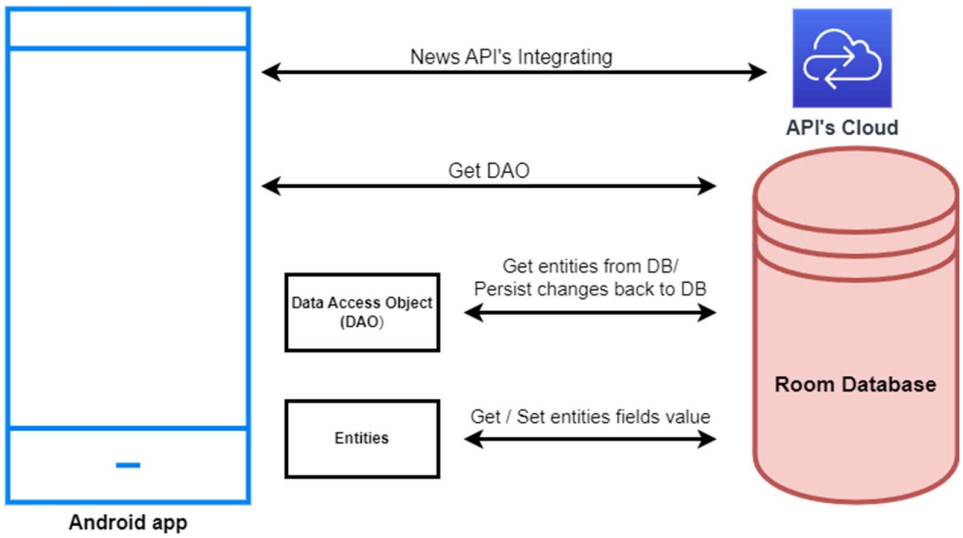
An Android Application for Keeping Up with the Latest Headlines

Project Based Experiential Learning Program

An Android Application for Keeping Up with the Latest Headlines :

The app's main feature is displaying a list of news articles, each with a title, image, and brief description. Users can scroll through the list of articles and tap on an article to view more details. The app uses the Jetpack Compose UI toolkit to build the UI and it uses the coil library to load images. The app fetches data from a remote server using Retrofit library and demonstrates how to use the Jetpack Compose UI toolkit for Android development.

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.
- You'll be able to integrate the API's accordingly.

Project Workflow:

- Users register into the application.
- After registration , user logins into the application.
- User enters into the main page

Tasks:

1.Required initial steps
2.Creating a new project.
3.Adding required dependencies.
4.Adding permissions
5. Creating the database classes.
6. Creating API Service and required classes for integrating API
7.Building application UI and connecting to database.
8.Modifying AndroidManifest.xml
9.Running the application.

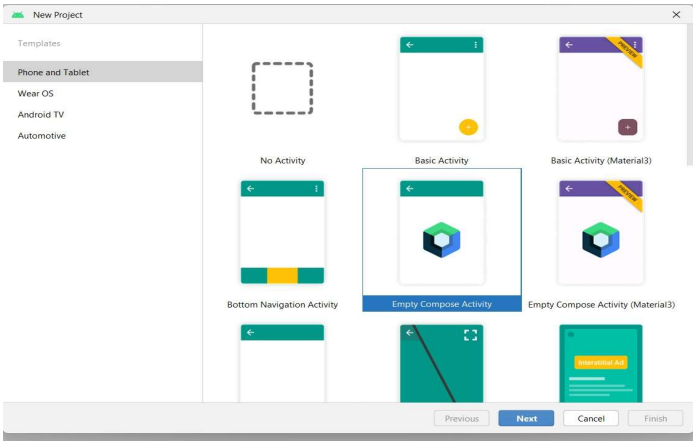
Task 1:

Required initial steps :

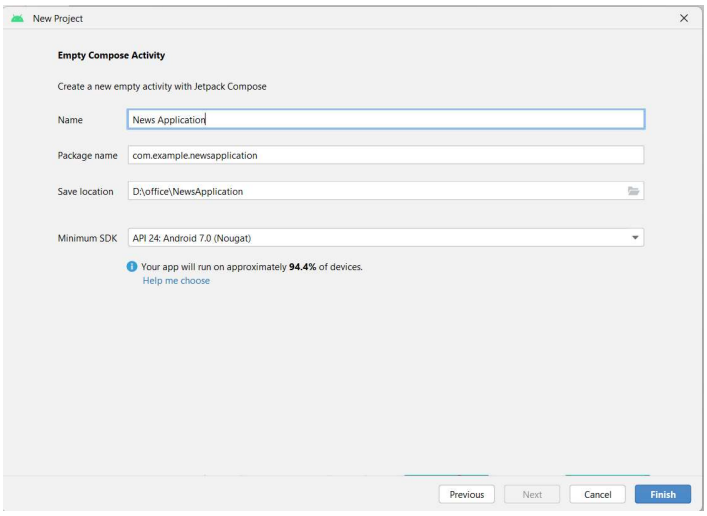
https://developer.android.com/studio/install

Task 2 :

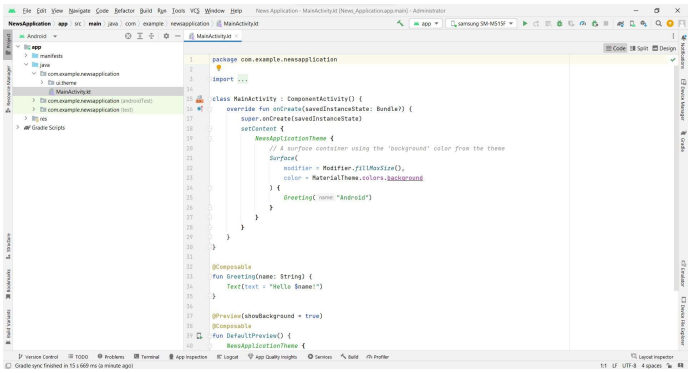
Creating a new project.
Step 1 : Android studio > File > New > New Project > Empty Compose Activity
Step 2 : Click on **Next** button.



Step 3 : Give name to the new project.
Step 4 : Give the Minimum SDK value
Step 5 : Click Finish



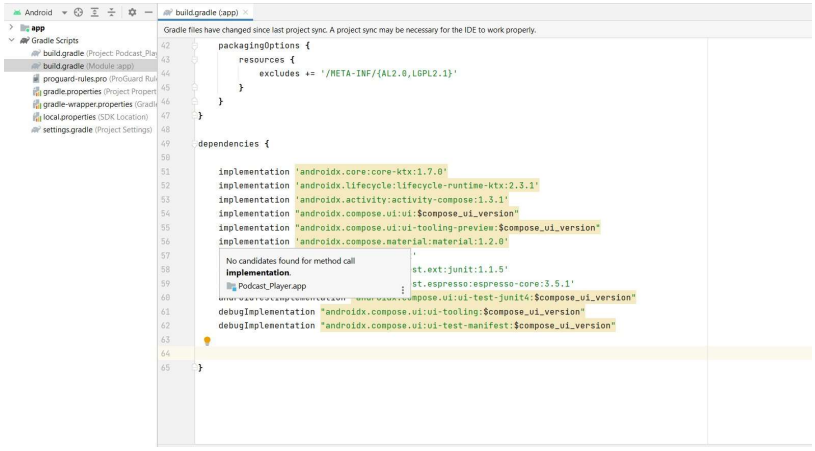
Main activity file



Task 3 :

Adding required dependencies.

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



Step 2 : Adding room dependencies.
Add the below code in dependencies

```
// Room Database
implementation 'androidx.room:room-common:2.5.0'
implementation 'androidx.room:room-ktx:2.5.0'
```

Step 3 : Adding Retrofit dependencies

```
// Retrofit
implementation 'com.squareup.retrofit2:retrofit:2.9.0'
implementation "com.squareup.okhttp3:okhttp:5.0.0-alpha.2"
implementation 'com.squareup.retrofit2:converter-gson:2.9.0'
```

Step 4 : Adding Coil dependencies

```
implementation("io.coil-kt:coil-compose:1.4.0")
```

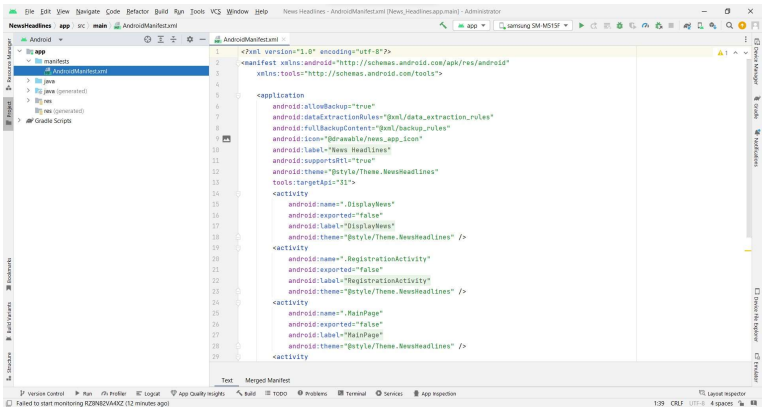
```
dependencies {

    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.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"


    // Room Database
    implementation 'androidx.room:room-common:2.5.0'
    implementation 'androidx.room:room-ktx:2.5.0'

    // Retrofit
    implementation 'com.squareup.retrofit2:retrofit:2.9.0'
    implementation "com.squareup.okhttp3:okhttp:5.0.0-alpha.2"
    implementation 'com.squareup.retrofit2:converter-gson:2.9.0'

    implementation("io.coil-kt:coil-compose:1.4.0")
}
```

Step 5 : Click on Sync now

Task 4:

Adding permissions

Step 1: Open AndroidManifest.xml.



Step 2: Add permission to access wifi and internet

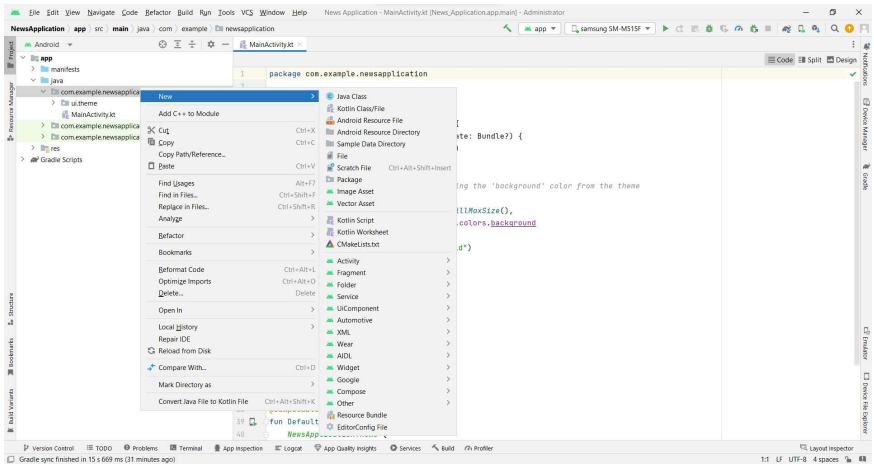
```
<!--    permissions-->
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
```

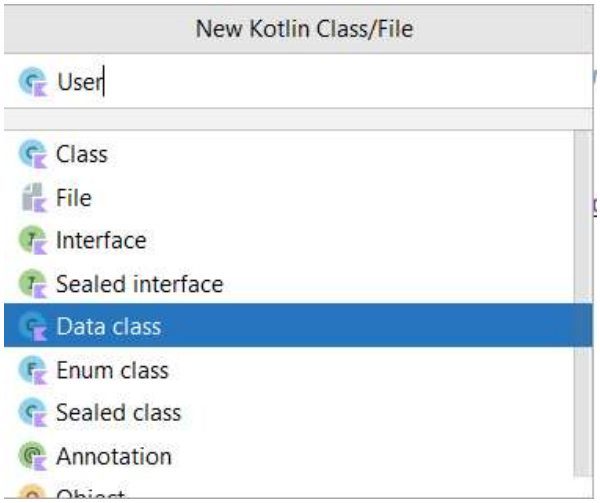
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
<!--    permissions-->
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@drawable/news_app_icon"
        android:label="News Headlines"
        android:supportsRtl="true"
        android:theme="@style/Theme.NewsHeadlines"
```

Task 5:

Creating the database classes.

Step 1 : Create User data class

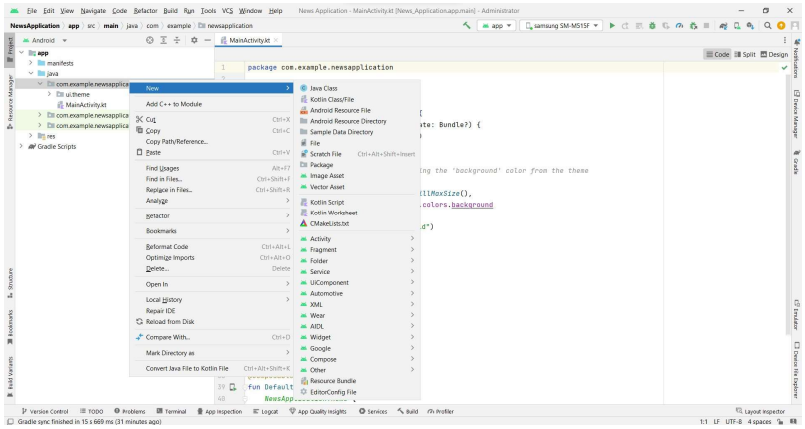


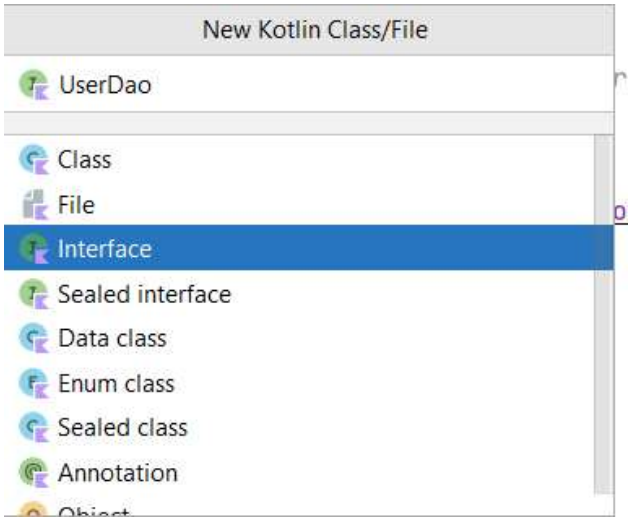


User class code:

https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/blob/master/app/src/main/java/com/example/newsheadlines/User.kt

Step 2 : Create an UserDao interface

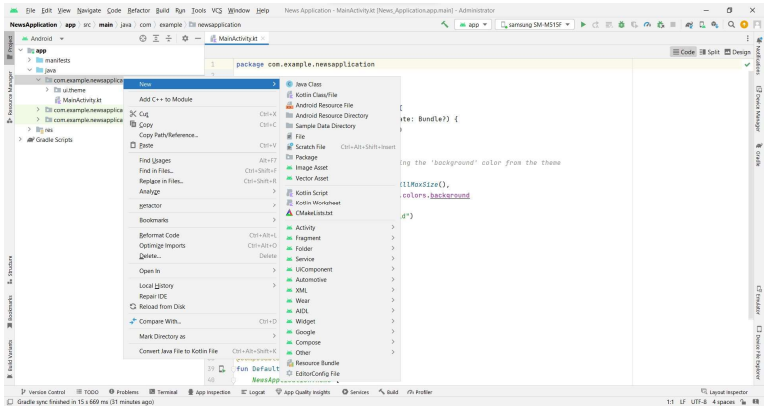


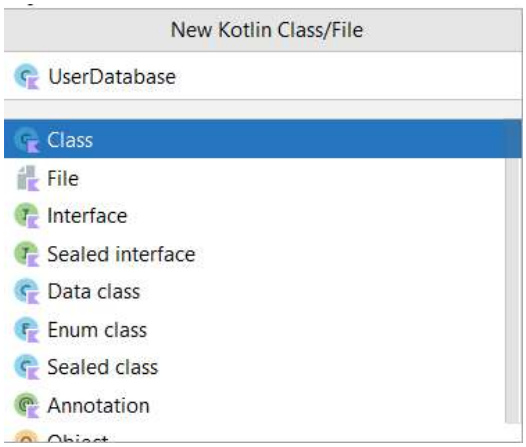


UserDao interface code:

https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/blob/master/app/src/main/java/com/example/newsheadlines/UserDao.kt

Step 3 : Create an UserDatabase class

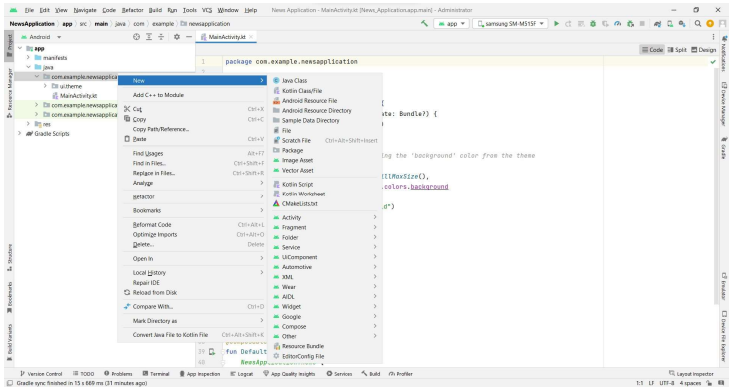


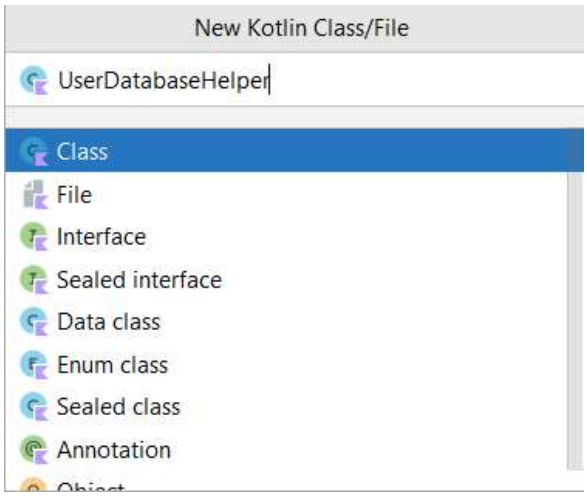


UserDatabase class code :

https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/blob/master/app/src/main/java/com/example/newsheadlines/UserDatabase.kt

Step 4 : Create an UserDatabaseHelper class





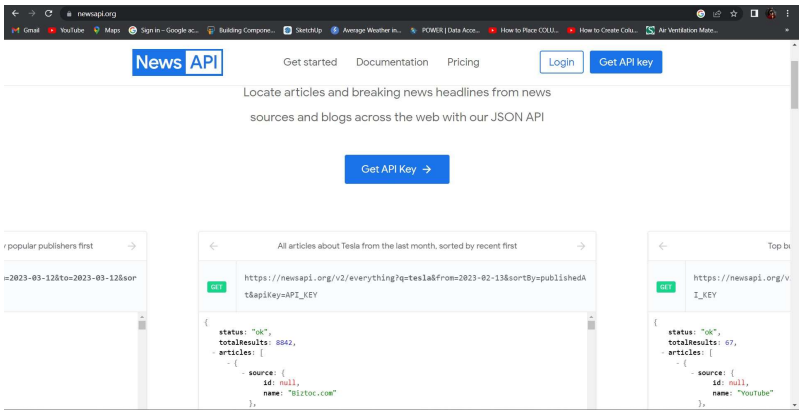
UserDatabaseHelper class code :

https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/blob/master/app/src/main/java/com/example/newsheadlines/UserDatabaseHelper.kt

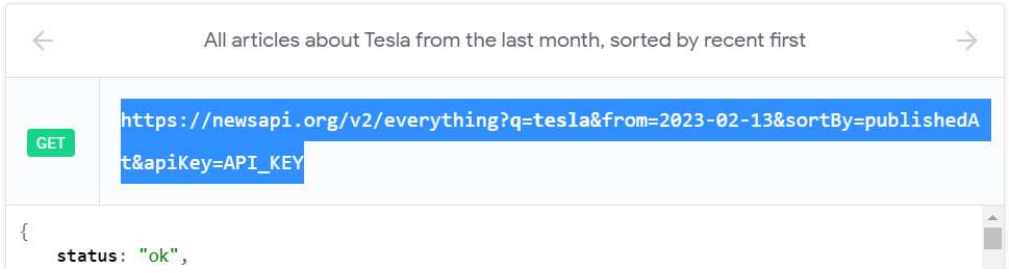
Task 6 :

Creating API Service and required classes for integrating API

Step 1: Create a API key for the required API

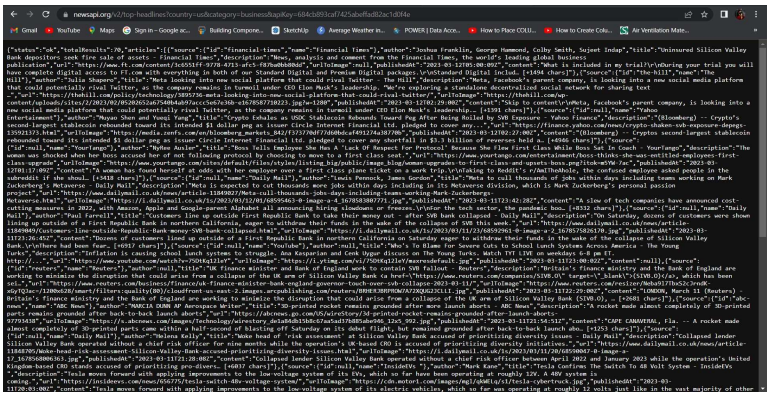


- Code which is needed to copy:

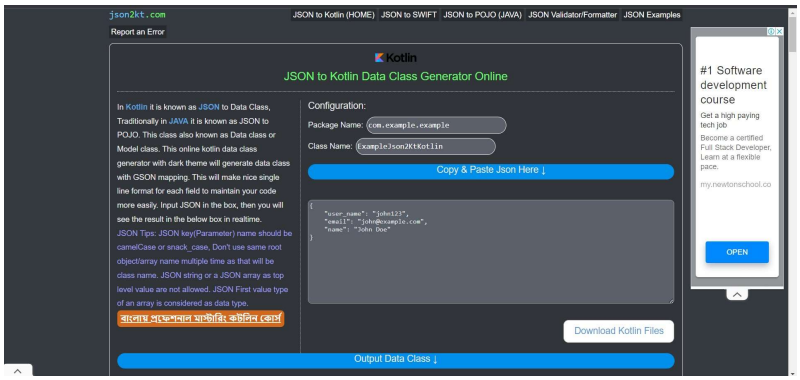


- Enter the API key at API_KEY to get the complete json file.

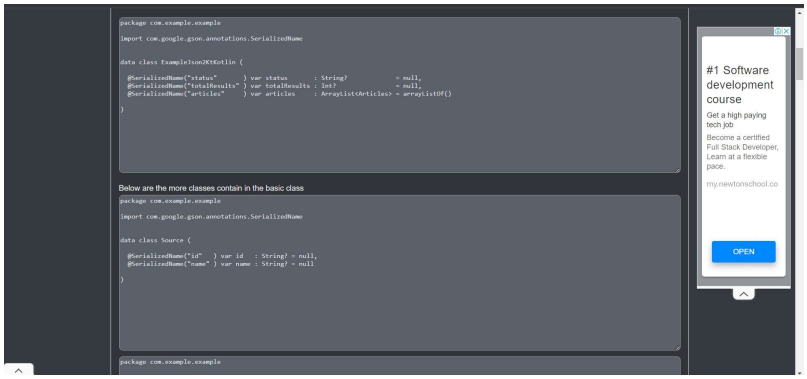
Json File



Step 2: Copy and paste the complete json file to json2kt.com



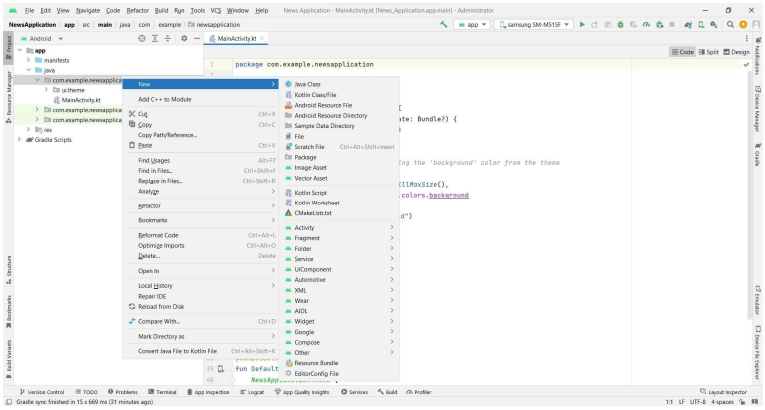
Click on Output Data class

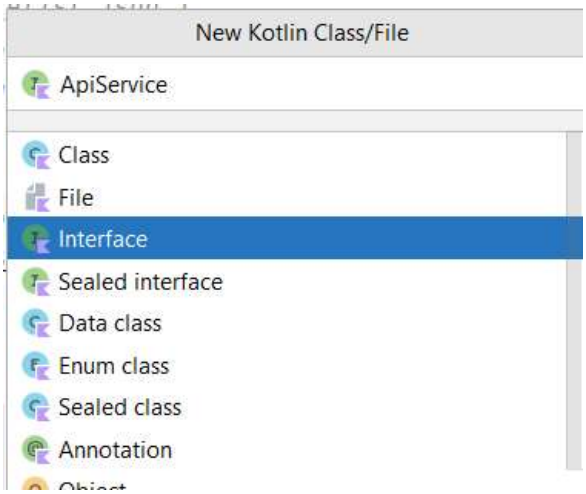


Now use this classes in android studio

Database for news integration into project

Step 3:
Create ApiService interface



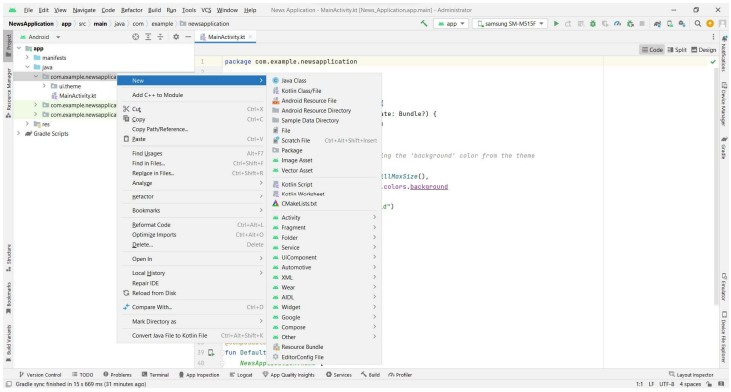


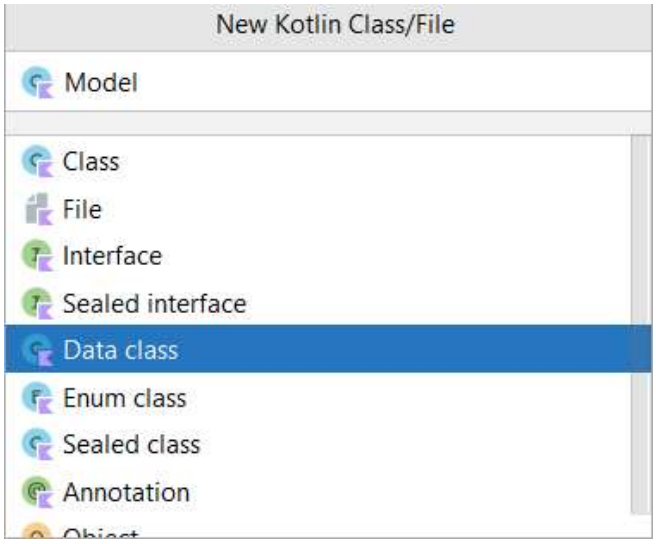

```kotlin
package com.example.newsheadlines

import ...

interface ApiService {

    @GET("top-headlines?country=us&category=business&apiKey=684cb893caf7425abeffad82ac1d0f4e")
    suspend fun getMovies() :News

    companion object {
        var apiService: ApiService? = null
        fun getInstance() : ApiService {
            if (apiService == null) {
                apiService = Retrofit.Builder()
                    .baseUrl("https://newsapi.org/v2/")

                    .addConverterFactory(GsonConverterFactory.create())
                    .build().create(ApiService::class.java)
            }
            return apiService!!
        }
    }

}
```

ApiService interface code:

https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/blob/master/app/src/main/java/com/example/newsheadlines/ApiService.kt

Step 4:
Create Model data class



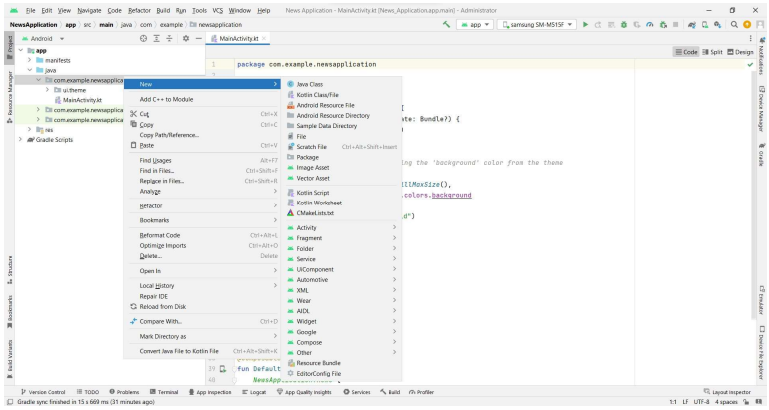


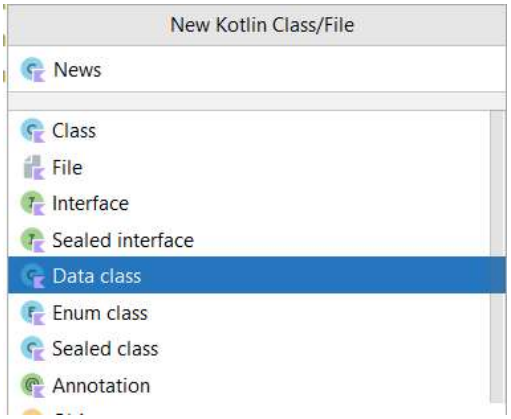

```
package com.example.newsheadlines


data class Movie(val name: String,
                 val imageUrl: String,
                 val desc: String,
                 val category: String)
```

Model class code:

https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/blob/master/app/src/main/java/com/example/newsheadlines/Model.kt

Step 5:
Create News data class



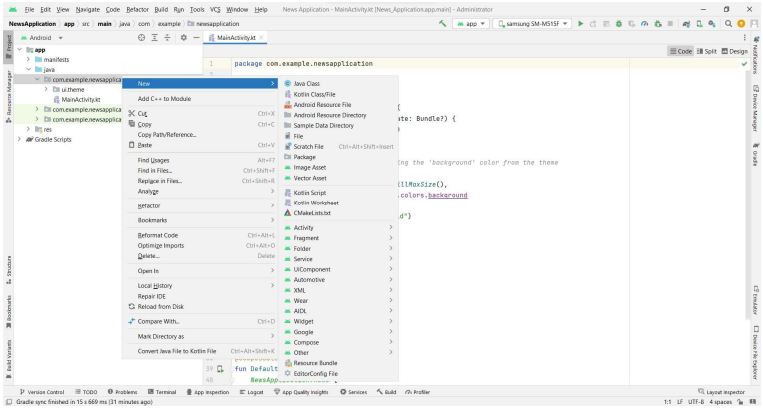


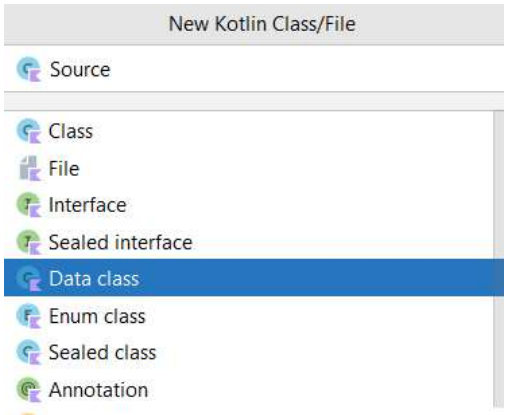
```kotlin
package com.example.newsheadlines

import ...


data class News (
    @SerializedName("status") var status:String?= null,
    @SerializedName("totalResults") var totalResults : Int?                 = null,
    @SerializedName("articles") var articles      : ArrayList<Articles> = arrayListOf()
)
```

News class code:

https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/blob/master/app/src/main/java/com/example/newsheadlines/News.kt

Step 6:
Create Source data class



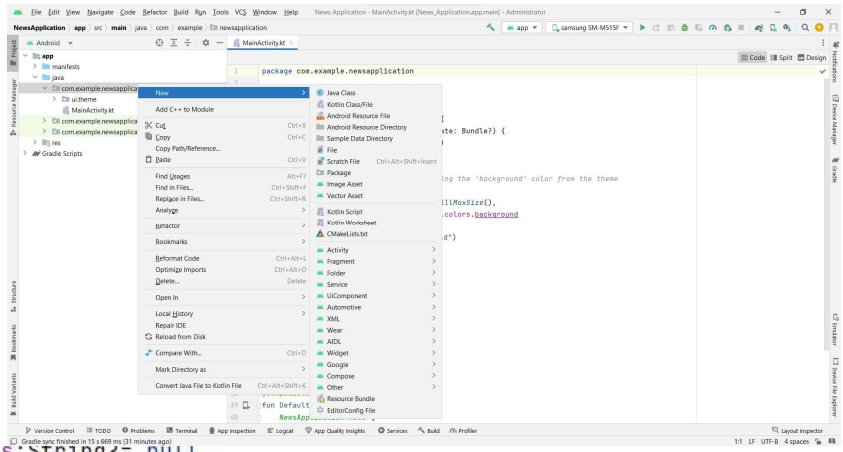


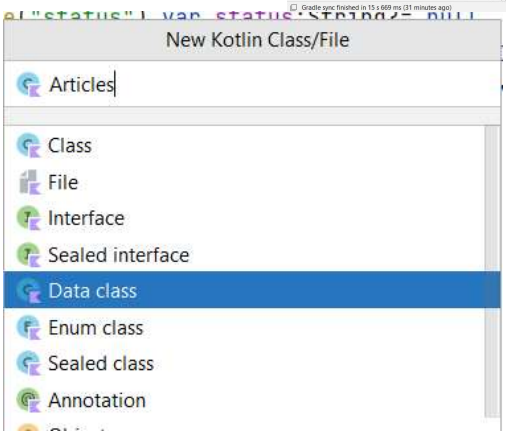
```kotlin
package com.example.example

import com.google.gson.annotations.SerializedName


data class Source (

  @SerializedName("id"   ) var id   : String? = null,
  @SerializedName("name" ) var name : String? = null

)
```

Model class code:

https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/blob/master/app/src/main/java/com/example/newsheadlines/Source.kt

Step 7:
Create Articles data class



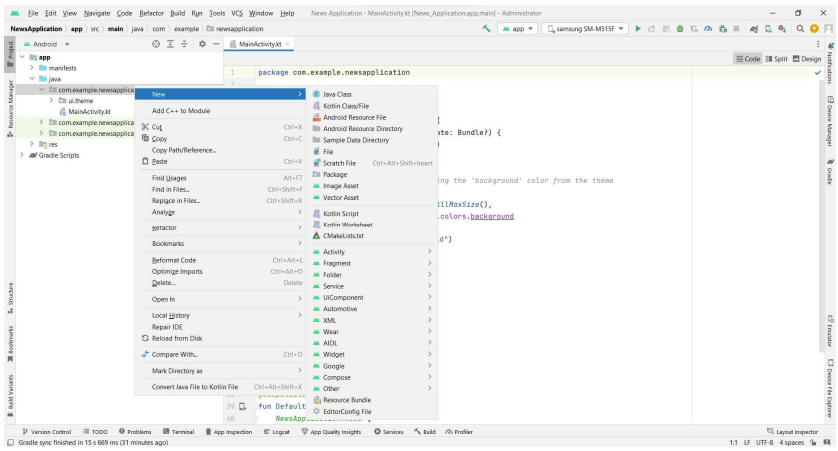


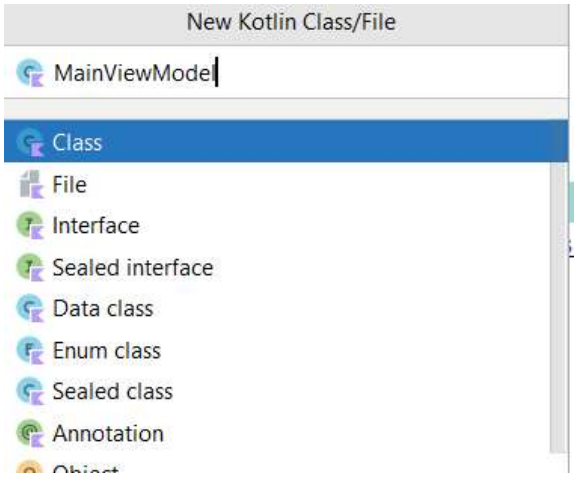
```kotlin
package com.example.example

import com.google.gson.annotations.SerializedName


data class Articles (

  @SerializedName("title"       ) var title       : String? = null,
  @SerializedName("description" ) var description : String? = null,
  @SerializedName("urlToImage"  ) var urlToImage  : String? = null,

)
```

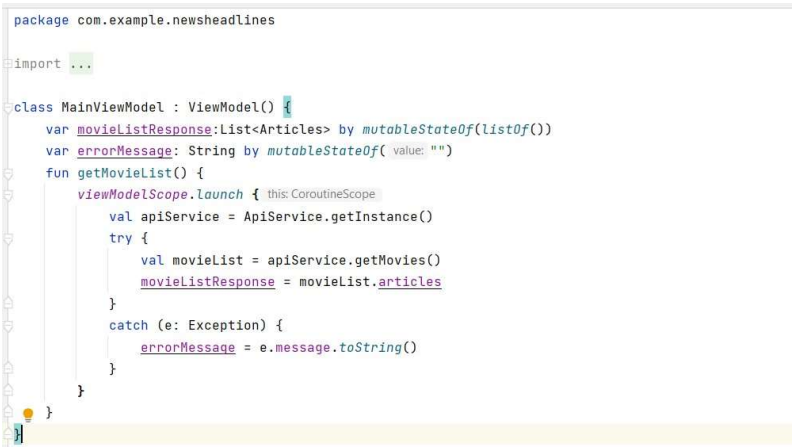
Article class code:

https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/blob/master/app/src/main/java/com/example/newsheadlines/Articles.kt

Step 8:
Create MainViewModel class







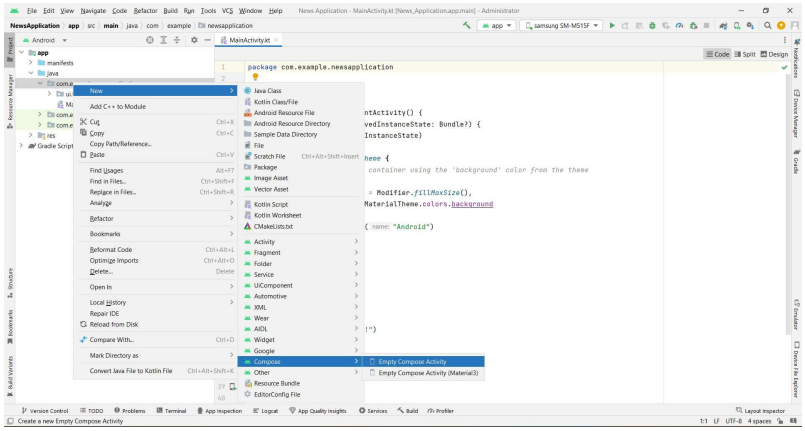
```
package com.example.newsheadlines

import ...

class MainViewModel : ViewModel() {
    var movieListResponse:List<Articles> by mutableStateOf(listOf())
    var errorMessage: String by mutableStateOf( value: "")
    fun getMovieList() {
        viewModelScope.launch { this: CoroutineScope
            val apiService = ApiService.getInstance()
            try {
                val movieList = apiService.getMovies()
                movieListResponse = movieList.articles
            }
            catch (e: Exception) {
                errorMessage = e.message.toString()
            }
        }
    }
}
```

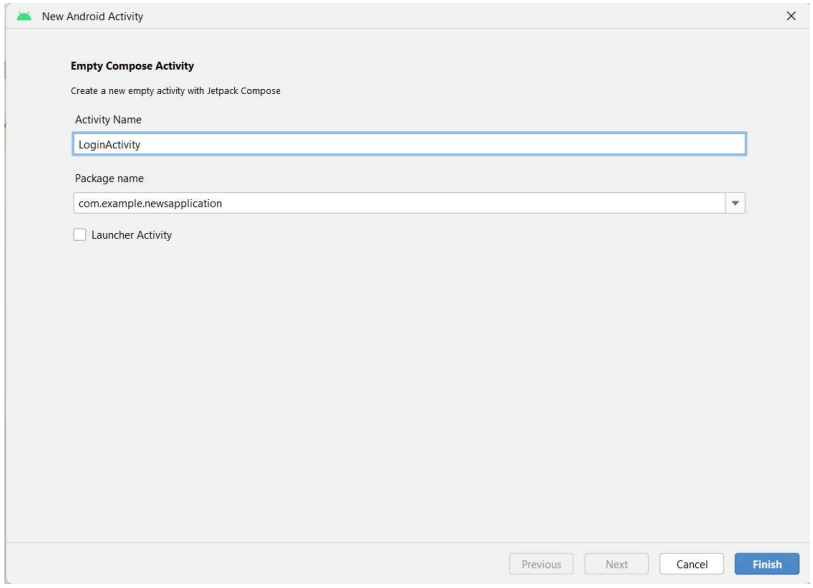
MainViewModel class code:

https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/blob/master/app/src/main/java/com/example/newsheadlines/MainViewModel.kt

Task 7:

Building application UI and connecting to database.
Step 1: Creating LoginActivity.kt with database





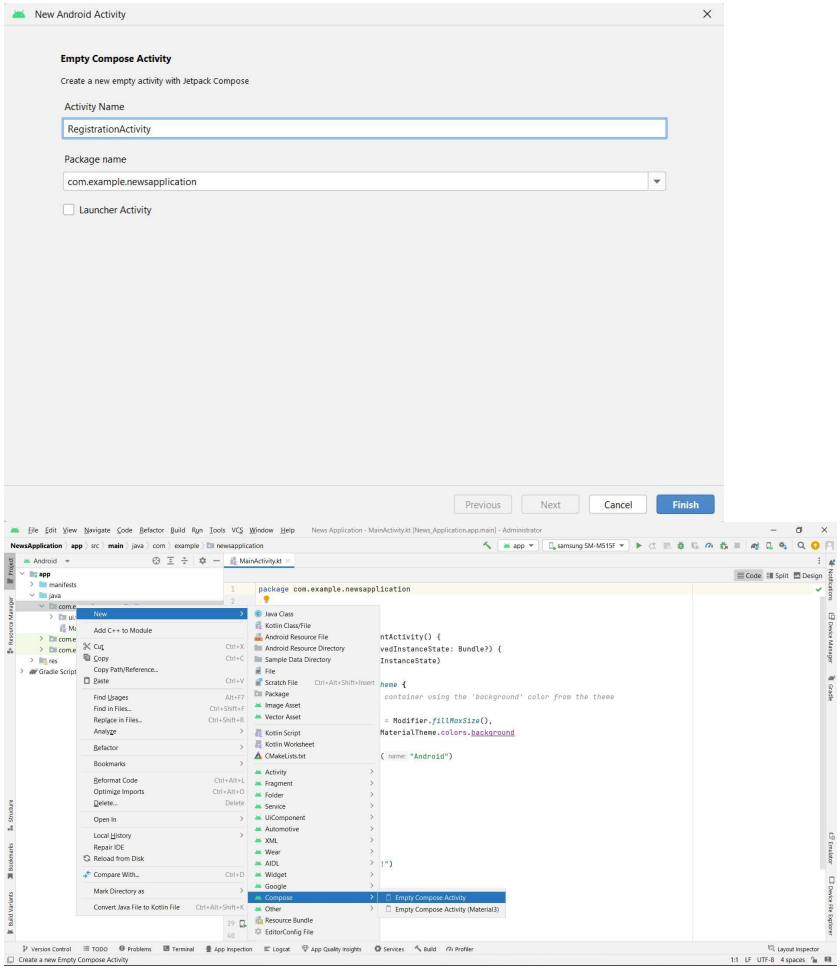
Database connection in LoginActivity.kt

```
package com.example.newsheadlines

import ...

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

            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: "") }
```

Complete code in below link

https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/blob/master/app/src/main/java/com/example/newsheadlines/LoginActivity.kt

Step 2 : Creating RegistrationActivity.kt with database



Database connection in RegistrationActivity.kt

```
package com.example.newsheadlines

import ...

class RegistrationActivity : 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( value: "") }
    var password by remember { mutableStateOf( value: "") }
    var email by remember { mutableStateOf( value: "") }
    var error by remember { mutableStateOf( value: "") }

    Column(
```

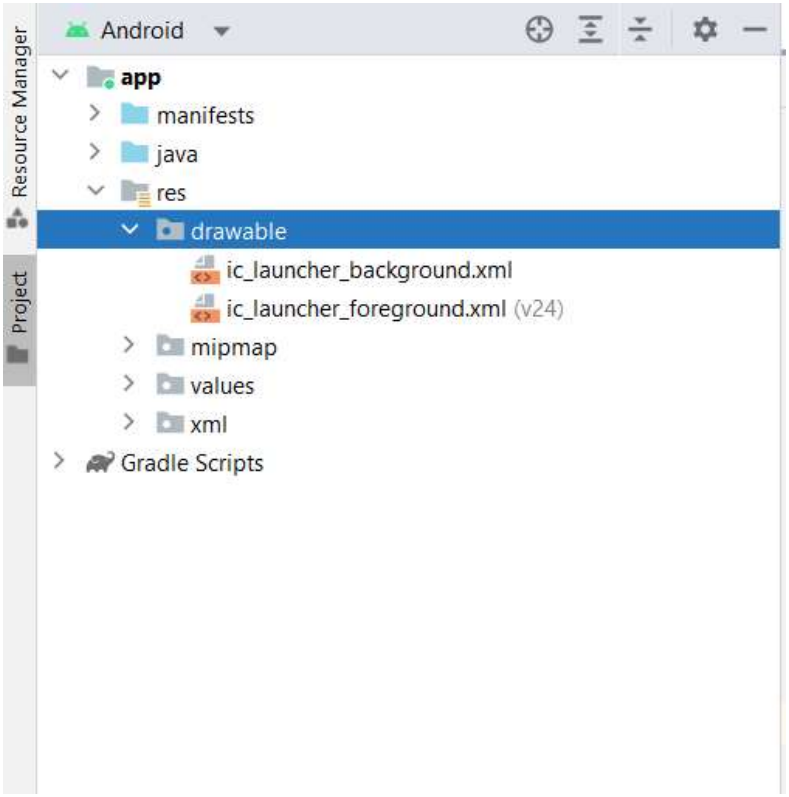
Complete code in below link:

https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/blob/master/app/src/main/java/com/example/newsheadlines/RegistrationActivity.kt

Step 3 : Creating MainActivity.kt file

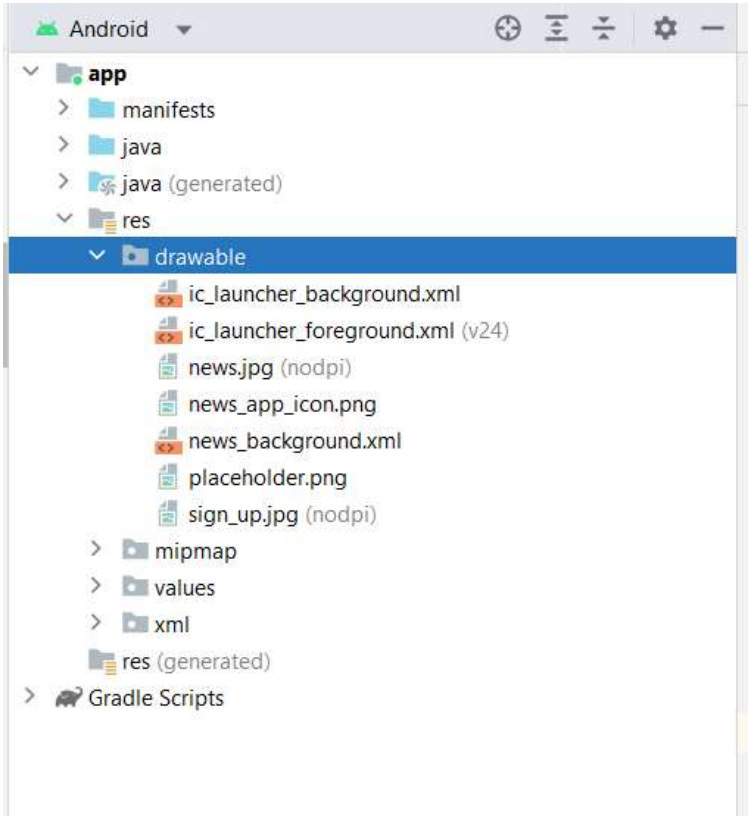
In MainActivity.kt file the main application is developed

- 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/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/tree/master/app/src/main/res/drawable

Required drawables



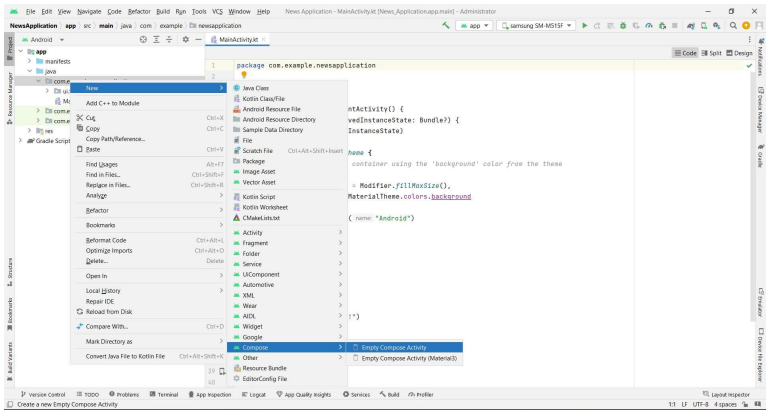
- After add all this we need to create the UI in MainActivity.kt file

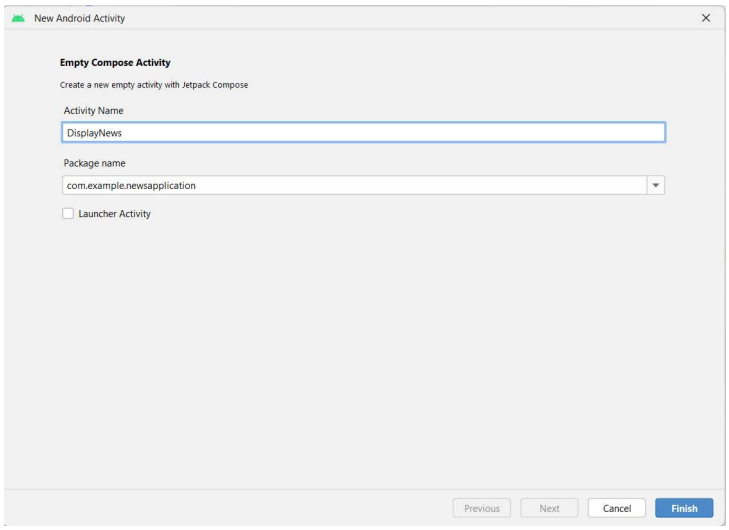
Mainpage.kt file
Complete MainActivity.kt code:
https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/blob/master/app/src/main/java/com/example/newsheadlines/MainPage.kt

Linking MainActivity with DisplayNews.kt

```
            modifier = Modifier
                .fillMaxHeight()
                .weight(0.3f)
        )


        Column(
            verticalArrangement = Arrangement.Center,
            modifier = Modifier
                .padding(4.dp)
                .fillMaxHeight()
                .weight(0.8f)
                .background(Color.Gray)
                .padding(20.dp)
                .selectable( selected: true,  enabled: true,  role: null,
                    onClick = {
                        Log.i( tag: "test123abc",  msg: "MovieItem: $index/n${movie.description}")
                        context.startActivity(
                            Intent(context, DisplayNews::class.java)
                                .setFlags(Intent.FLAG_ACTIVITY_NEW_TASK)
                                .putExtra( name: "desk", movie.description.toString())
                                .putExtra( name: "urlToImage", movie.urlToImage)
                                .putExtra( name: "title", movie.title)
                        )
                    })
        ) { this: ColumnScope
```

Step 4 : Creating DisplayNews.kt file





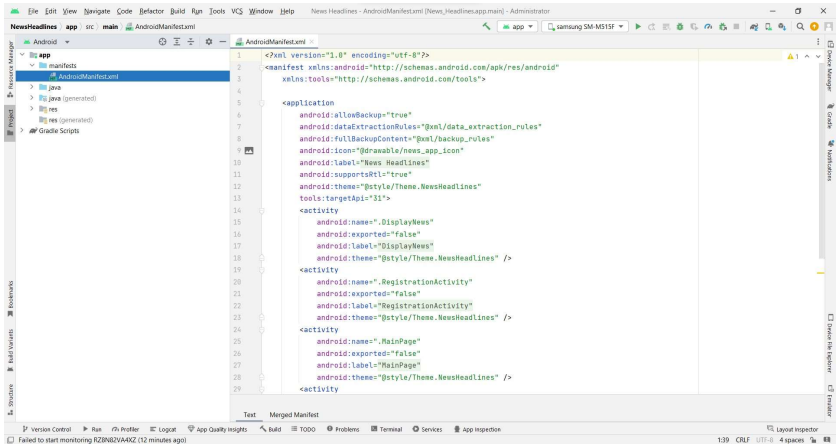

```
class DisplayNews : ComponentActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContent {
            NewsHeadlinesTheme {
                // A surface container using the 'background' color from the theme
                Surface(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

                    var desk = getIntent().getStringExtra( name: "desk")
                    var title = getIntent().getStringExtra( name: "title")
                    var uriImage = getIntent().getStringExtra( name: "urlToImage")
                    Log.i( tag: "test123abc",  msg: "MovieItem: $desk")

                    Column(Modifier.background(Color.Gray).padding(20.dp), horizontalAlignment = Alignment.CenterHorizontally
                        Text(text = ""+title, fontSize = 32.sp)
                        HtmlText(html = desk.toString())
                        /*  AsyncImage(
                              model = "https://example.com/image.jpg",
                              contentDescription = "Translated description of what the image contains"
                          )*/
                        Image(
                            painter = rememberImagePainter(uriImage),
                            contentDescription = "My content description",
                        )
                    }
                    //  Greeting(desk.toString())
```

Complete DisplayNews.kt code:

https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/blob/master/app/src/main/java/com/example/newsheadlines/DisplayNews.kt

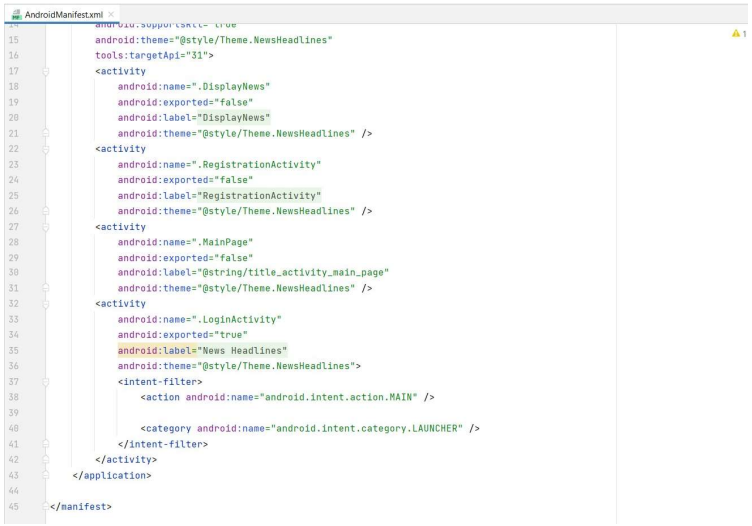
Task 8:

Modifying AndroidManifest.xml



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.



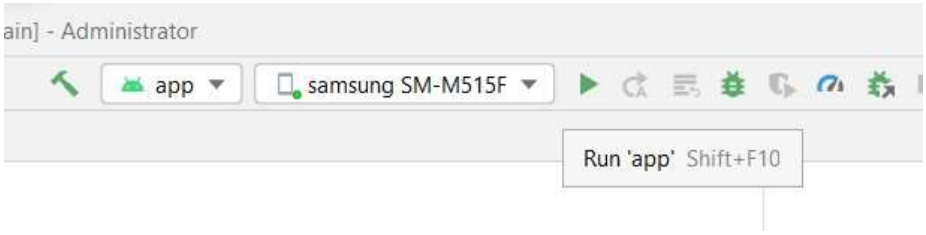
Complete code is given below:
https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines/blob/master/app/src/main/AndroidManifest.xml

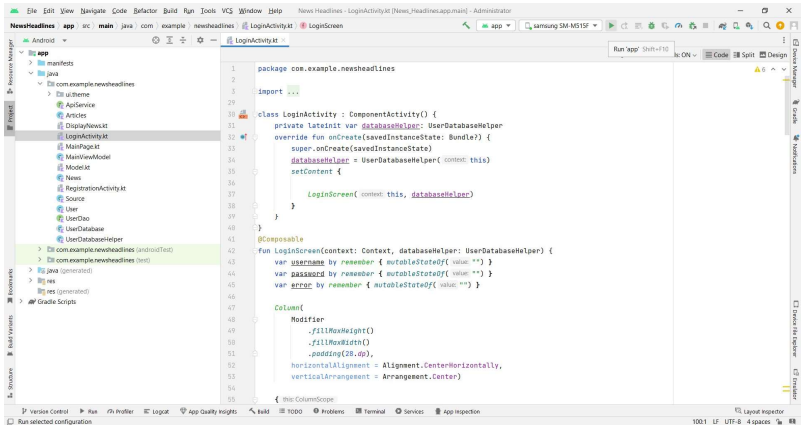
Task 9:

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



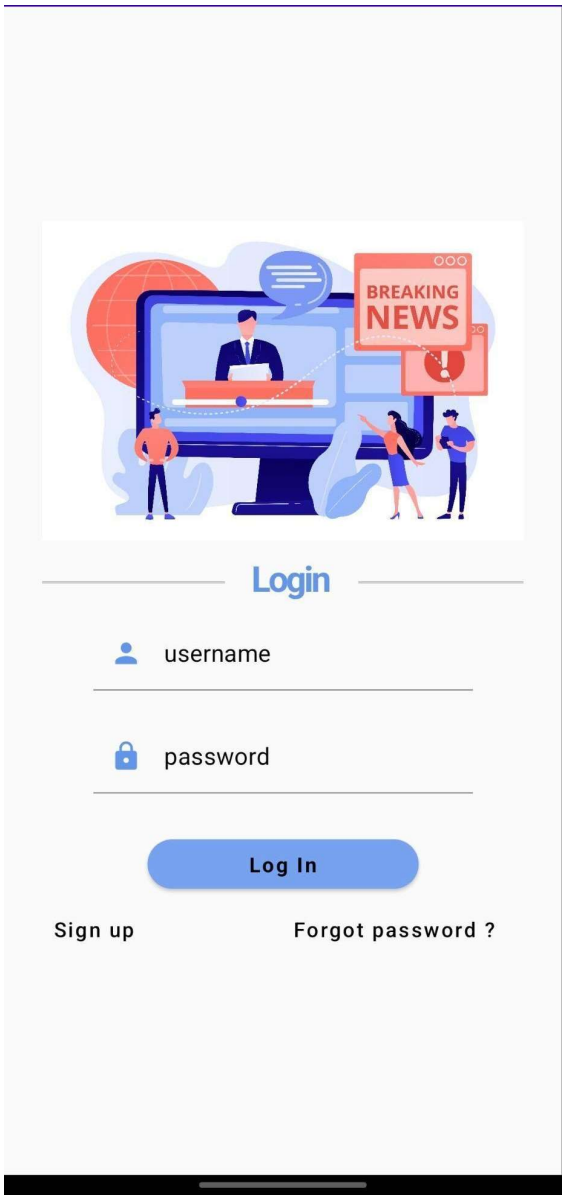


Complete Project Link:

https://github.com/pravalikalagadapati/an-android-application-for-keeping-up-with-the-latest-headlines

Final Output of the Application :

Login Page:



Register Page :

Sign Up

username

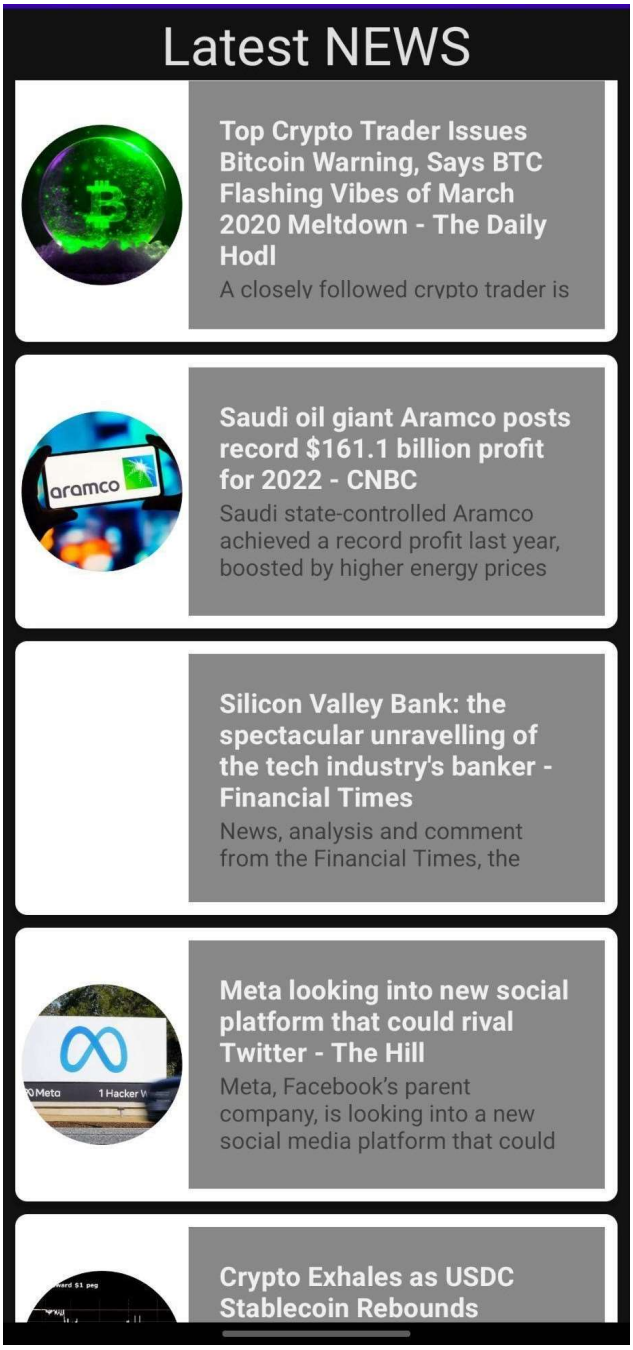
password

email

Register

Have an account? Log in

Main News Headlines Page :



Display News Page :

Team ID : LTVIP2023TMID04305

Team Size : 5

Team Leader : Lagadapati Pravallika

Team member : Kuruva Sravani

Team member : Konga Swathi

Team member : Lekkala Maniteja

Team member : Adaveni Nirmala