A SLEEP TRACKING APP FOR BETTER NIGHT'S REST

1.INTRODUCTION

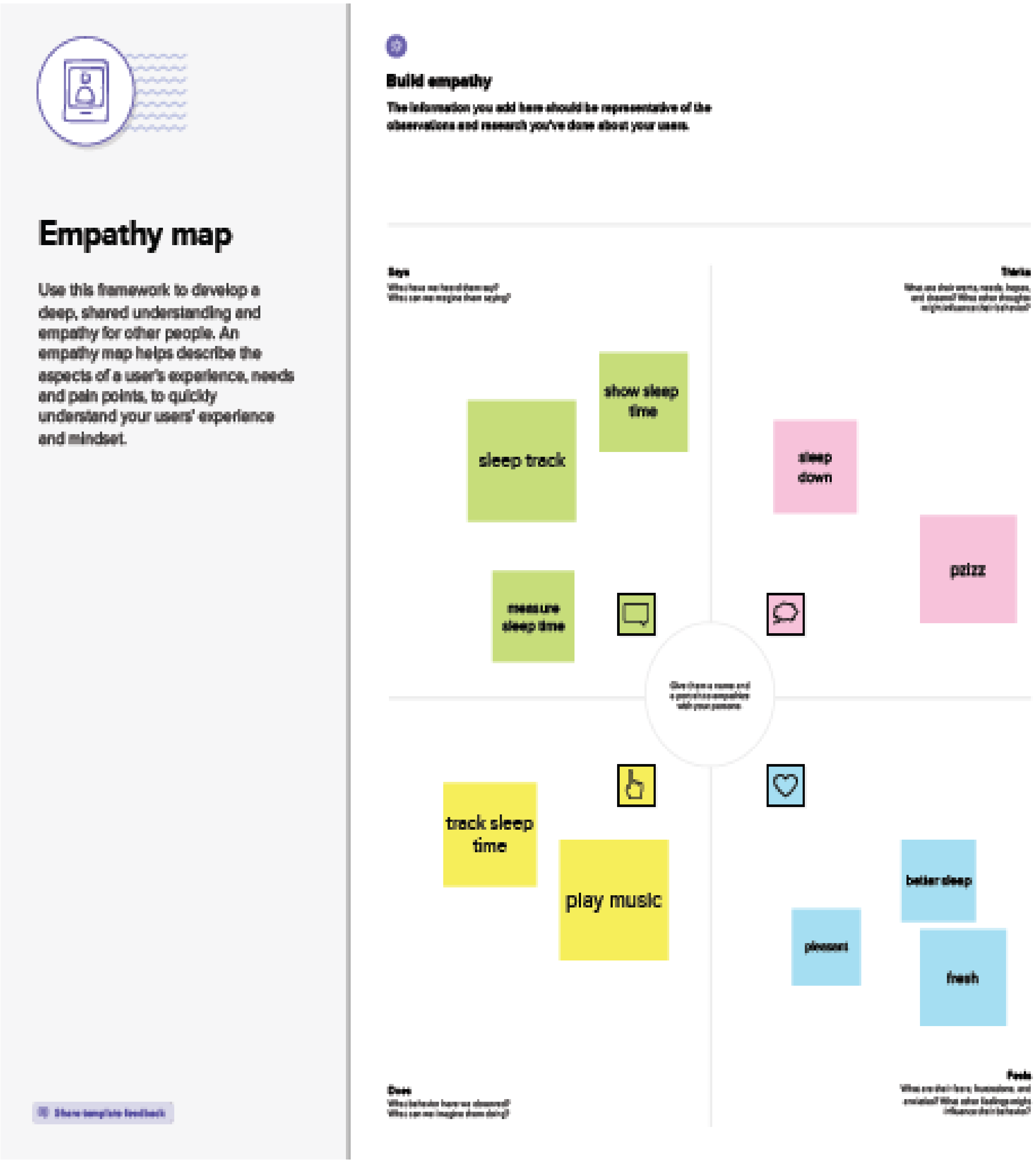
* The vast growth of advancements in Technology has pushed more and more people to incorporate them into their daily life. An industry that has faced a significant disruption due to this is the health and wellness industry. As smartphones are now much more than a phone and wearable technologies reinforce their use, healthcare-related mobile apps are becoming rapidly popular.

*Healthcare companies and wellness organizations one by one, adapt to this change and try to provide a variety of health-related services to the users of smartphones, tablets, or PCs.

*Irregular sleeping patterns are a common problem. This web app will fulfill the user's needs in tracking their sleeping patterns, including duration and timings. You will create a web app to track three parameters: sleep time, wake up time, and sleep duration. Users can add, edit, or remove any sleep entries.

2.PROBLEM DEFINITION & DESIGN THINKING

2.1 EMPATHY MAP:



Empathy map

Use this framework to develop a deep, shared understanding and empathy for other people. An empathy map helps describe the aspects of a user's experience, needs and pain points, to quickly understand your users' experience and mindset.

Build empathy

The information you add here should be representative of the observations and research you've done about your users.

Says
What have we heard them say?
What can we imagine them saying?

sleep track

show sleep time

measure sleep time

Thinks
What are their wants, needs, hopes, and dreams? What other thoughts might influence their behavior?

sleep down

pzizz

[illegible]

Does
What behavior have we observed?
What can we imagine them doing?

track sleep time

play music

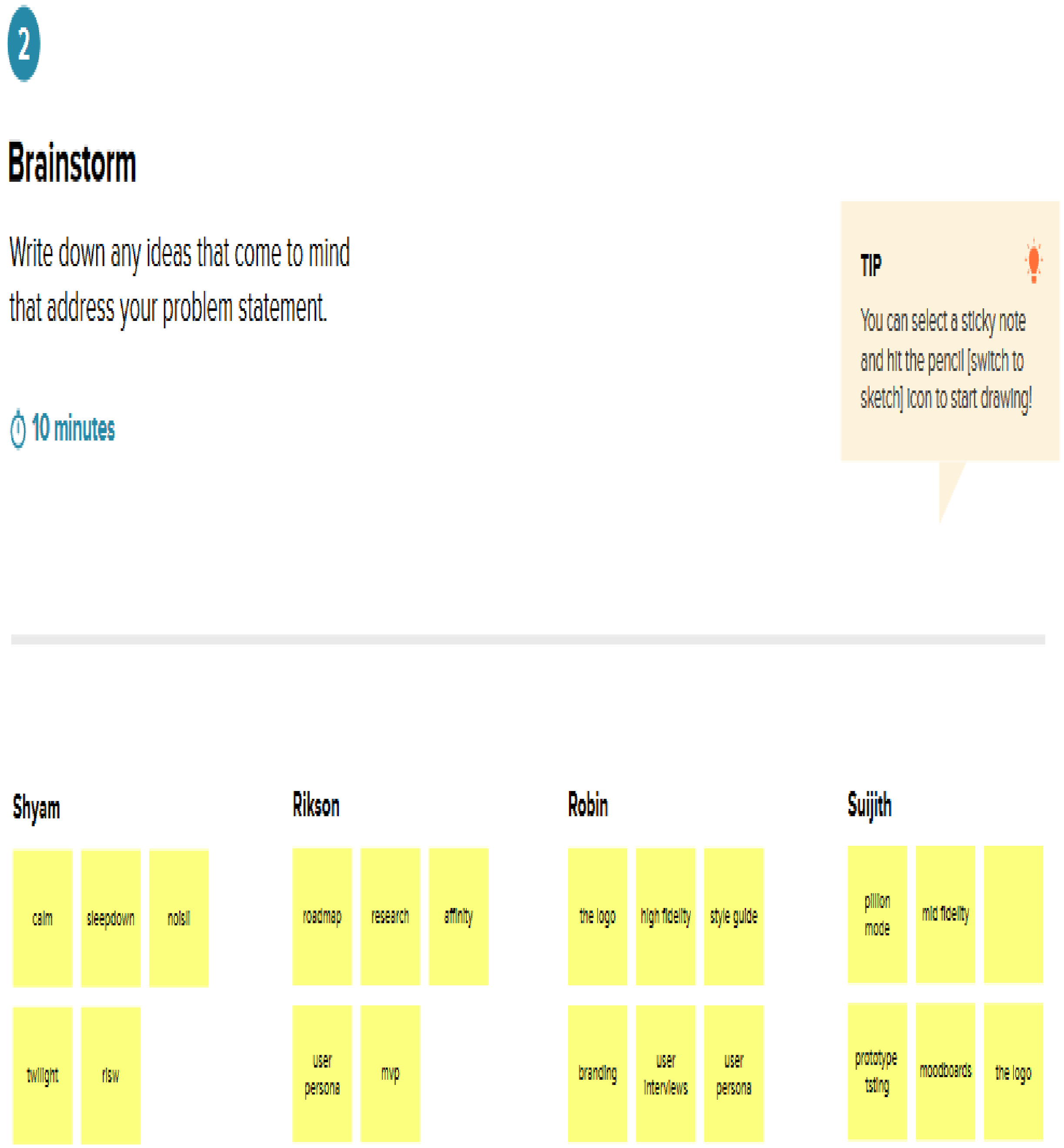
Feels
What are their fears, frustrations, and anxieties? What other feelings might influence their behavior?

better sleep

pleasant

fresh

2.2 IDEATION & BRAINSTORMING MAPS:



2

Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Shyam

calm
sleepdown
noisli
twilight
risw

Rikson

roadmap
research
affinity
user persona
mvp

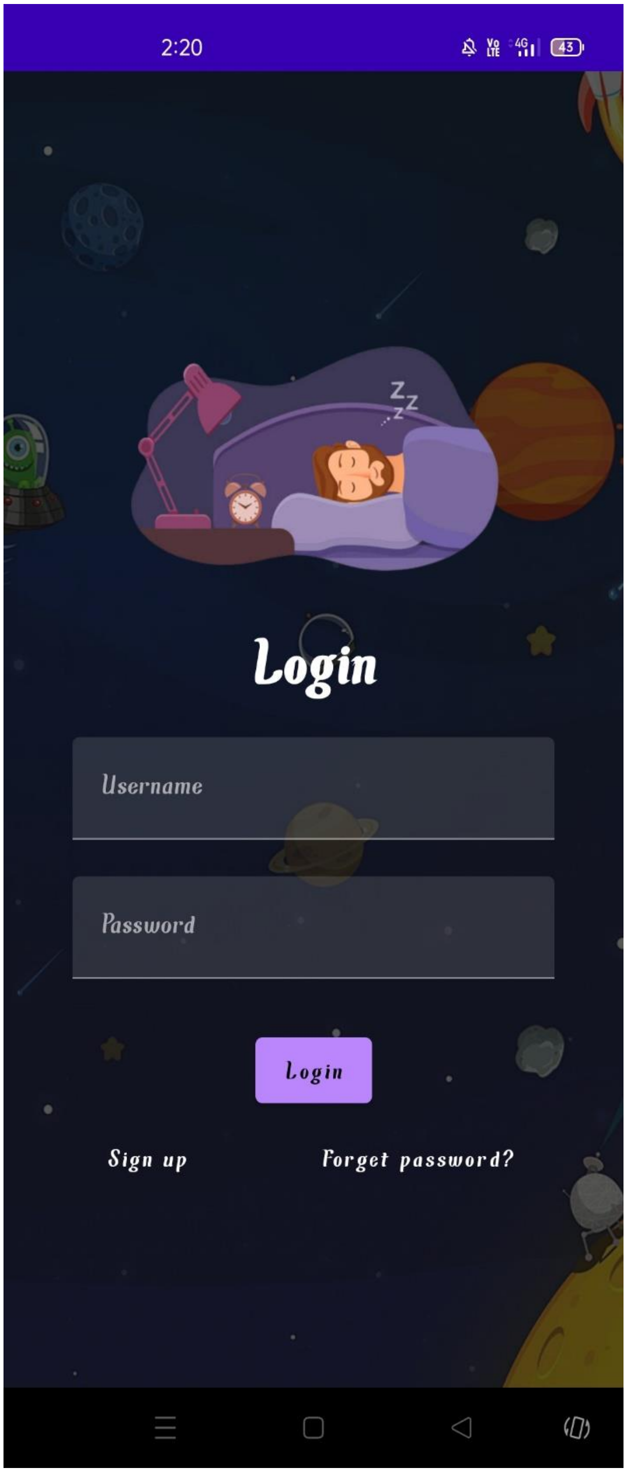
Robin

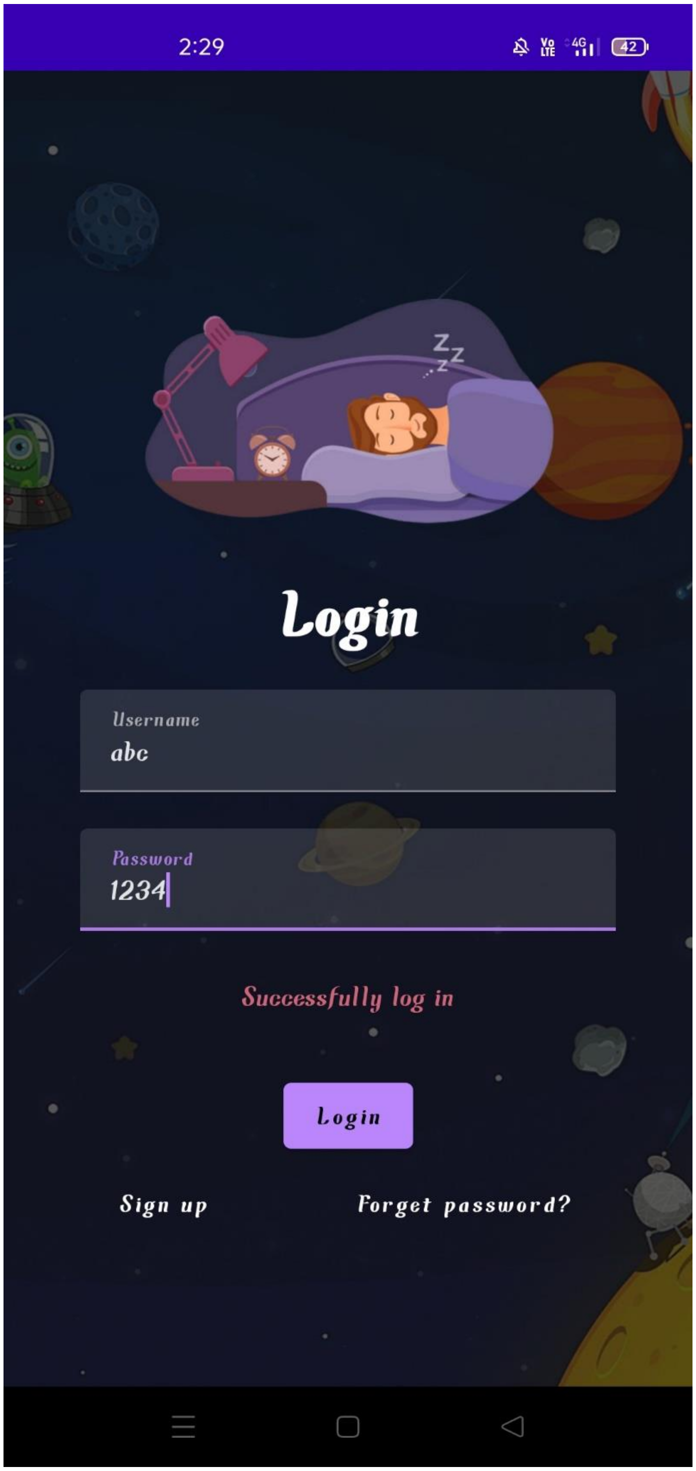
the logo
high fidelity
style guide
branding
user interviews
user persona

Suijith

pillion mode
mid fidelity
prototype tsting
moodboards
the logo

3.RESULT:

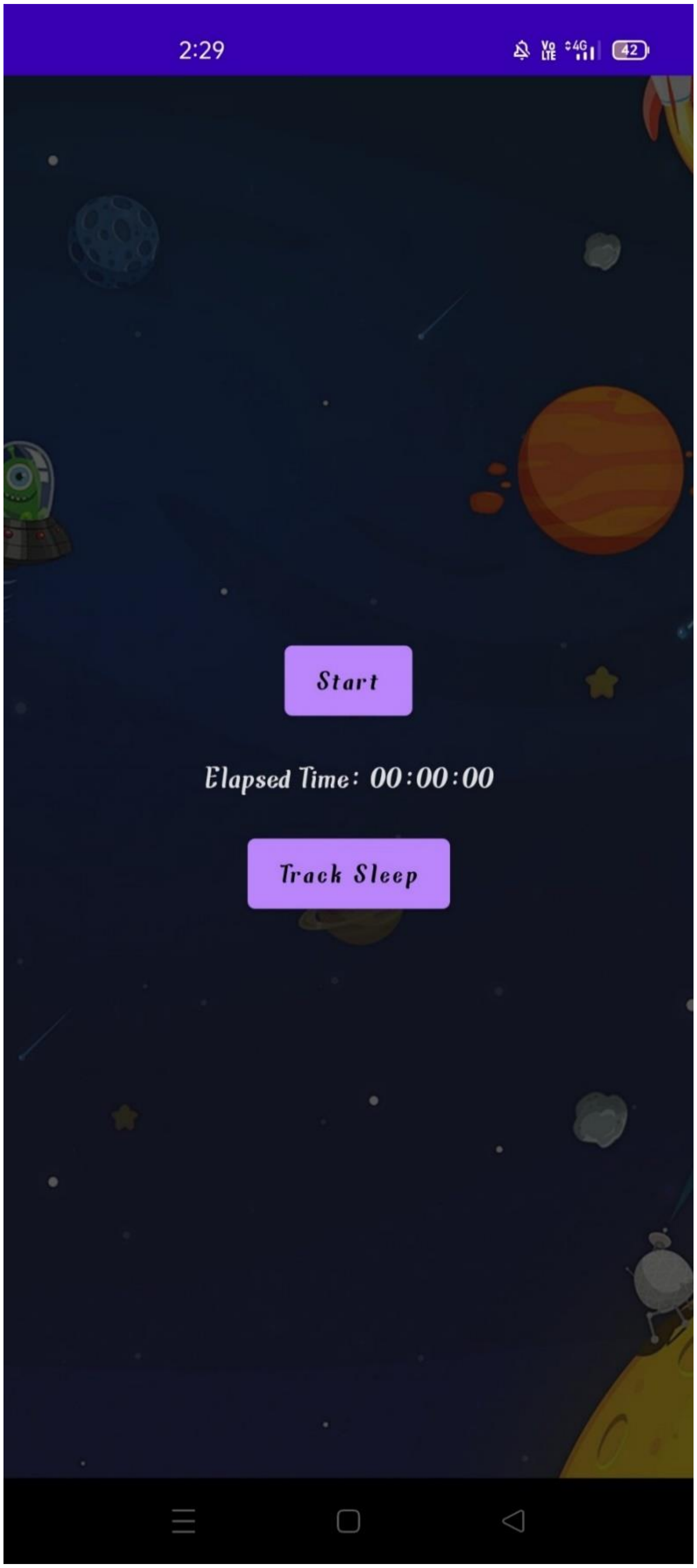




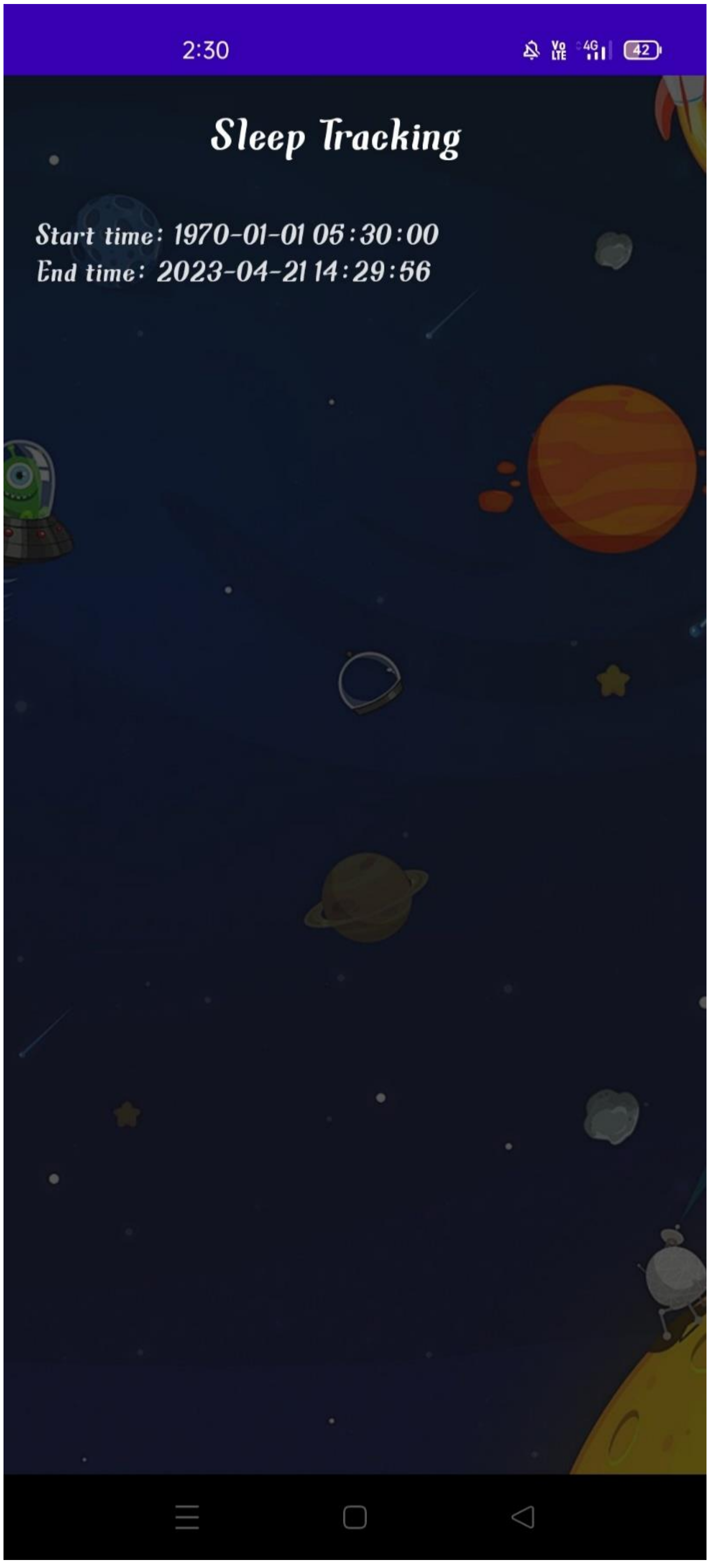
Username
abc

Password
1234

Successfully log in



Elapsed Time: 00:00:00



Sleep Tracking

Start time: 1970-01-01 05:30:00
End time: 2023-04-21 14:29:56

4.ADVANTAGES AND DISADVANTAGE:

Advantages:

1. Helps You Better Understand Your Sleep Patterns
2. Tracks Your Different Sleep Phases
3. Captures Possible Sleep Interruptions and Disruptions
4. Gently Wakes You Up at the Ideal Time
5. Convenient Compared to Wearing a Sleep Tracker Device

Disadvantage:

1. Hinders Sleep More Than It Helps
2. Promotes Nighttime Screen Use Before Bed
3. Incorrect or Unreliable Sleep Tracking Data
4. Creates an Unhealthy Sleep Obsession

5.Applications:

The area where the sleep tracker is used in health and medical field.

6.Conclusion

Apps available for sleep self-management and tracking may be valuable tools for self-management of sleep disorder or improving sleep quality, yet they require improvement in terms of quality and content, highlighting the need for further validity studies.

7.Future Scope:

1.ARTIFICIAL INTELLIGENCE-BASED SLEEP MODELLING

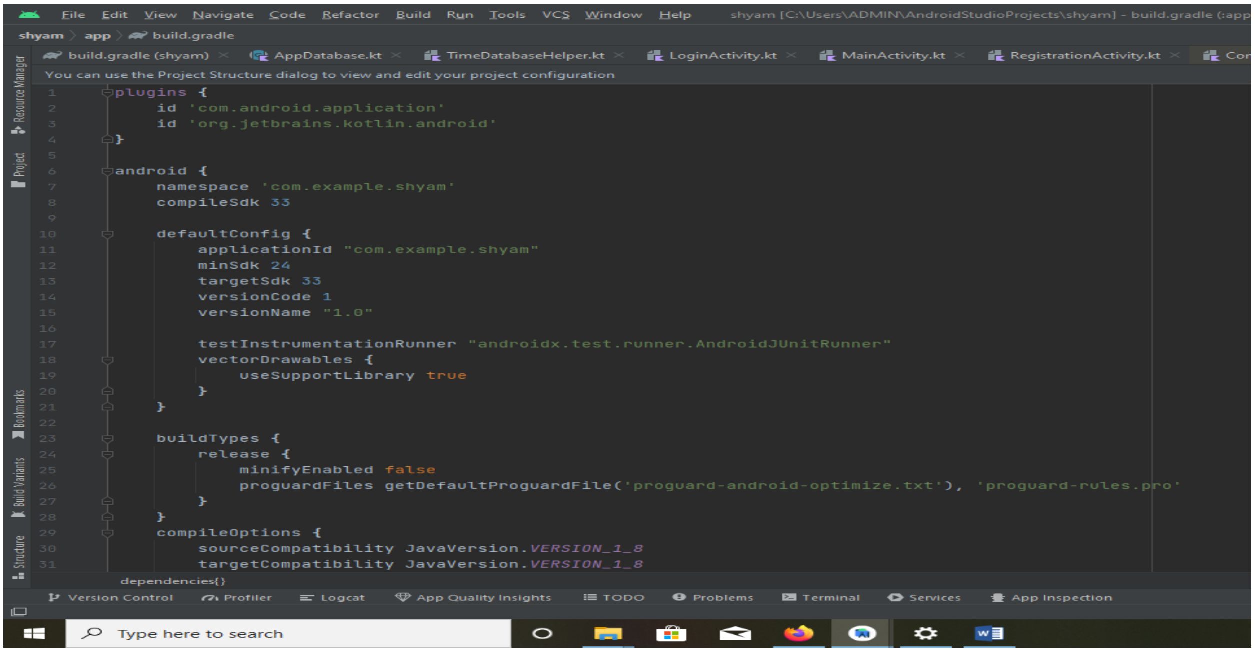
2.SLEEP DATA ACQUISITION

3.SLEEP DATA STORAGE AND CURATION

4.DATA PRE-PROCESSING 5.DATA VISUALISATION AND VISUAL ANALYTICS

8.APPENDIX

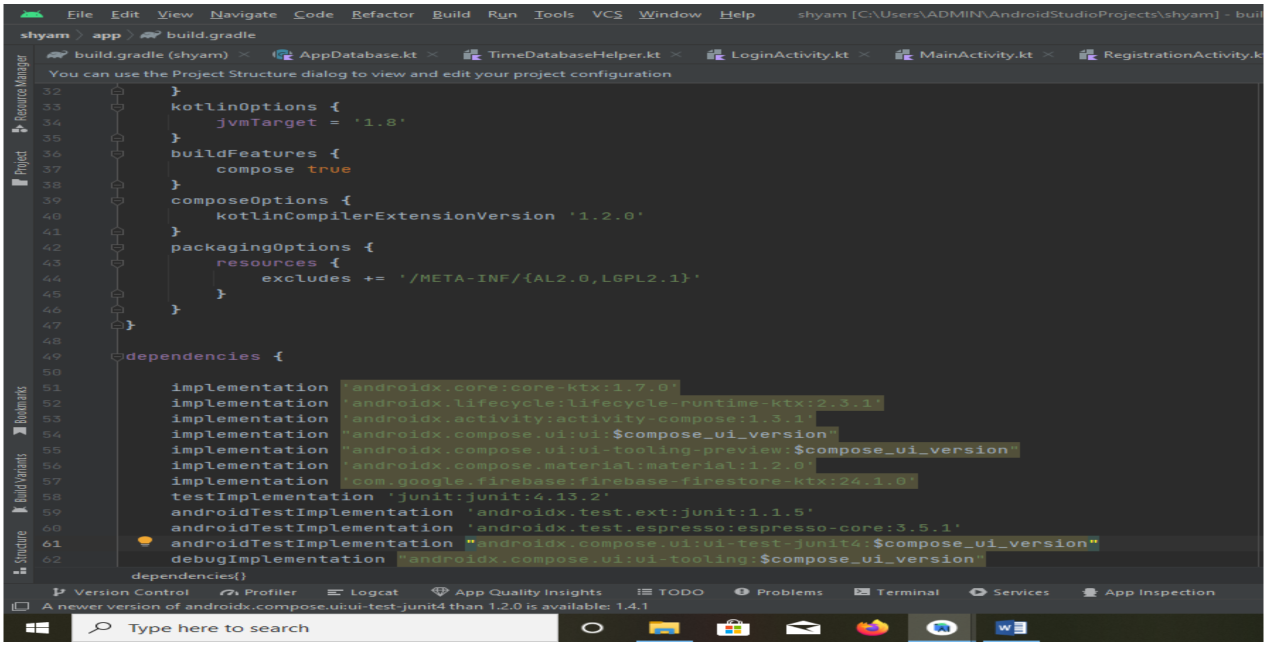
*BUILD.GRADLE



You can use the Project Structure dialog to view and edit your project configuration

```
plugins {
    id 'com.android.application'
    id 'org.jetbrains.kotlin.android'
}

android {
    namespace 'com.example.shyam'
    compileSdk 33

    defaultConfig {
        applicationId "com.example.shyam"
        minSdk 24
        targetSdk 33
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
        vectorDrawables {
            useSupportLibrary true
        }
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
```

dependencies{}

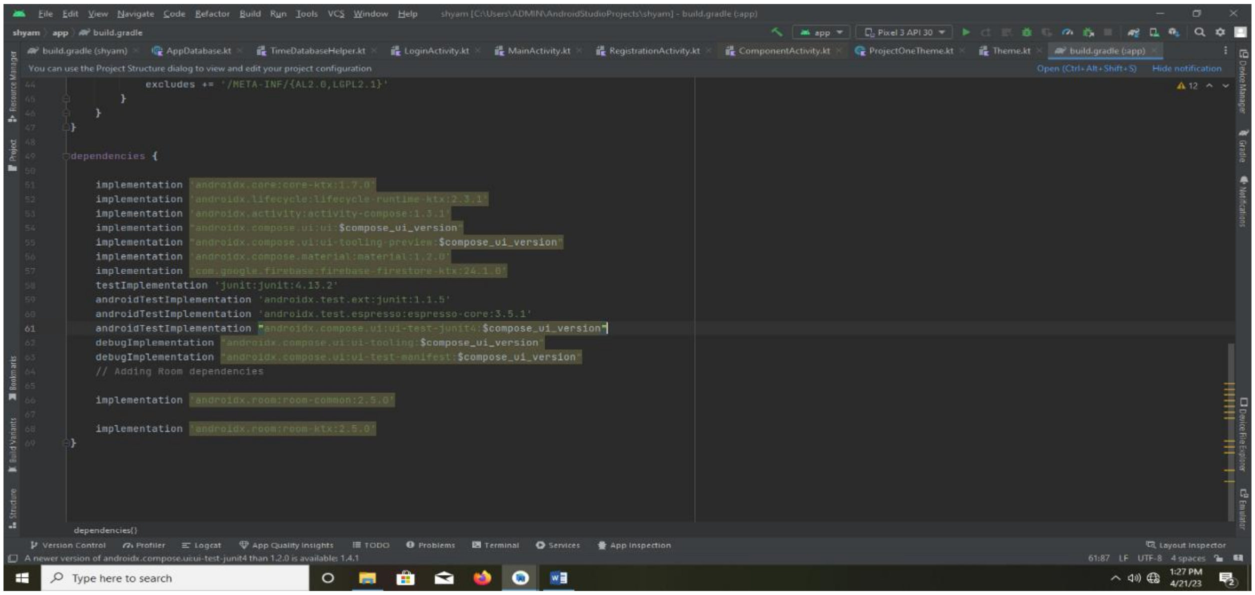


You can use the Project Structure dialog to view and edit your project configuration

```
    }
    kotlinOptions {
        jvmTarget = '1.8'
    }
    buildFeatures {
        compose true
    }
    composeOptions {
        kotlinCompilerExtensionVersion '1.2.0'
    }
    packagingOptions {
        resources {
            excludes += '/META-INF/{AL2.0,LGPL2.1}'
        }
    }
}

dependencies {

    implementation 'androidx.core:core-ktx:1.7.0'
    implementation 'androidx.lifecycle:lifecycle-runtime-ktx:2.3.1'
    implementation 'androidx.activity:activity-compose:1.3.1'
    implementation "androidx.compose.ui:ui:$compose_ui_version"
    implementation "androidx.compose.ui:ui-tooling-preview:$compose_ui_version"
    implementation 'androidx.compose.material:material:1.2.0'
    implementation 'com.google.firebase:firebase-firestore-ktx:24.1.0'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.5'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'
    androidTestImplementation "androidx.compose.ui:ui-test-junit4:$compose_ui_version"
    debugImplementation "androidx.compose.ui:ui-tooling:$compose_ui_version"
```

dependencies{}

A newer version of androidx.compose.ui:ui-test-junit4 than 1.2.0 is available: 1.4.1

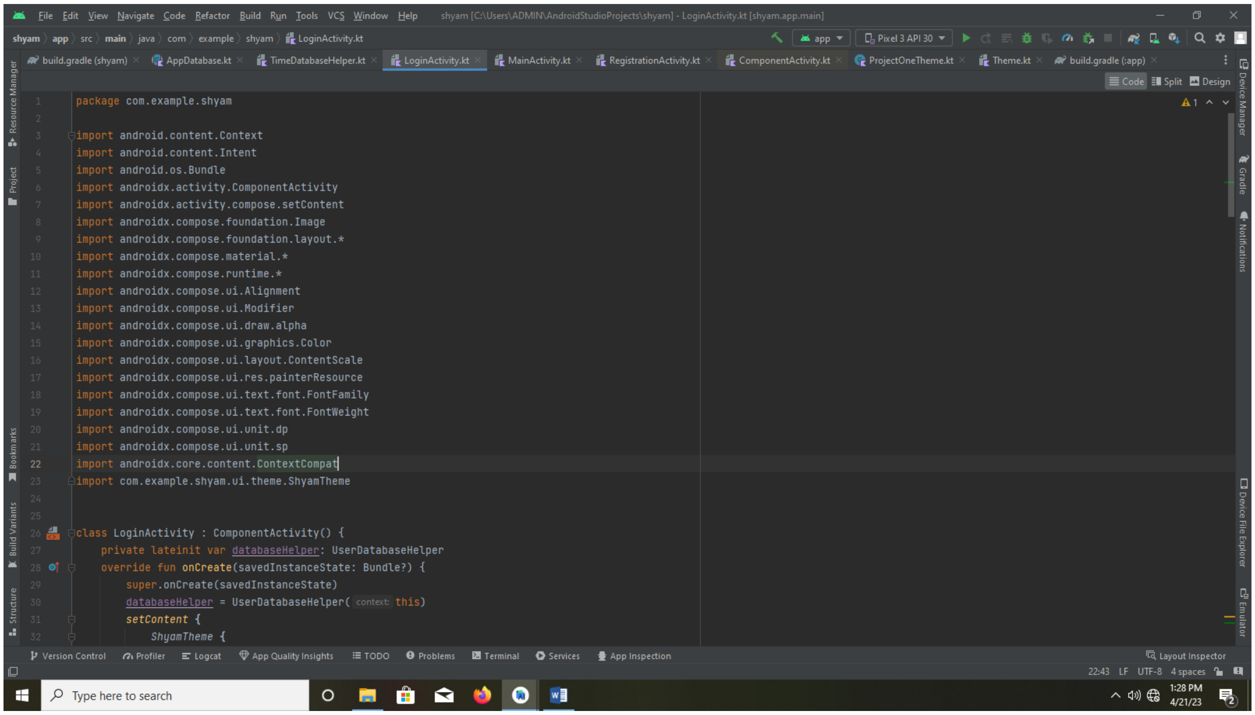


You can use the Project Structure dialog to view and edit your project configuration

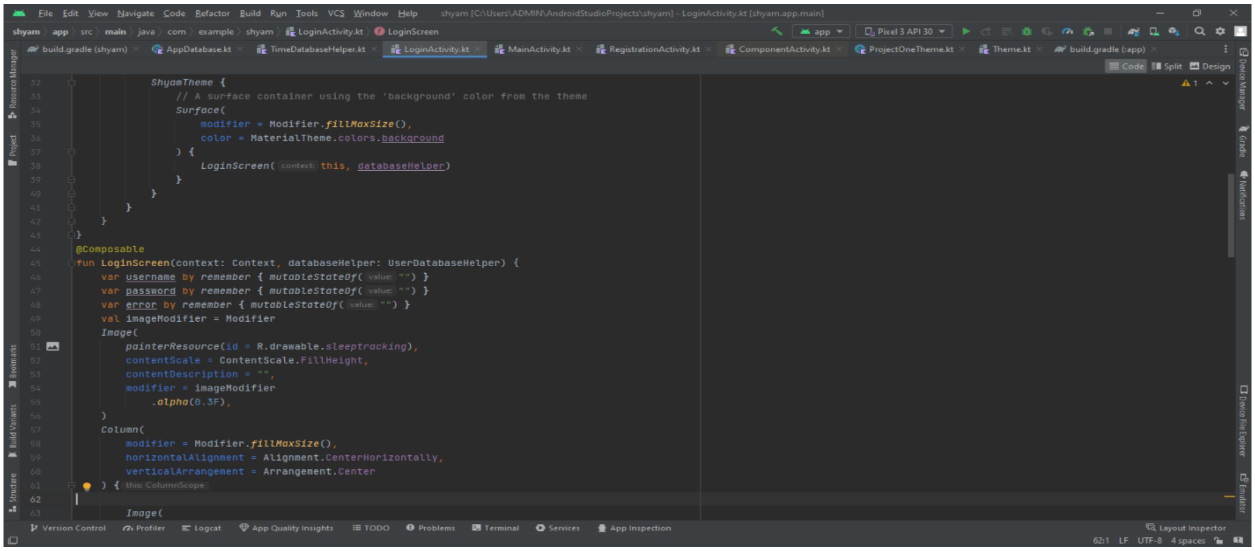
```
                excludes += '/META-INF/{AL2.0,LGPL2.1}'
            }
        }
}

dependencies {

    implementation 'androidx.core:core-ktx:1.7.0'
    implementation 'androidx.lifecycle:lifecycle-runtime-ktx:2.3.1'
    implementation 'androidx.activity:activity-compose:1.3.1'
    implementation "androidx.compose.ui:ui:$compose_ui_version"
    implementation "androidx.compose.ui:ui-tooling-preview:$compose_ui_version"
    implementation 'androidx.compose.material:material:1.2.0'
    implementation 'com.google.firebase:firebase-firestore-ktx:24.1.0'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.5'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.5.1'
    androidTestImplementation "androidx.compose.ui:ui-test-junit4:$compose_ui_version"
    debugImplementation "androidx.compose.ui:ui-tooling:$compose_ui_version"
    debugImplementation "androidx.compose.ui:ui-test-manifest:$compose_ui_version"
    // Adding Room dependencies

    implementation 'androidx.room:room-common:2.5.0'

    implementation 'androidx.room:room-ktx:2.5.0'
}
```

A newer version of androidx.compose.ui:ui-test-junit4 than 1.2.0 is available: 1.4.1

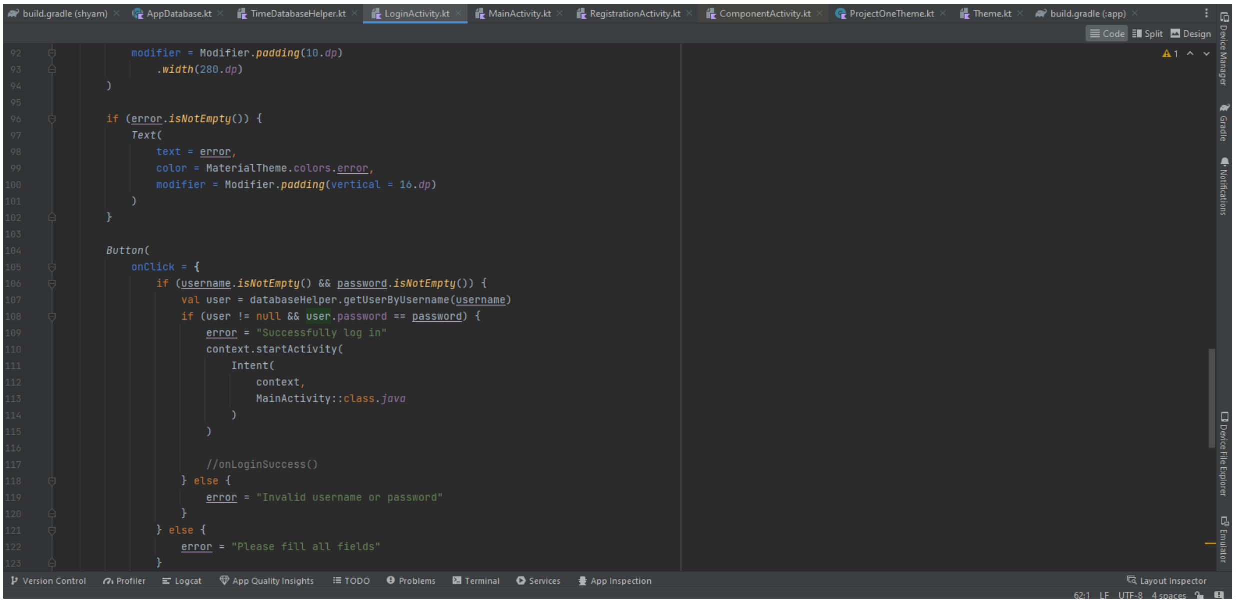
*LOGINACTIVITY.KT



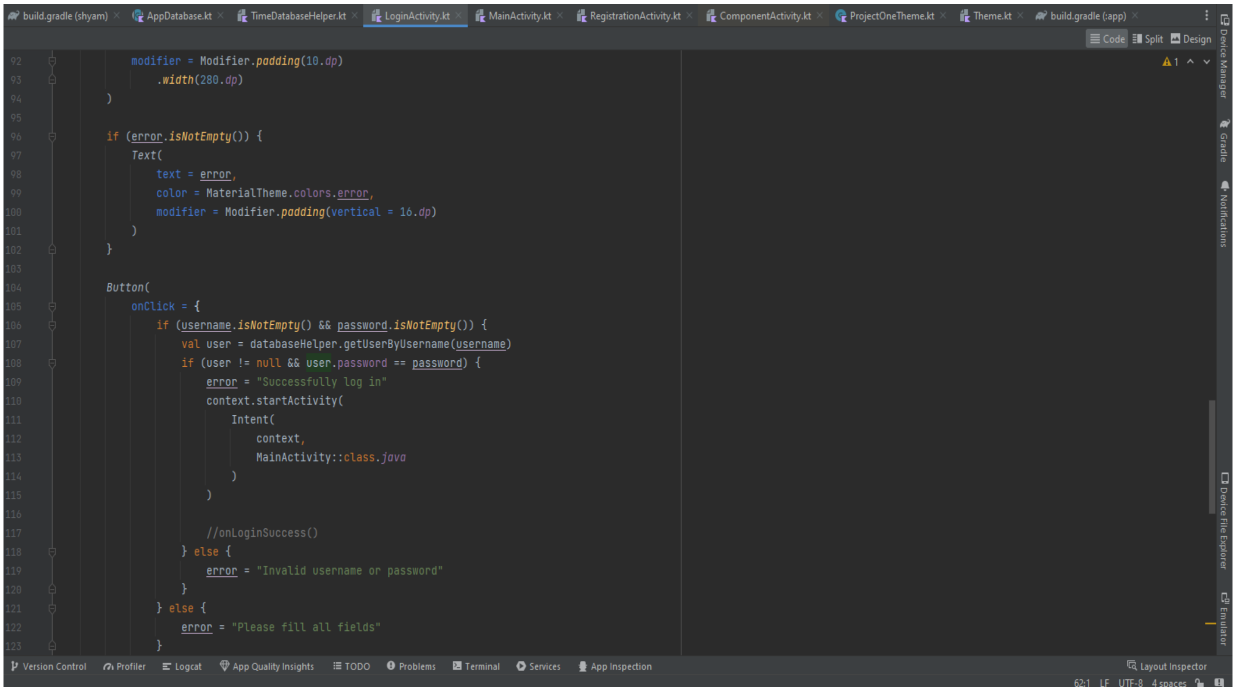
```
package com.example.shyam

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.shyam.ui.theme.ShyamTheme


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



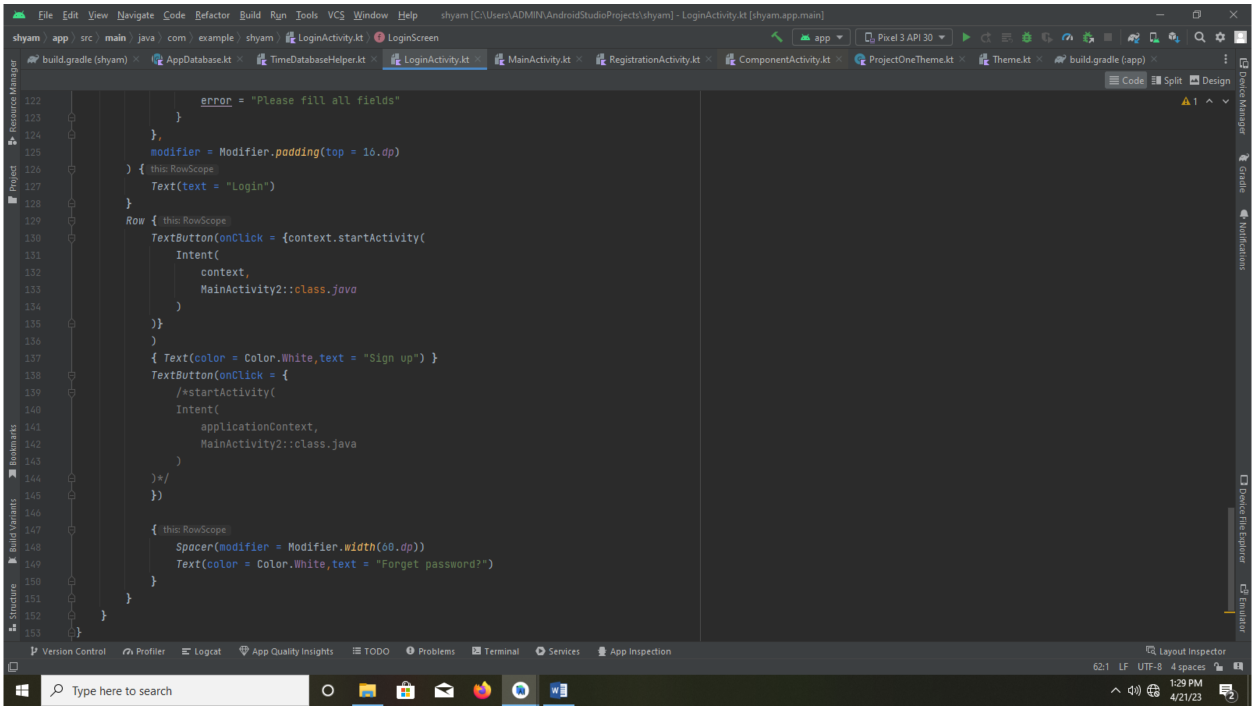
```
            ShyamTheme {
                // A surface container using the 'background' color from the theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background
                ) {
                    LoginScreen( 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(
        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

        Image(
```



```
            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"
                }
```

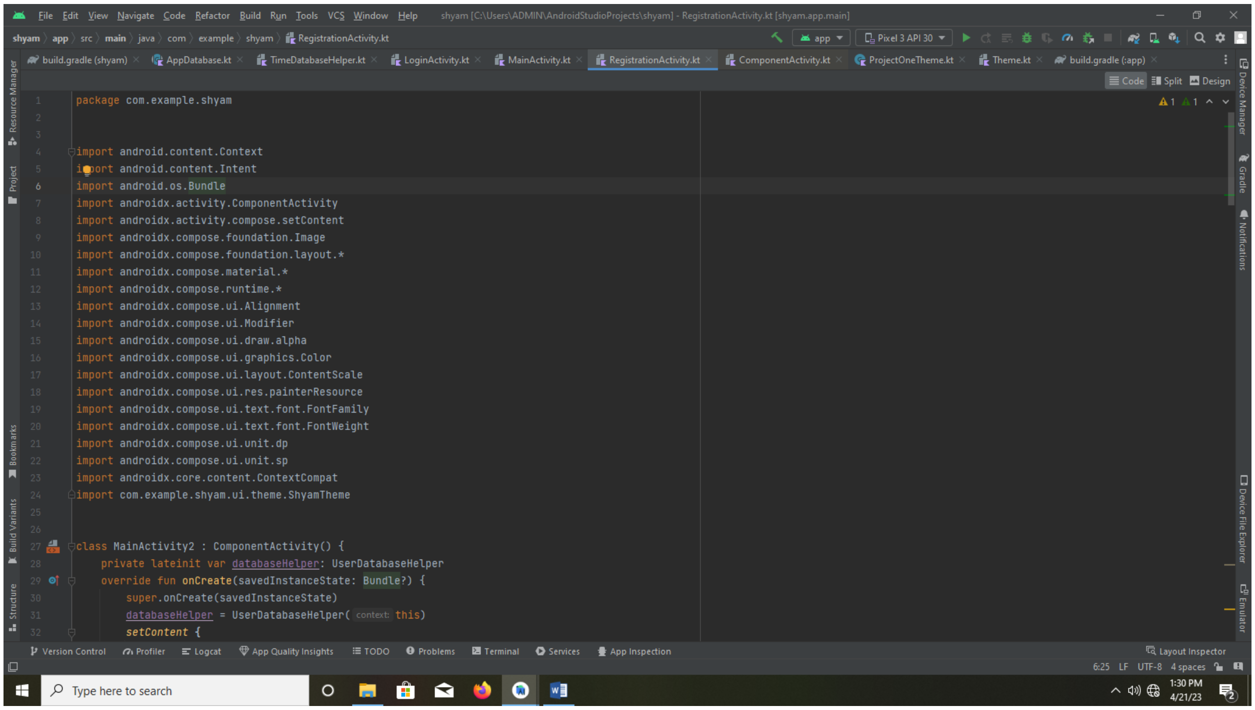


```kotlin
            modifier = Modifier.padding(10.dp)
                .width(280.dp)
        )

        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"
                }
```

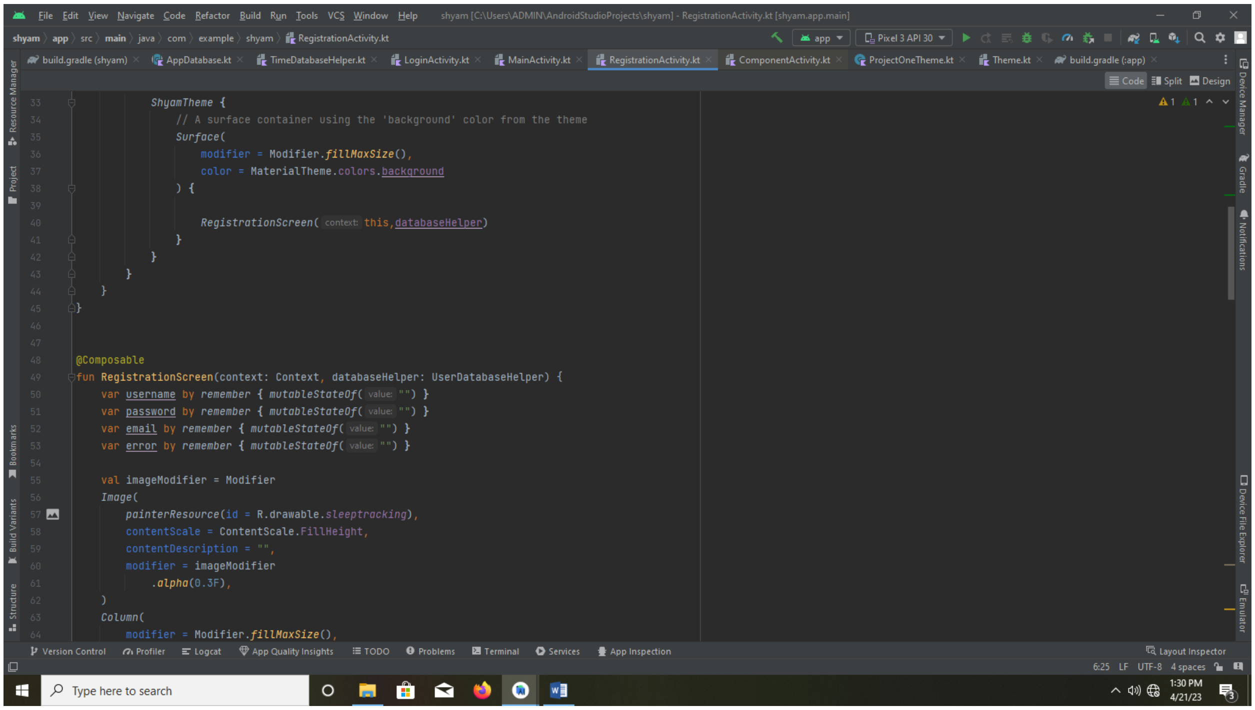


```kotlin
                    error = "Please fill all fields"
                }
            },
            modifier = Modifier.padding(top = 16.dp)
        ) {
            Text(text = "Login")
        }
        Row {
            TextButton(onClick = {context.startActivity(
                Intent(
                    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?")
            }
        }
    }
}
```

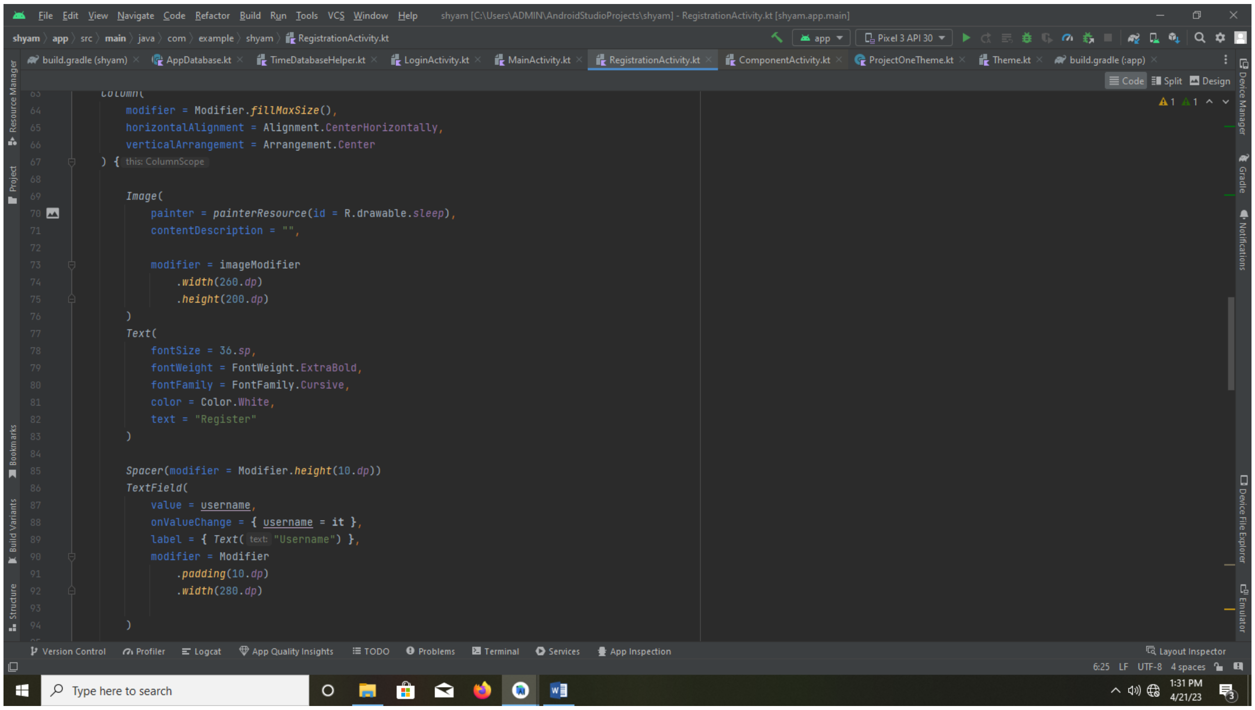
*REGISTRATIONACTIVITY.KT



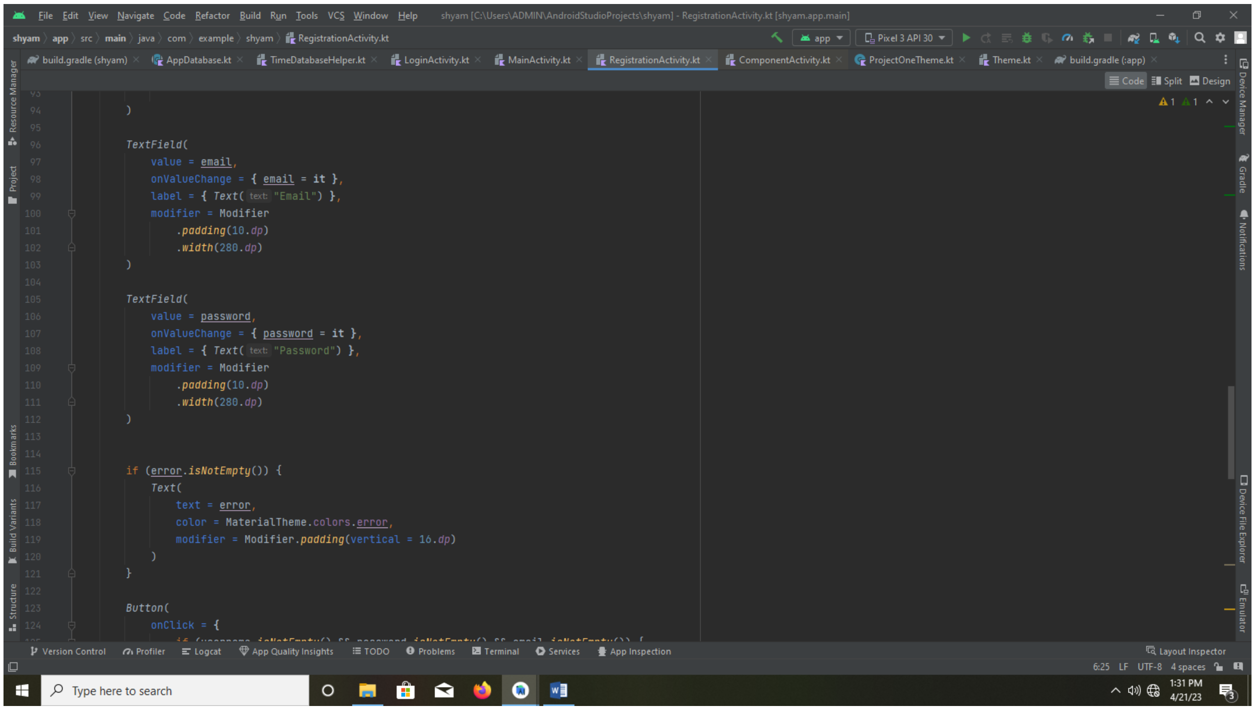
```kotlin
package com.example.shyam


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.shyam.ui.theme.ShyamTheme


class MainActivity2 : ComponentActivity() {
    private lateinit var databaseHelper: UserDatabaseHelper
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        databaseHelper = UserDatabaseHelper(this)
        setContent {
```



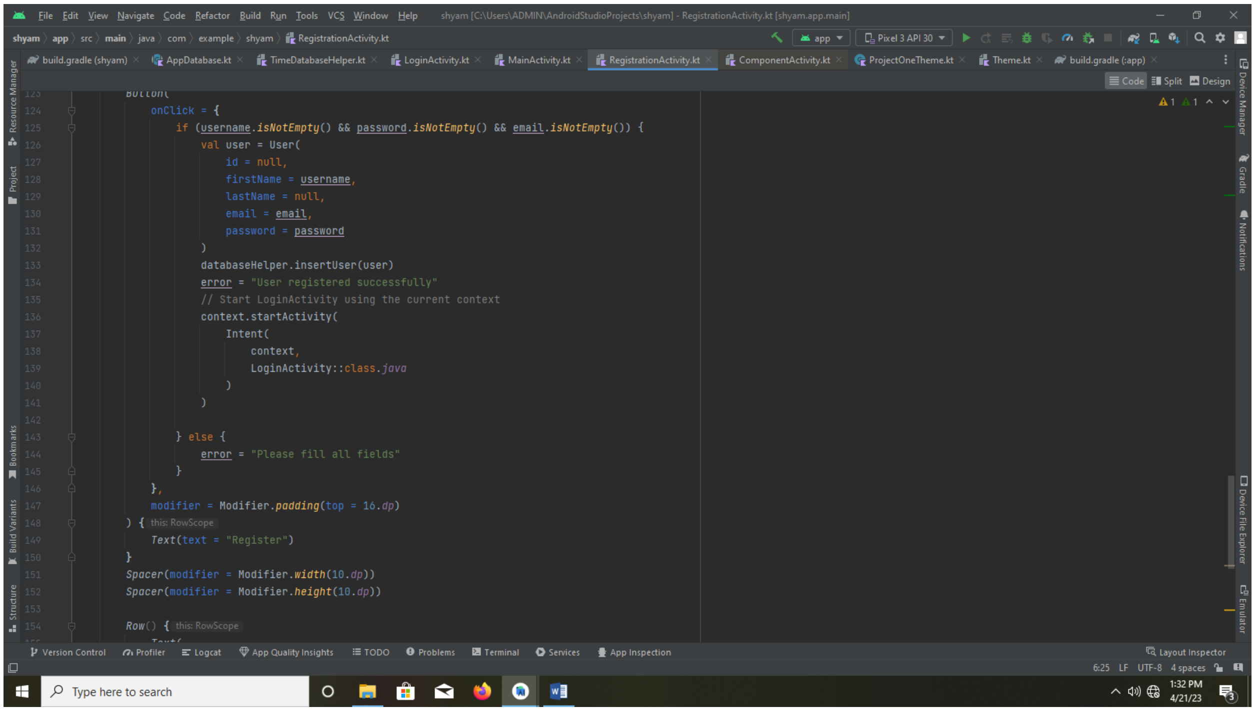
```kotlin
            ShyamTheme {
                // A surface container using the 'background' color from the theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background
                ) {

                    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(),
```



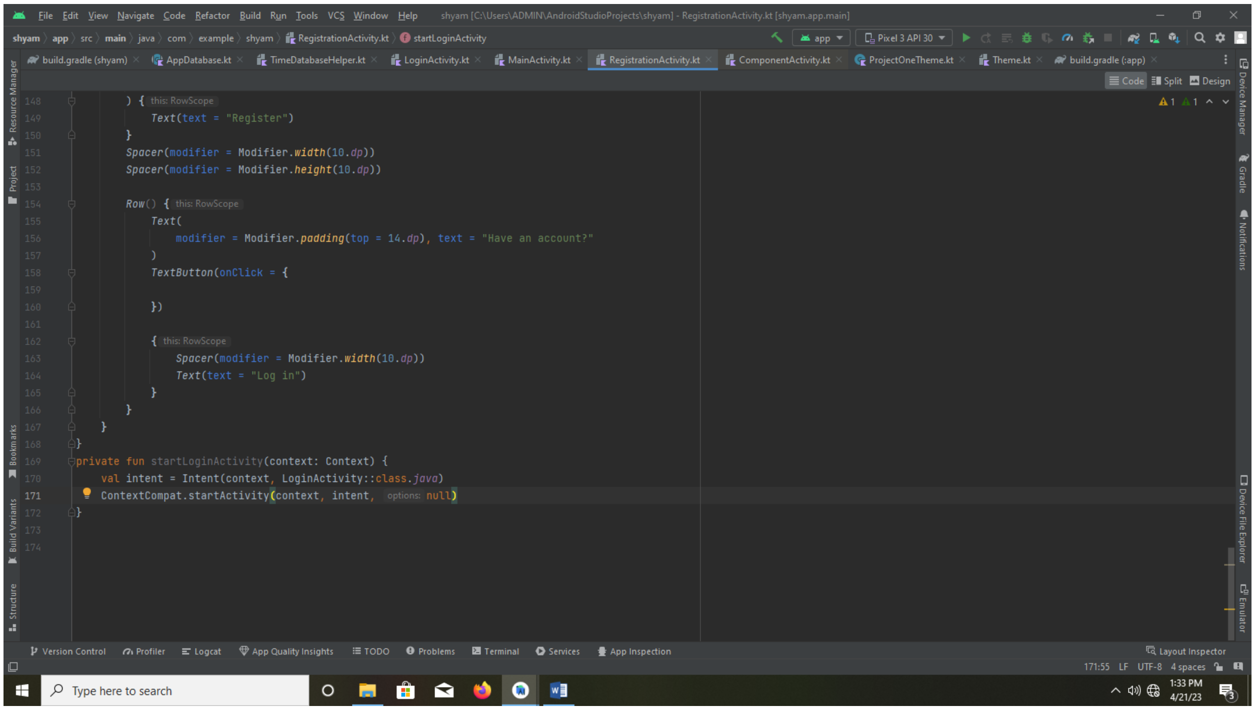
```kotlin
        Column(
            modifier = Modifier.fillMaxSize(),
            horizontalAlignment = Alignment.CenterHorizontally,
            verticalArrangement = Arrangement.Center
        ) {

            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)

            )
```



```kotlin
            )

            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()) {
```

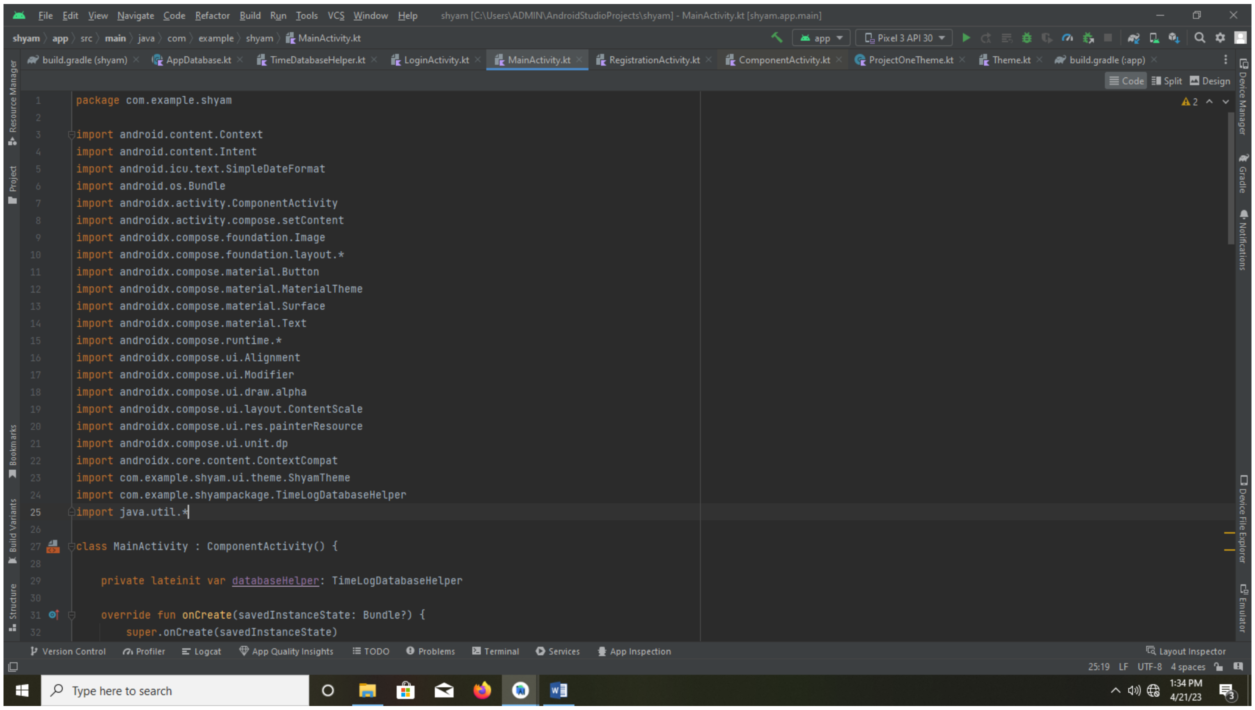


```
        Button(
            onClick = {
                if (username.isNotEmpty() && password.isNotEmpty() && email.isNotEmpty()) {
                    val user = User(
                        id = null,
                        firstName = username,
                        lastName = null,
                        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(
```

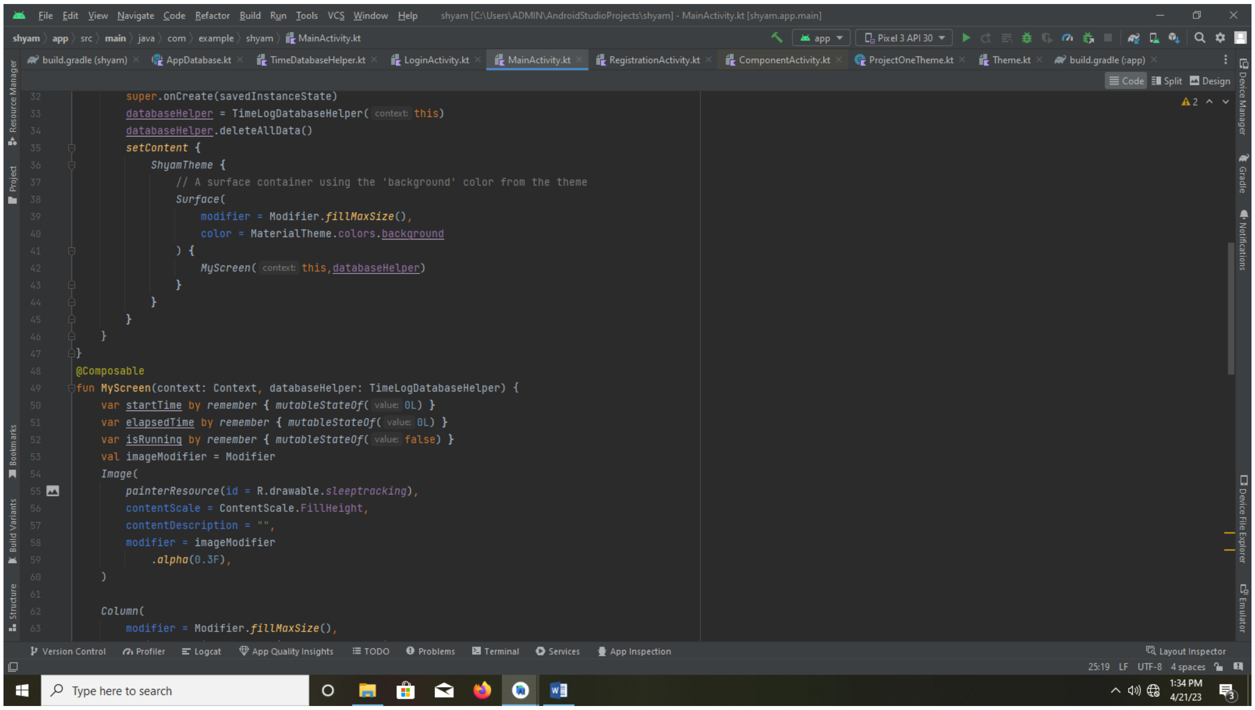


```
        ) {
            Text(text = "Register")
        }
        Spacer(modifier = Modifier.width(10.dp))
        Spacer(modifier = Modifier.height(10.dp))

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

            })

            {
                Spacer(modifier = Modifier.width(10.dp))
                Text(text = "Log in")
            }
        }
    }
}
private fun startLoginActivity(context: Context) {
    val intent = Intent(context, LoginActivity::class.java)
    ContextCompat.startActivity(context, intent, null)
}
```

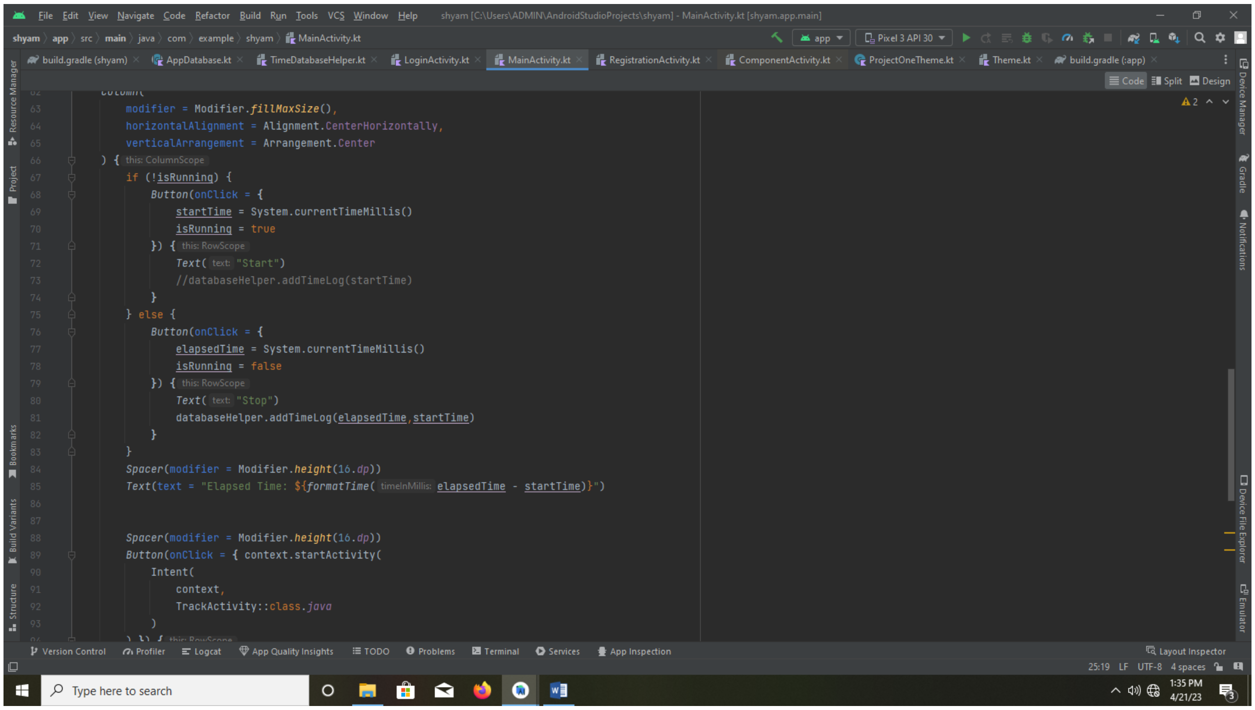
*MAINACTIVITY.KT



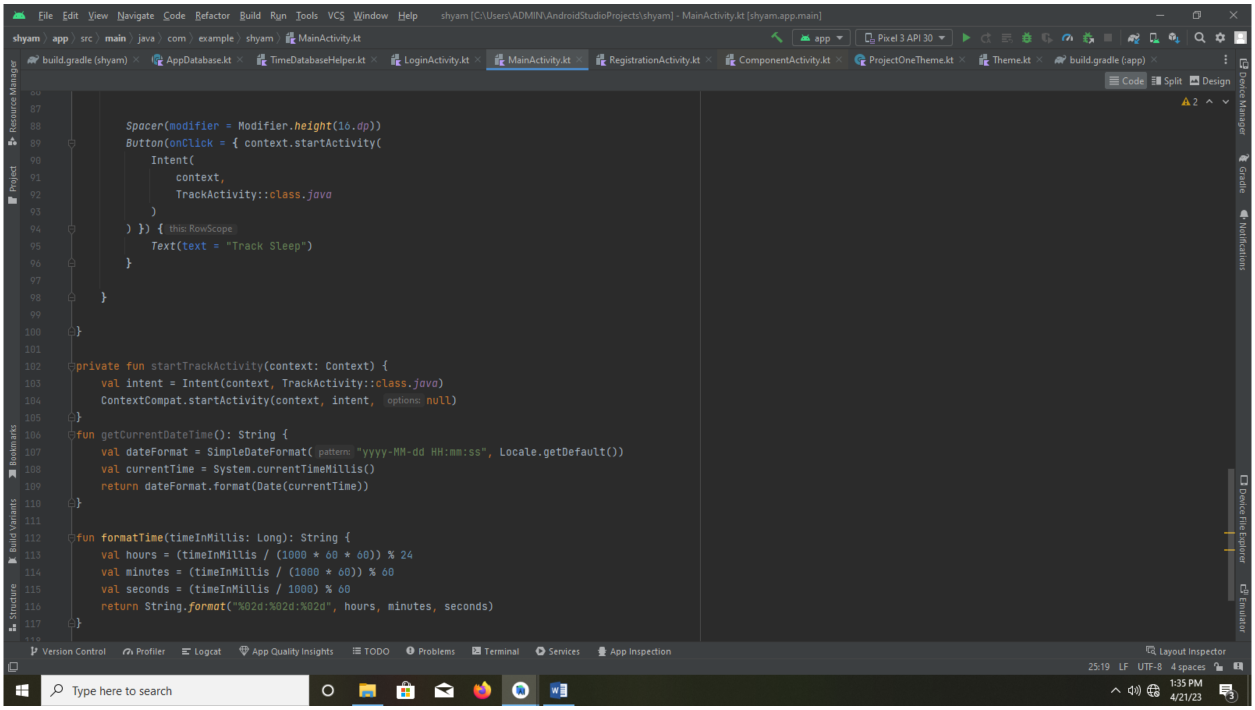
```kotlin
package com.example.shyam

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.shyam.ui.theme.ShyamTheme
import com.example.shyampackage.TimeLogDatabaseHelper
import java.util.*

class MainActivity : ComponentActivity() {

    private lateinit var databaseHelper: TimeLogDatabaseHelper

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



```kotlin
        super.onCreate(savedInstanceState)
        databaseHelper = TimeLogDatabaseHelper(this)
        databaseHelper.deleteAllData()
        setContent {
            ShyamTheme {
                // 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(),
```

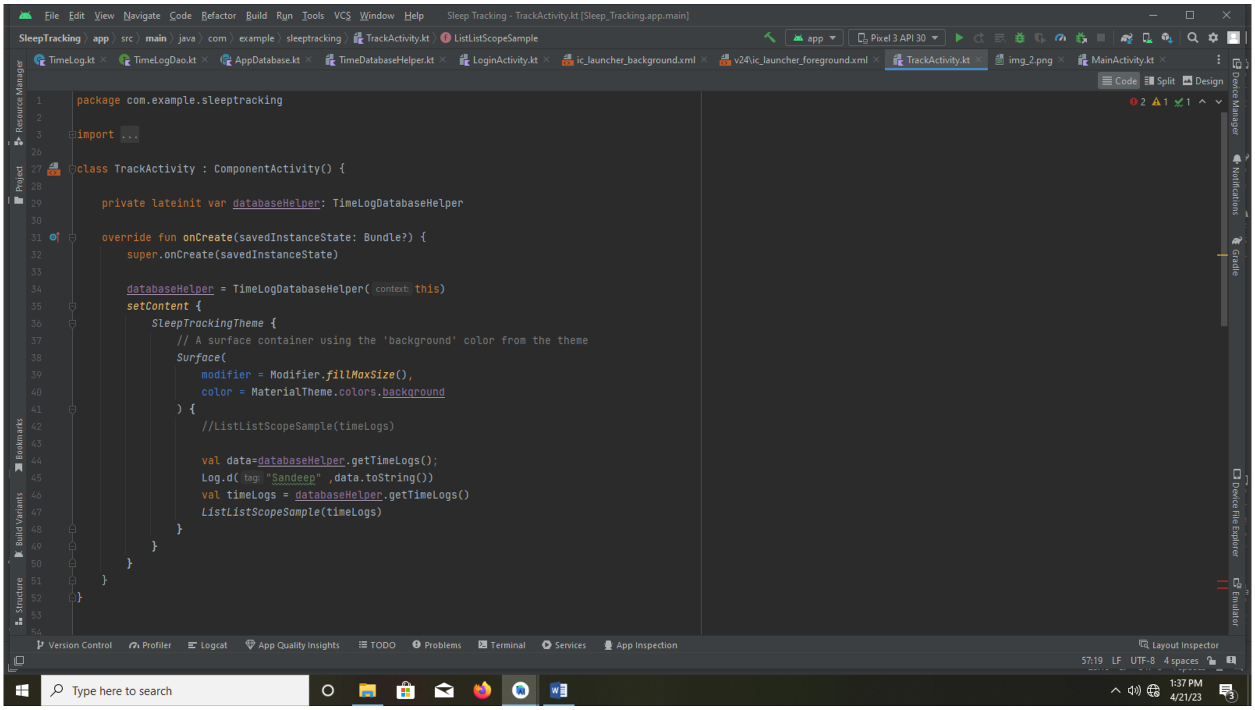


```kotlin
        Column(
            modifier = Modifier.fillMaxSize(),
            horizontalAlignment = Alignment.CenterHorizontally,
            verticalArrangement = Arrangement.Center
        ) { this: ColumnScope
            if (!isRunning) {
                Button(onClick = {
               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


            Spacer(modifier = Modifier.height(16.dp))
            Button(onClick = { context.startActivity(
                Intent(
                    context,
                    TrackActivity::class.java
                )
            ) }) { this: RowScope
```



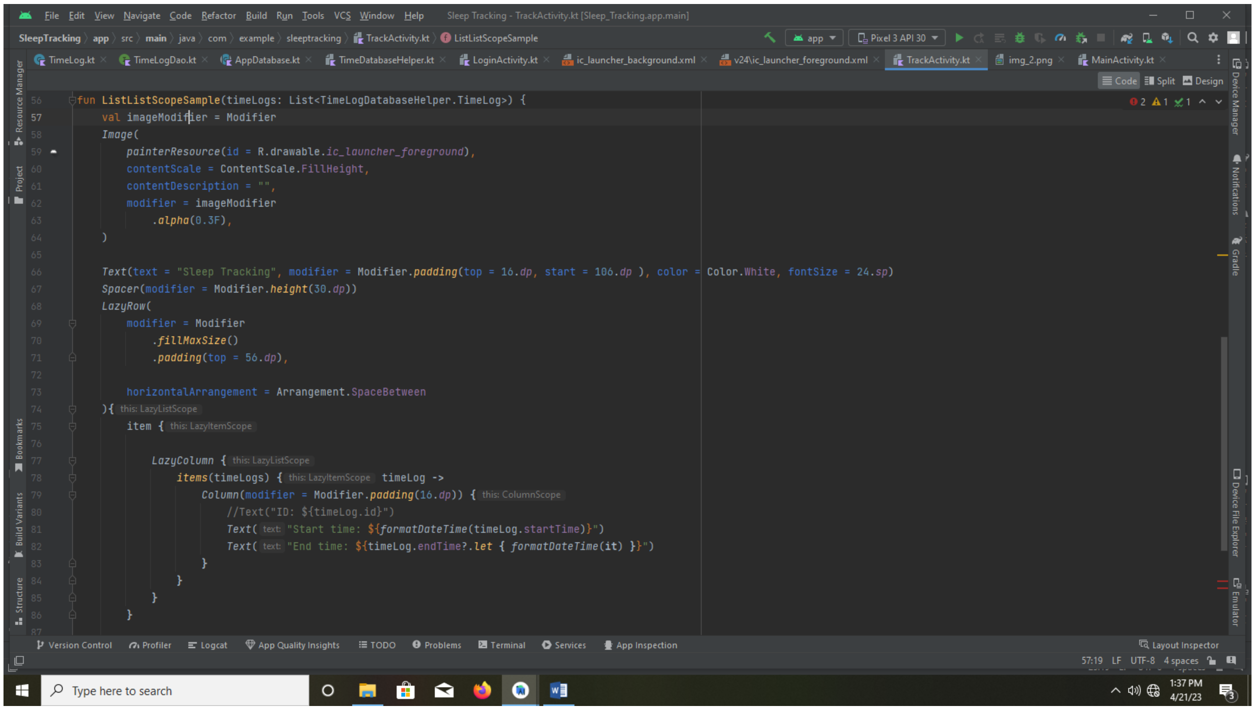
```kotlin

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

        }

}

private fun startTrackActivity(context: Context) {
    val intent = Intent(context, TrackActivity::class.java)
    ContextCompat.startActivity(context, intent,  options: null)
}
fun getCurrentDateTime(): String {
    val dateFormat = SimpleDateFormat( pattern: "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)
}
```

*TRACKAACTIVITY.KT



File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
Sleep Tracking - TrackActivity.kt [Sleep_Tracking.app.main]
SleepTracking › app › src › main › java › com › example › sleeptracking › TrackActivity.kt › ListListScopeSample
TimeLog.kt
TimeLogDao.kt
AppDatabase.kt
TimeDatabaseHelper.kt
LoginActivity.kt
ic_launcher_background.xml
v24\ic_launcher_foreground.xml
TrackActivity.kt
img_2.png
MainActivity.kt
Version Control
Profiler
Logcat
App Quality Insights
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Type here to search
1:37 PM
4/21/23

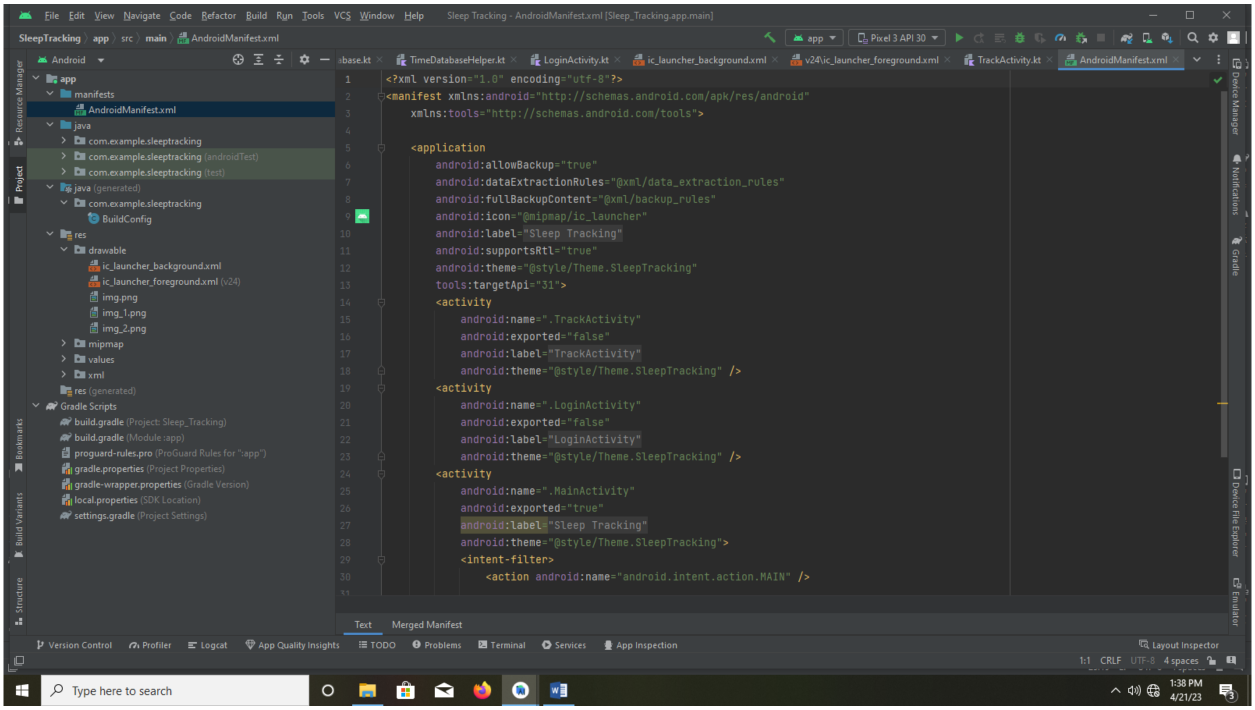
```kotlin
package com.example.sleeptracking

import ...

class TrackActivity : ComponentActivity() {

    private lateinit var databaseHelper: TimeLogDatabaseHelper

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

        databaseHelper = TimeLogDatabaseHelper( context: this)
        setContent {
            SleepTrackingTheme {
                // 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)
                }
            }
        }
    }
}
```



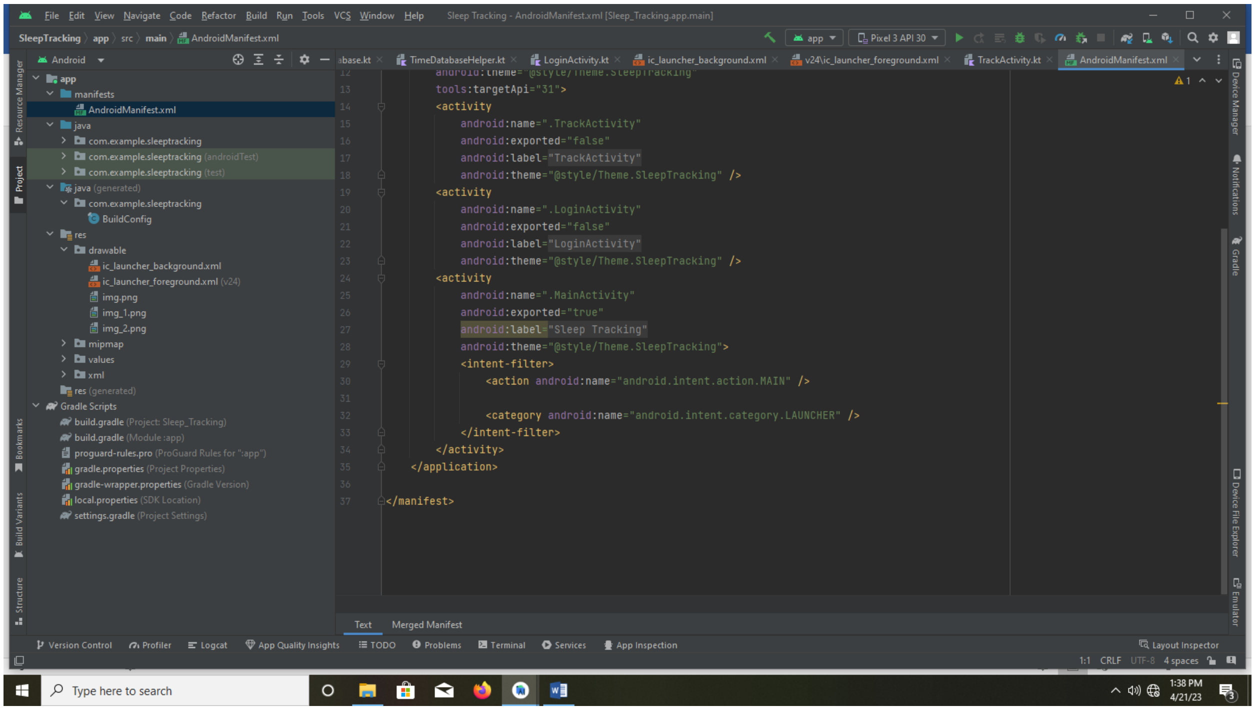
File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
Sleep Tracking - TrackActivity.kt [Sleep_Tracking.app.main]
SleepTracking › app › src › main › java › com › example › sleeptracking › TrackActivity.kt › ListListScopeSample
Version Control
Profiler
Logcat
App Quality Insights
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Type here to search
1:37 PM
4/21/23

```kotlin
fun ListListScopeSample(timeLogs: List<TimeLogDatabaseHelper.TimeLog>) {
    val imageModifier = Modifier
    Image(
        painterResource(id = R.drawable.ic_launcher_foreground),
        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
    ){ this: LazyListScope
        item { this: LazyItemScope

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

*ANDROID MAINFEST.XML



```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="Sleep Tracking"
        android:supportsRtl="true"
        android:theme="@style/Theme.SleepTracking"
        tools:targetApi="31">
        <activity
            android:name=".TrackActivity"
            android:exported="false"
            android:label="TrackActivity"
            android:theme="@style/Theme.SleepTracking" />
        <activity
            android:name=".LoginActivity"
            android:exported="false"
            android:label="LoginActivity"
            android:theme="@style/Theme.SleepTracking" />
        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:label="Sleep Tracking"
            android:theme="@style/Theme.SleepTracking">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
```



```xml
        android:theme="@style/Theme.SleepTracking"
        tools:targetApi="31">
        <activity
            android:name=".TrackActivity"
            android:exported="false"
            android:label="TrackActivity"
            android:theme="@style/Theme.SleepTracking" />
        <activity
            android:name=".LoginActivity"
            android:exported="false"
            android:label="LoginActivity"
            android:theme="@style/Theme.SleepTracking" />
        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:label="Sleep Tracking"
            android:theme="@style/Theme.SleepTracking">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
```

```xml

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

</manifest>
```