

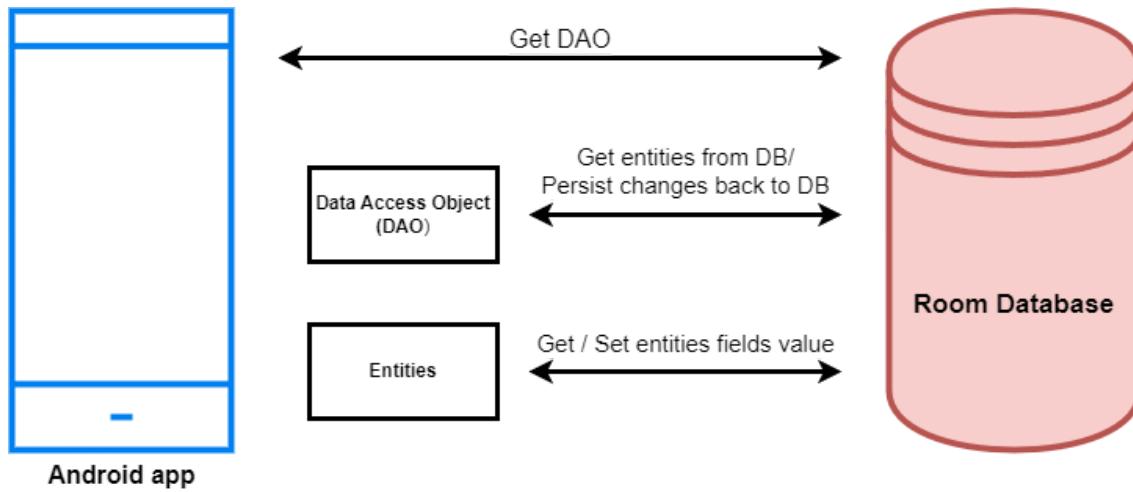


**Wanderlust: A Personalized Travel Planning and
Tracking App**
Project Based Experiential Learning Program

Wanderlust: A Personalized Travel Planning and Tracking App

A project built using the Android Compose UI toolkit. It demonstrates how to create a simple travel app using the Compose libraries. It also features a personalized feed of recommended accommodations based on the locations.

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 logs in into the application.
- User enters into the main page

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.

Task 1:

Required initial steps :

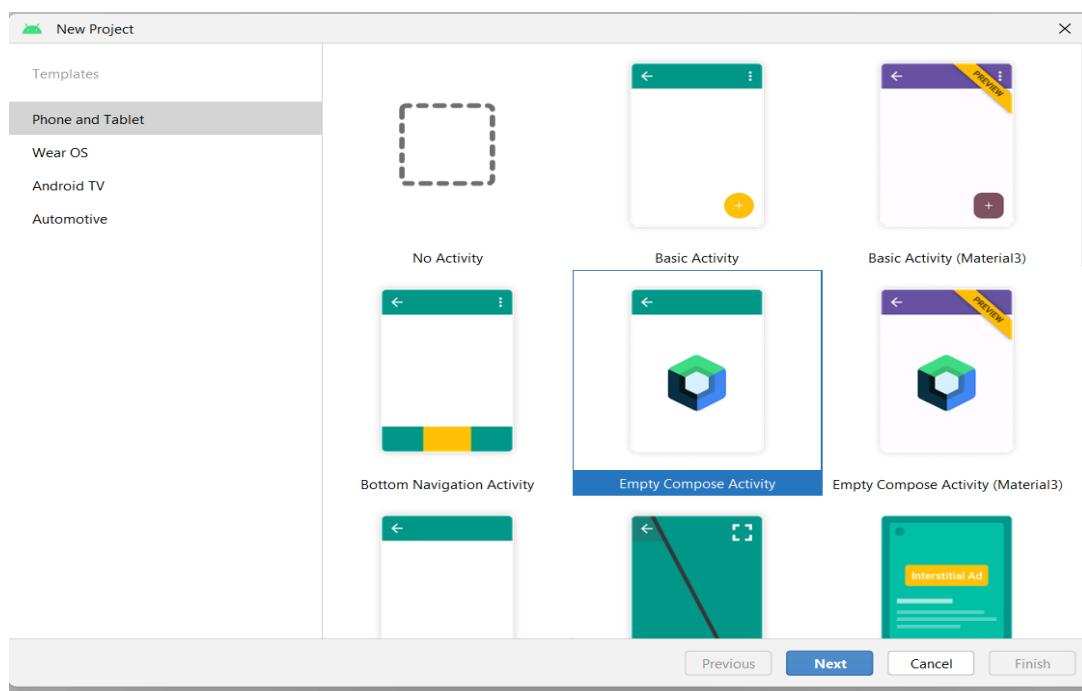
<https://developer.android.com/studio/install>

Task 2 :

Creating a new project.

Step 1 : Android studio > File > New > New Project > Empty Compose Activity

Step 2 : Click on **Next** button.

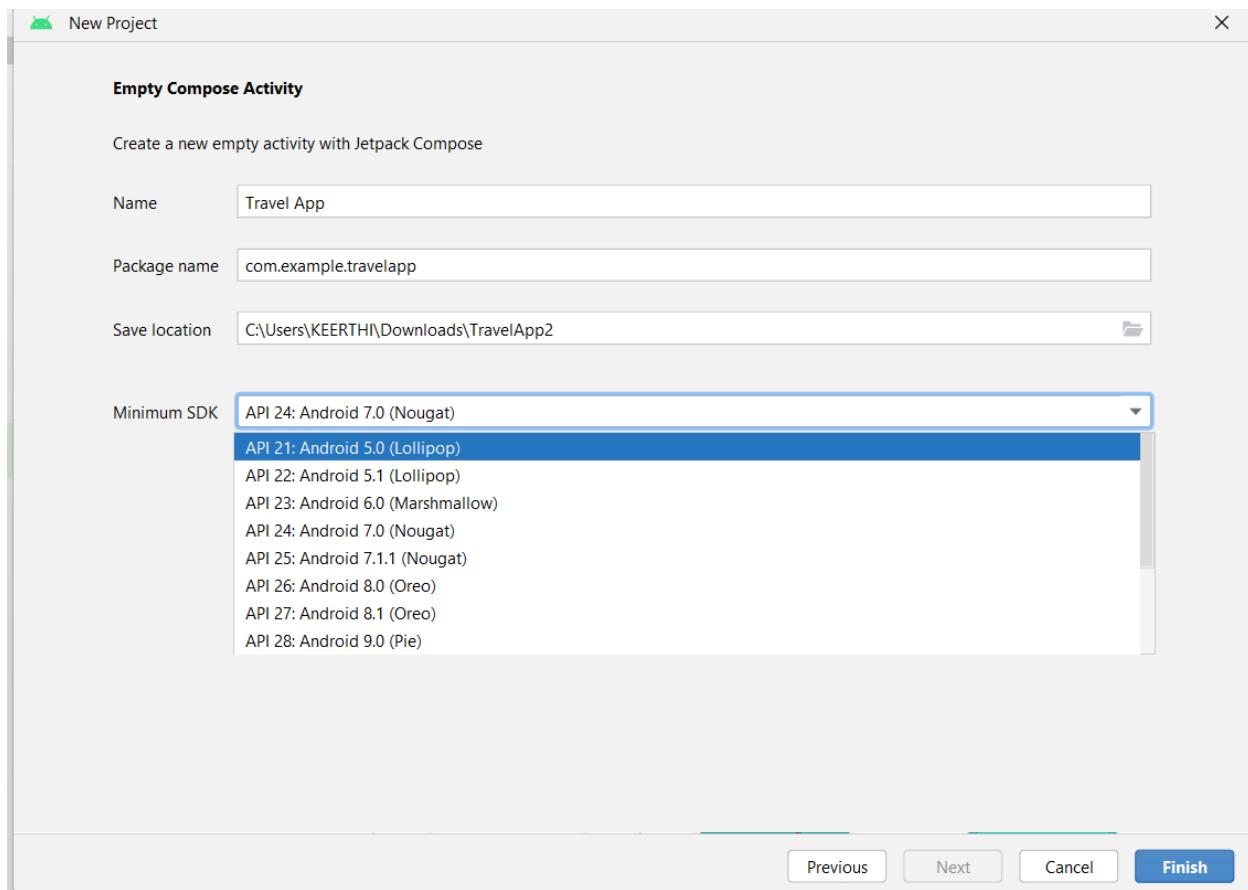


Step 3 :

Give name to the new project.

Step 4 : Give the Minimum SDK value

Step 5 : Click Finish



Main activity file

```

1 package com.example.travellapp
2
3 import ...
4
5 class MainActivity : ComponentActivity() {
6     override fun onCreate(savedInstanceState: Bundle?) {
7         super.onCreate(savedInstanceState)
8         setContent {
9             TravelApp(context: this)
10        }
11    }
12
13 @Composable
14 fun TravelApp(context: Context) {
15     Column(
16         modifier = Modifier
17             .padding(20.dp)
18             .verticalScroll(rememberScrollState())
19     ) {
20         this@ColumnScope
21
22         Text(
23             fontSize = 40.sp,
24             color = Color(android.graphics.Color.rgb( red: 128, green: 40, blue: 251)),
25             fontFamily = FontFamily.Cursive,
26             text = "Wanderlust Travel"
27         )
28
29         Spacer(modifier = Modifier.height(20.dp))
30     }
31 }
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53

```

Task 3 :

Adding required dependencies.

Step 1 : Gradle scripts > build.gradle(Module :app)

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65

```

Step 2 : Adding room dependencies.

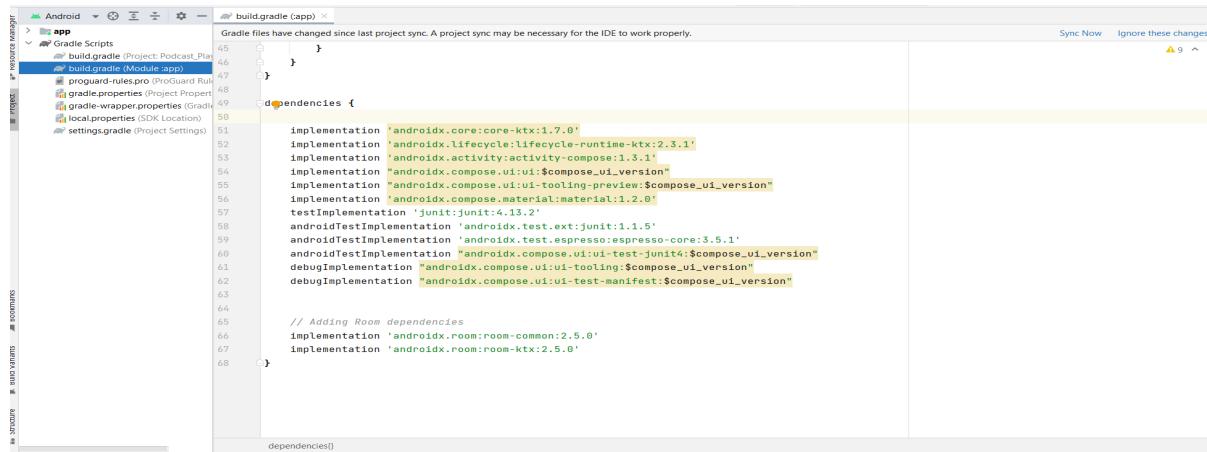
Add the below code in dependencies

```

// Adding Room dependencies
implementation 'androidx.room:room-common:2.5.0'

```

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



The screenshot shows the Android Studio interface with the build.gradle (app) file open. The code editor displays the following Gradle configuration:

```
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'
    testImplementation 'junit:junit:4.13.2'
    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"

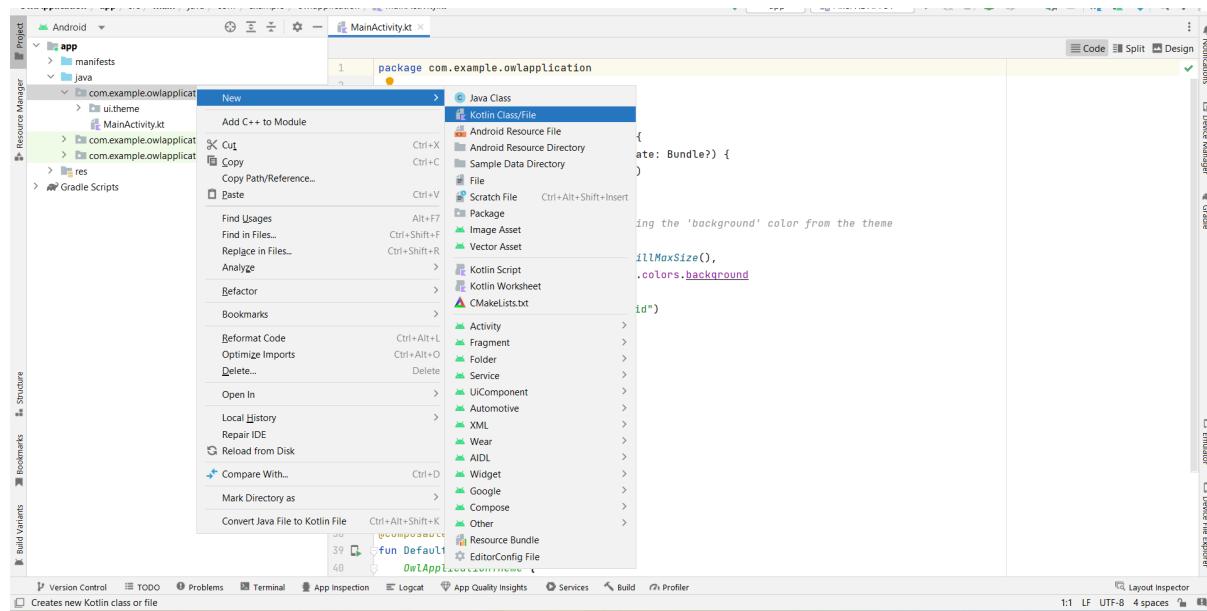
    // Adding Room dependencies
    implementation 'androidx.room:room-common:2.5.0'
    implementation 'androidx.room:room-ktx:2.5.0'
}
```

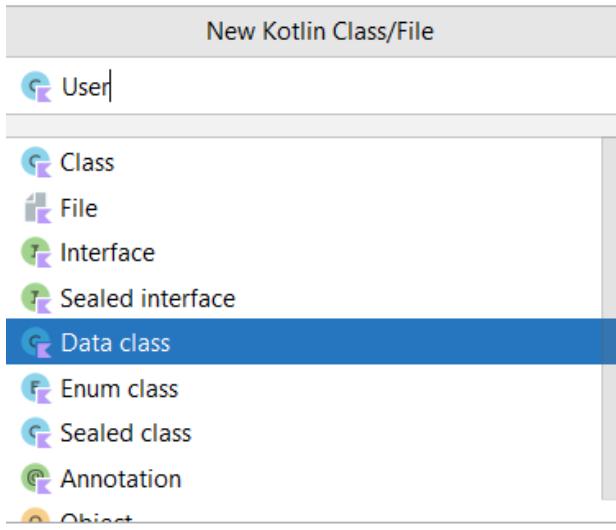
Step 3 : Click on Sync now

Task 4:

Creating the database classes.

Step 1 : Create User data class

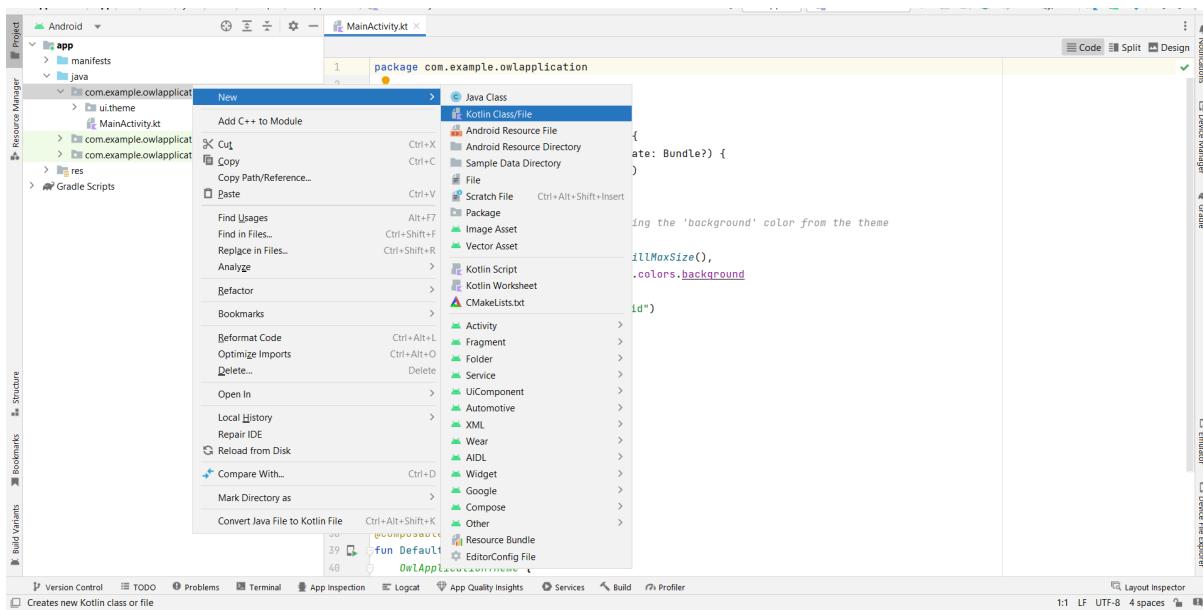


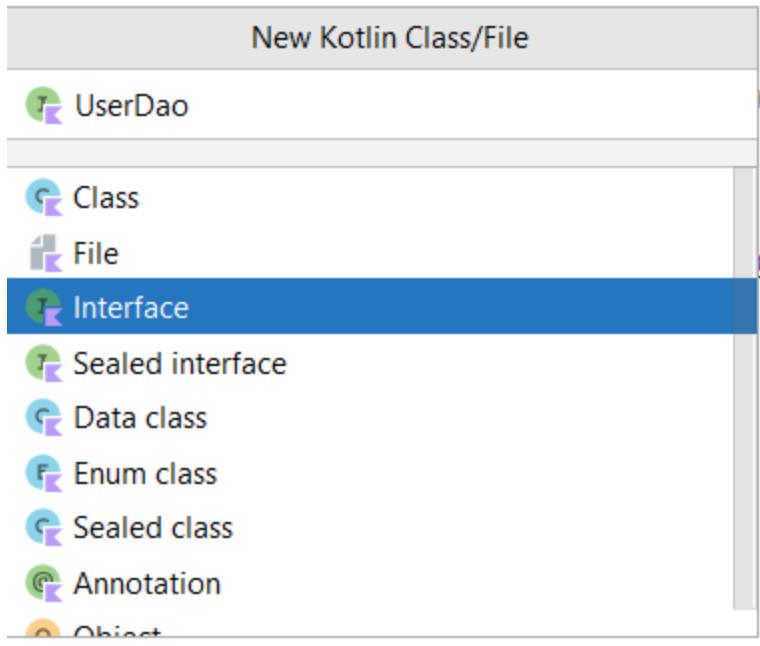


User data class code:

<https://github.com/smartinternz02/Travel-Plan-App/blob/master/app/src/main/java/com/example/travelapp/User.kt>

Step 2 : Create an UserDao interface

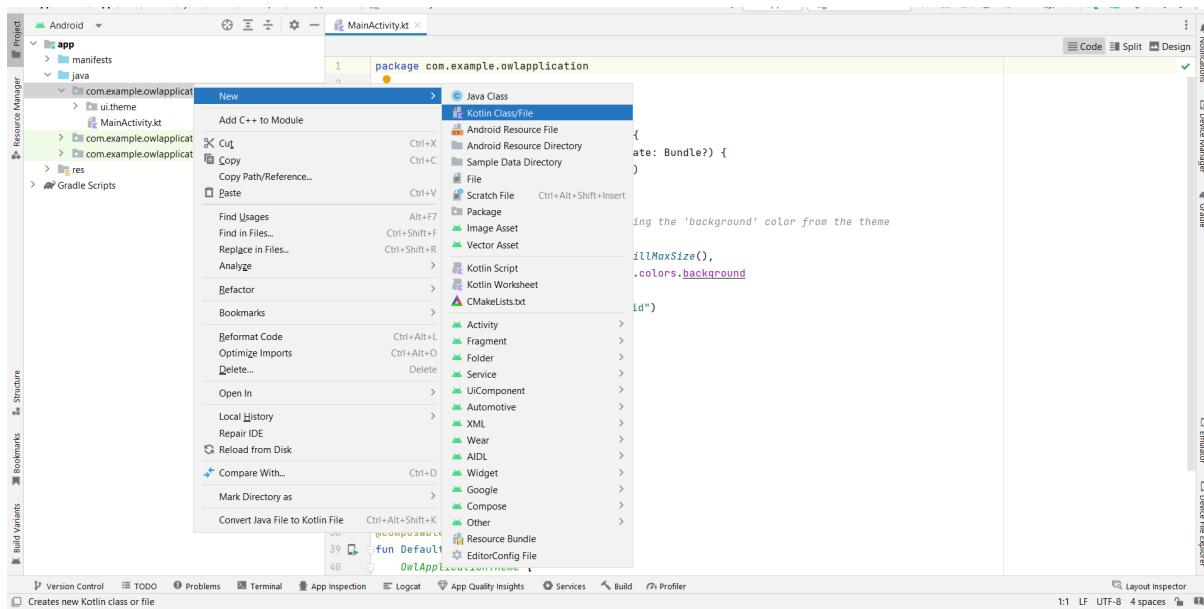


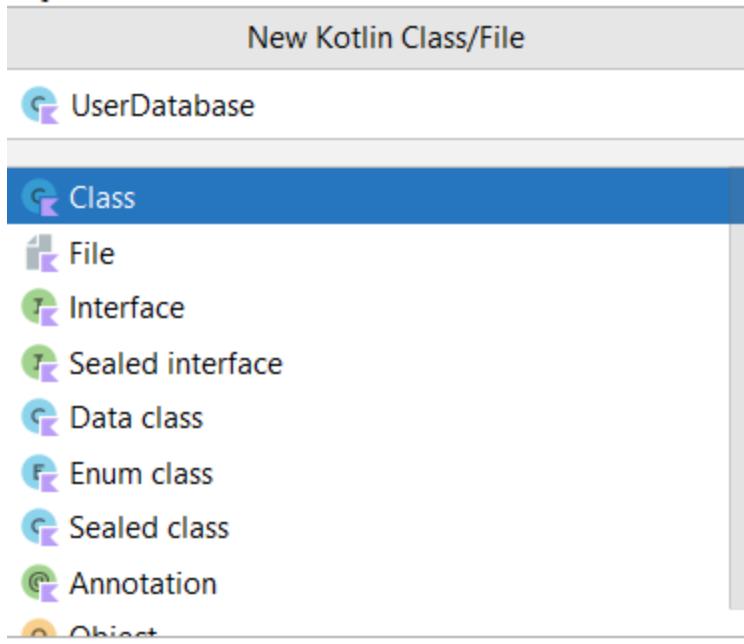


UserDao interface code :

<https://github.com/smartzinternz02/Travel-Plan-App/blob/master/app/src/main/java/com/example/travelapp/UserDao.kt>

Step 3 : Create an UserDatabase class

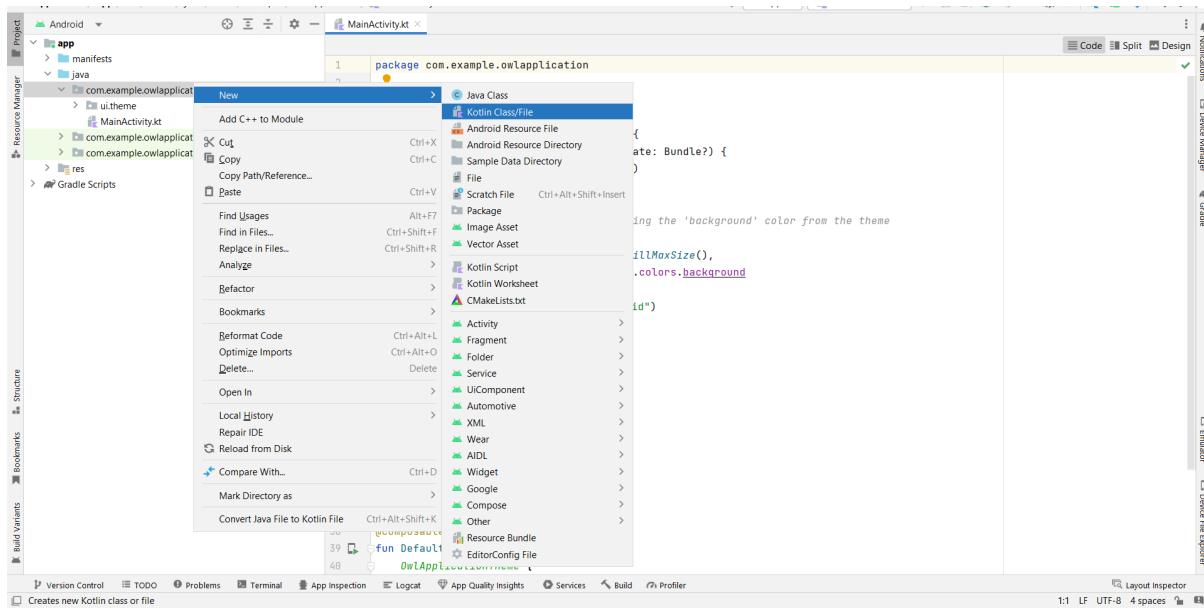


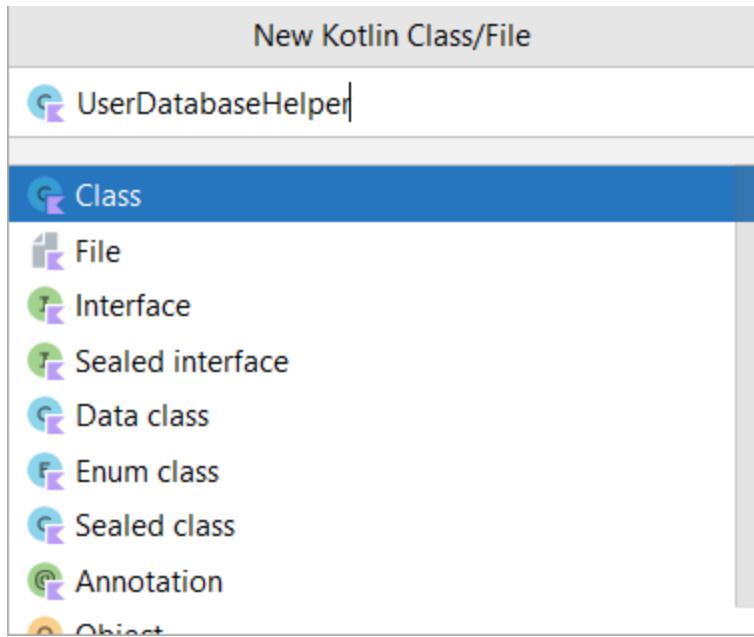


UserDatabase class code :

<https://github.com/smarterinternz02/Travel-Plan-App/blob/master/app/src/main/java/com/example/travelapp/UserDatabase.kt>

Step 4 : Create an UserDatabaseHelper class





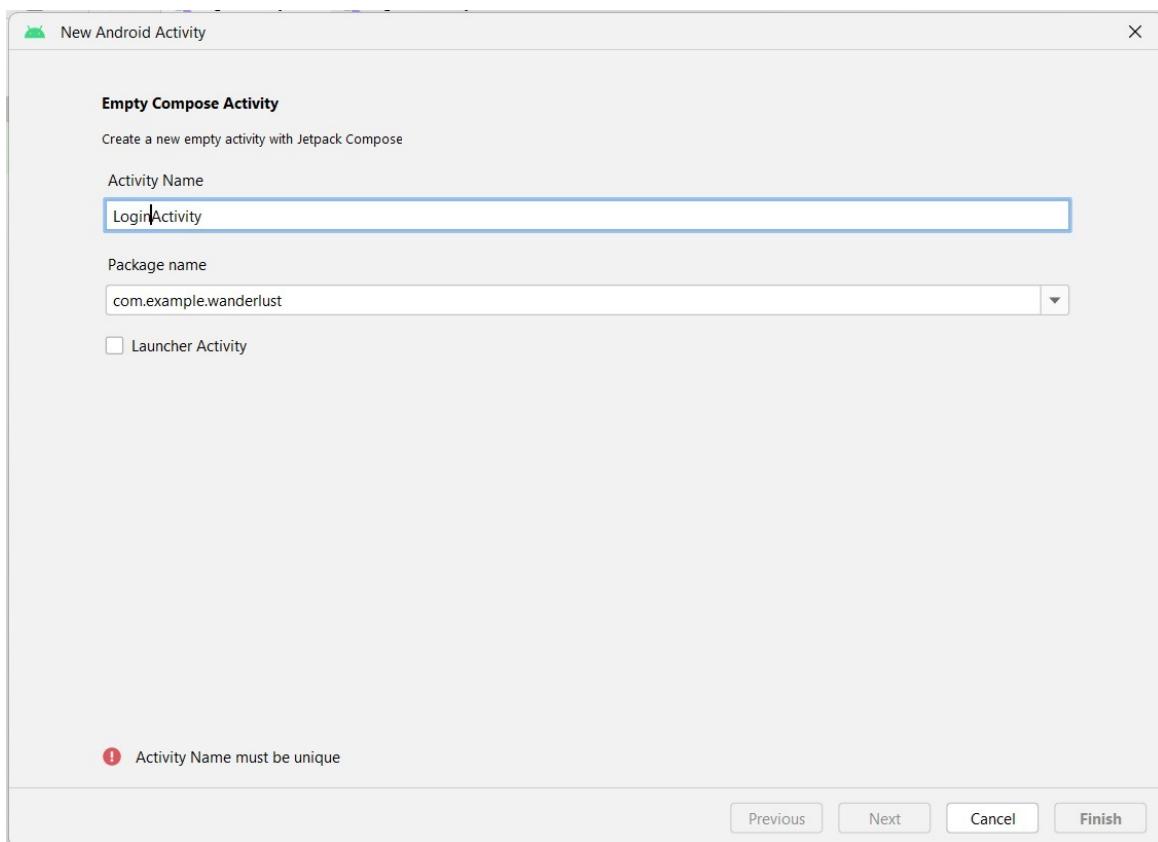
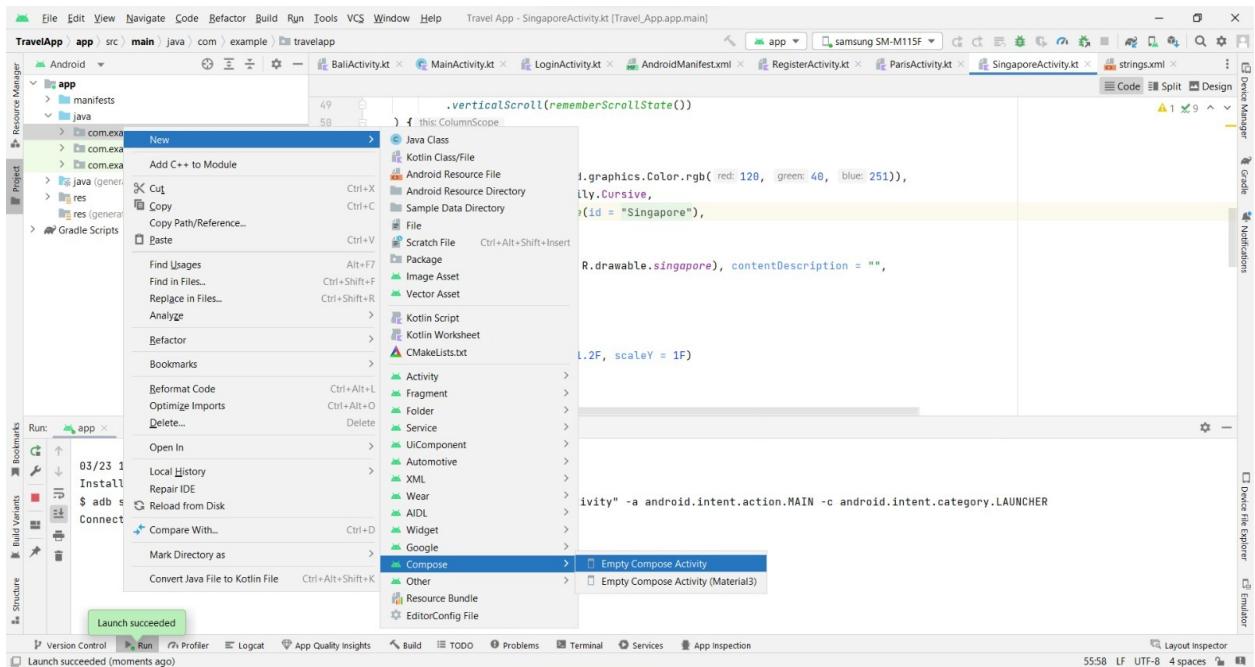
UserDatabaseHelper class code :

<https://github.com/smartinternz02/Travel-Plan-App/blob/master/app/src/main/java/com/example/travelapp/UserDatabaseHelper.kt>

Task 5:

Building application UI and connecting to database.

Step 1: Creating LoginActivity.kt with database



Database connection in LoginActivity.kt

```

package com.example.travelapp

import ...

class LoginActivity : ComponentActivity() {
    private lateinit var databaseHelper: UserDatabaseHelper
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        databaseHelper = UserDatabaseHelper(context)
        setContent {
            LoginScreen(context, 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 = "") }

    Column(
        modifier = Modifier.fillMaxSize().background(Color.White),
        horizontalAlignment = Alignment.CenterHorizontally,
        verticalArrangement = Arrangement.Center
    ) { this@ColumnScope

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

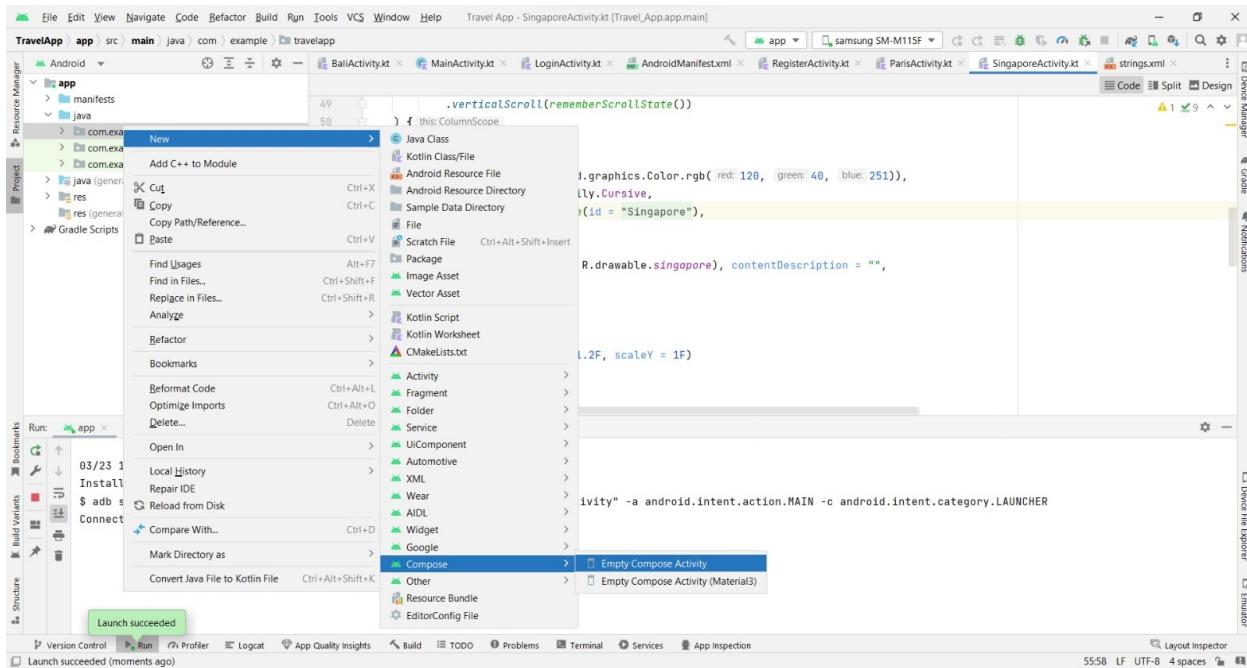
        Text(

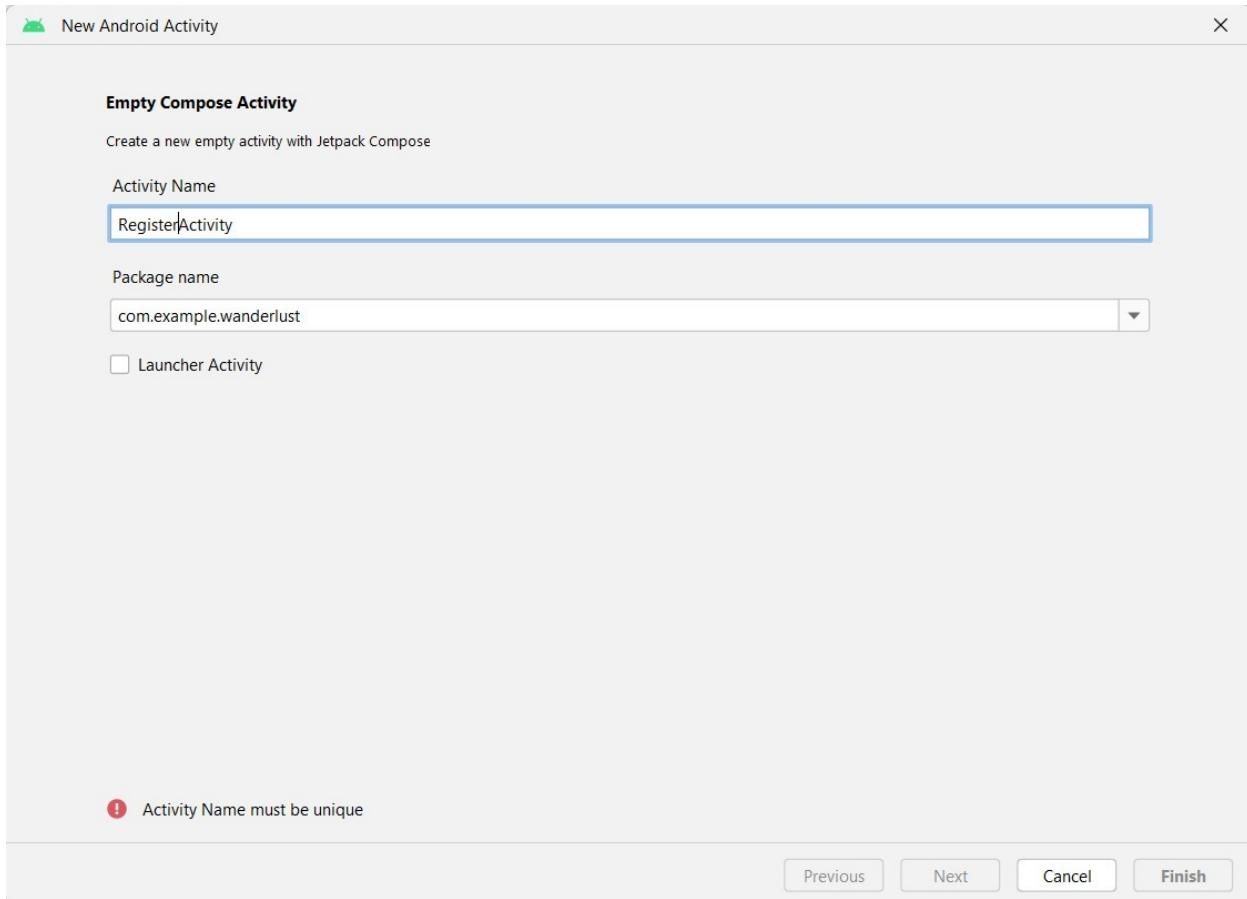
```

Complete code in below link:

<https://github.com/smartinternz02/Travel-Plan-App/blob/master/app/src/main/java/com/example/travelapp/LoginActivity.kt>

Step 2 : Creating RegisterActivity.kt with database





Database connection in RegisterActivity.kt

```

package com.example.travelapp

import ...

class RegisterActivity : ComponentActivity() {
    private lateinit var databaseHelper: UserDatabaseHelper
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        databaseHelper = UserDatabaseHelper(context: this)
        setContent {
            RegistrationScreen(context: 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("") }

    Column(
        modifier = Modifier.fillMaxSize().background(Color.White),
        horizontalAlignment = Alignment.CenterHorizontally,
        verticalArrangement = Arrangement.Center
    ) { this@ColumnScope
        Image(painterResource(id = R.drawable.tre), contentDescription = "")
    }
}

```

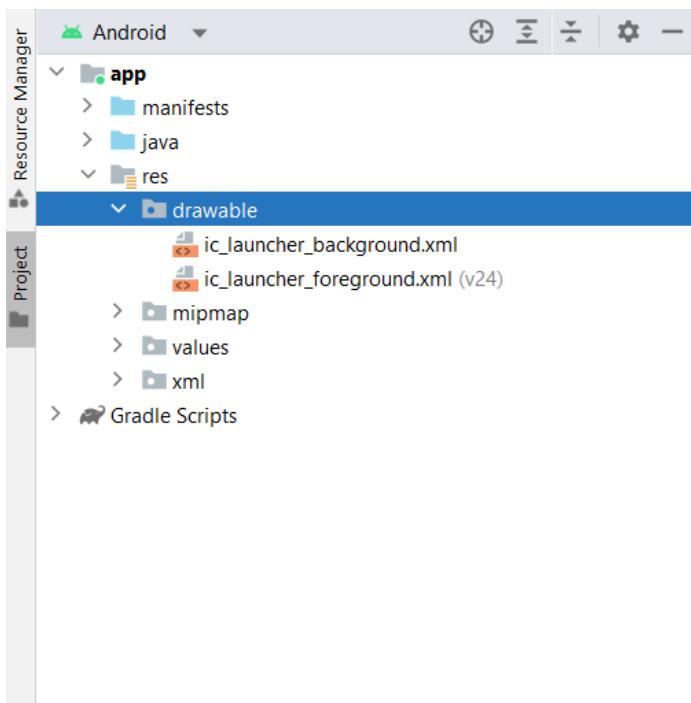
Complete code in below link:

<https://github.com/smartinternz02/Travel-Plan-App/blob/master/app/src/main/java/com/example/travelapp/RegisterActivity.kt>

Step 3 : Creating MainActivity.kt file

In MainActivity.kt file the main application is developed

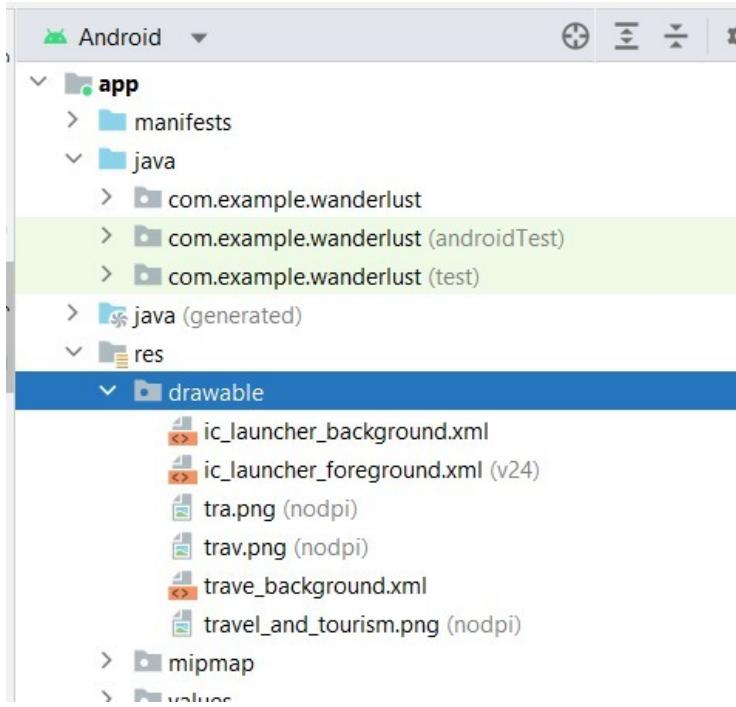
- Before creating UI we need to add some images in drawables which are in res



Download the required drawable from the code:

<https://github.com/smartinternz02/Travel-Plan-App/tree/master/app/src/main/res/drawable-nodpi>

<https://github.com/smartinternz02/Travel-Plan-App/tree/master/app/src/main/res/drawable>



MainActivity.kt

```

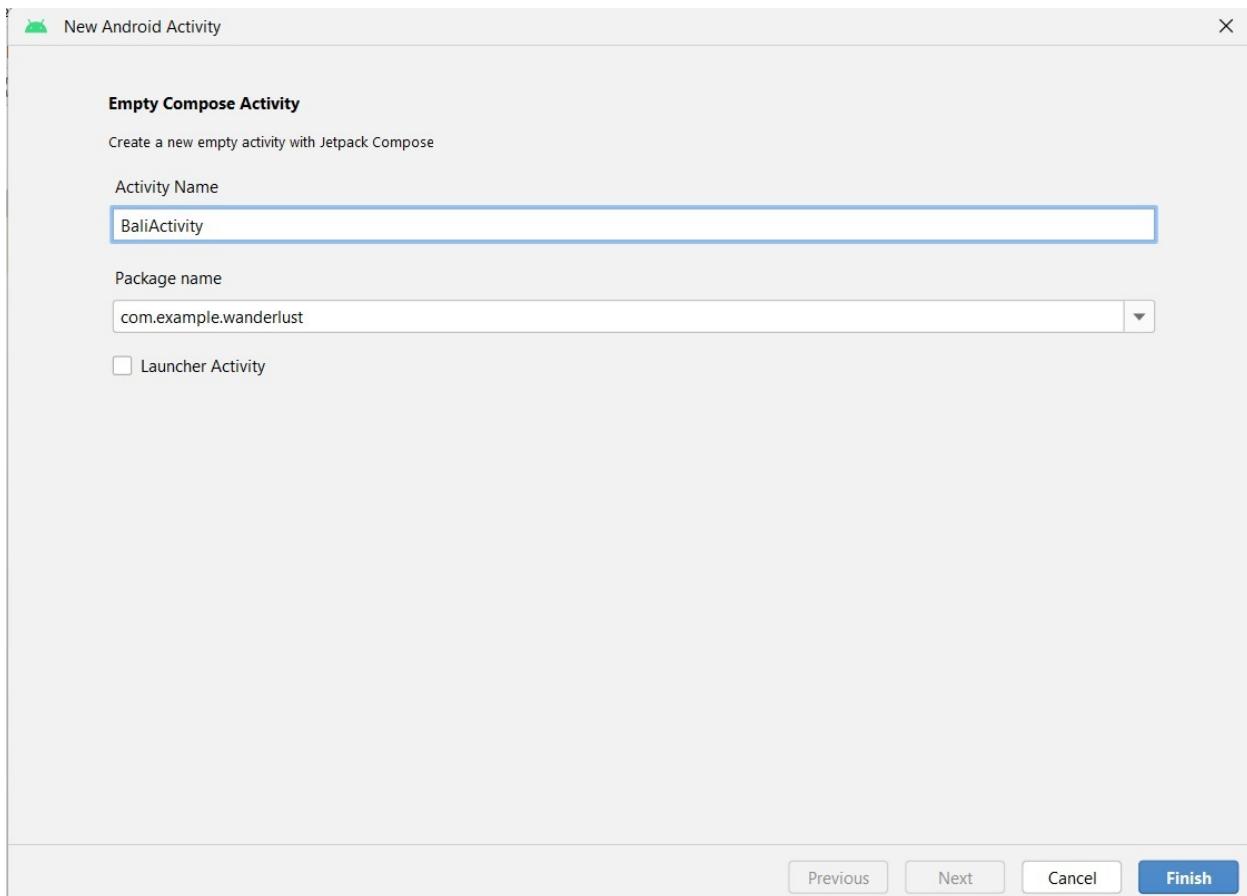
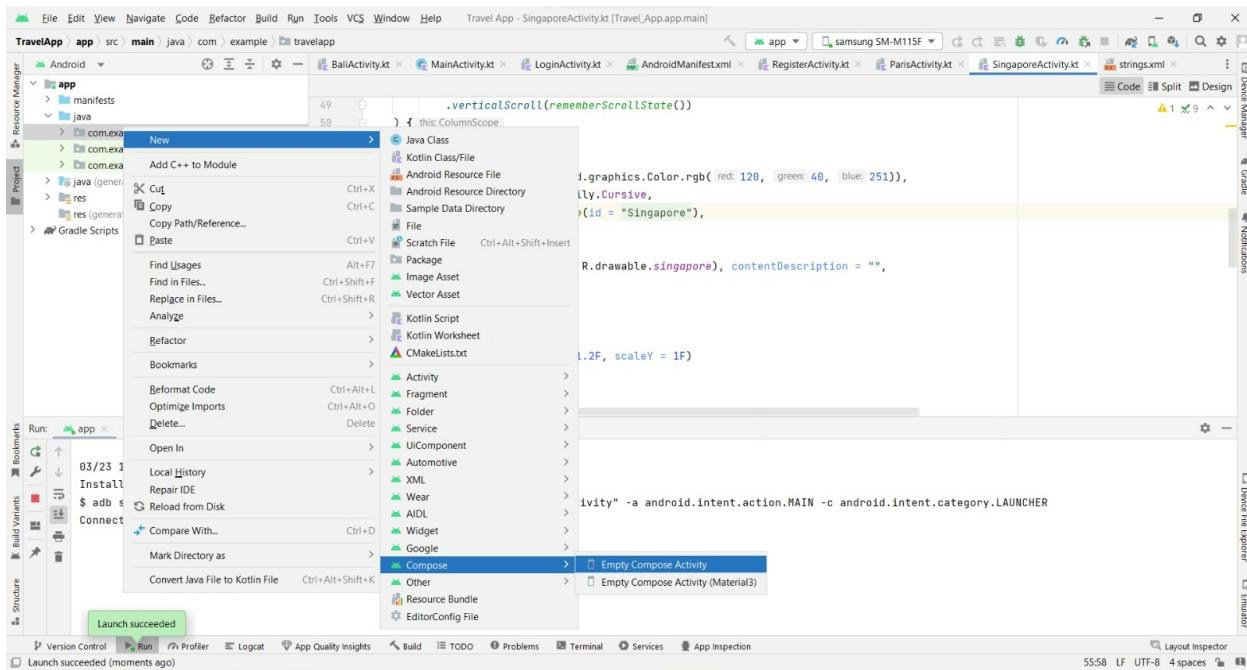
1 package com.example.travelapp
2
3 import ...
4
5 class MainActivity : ComponentActivity() {
6     override fun onCreate(savedInstanceState: Bundle?) {
7         super.onCreate(savedInstanceState)
8         setContent {
9             TravelApp(context: this)
10        }
11    }
12
13 @Composable
14 fun TravelApp(context: Context) {
15     Column(
16         modifier = Modifier
17             .padding(20.dp)
18             .verticalScroll(rememberScrollState())
19     ) {
20         this: ColumnScope
21
22         Text(
23             fontSize = 40.sp,
24             color = Color(android.graphics.Color.rgb(red: 128, green: 40, blue: 251)),
25             fontFamily = FontFamily.Cursive,
26             text = "Wanderlust Travel"
27         )
28
29         Spacer(modifier = Modifier.height(20.dp))
30     }
31 }
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53

```

Complete code in below link:

<https://github.com/smartinternz02/Travel-Plan-App/blob/master/app/src/main/java/com/example/travelapp/MainActivity.kt>

Step 4 : Creating BaliActivity.kt file



BaliActivity.kt

The screenshot shows the Android Studio interface with the BaliActivity.kt file open in the mainActivity.kt tab. The code is a Jetpack Compose Activity. The code block is as follows:

```
package com.example.travelapp

import ...

class BaliActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            TravelAppTheme {
                // A surface container using the 'background' color from the theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background
                ) {
                    PlaceOne()
                }
            }
        }
    }

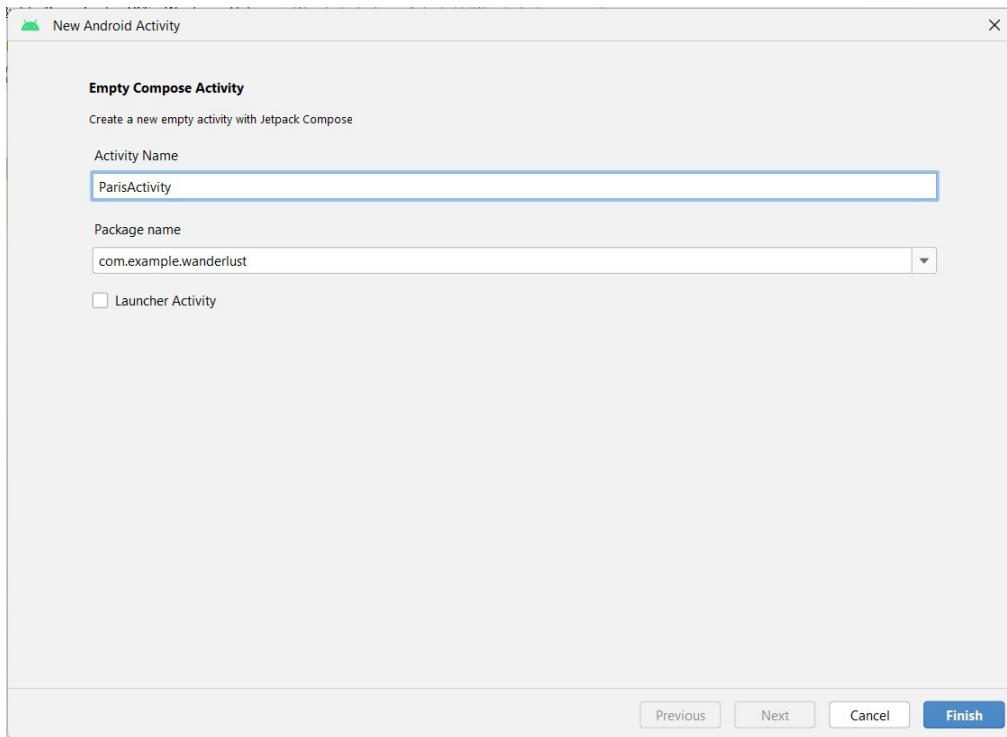
    @Composable
    fun PlaceOne() {
        Column(modifier = Modifier.background(color = Color.White)
            .padding(20.dp)
            .verticalScroll(rememberScrollState()))
        ) { this@ColumnScope
            Text(
                fontSize = 40.sp,
                color = Color(android.graphics.Color.rgb(red: 120, green: 40, blue: 251)),
            )
        }
    }
}
```

The Project structure on the left shows the app module with various activities like LoginActivity, MainActivity, ParisActivity, RegisterActivity, and SingaporeActivity. The code editor has syntax highlighting and code completion suggestions.

Complete code in below link:

<https://github.com/smartinternz02/Travel-Plan-App/blob/master/app/src/main/java/com/example/travelapp/BaliActivity.kt>

Step 5 : Creating ParisActivity.kt file



ParisActivity.kt file

The screenshot shows the Android Studio interface with the code editor open to the `ParisActivity.kt` file. The code is written in Jetpack Compose. The project structure on the left includes files like `MainActivity.kt`, `BaliActivity.kt`, and `Greeting.kt`. The code itself defines a `ParisActivity` class that extends `ComponentActivity`. It overrides the `onCreate` method to set the content view to a `Surface` component with a white background and a `Greeting` composable. The `Greeting` composable contains a `Column` with a `Text` element.

```
package com.example.travelapp

import ...

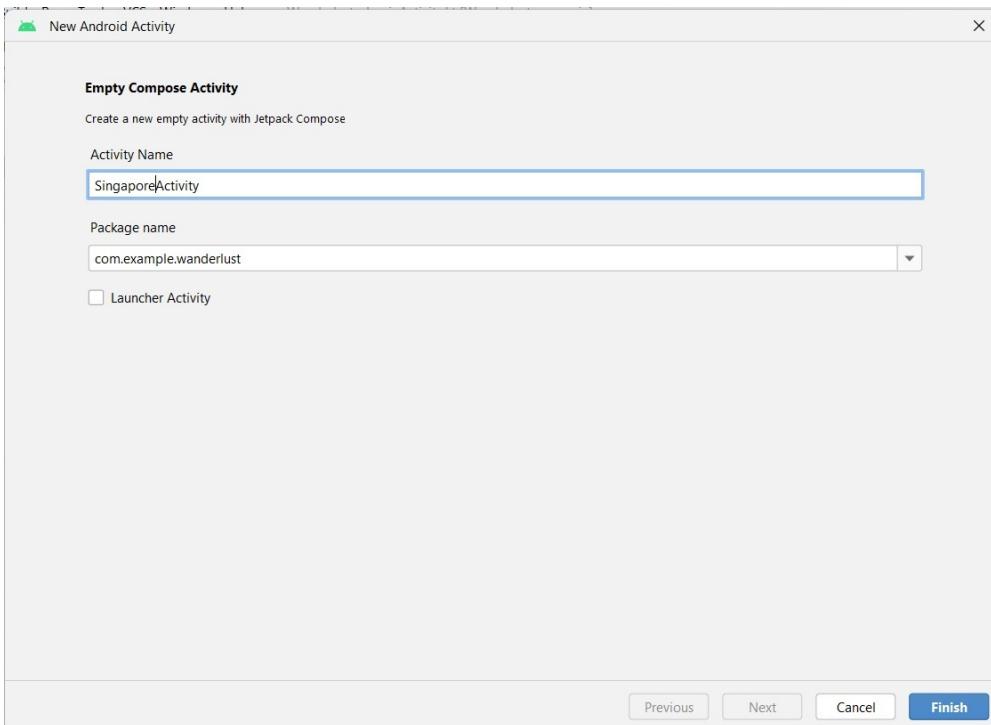
class ParisActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            TravelAppTheme {
                // A surface container using the 'background' color from the theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background
                ) {
                    Greeting()
                }
            }
        }
    }

    @Composable
    fun Greeting() {
        Column(
            modifier = Modifier.background(color = Color.White)
                .padding(20.dp)
                .verticalScroll(rememberScrollState())
        ) {
            Text(
                ...
            )
        }
    }
}
```

Complete code in below link:

<https://github.com/smarterinternz02/Travel-Plan-App/blob/master/app/src/main/java/com/example/travelapp/ParisActivity.kt>

Step 6 : Creating SingaporeActivity.kt file



SingaporeActivity.kt file

The screenshot shows the Android Studio interface with the code editor open to the `SingaporeActivity.kt` file. The code is a Kotlin file defining a class `SingaporeActivity` that extends `ComponentActivity`. It overrides the `onCreate` method and sets the content view to a `TravelAppTheme` surface with a white background and padding. It also defines a `Greeting2` composable function that displays a column of text.

```
package com.example.travelapp

import ...

class SingaporeActivity : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            TravelAppTheme {
                // A surface container using the 'background' color from the theme
                Surface(
                    modifier = Modifier.fillMaxSize(),
                    color = MaterialTheme.colors.background
                ) {
                    Greeting2()
                }
            }
        }
    }

    @Composable
    fun Greeting2() {
        Column(
            modifier = Modifier.background(color = Color.White)
                .padding(20.dp)
                .verticalScroll(rememberScrollState())
        ) { this@ColumnScope
            Text(
                text = "Hello from Singapore!"
            )
        }
    }
}
```

Complete code in below link:

<https://github.com/smarterinternz02/Travel-Plan-App/blob/master/app/src/main/java/com/example/travelapp/SingaporeActivity.kt>

Task 6:

Modifying AndroidManifest.xml

```

<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="Travel App"
        android:supportsRtl="true"
        android:theme="@style/Theme.TravelApp"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:label="MainActivity"
            android:theme="@style/Theme.TravelApp" />
        <activity
            android:name=".BaliActivity"
            android:exported="false"
            android:label="BaliActivity"
            android:theme="@style/Theme.TravelApp" />
        <activity
            android:name=".ParisActivity"
            android:exported="false"
            android:label="ParisActivity"
            android:theme="@style/Theme.TravelApp" />
        <activity
            android:name=".SingaporeActivity"
            android:exported="false"
            android:label="SingaporeActivity"
            android:theme="@style/Theme.TravelApp" />
    </application>
</manifest>

```

When we run the app we will get the `MainActivity.kt` file as our first screen , but we want `LoginActivity.kt` , So we need to change in `AndroidManifest.xml`.

Changed AndroidManifest.xml

```

<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="Travel App"
        android:supportsRtl="true"
        android:theme="@style/Theme.TravelApp"
        tools:targetApi="31">
        <activity
            android:name=".BaliActivity"
            android:exported="false"
            android:label="BaliActivity"
            android:theme="@style/Theme.TravelApp" />
        <activity
            android:name=".ParisActivity"
            android:exported="false"
            android:label="ParisActivity"
            android:theme="@style/Theme.TravelApp" />
        <activity
            android:name=".SingaporeActivity"
            android:exported="false"
            android:label="SingaporeActivity"
            android:theme="@style/Theme.TravelApp" />
        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:label="Travel App"
            android:theme="@style/Theme.TravelApp" />
        <activity
            android:name=".LoginActivity"
            android:exported="true"
            android:label="Travel App"
            android:theme="@style/Theme.TravelApp" />
        <activity
            android:name=".RegisterActivity"
            android:exported="false"
            android:label="RegisterActivity"
            android:theme="@style/Theme.TravelApp" />
    </application>
</manifest>

```

Complete `AndroidManifest.xml` code:

<https://github.com/smartinternz02/Travel-Plan-App/blob/master/app/src/main/AndroidManifest.xml>

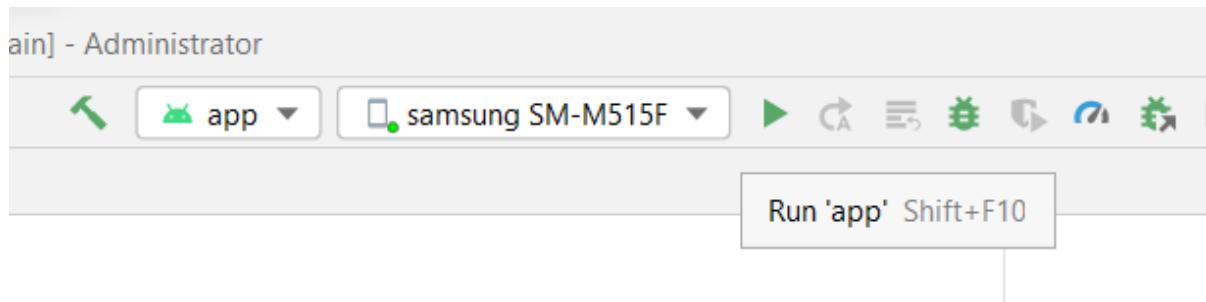
Task 7:

Running the application.

Step 1: Run apps on a hardware device

<https://developer.android.com/studio/run/device>

Step 2: Run the application in Mobile



Complete Project Link: <https://github.com/smartzinternz02/Travel-Plan-App>

Final Output of the Application :

Login Page :



Login

Login

[Register](#)

[Forget password?](#)

RegisterPage :



Register

Username

Email

Password

Register

Have an account? [Log in](#)



MainPage :

Wanderlust Travel



Bali

Super saver pack with less than \$10000
7days/2persons



Paris

Super saver pack with less than \$10000
7days/2persons



Singapore

Location Page:

Bali



Day 1: Arrival and Relaxation

Arrive in Bali and check into your hotel or accommodation.

Spend the day relaxing and getting acclimated to the island.

If you have time, explore the nearby area or head to the beach.

Day 2: Ubud Tour

Start your day early and head to Ubud, a cultural and artistic hub in Bali.

Visit the Monkey Forest and the Ubud Palace.

Take a tour of the Tegalalang Rice Terrace, a beautiful UNESCO World Heritage Site.

End your day with a traditional Balinese dance performance.

Day 3: Temple Hopping

Visit some of Bali's most famous temples, such as Tanah Lot and Uluwatu.

Take in the stunning views of the ocean and cliffs.

Enjoy a sunset dinner at one of the many restaurants near the temples.



