



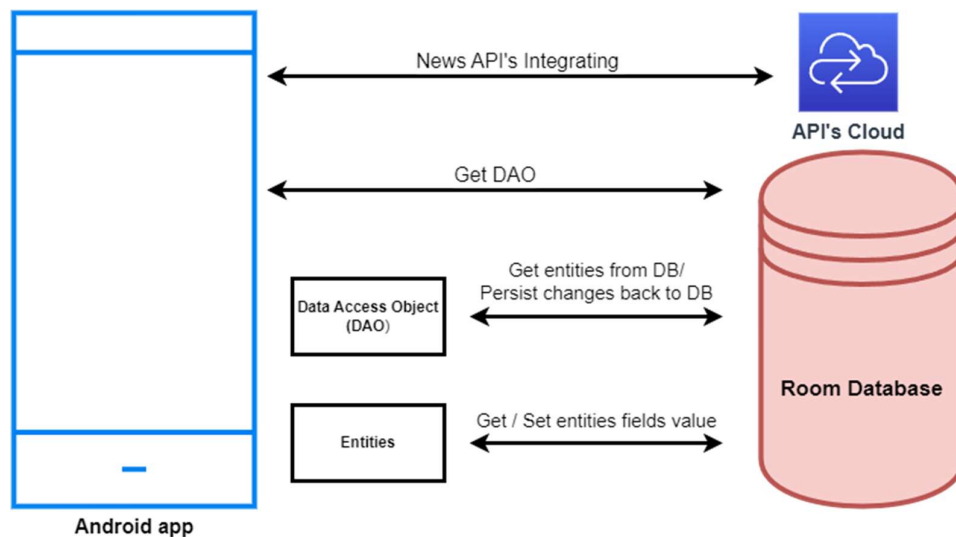
## **An Android Application for Keeping Up with the Latest Headlines**

**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 logs 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 :

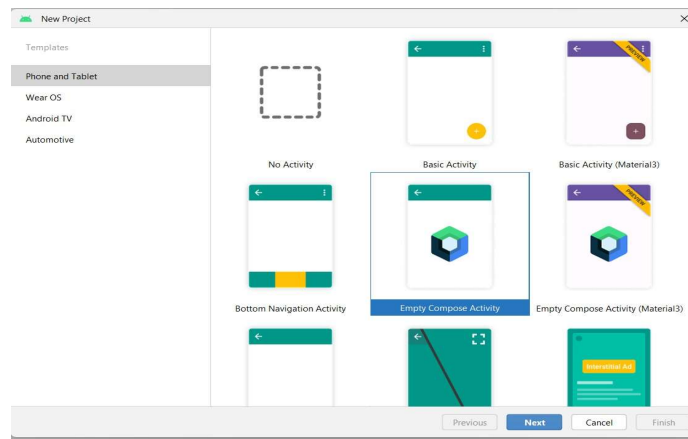
<https://developer.android.com/studio/install>

### Task 2 :

Creating a new project.

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

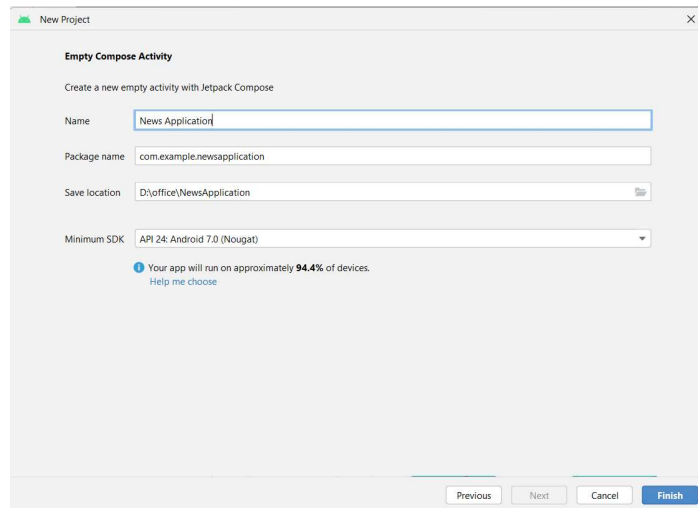
Step 2 : Click on **Next** button.



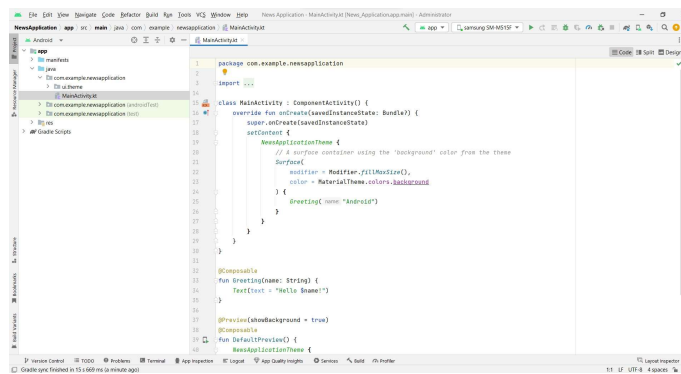
Step 3 : Give name to the new project.

Step 4 : Give the Minimum SDK value

Step 5 : Click Finish



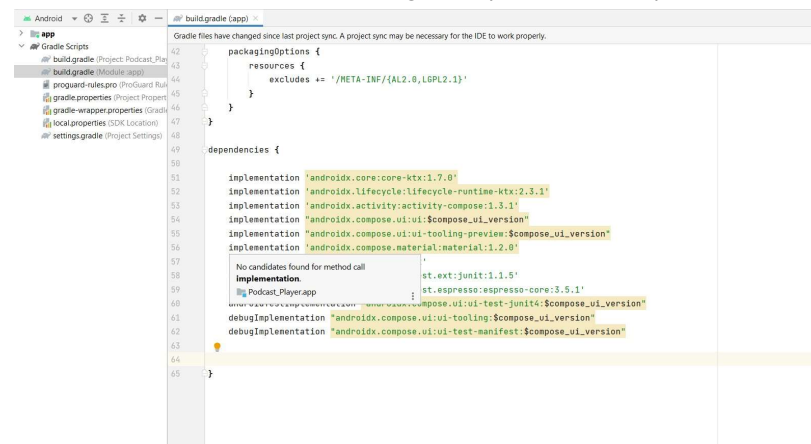
Main activity file



## 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'

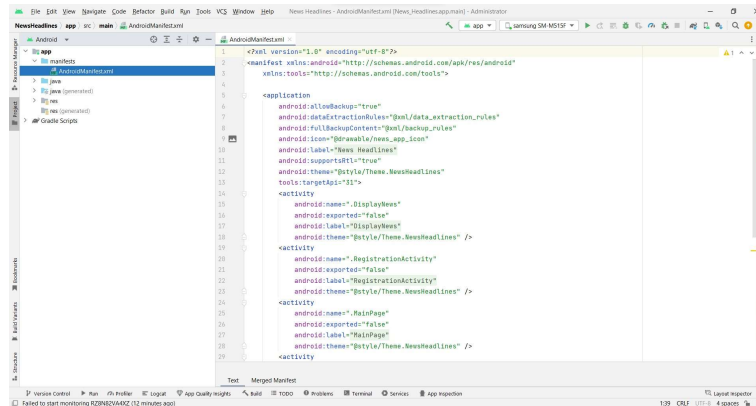
    implementation("io.coil-kt:coil-compose:1.4.0")
}
```

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">

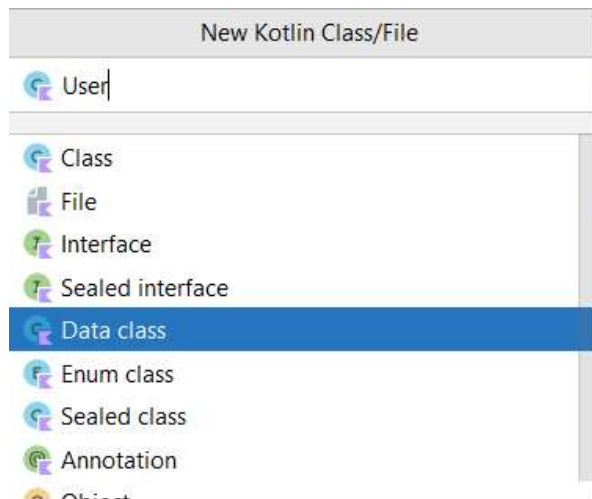
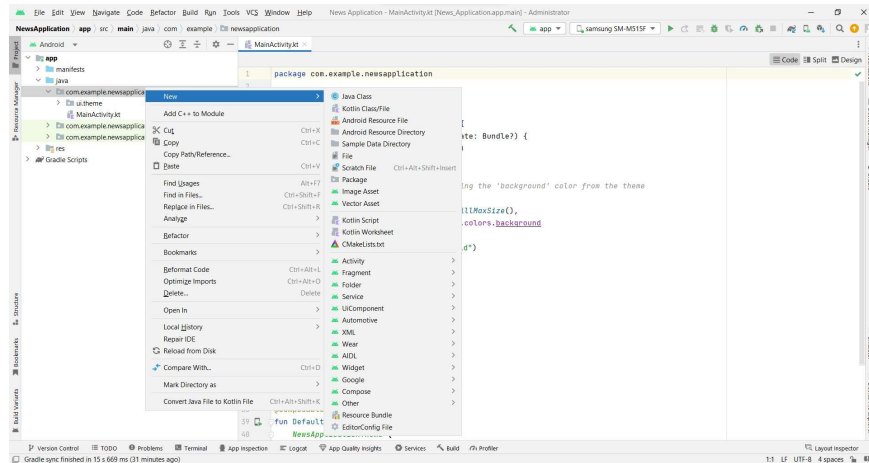
    <!--      permissions-->
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>

    <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:supportRtl="true"
        android:theme="@style/Theme.NewsHeadlines"
```

## Task 5:

Creating the database classes.

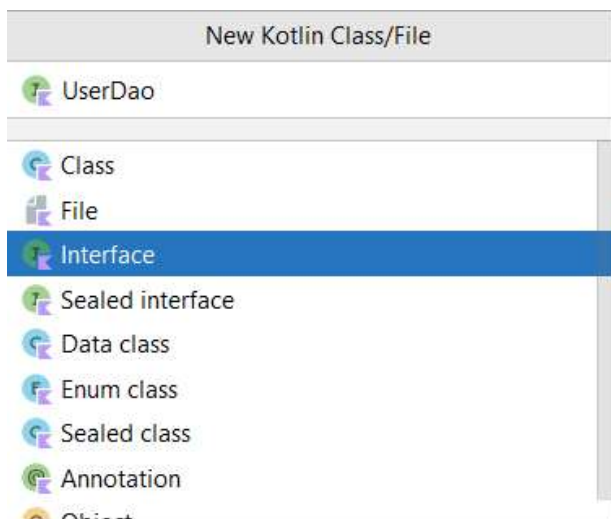
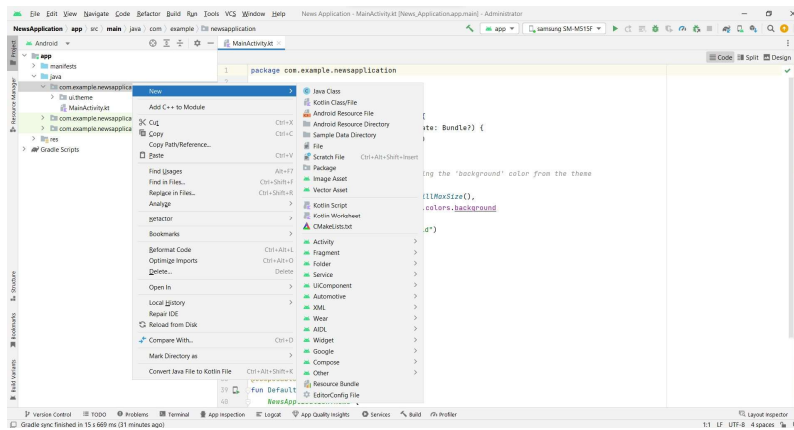
### Step 1 : Create User data class



User class code:

<https://github.com/RohithDurgala/Latest-Headlines-App/blob/master/app/src/main/java/com/example/newsheadlines/User.kt>

### Step 2 : Create an UserDao interface

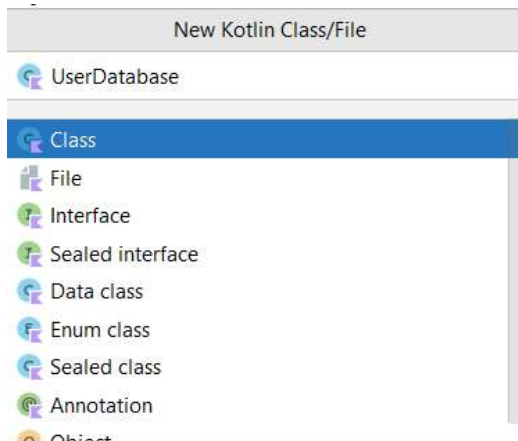
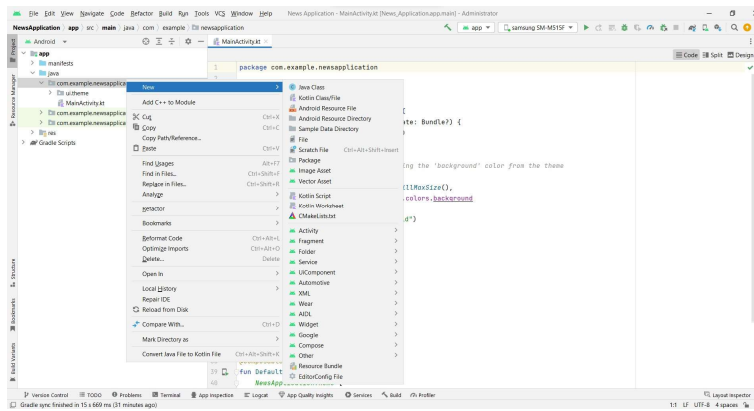


UserDao interface code:

<https://github.com/RohithDurgala/Latest-Headlines-App/blob/master/app/src/main/java/com/example/newsheadlines/UserDao.kt>

Step 3 : Create an UserDatabase class

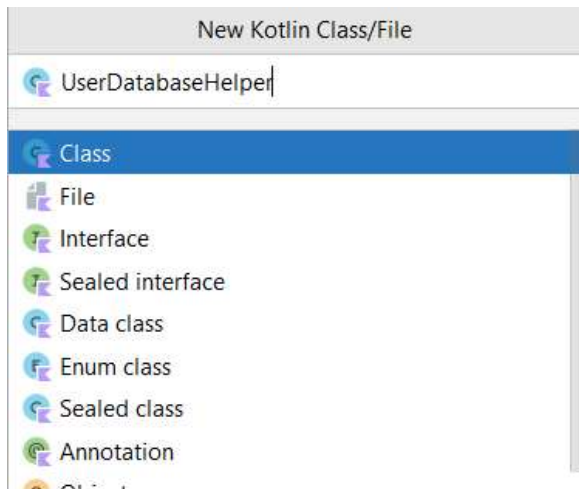
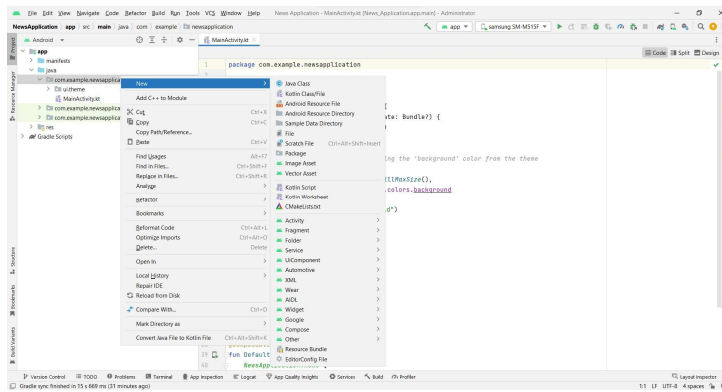




UserDatabase class code

: <https://github.com/RohithDurgala/Latest-Headlines-App/blob/master/app/src/main/java/com/example/newsheadlines/UserDatabase.kt>

Step 4 : Create an UserDatabaseHelper class



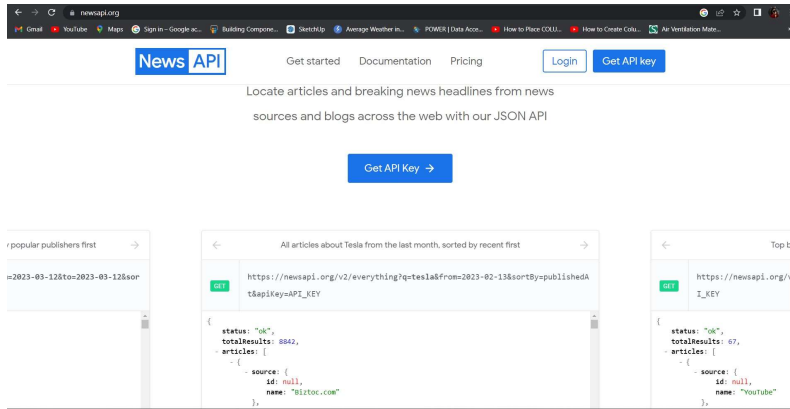
UserDatabaseHelper class code :

<https://github.com/RohithDurgala/Latest-Headlines-App/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

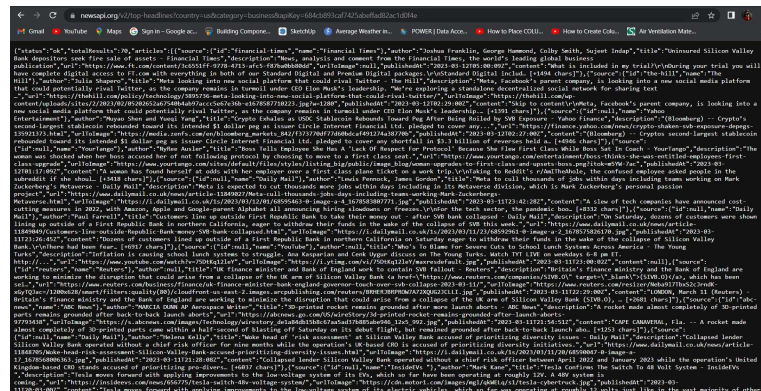


- Code which is needed to copy:

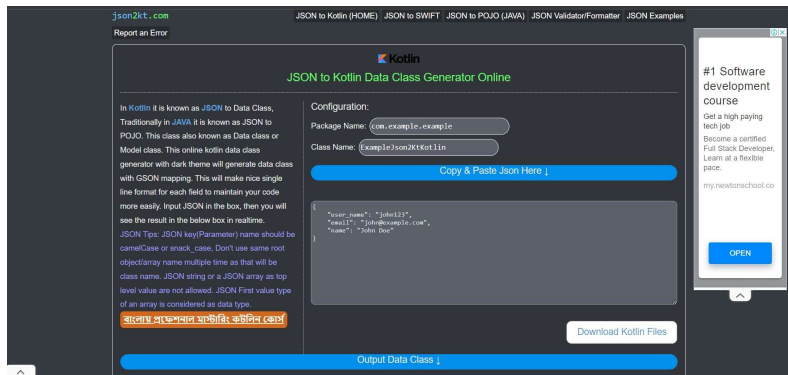


- Enter the API key at API\_KEY to get the complete json file.

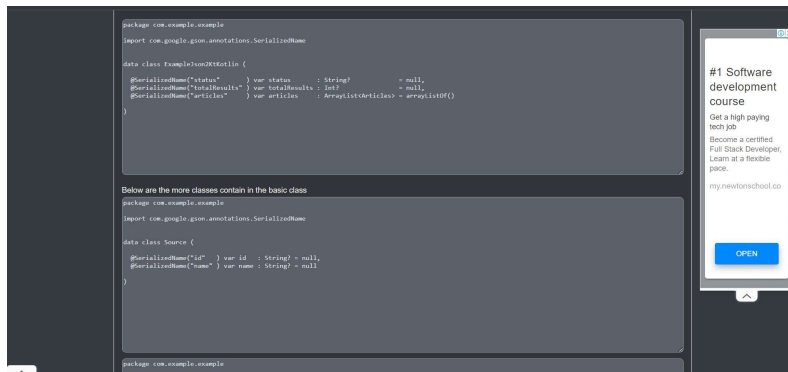
## Json File



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



Click on Output Data class

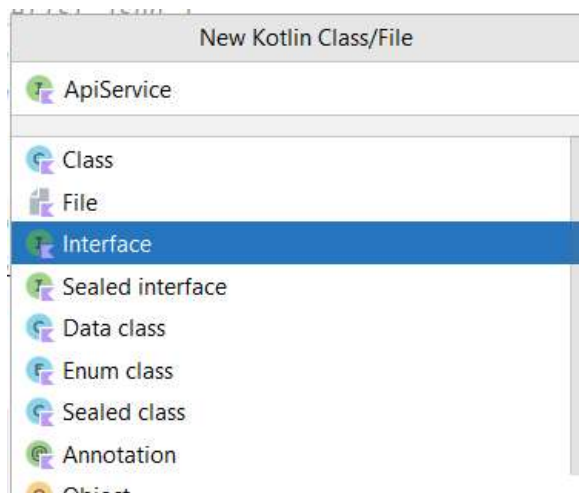
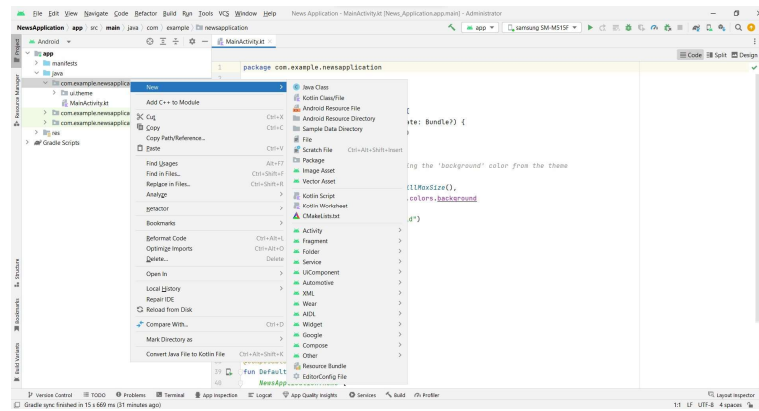


Now use this classes in android studio

## Database for news integration into project

Step 3:

Create ApiService interface



```
package com.example.newsheadlines

import ...

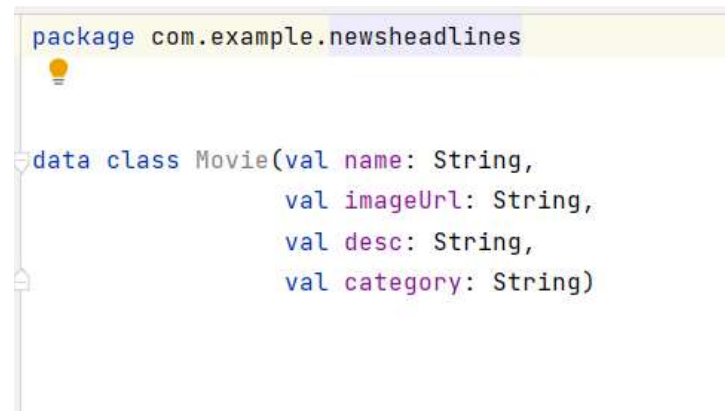
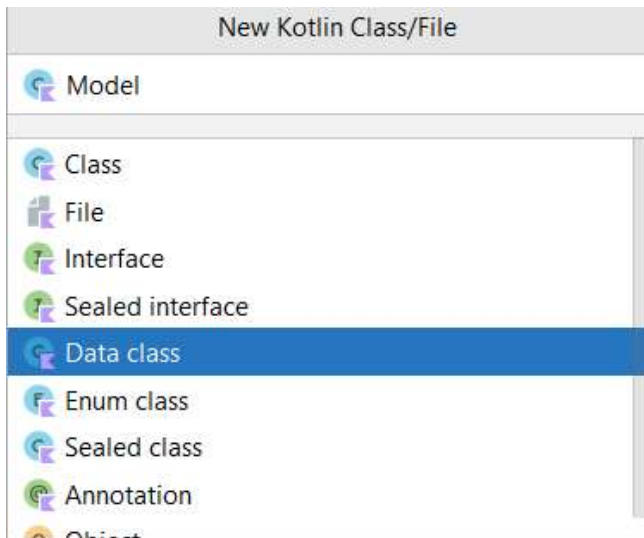
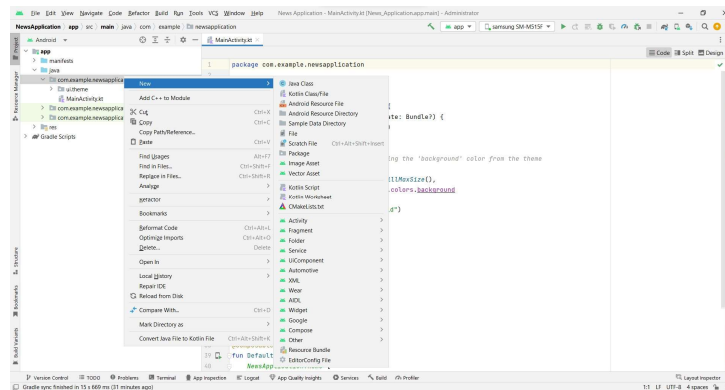
interface ApiService {
    @GET("top-headlines?country=us&category=business&apiKey=684cb893ca77425abeffad82ac1d0f4e")
    suspend fun getMovies() : News
}

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/RohithDurgala/Latest-Headlines-App/blob/master/app/src/main/java/com/example/newsheadlines/ApiService.kt>

## Step 4: Create Model data class

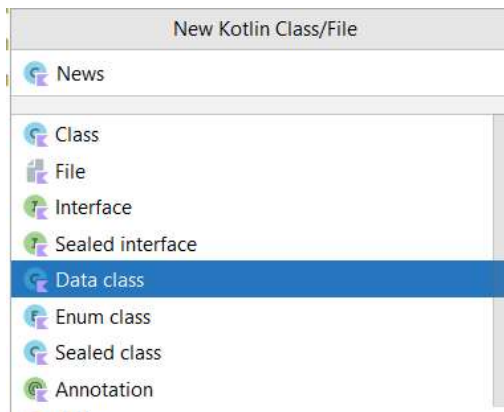
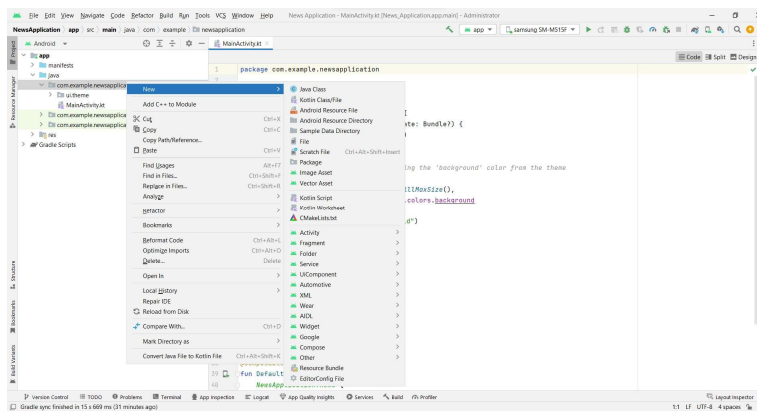


Model class code:

<https://github.com/RohithDurgala/Latest-Headlines-App/blob/master/app/src/main/java/com/example/newsheadlines/Model.kt>

Step 5:

Create News data class





```

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()
)

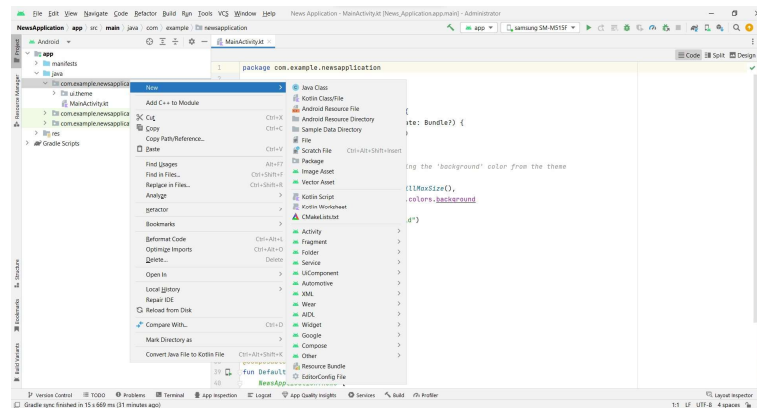
```

News class code:

<https://github.com/RohithDurgala/Latest-Headlines-App/blob/master/app/src/main/java/com/example/newsheadlines/News.kt>

Step 6:

Create Source data class



### New Kotlin Class/File

Source

Class

File

Interface

Sealed interface

Data class

Enum class

Sealed class

Annotation

```

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
)

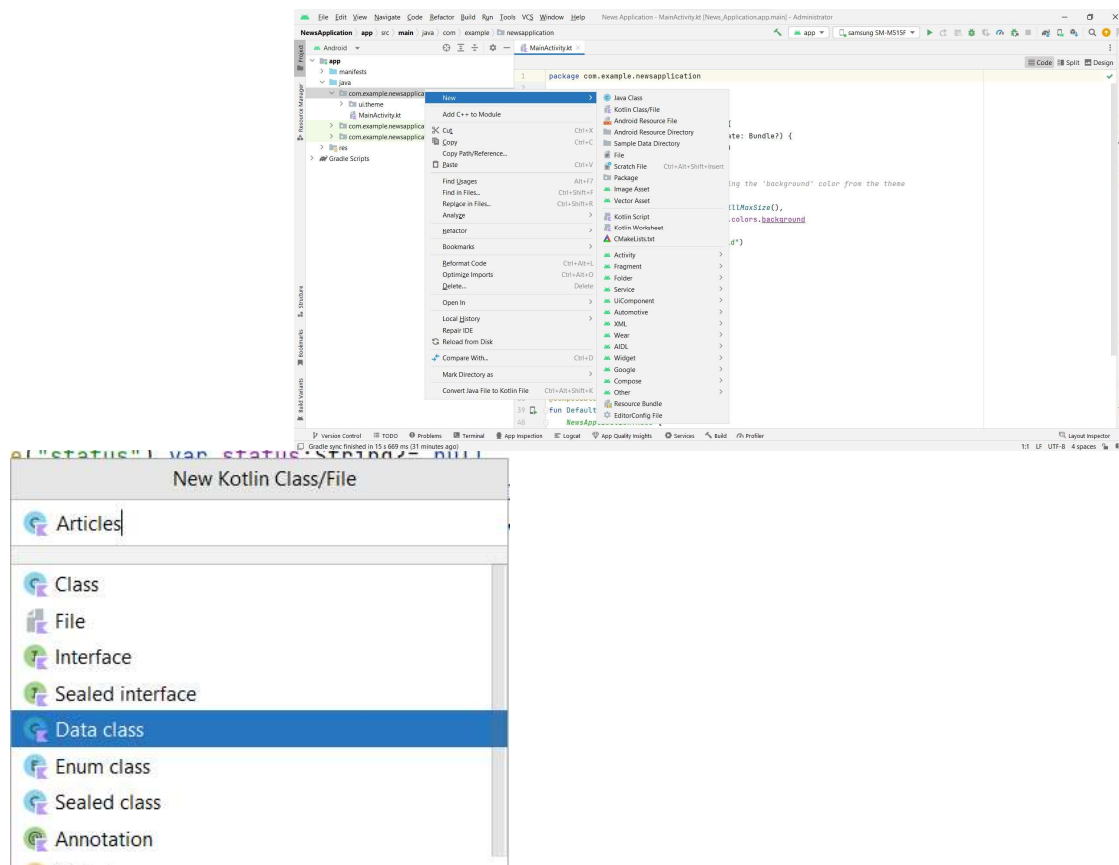
```

Model class code:

<https://github.com/RohithDurgala/Latest-Headlines-App/blob/master/app/src/main/java/com/example/newsheadlines/Source.kt>

Step 7:

Create Articles data class



```

package com.example.example

import com.google.gson.annotations.SerializedName

data class Articles {

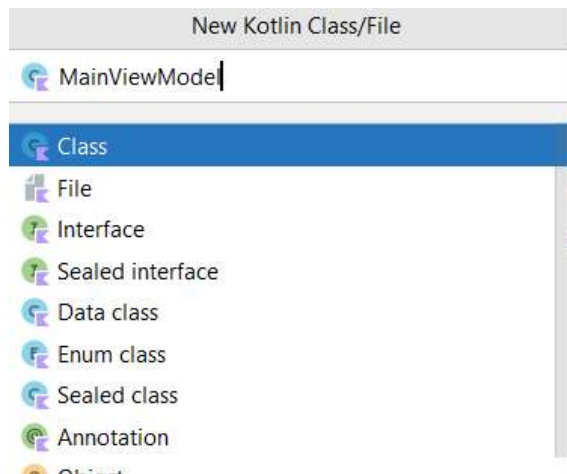
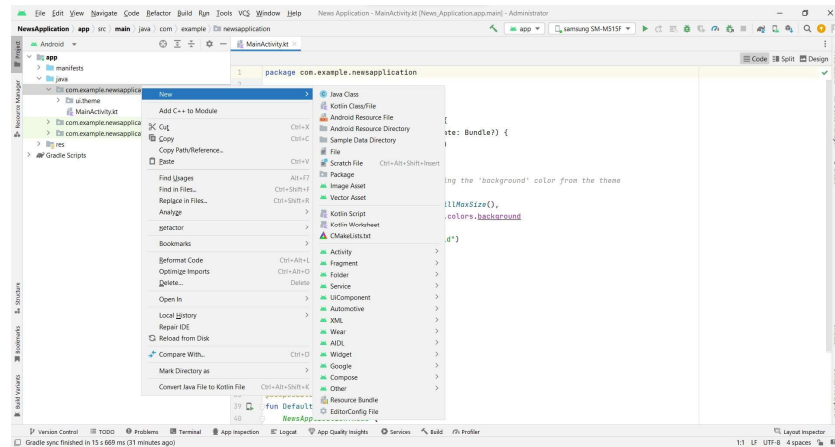
    @SerializedName("title" ) var title : String? = null,
    @SerializedName("description" ) var description : String? = null,
    @SerializedName("urlToImage" ) var urlToImage : String? = null,

```

Article class code:

<https://github.com/RohithDurgala/Latest-Headlines-App/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()
            }
        }
    }
}

```

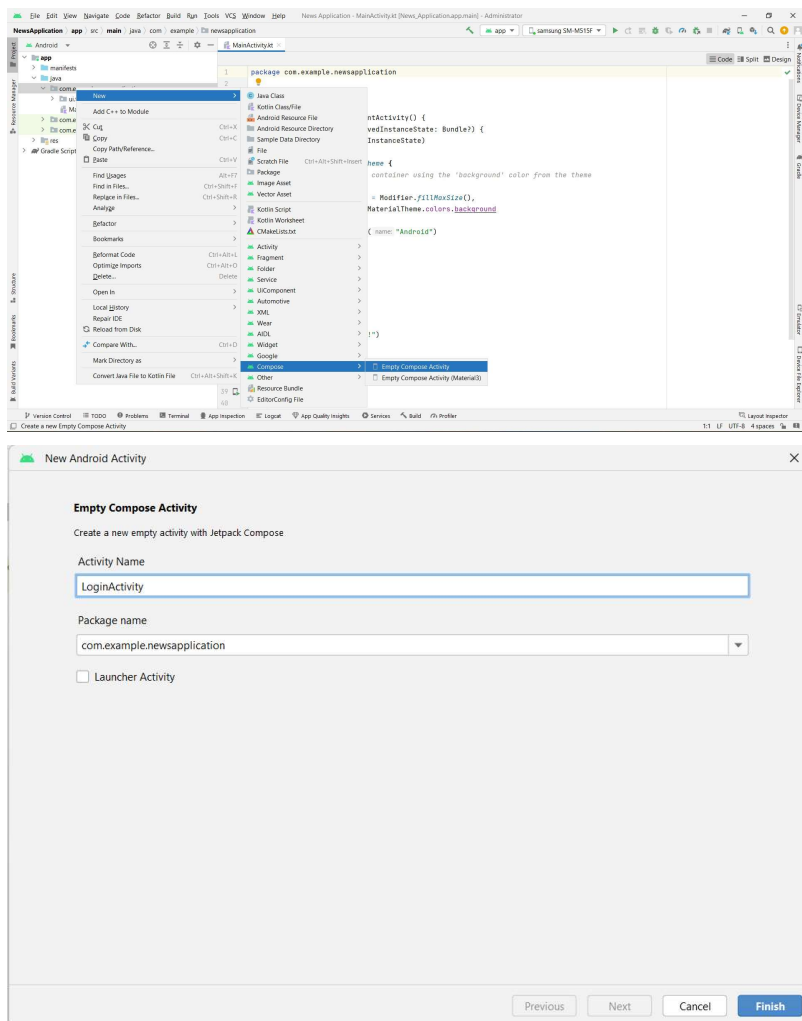
MainViewModel class code:

<https://github.com/RohithDurgala/Latest-Headlines-App/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)
        }
    }
}

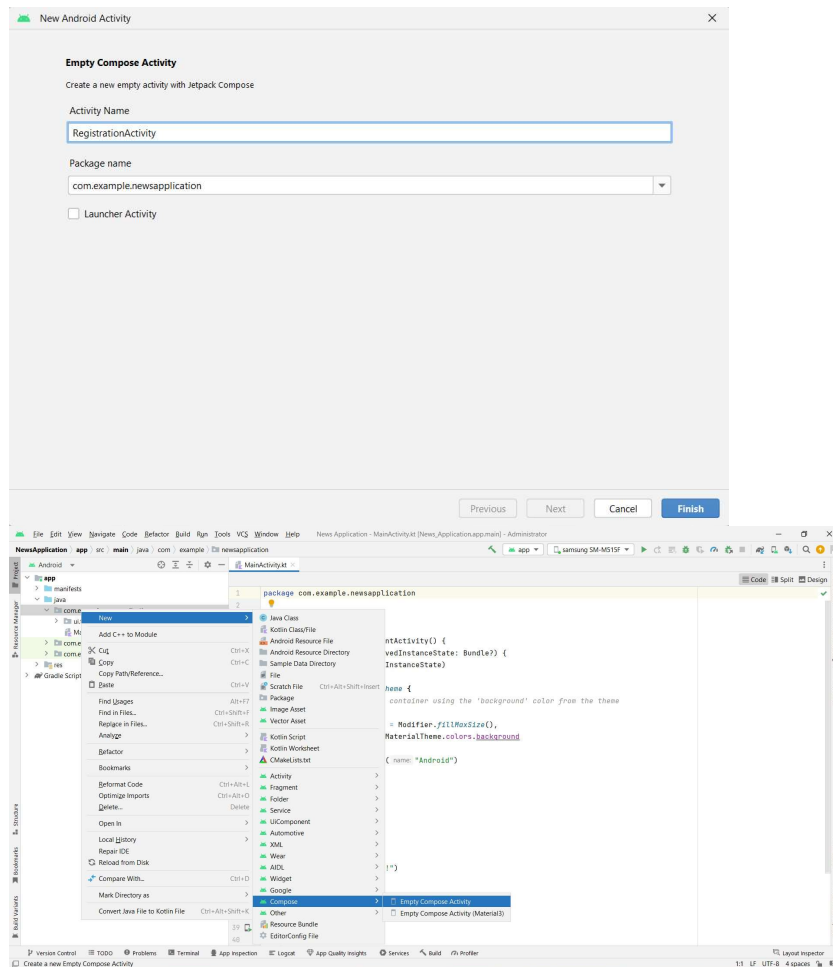
@Composable
fun LoginScreen(context: Context, databaseHelper: UserDatabaseHelper) {
    var username by remember { mutableStateOf(value: "") }
    var password by remember { mutableStateOf(value: "") }
    var error by remember { mutableStateOf(value: "") }
}

```

Complete code in below link

<https://github.com/RohithDurgala/Latest-Headlines-App/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<String>("") }
    var password by remember { mutableStateOf<String>("") }
    var email by remember { mutableStateOf<String>("") }
    var error by remember { mutableStateOf<String>("") }
}

```

Complete code in below link:

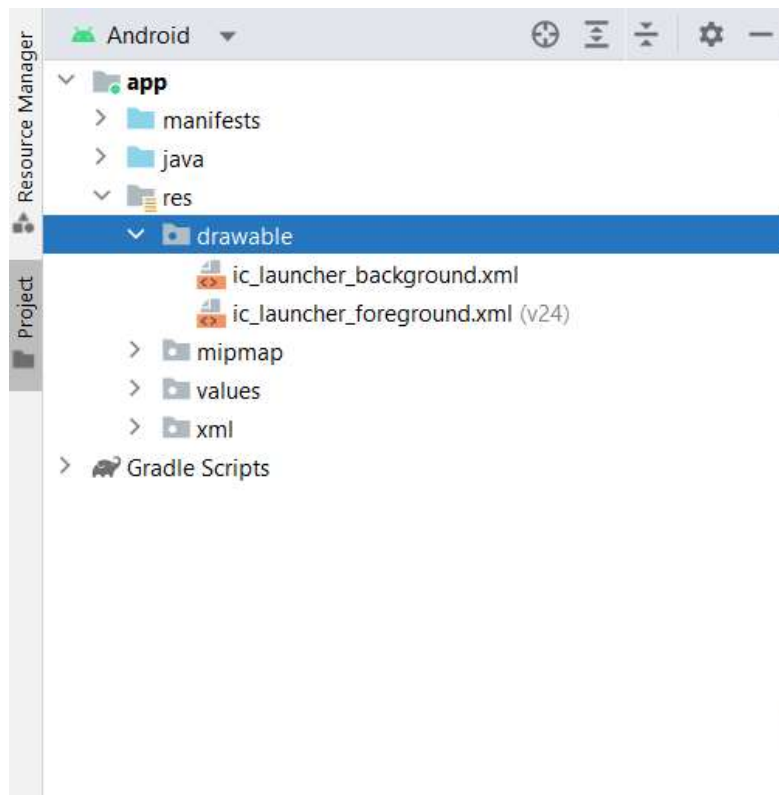
<https://github.com/RohithDurgala/Latest-Headlines-App/blob/master/app/src/main/java/com/example/newsheadlines/RegistrationActivity.kt>

### Step 3 : Creating MainActivity.kt file

In MainActivity.kt file the main application is developed

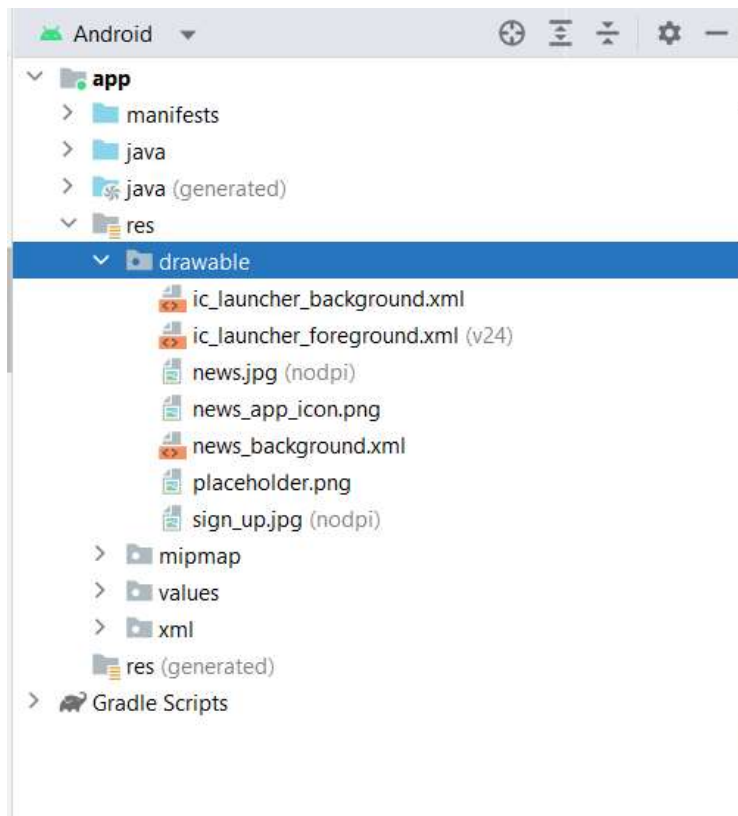
- Before creating UI we need to add some images in drawables which are in res





Download the required drawable from the code:  
<https://github.com/RohithDurgala/Latest-Headlines-App/tree/master/app/src/main/res/drawable>

Required drawables



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

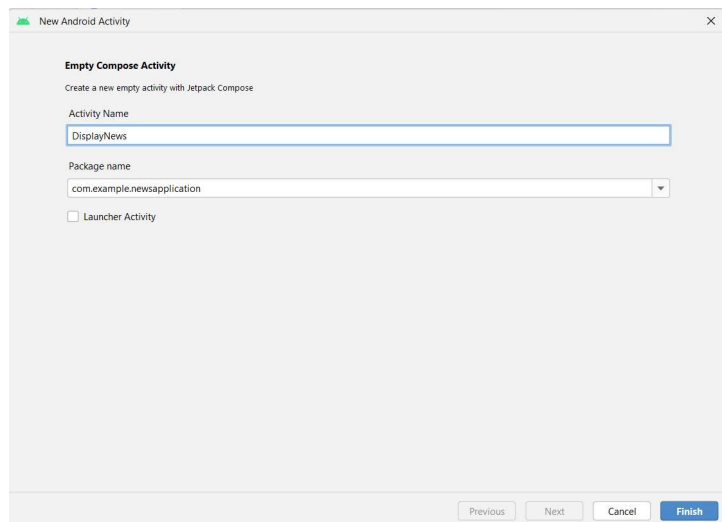
Mainpage.kt file

Complete MainActivity.kt code:

<https://github.com/RohithDurgala/Latest-Headlines-App/blob/master/app/src/main/java/com/example/newsheadlines/MainPage.kt>

Linking MainActivity with DisplayNews.kt





```
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 urlImage = getIntent().getStringExtra( name: "urlToImage")
                    Log.i( tag: "test123abc", msg: "MovieItem: $desk")

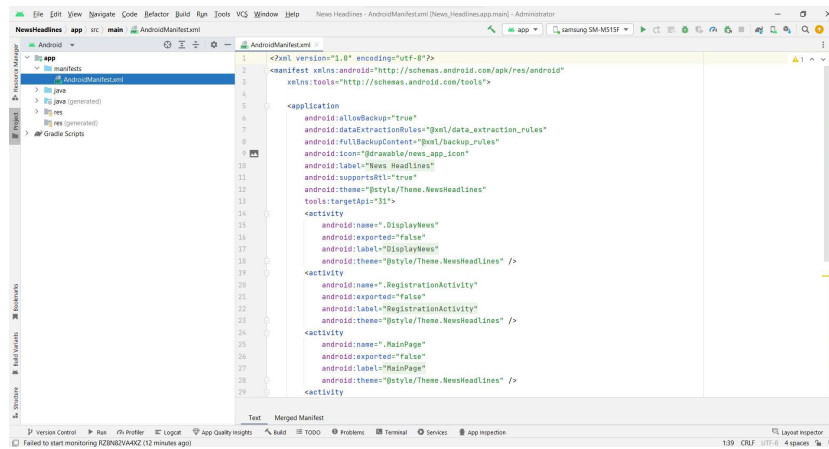
                    Column(modifier.background(Color.Gray).padding(28.dp), horizontalAlignment = Alignment.CenterHorizontally) {
                        Text(text = "$title", fontSize = 32.sp)
                        HtmlText(html = $desk.toString())
                        /* AsyncImage(
                            model = "https://example.com/image.jpg",
                            contentDescription = "Translated description of what the image contains"
                        )*/
                        Image(
                            painter = rememberImagePainter(urlImage),
                            contentDescription = "My content description",
                        )
                    }
                }
            }
        }
    }
}
```

Complete DisplayNews.kt code:

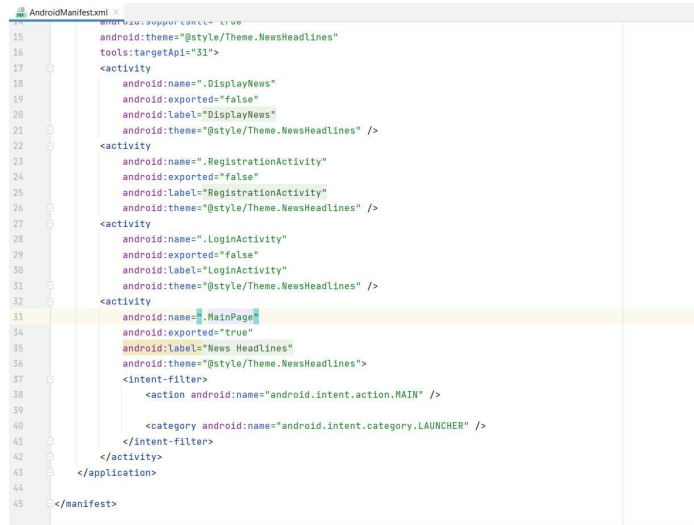
<https://github.com/RohithDurgala/Latest-Headlines-App/blob/master/app/src/main/java/com/example/newsheadlines/DisplayNews.kt>

## Task 8:

Modifying AndroidManifest.xml

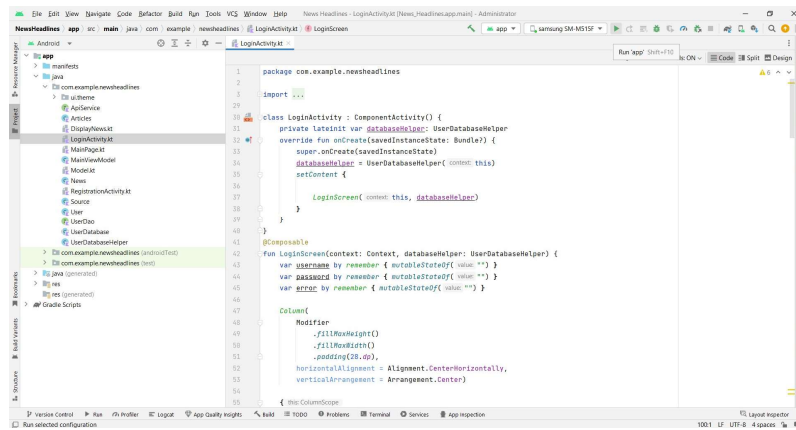


When we run the app we will get the MainActivity.kt file as our first screen , but we want LoginActivity.kt , So we need to change in AndroidManifest.xml.



Changed AndroidManifest.xml.






**Complete Project Link:**


<https://github.com/RohithDurgala/Latest-Headlines-App>


**Final Output of the Application :**

Login Page:



## Login

 username

 password

**Log In**

[Sign up](#) [Forgot password ?](#)


Register Page :




## Sign Up



 username

 password


 email

**Register**

Have an account? [Log in](#)


Main News Headlines Page :

## Latest NEWS




### Top Crypto Trader Issues Bitcoin Warning, Says BTC Flashing Vibes of March 2020 Meltdown - The Daily Hodl

A closely followed crypto trader is




### Saudi oil giant Aramco posts record \$161.1 billion profit for 2022 - CNBC

Saudi state-controlled Aramco achieved a record profit last year, boosted by higher energy prices




### Silicon Valley Bank: the spectacular unravelling of the tech industry's banker - Financial Times

News, analysis and comment from the Financial Times, the



### Meta looking into new social platform that could rival Twitter - The Hill

Meta, Facebook's parent company, is looking into a new social media platform that could



### Crypto Exhales as USDC Stablecoin Rebounds

Display News Page :

# Top Crypto Trader Issues Bitcoin Warning, Says BTC Flashing Vibes of March 2020 Meltdown - The Daily Hodl

A closely followed crypto trader is issuing an alert to Bitcoin (BTC) holders, saying that the king crypto's current market structure looks similar to its price action prior to a deep plunge about three years ago.

