking App project aims to address the need for better sleep quality by offering a user-friendly solution. The project identified existing issues with sleep tracking apps, proposed a comprehensive set of functional and non-functional requirements, and designed key features like user registration, login, and sleep session tracking. The results include output screenshots showcasing the app's interface and functionality.

While the app provides advantages such as easy access to sleep information and heart rate monitoring, it comes with potential drawbacks, including time expenditure and the temptation to check the app during the night. The success of the project lies in its ability to empower users to improve their sleep habits, acknowledging the balance between benefits and challenges in the pursuit of better overall well-being.

1. **FUTURE SCOPE**
2. **Advanced Sleep Analysis Algorithms:**

Invest in research and development to improve the accuracy of sleep analysis. Implement advanced algorithms that can better interpret sleep patterns, taking into account factors such as sleep cycles, stages, and interruptions.

1. **Integration with Health Data:**

Collaborate with health data platforms or APIs to integrate additional health-related information. This could include factors like physical activity, nutrition, or stress levels, providing users with a comprehensive view of their overall well-being.

1. **Community and Social Features:**

Introduce a community aspect to the app where users can share their sleep improvement journeys, tips, and challenges. Adding social features can enhance user engagement and create a supportive environment.

1. **Smart Home Integration:**

Explore integration with smart home devices to create an optimal sleep environment. This could involve connecting with smart lights, thermostats, or other devices that contribute to creating a conducive atmosphere for better sleep.

1. **Real-time Sleep Tracking:**

Develop real-time sleep tracking capabilities, allowing users to monitor their sleep patterns as they happen. This feature can provide instant feedback and may be particularly useful for those trying to implement immediate changes to their sleep habits.

**APPENDIX**

**Source Code**

1. **Main Activity**

package com.example.sleep\_tracking\_app  
  
import android.content.Intent  
import android.os.Bundle  
import android.os.Handler  
import android.os.Looper  
import android.widget.Button  
import android.widget.TextView  
import androidx.appcompat.app.AppCompatActivity  
import com.google.firebase.FirebaseApp  
import java.text.SimpleDateFormat  
import java.util.Date  
import java.util.Locale  
  
class MainActivity : AppCompatActivity() {

private lateinit var startSleepTrackingButton: Button  
 private lateinit var elapsedTimeTextView: TextView  
 private lateinit var trackSleepButton: Button  
  
 private val handler = Handler(Looper.getMainLooper())  
 private var startTimeMillis = 0L  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
  
 FirebaseApp.initializeApp(this)  
  
 setContentView(R.layout.*activity\_main*)  
  
 startSleepTrackingButton = findViewById(R.id.*startSleepTrackingButton*)  
 elapsedTimeTextView = findViewById(R.id.*elapsedTimeTextView*)  
 trackSleepButton = findViewById(R.id.*trackSleepButton*)  
  
 startSleepTrackingButton.setOnClickListener **{**  
startTimeMillis = System.currentTimeMillis()  
 handler.postDelayed(updateElapsedTime, 1000)  
 **}**  
  
trackSleepButton.setOnClickListener **{**  
val intent = Intent(this, TrackSleepActivity::class.*java*)  
  
 // Set start and end times as extras  
 val currentTimeMillis = System.currentTimeMillis()  
 val endTimeMillis = currentTimeMillis + (8 \* 60 \* 60 \* 1000) // Assuming 8 hours of sleep  
  
 intent.putExtra("startTime", currentTimeMillis)  
 intent.putExtra("endTime", endTimeMillis)  
  
 startActivity(intent)  
 **}**  
}  
  
 private val updateElapsedTime = object : Runnable {  
 override fun run() {  
 val currentTimeMillis = System.currentTimeMillis()  
 val elapsedTimeMillis = currentTimeMillis - startTimeMillis  
  
 val hours = (elapsedTimeMillis / (1000 \* 60 \* 60)) % 24  
 val minutes = (elapsedTimeMillis / (1000 \* 60)) % 60  
 val seconds = (elapsedTimeMillis / 1000) % 60  
  
 val formattedTime = String.*format*("%02d:%02d:%02d", hours, minutes, seconds)  
  
 elapsedTimeTextView.*text* = "Elapsed Time: $formattedTime"  
  
 handler.postDelayed(this, 1000)  
 }  
 }  
  
 override fun onDestroy() {  
 super.onDestroy()  
 handler.removeCallbacks(updateElapsedTime)  
 }  
}

1. **Login Activity**

package com.example.sleep\_tracking\_app  
  
import android.content.Intent  
import android.os.Bundle  
import android.widget.Button  
import android.widget.EditText  
import android.widget.TextView  
import android.widget.Toast  
import androidx.appcompat.app.AppCompatActivity  
  
class LoginActivity : AppCompatActivity() {  
  
 private lateinit var usernameEditText: EditText  
 private lateinit var passwordEditText: EditText  
 private lateinit var signInButton: Button  
 private lateinit var forgetPasswordTextView: TextView  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_login*)  
  
 usernameEditText = findViewById(R.id.*usernameEditText*)  
 passwordEditText = findViewById(R.id.*passwordEditText*)  
 signInButton = findViewById(R.id.*signInButton*)  
 forgetPasswordTextView = findViewById(R.id.*forgetPasswordTextView*)  
  
 signInButton.setOnClickListener **{**  
val username = usernameEditText.*text*.toString()  
 val password = passwordEditText.*text*.toString()  
  
 if (username.*isNotEmpty*() && password.*isNotEmpty*()) {  
 // Valid username and password, open RegistrationActivity  
 val intent = Intent(this, RegistrationActivity::class.*java*)  
 startActivity(intent)  
 } else {  
 // Invalid username or password  
 Toast.makeText(*baseContext*, "Could not sign in. Please try again.", Toast.*LENGTH\_SHORT*).show()  
 }  
 **}**  
  
forgetPasswordTextView.setOnClickListener **{**  
val intent = Intent(this, ForgetPasswordActivity::class.*java*)  
 startActivity(intent)  
 **}**  
}  
}

1. **Register Activity**

package com.example.sleep\_tracking\_app  
  
import android.content.Intent  
import android.os.Bundle  
import android.widget.Button  
import android.widget.EditText  
import android.widget.Toast  
import androidx.appcompat.app.AppCompatActivity  
  
class RegistrationActivity : AppCompatActivity() {  
  
 private lateinit var nameEditText: EditText  
 private lateinit var emailEditText: EditText  
 private lateinit var passwordEditText: EditText  
 private lateinit var registerButton: Button  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_registration*)  
  
 nameEditText = findViewById(R.id.*nameEditText*)  
 emailEditText = findViewById(R.id.*emailEditText*)  
 passwordEditText = findViewById(R.id.*passwordEditText*)  
 registerButton = findViewById(R.id.*registerButton*)  
  
 registerButton.setOnClickListener **{**  
val name = nameEditText.*text*.toString()  
 val email = emailEditText.*text*.toString()  
 val password = passwordEditText.*text*.toString()  
  
 if (name.*isNotEmpty*() && email.*isNotEmpty*() && password.*isNotEmpty*()) {  
 // Valid input, open MainActivity  
 val intent = Intent(this, MainActivity::class.*java*)  
 startActivity(intent)  
 finish()  
 } else {  
 // Invalid input  
 Toast.makeText(*baseContext*, "Please fill in all fields.", Toast.*LENGTH\_SHORT*).show()  
 }  
 **}**  
}S  
}

1. **ForgotPassword Activity**

package com.example.sleep\_tracking\_app  
  
import android.os.Bundle  
import android.widget.Button  
import android.widget.EditText  
import android.widget.Toast  
import androidx.appcompat.app.AppCompatActivity  
import com.google.firebase.auth.FirebaseAuth  
  
class ForgetPasswordActivity : AppCompatActivity() {  
  
 private lateinit var emailEditText: EditText  
 private lateinit var resetButton: Button  
  
 private lateinit var auth: FirebaseAuth  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_forget\_password*)  
  
 auth = FirebaseAuth.getInstance()  
  
 emailEditText = findViewById(R.id.*emailEditText*)  
 resetButton = findViewById(R.id.*resetButton*)  
  
 resetButton.setOnClickListener **{**  
val email = emailEditText.*text*.toString()  
  
 auth.sendPasswordResetEmail(email)  
 .addOnCompleteListener **{** task **->**  
if (task.*isSuccessful*) {  
 // Password reset email sent successfully  
 Toast.makeText(this, "Password reset email sent!", Toast.*LENGTH\_SHORT*).show()  
 finish()  
 } else {  
 // If sending email fails, display a message to the user.  
 Toast.makeText(this, "Password reset failed. Please try again.", Toast.*LENGTH\_SHORT*).show()  
 }  
 **}**  
 **}**  
}  
}

1. **TrackSleep Activity**

package com.example.sleep\_tracking\_app  
  
import android.os.Bundle  
import android.widget.TextView  
import androidx.appcompat.app.AppCompatActivity  
import java.text.SimpleDateFormat  
import java.util.Date  
import java.util.Locale  
  
class TrackSleepActivity : AppCompatActivity() {  
  
 private lateinit var startTimeTextView: TextView  
 private lateinit var endTimeTextView: TextView  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_track\_sleep*)  
  
 startTimeTextView = findViewById(R.id.*startTimeTextView*)  
 endTimeTextView = findViewById(R.id.*endTimeTextView*)  
  
 val startTimeMillis = *intent*.getLongExtra("startTime", 0)  
 val endTimeMillis = *intent*.getLongExtra("endTime", 0)  
  
 val startTime = formatTime(startTimeMillis)  
 val endTime = formatTime(endTimeMillis)  
  
 startTimeTextView.*text* = "Start Time: $startTime"  
 endTimeTextView.*text* = "End Time: $endTime"  
 }  
  
 private fun formatTime(timeMillis: Long): String {  
 val dateFormat = SimpleDateFormat("dd-MM-yyyy", Locale.getDefault())  
 val timeFormat = SimpleDateFormat("HH:mm:ss", Locale.getDefault())  
  
 val date = Date(timeMillis)  
 val formattedDate = dateFormat.format(date)  
 val formattedTime = timeFormat.format(date)  
  
 return "$formattedDate $formattedTime"  
 }  
}

1. **Manifest.xml**

<?xml version="1.0" encoding="utf-8"?>  
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 package="com.example.sleep\_tracking\_app">  
  
 <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:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 [android:theme="@style/Theme.Sleep\_tracking\_app](mailto:android:theme="@style/Theme.Sleep_tracking_app)"  
 tools:targetApi="31">  
  
 <activity  
 android:name=".ForgetPasswordActivity"  
 android:exported="false" />  
  
 <activity  
 android:name=".TrackSleepActivity"  
 android:exported="false" />  
  
 <activity  
 android:name=".LoginActivity"  
 android:exported="true">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
 </activity>  
  
 <activity  
 android:name=".RegistrationActivity"  
 android:exported="false" />  
  
 <activity  
 android:name=".MainActivity"  
 android:exported="false" />  
  
 <meta-data  
 android:name="preloaded\_fonts"  
 android:resource="@array/preloaded\_fonts" />  
 </application>  
  
</manifest>

1. **Colors.xml**

<?xml version="1.0" encoding="utf-8"?>  
<resources>  
 <color name="black">#FF000000</color>  
 <color name="white">#FFFFFFFF</color>  
</resources>

1. **String.xml**

<resources>  
 <string name="app\_name">Sleep\_tracking\_app</string>  
</resources>

**Git Hub Link: https://github.com/vikash-kumar-sinha/Sleep\_tracking\_app**

**Project Demo Link: https://drive.google.com/file/d/1kXlQBAN1un-X6SWnjTTzz7bOiCNcXB8u/view?usp=sharing**