**WEATHER APP with Retrofit and Navigation**

Here's a real-time case study for implementing **Retrofit** and **Navigation** in an Android weather app using **Kotlin**.

**1. Project Setup**

* **Retrofit**: Used to handle network requests and fetch weather data from an API.
* **Navigation**: Used to handle transitions between different screens.

**Dependencies:**

In your build.gradle (Module: app), add the following dependencies:

gradle

CopyEdit

implementation "com.squareup.retrofit2:retrofit:2.9.0"

implementation "com.squareup.retrofit2:converter-gson:2.9.0"

implementation "androidx.navigation:navigation-fragment-ktx:2.5.1"

implementation "androidx.navigation:navigation-ui-ktx:2.5.1"

implementation "androidx.recyclerview:recyclerview:1.2.1"

**2. API Setup (Weather API)**

Create a simple API interface using Retrofit to get weather data. For this example, we'll use a sample API like https://api.openweathermap.org/data/2.5/weather.

kotlin

CopyEdit

interface WeatherApi {

@GET("weather")

suspend fun getWeather(@Query("q") city: String, @Query("appid") apiKey: String): Response<WeatherResponse>

}

**3. Weather Response Model**

Create a data class that models the API response.

kotlin

CopyEdit

data class WeatherResponse(

val main: Main,

val weather: List<Weather>

)

data class Main(

val temp: Double

)

data class Weather(

val description: String

)

**4. Retrofit Setup**

Initialize Retrofit and make a network call.

kotlin

CopyEdit

object RetrofitInstance {

private const val BASE\_URL = "https://api.openweathermap.org/data/2.5/"

val weatherApi: WeatherApi by lazy {

Retrofit.Builder()

.baseUrl(BASE\_URL)

.addConverterFactory(GsonConverterFactory.create())

.build()

.create(WeatherApi::class.java)

}

}

**5. Fetching Weather Data**

In your main activity, create a function to call the API and display the result.

kotlin

CopyEdit

class MainActivity : AppCompatActivity() {

private lateinit var binding: ActivityMainBinding

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

binding = ActivityMainBinding.inflate(layoutInflater)

setContentView(binding.root)

//val btnFetchWeather = findViewById<Button>(R.id.btnFetchWeather)

// Navigate to weather details

binding.btnFetchWeather.setOnClickListener {

fetchWeather("London")

}

}

private fun fetchWeather(city: String) {

val apiKey = "your\_api\_key\_here"

CoroutineScope(Dispatchers.IO).launch {

val response = RetrofitInstance.weatherApi.getWeather(city, apiKey)

withContext(Dispatchers.Main) {

if (response.isSuccessful) {

val weatherResponse = response.body()

displayWeather(weatherResponse)

} else {

showError()

}

}

}

}

private fun displayWeather(weatherResponse: WeatherResponse?) {

if (weatherResponse != null) {

binding.tvWeather.text = "Temp: ${weatherResponse.main.temp}°C, ${weatherResponse.weather[0].description}"

}

}

private fun showError() {

binding.tvWeather.text = "Error fetching weather data"

}

}

**6. Navigation Setup**

You can add a second fragment that will display detailed weather information.

1. **Create a fragment (WeatherDetailsFragment)**

kotlin

CopyEdit

class WeatherDetailsFragment : Fragment(R.layout.fragment\_weather\_details) {

private lateinit var binding: FragmentWeatherDetailsBinding

override fun onViewCreated(view: View, savedInstanceState: Bundle?) {

super.onViewCreated(view, savedInstanceState)

binding = FragmentWeatherDetailsBinding.bind(view)

val weatherData = arguments?.getString("weather\_data")

binding.tvWeatherDetails.text = weatherData

}

}

1. **Add Navigation Graph**

In res/navigation/nav\_graph.xml:

xml

CopyEdit

<navigation xmlns:android="http://schemas.android.com/apk/res/android"

android:id="@+id/nav\_graph"

app:startDestination="@id/mainFragment">

<fragment

android:id="@+id/mainFragment"

android:name="com.example.weatherapp.MainFragment"

android:label="Weather"

app:destination="@id/weatherDetailsFragment"/>

<fragment

android:id="@+id/weatherDetailsFragment"

android:name="com.example.weatherapp.WeatherDetailsFragment"

android:label="Weather Details">

</fragment>

</navigation>

1. **Navigate to the Details Fragment**

In MainActivity:

kotlin

CopyEdit

val action = MainFragmentDirections.actionMainFragmentToWeatherDetailsFragment(weatherResponse.toString())

findNavController().navigate(action)

**7. Handling Navigation in MainFragment**

Create a simple button to navigate between fragments:

kotlin

CopyEdit

class MainFragment : Fragment(R.layout.fragment\_main) {

private lateinit var binding: FragmentMainBinding

override fun onViewCreated(view: View, savedInstanceState: Bundle?) {

super.onViewCreated(view, savedInstanceState)

binding = FragmentMainBinding.bind(view)

binding.btnNavigate.setOnClickListener {

val action = MainFragmentDirections.actionMainFragmentToWeatherDetailsFragment("Weather details here")

findNavController().navigate(action)

}

}

}

**8. Conclusion**

This case study shows how to use **Retrofit** to fetch weather data and navigate between fragments in an Android app using **Navigation**