

SL.No	Folder & Description
1	<p>AndroidManifest.xml</p> <p>This is the manifest file which describes the fundamental characteristics of the app and defines each of its components. It defines each part of the app, such as activities, services, broadcast receivers, and content providers. It lists permissions required by the app to access device features or data.</p>
2	<p>Java</p> <p>This contains the .java source files for your project. By default, it includes an MainActivity.java source file having an activity class that runs when your app is launched using the app icon.</p>
3	<p>res/drawable</p> <p>This is a directory for drawable objects that are designed for high-density Screens. It is also used to import images, custom resources for the projects</p>
4	<p>res/layout</p> <p>This is a directory for files that define your app's user interface.</p>
5	<p>res/values</p> <p>This is a directory for other various XML files that contain a collection of resources, such as strings and colours definitions.</p>
6	<p>Build.gradle</p> <p>This is an auto generated file which contains compileSdkVersion, buildToolsVersion, applicationId, minSdkVersion, targetSdkVersion, versionCode and versionName</p>

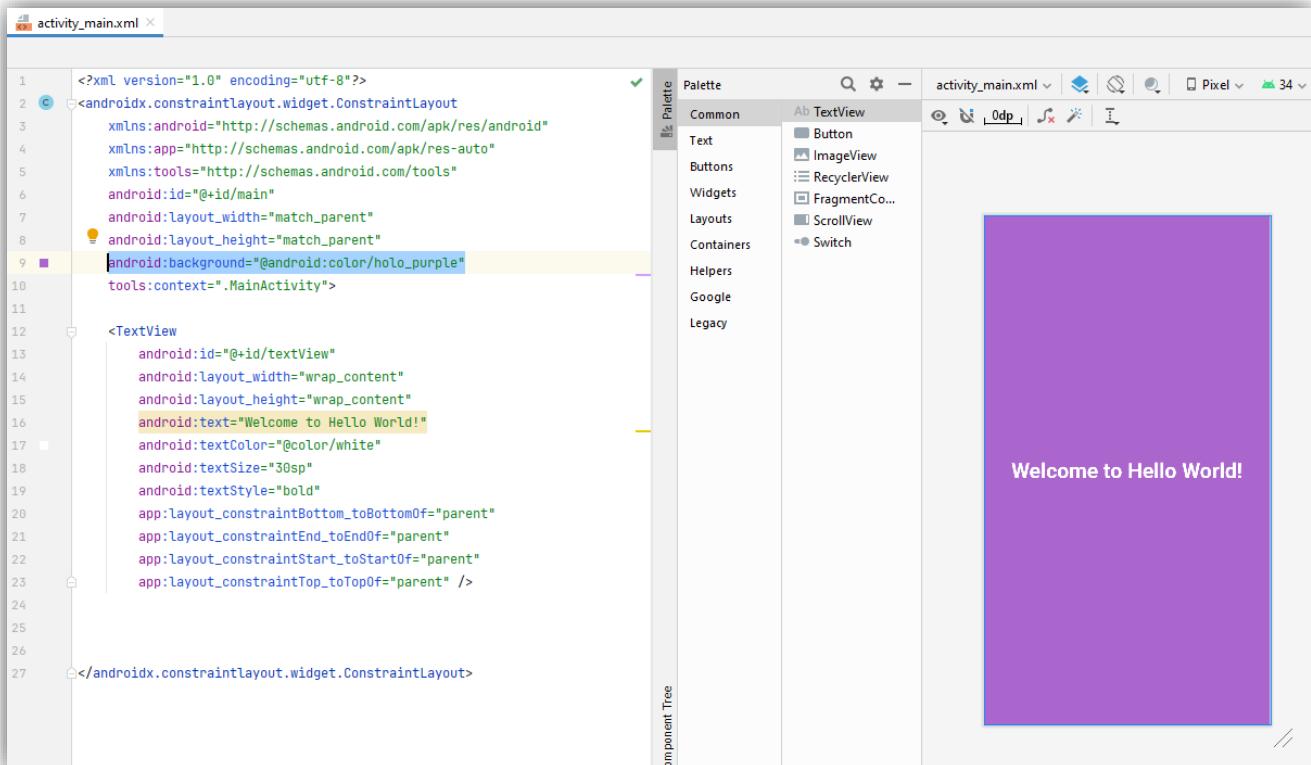
MOBILE APPLICATION DEVELOPMENT LAB PROGRAM

1. [Creating “Hello world” Application.](#)
2. [Creating an application that displays message based on the screen orientation.](#)
3. [Create an application to develop Login window using UI controls.](#)
4. [Create an application to implement new activity using explicit intent, implicit intent and content provider.](#)
5. [Create an application that displays custom designed Opening Screen.](#)
6. [Create an UI with all views.](#)
7. [Create menu in Application](#)
8. [Read/ write the Local data.](#)
9. [Create / Read / Write data with database \(SQLite\).](#)
10. [Create an application to send SMS and receive SMS](#)
11. [Create an application to send an e-mail.](#)
12. [Display Map based on the Current/given location.](#)
13. [Create a sample application with login module\(check user name and password\) On successful login change TextView “Login Successful”. On login fail alert using Toast “login fail”](#)
14. [Learn to deploy Android applications](#)

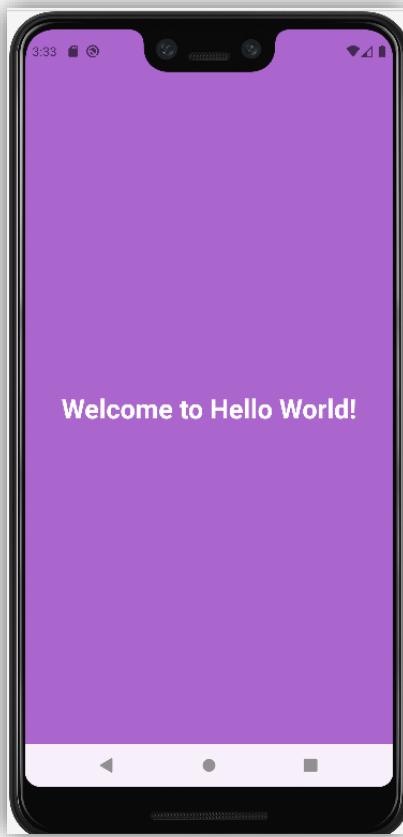
1. Creating “Hello world” Application.

1. Click Start ->**Android Studio**, “a Welcome to Android Studio dialog box will appear. Click **New Project**, the New Project Dialog box appears.
2. Choose **Empty Views Activity** then click next.
3. Specify the **Name of your project**, Select the Language as **Java**, and Select the **Minimum SDK** as API 16 (Jelly Bean, Android 4.1). Click Finish Button.
4. Click on **activity_main.xml file**, Select split window i.e. (both code and design)
5. Change the background color of layout in **ConstraintLayout**
6. **android:background="@android:color/holo_purple"**
7. Go to palette option-> Common-> **TextView**, Then Drag drop the Text View into design window
8. Edit the text View content in **activity_main.xml** code file as follows:
9. **<TextView**
android:id="@+id/textView"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Welcome to Hello world!"
android:textColor="@color/white"
android:textSize="30sp"/>

10. Align the Text View to the center of the layout



11. Press **Shift F10** to run the application Or, Press the run button on top of the interface



2. Creating an application that displays message based on the screen orientation.

1. Create a **new project** and choose **Empty Views Activity** then click **Next**.
2. Change the Text View content in the code as follows:

```
<TextView  
    android:id="@+id/textView"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="screen orientation"  
    android:textSize="30sp"/>
```

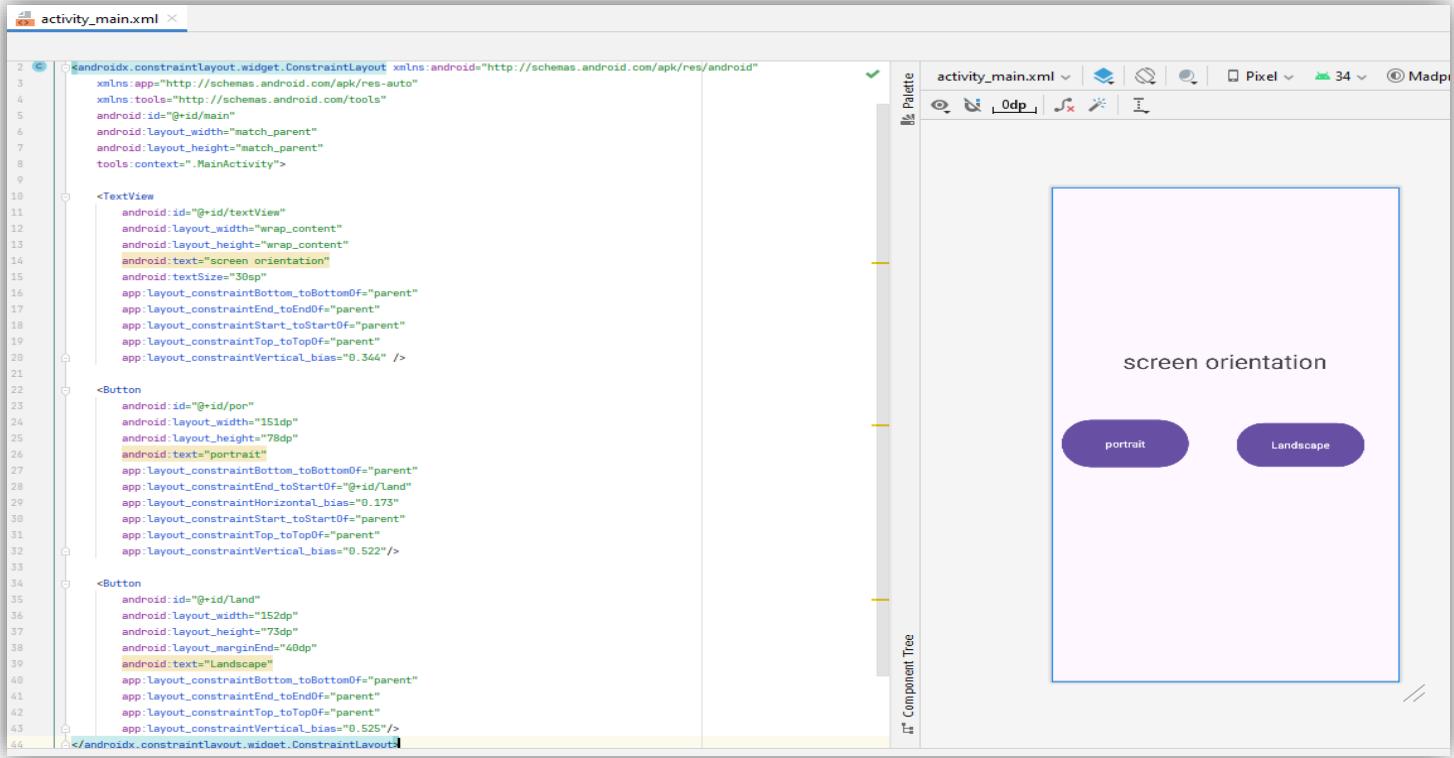
3. Go to palette option-> Common-> **ButtonView**, Then Drag drop the Button View into design window

4. Create two **Button** resources in **activity_main.xml** and update the following code.

```
<Button  
    android:id="@+id/por"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="portrait" />
```

```
<Button  
    android:id="@+id/lan"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Landscape" />
```

5. Align the Text View and Buttons to the center of the layout



6. Update the following code in **MainActivity.java** file

- Importing necessary packages:

```
import android.view.View;
import android.widget.Button;
import android.widget.Toast;
```

- Create two **Button** object, create **clickListener** event

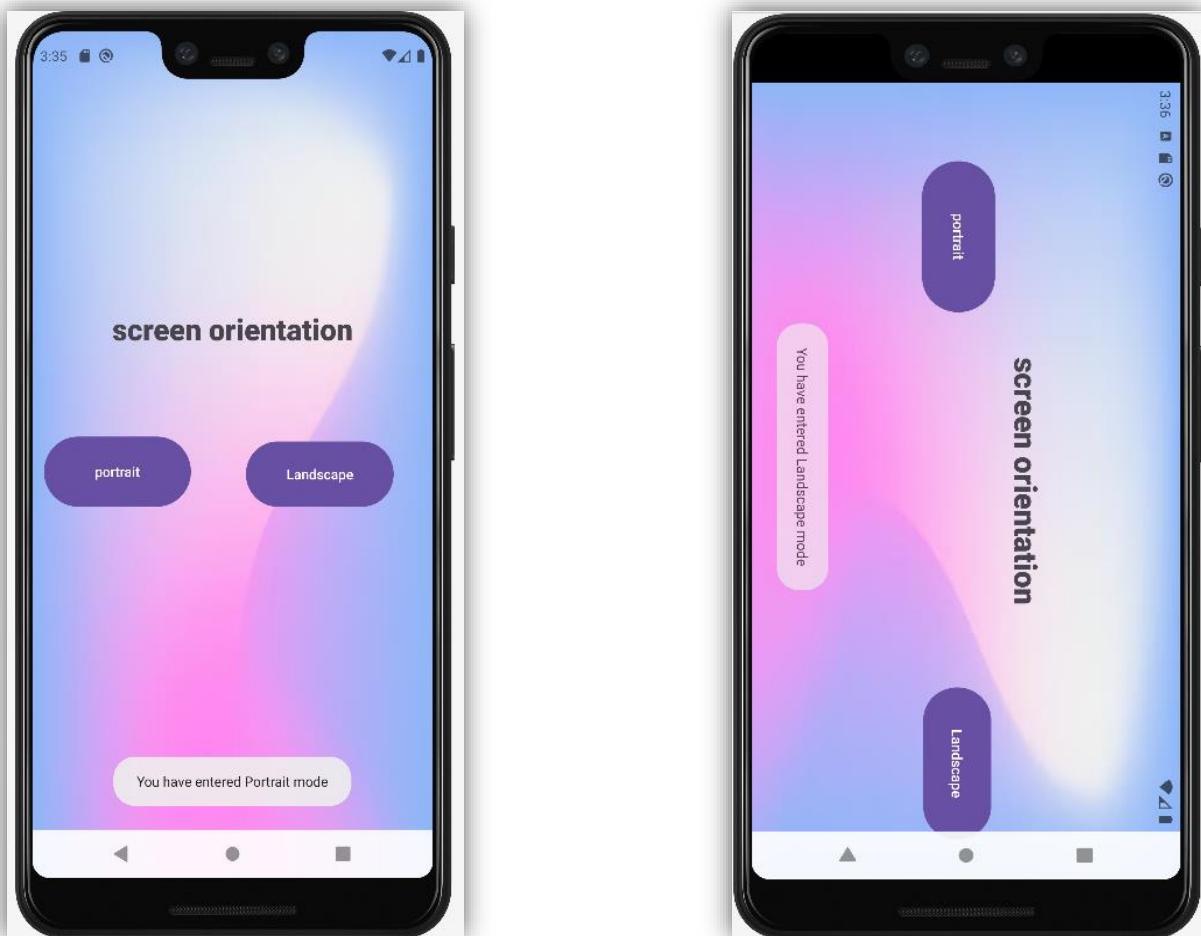
Button l,p;

```
l=findViewById(R.id.land);
p=findViewById(R.id.por);
l.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_LANDSCAPE);
        Toast.makeText(MainActivity.this,"You have entered Landscape
mode",Toast.LENGTH_SHORT).show();
    }
});
```

```
p.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT);
        Toast.makeText(MainActivity.this,"You have entered Portrait
mode",Toast.LENGTH_SHORT).show();
    }
});
```

```
 MainActivity.java
4 import android.os.Bundle;
5 import android.view.View;
6 import android.widget.Button;
7 import android.widget.Toast;
8
9 import androidx.activity.EdgeToEdge;
10 import androidx.appcompat.app.AppCompatActivity;
11
12 public class MainActivity extends AppCompatActivity {
13     @Override
14     protected void onCreate(Bundle savedInstanceState) {
15         super.onCreate(savedInstanceState);
16         EdgeToEdge.enable( thisEnableEdgeToEdge: this );
17         setContentView(R.layout.activity_main);
18
19         Button l,p;
20         l=findViewById(R.id.land);
21         p=findViewById(R.id.por);
22         l.setOnClickListener(new View.OnClickListener() {
23             @Override
24             public void onClick(View v) {
25                 setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_Landscape);
26                 Toast.makeText( context: MainActivity.this, text: "You have entered Landscape mode",Toast.LENGTH_SHORT).show();
27             }
28         });
29
30         p.setOnClickListener(new View.OnClickListener() {
31             @Override
32             public void onClick(View v) {
33                 setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_Portrait);
34                 Toast.makeText( context: MainActivity.this, text: "You have entered Portrait mode",Toast.LENGTH_SHORT).show();
35             }
36         });
37     }
38 }
```

7. Press **Shift F10** to run the application Or, Press the run button on top of the interface

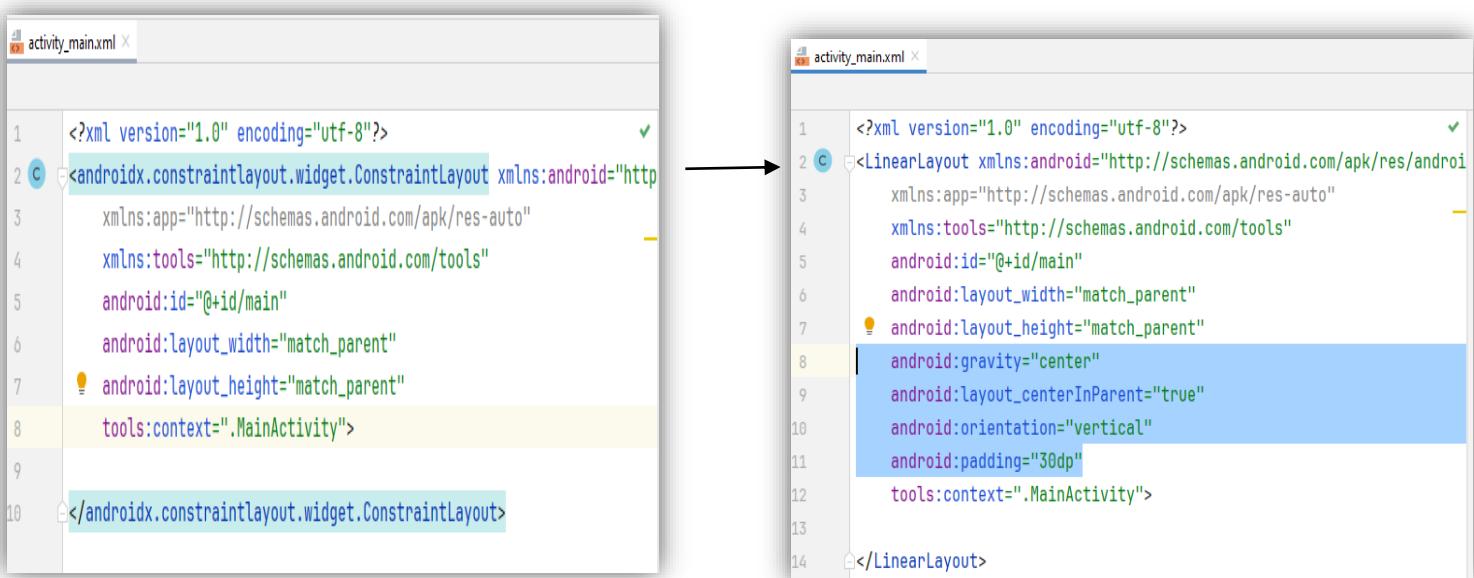


3. Create an application to develop Login window using UI controls.

1. Create a new project and choose **Empty Views Activity** then click **Next**.
2. Change the layout from constraint Layout to Relative Layout in **activity_main.xml** file.
`<androidx.constraintlayout.widget.ConstraintLayout..`
To
`<LinearLayout..`

3. update the following code in the linearlayout tag.

```
    android:gravity="center"  
    android:layout_centerInParent="true"  
    android:orientation="vertical"  
    android:padding="30dp"
```



4. Go to palette option-> Common-> **TextView**, Then Drag drop the Button View into design window

5. Edit the text View content in **activity_main.xml** code file as follows:

```
<TextView  
    android:id="@+id/textView3"  
    android:layout_width="match_parent"  
    android:layout_height="48sp"  
    android:text="LOGIN PAGE"  
    android:textAlignment="center"  
    android:textSize="30sp"/>
```

6. Add two edit text one after the other for username and password

7. Go to palette option-> Text-> **Plain Text**, Then Drag drop the Plain text into design window next to textView and update the following code

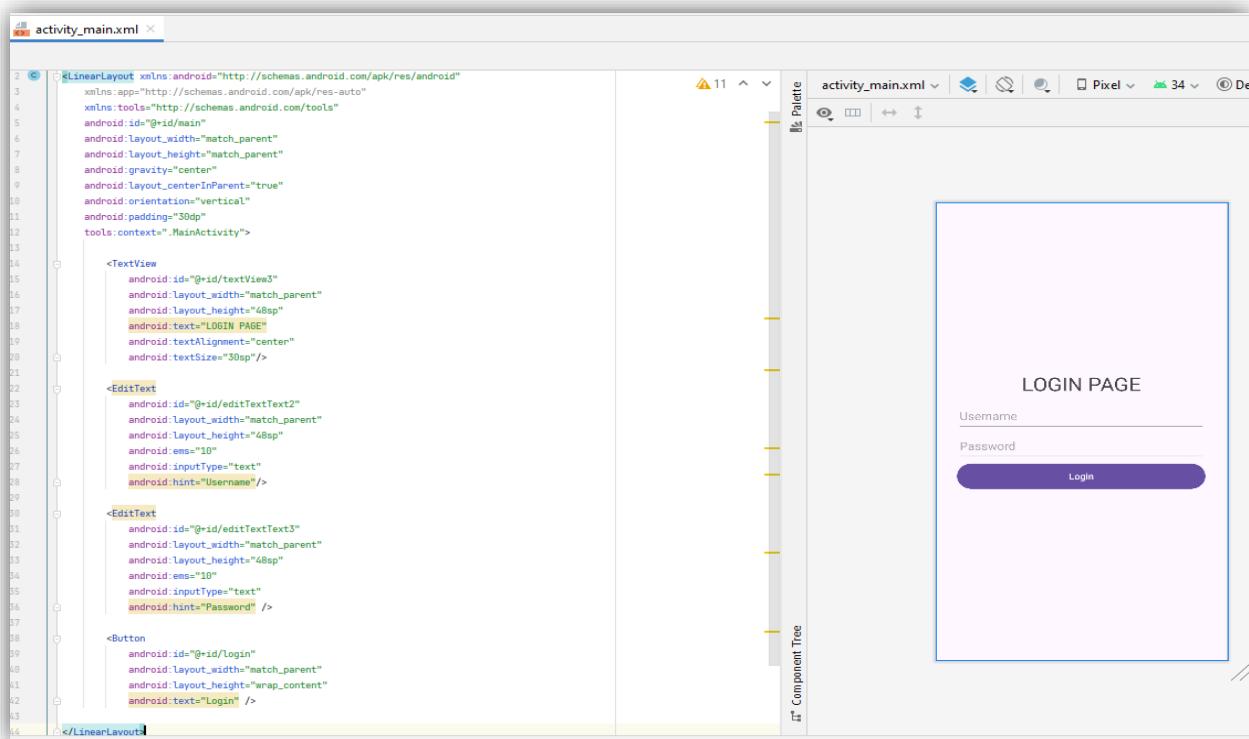
```
<EditText  
    android:id="@+id/editTextText2"  
    android:layout_width="match_parent"  
    android:layout_height="48sp"  
    android:ems="10"  
    android:inputType="text"  
    android:hint="Username"/>
```

8. Go to palette option-> Text-> **Plain Text**, Then Drag drop the Plain text into design window at bottom of previous Plain text and update the following code

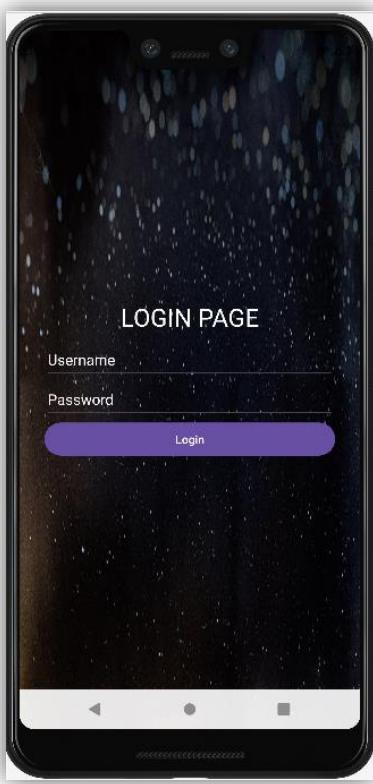
```
<EditText
    android:id="@+id/editTextText2"
    android:layout_width="match_parent"
    android:layout_height="48sp"
    android:ems="10"
    android:inputType="text"
    android:hint="Password"/>
```

9. Add a **ButtonView** for login and update the code.

```
<Button
    android:id="@+id/login"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Login" />
```



10. Press **Shift F10** to run the application Or, Press the run button on top of the interface



4. Create an application to implement new activity using explicit intent and implicit intent.

- **Explicit Intent:** This involves navigating from one activity to another within the same application.
 - **Implicit Intent:** This involves triggering an action that can be handled by another application.
1. Create a **new project** and choose **Empty Views Activity** then click **Next**.
 2. Change the layout from constraint Layout to Relative Layout in **activity_main.xml** file.
 From
`<androidx.constraintlayout.widget.ConstraintLayout..`
 To
`<LinearLayout..`
 3. update the following code in the linearlayout tag.
 `android:orientation="vertical"
 android:gravity="center"`

```

activity_main.xml
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res-auto"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:id="@+id/main"
6     android:layout_width="match_parent"
7     android:layout_height="match_parent"
8     tools:context=".MainActivity">
9
10 </androidx.constraintlayout.widget.ConstraintLayout>

```

```

activity_main.xml
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res-auto"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:id="@+id/main"
6     android:layout_width="match_parent"
7     android:layout_height="match_parent"
8     android:orientation="vertical" // This line is highlighted in blue
9     android:gravity="center" // This line is highlighted in blue
10    tools:context=".MainActivity">

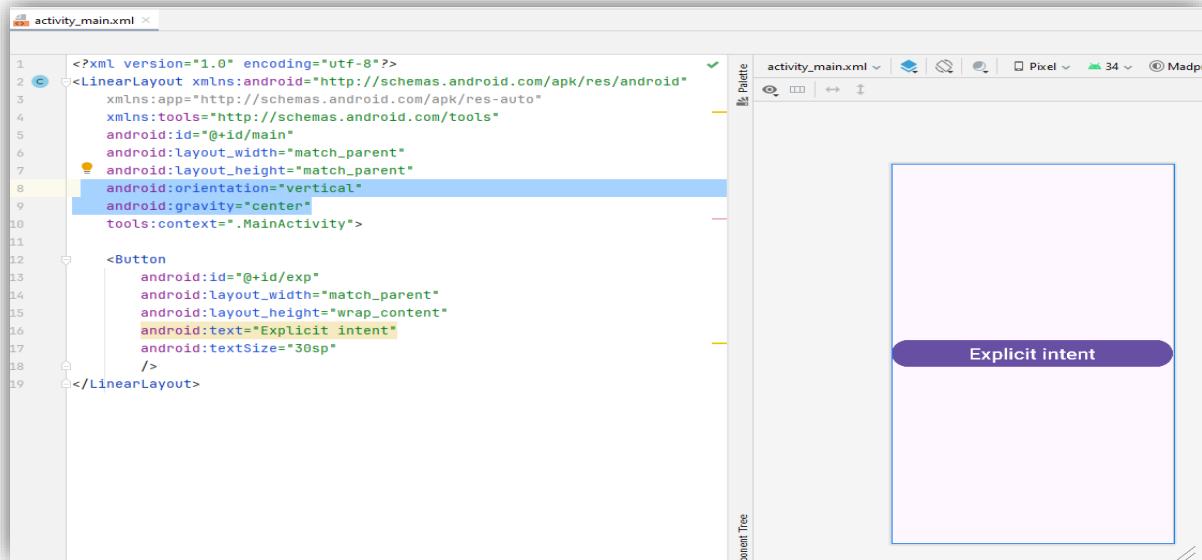
```

4. Add a **ButtonView** for explicit intent in **activity_main.xml** and update the code.

```

<Button
    android:id="@+id/exp"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Explicit intent"
    android:textSize="30sp"/>

```



5. Perform a logic for explicit intent in **MainActivity.java** file

- Importing necessary packages

```

import android.widget.Button;
import android.content.Intent;

```

- Add a object for explicit button

```
Button explicit;
```

- Add logic for button view using intent object

```

explicit=findViewById(R.id.exp);
explicit.setOnClickListener(logic -> {
    Intent intent = new Intent(MainActivity.this, SecondActivity.class);
    startActivity(intent);
});

```

```

package com.mad.madprog4;

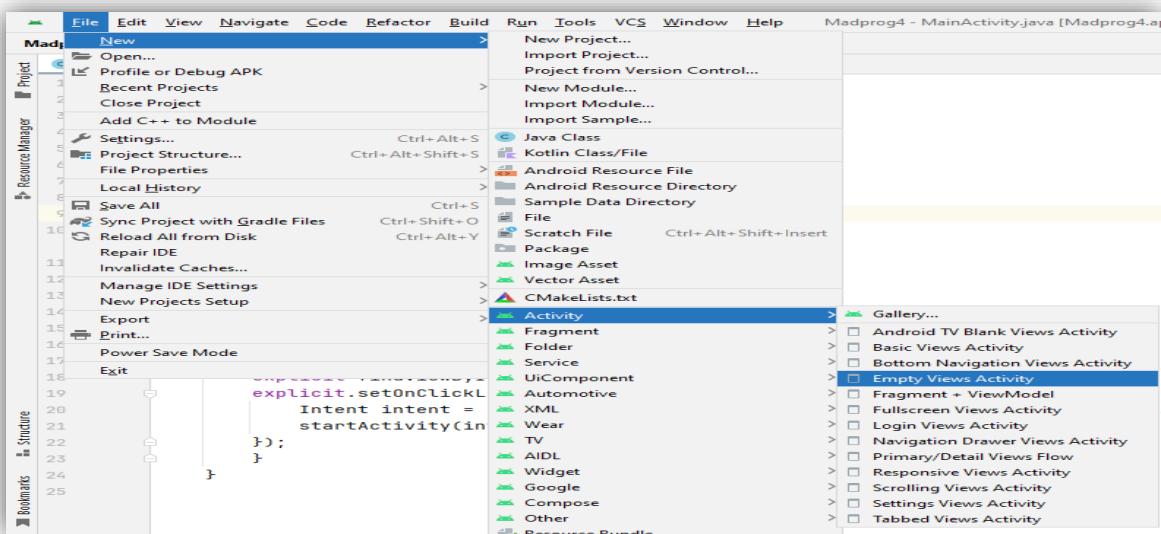
import android.os.Bundle;
import android.widget.Button;
import android.content.Intent;
import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    Button explicit;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable( SthisEnableEdgeToEdge: this );
        setContentView(R.layout.activity_main);
        explicit=findViewById(R.id.exp);
        explicit.setOnClickListener(logic -> {
            Intent intent = new Intent( packageContext: MainActivity.this, SecondActivity.class );
            startActivity(intent);
        });
    }
}

```

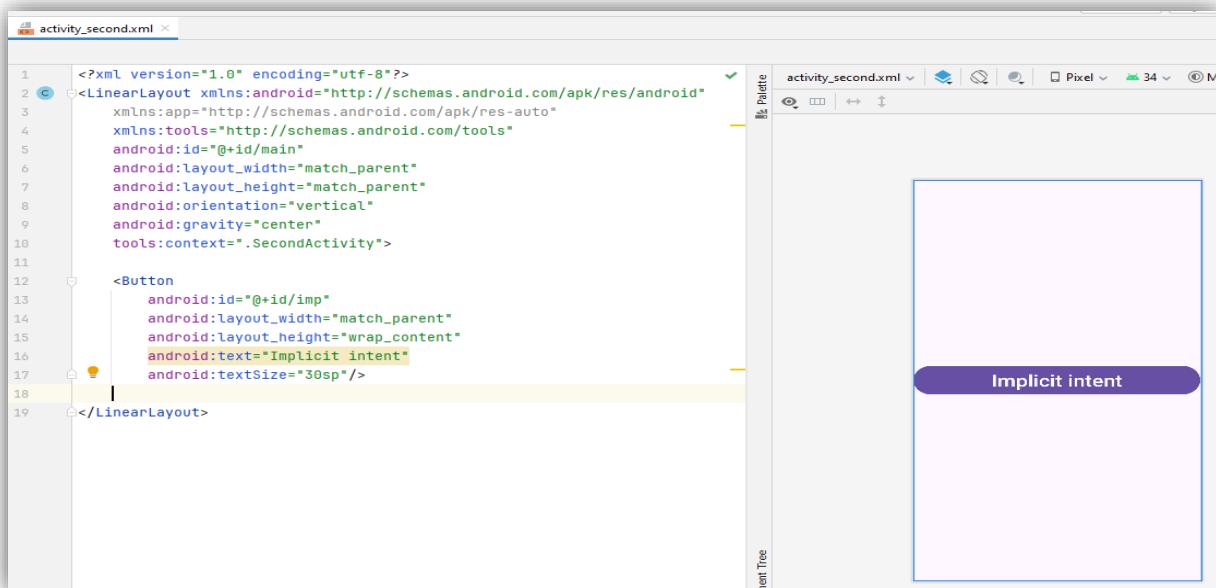
- Add an another empty view activity and rename it as **SecondActivity**
Go to file -> New -> Activity -> Empty View Activity -> (rename as SecondActivity) -> Finish
(It will create SecondActivity.java in java folder and activity_second.xml in res folder)



- Go to activity_second.xml file and change the layout to linear and add the following code
android:orientation="vertical"
android:gravity="center"

- Now add a **ButtonView** for implicit intent in **activity_second.xml** and update the code.

```
<Button
    android:id="@+id/imp"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Implicit intent"
    android:textSize="30sp"/>
```



- Perform a logic for implicit intent in **SecondActivity.java** file

- Importing necessary packages

```
import android.content.Intent;
import android.net.Uri;
import android.view.View;
import android.widget.Button;
```

- Add a object for explicit button

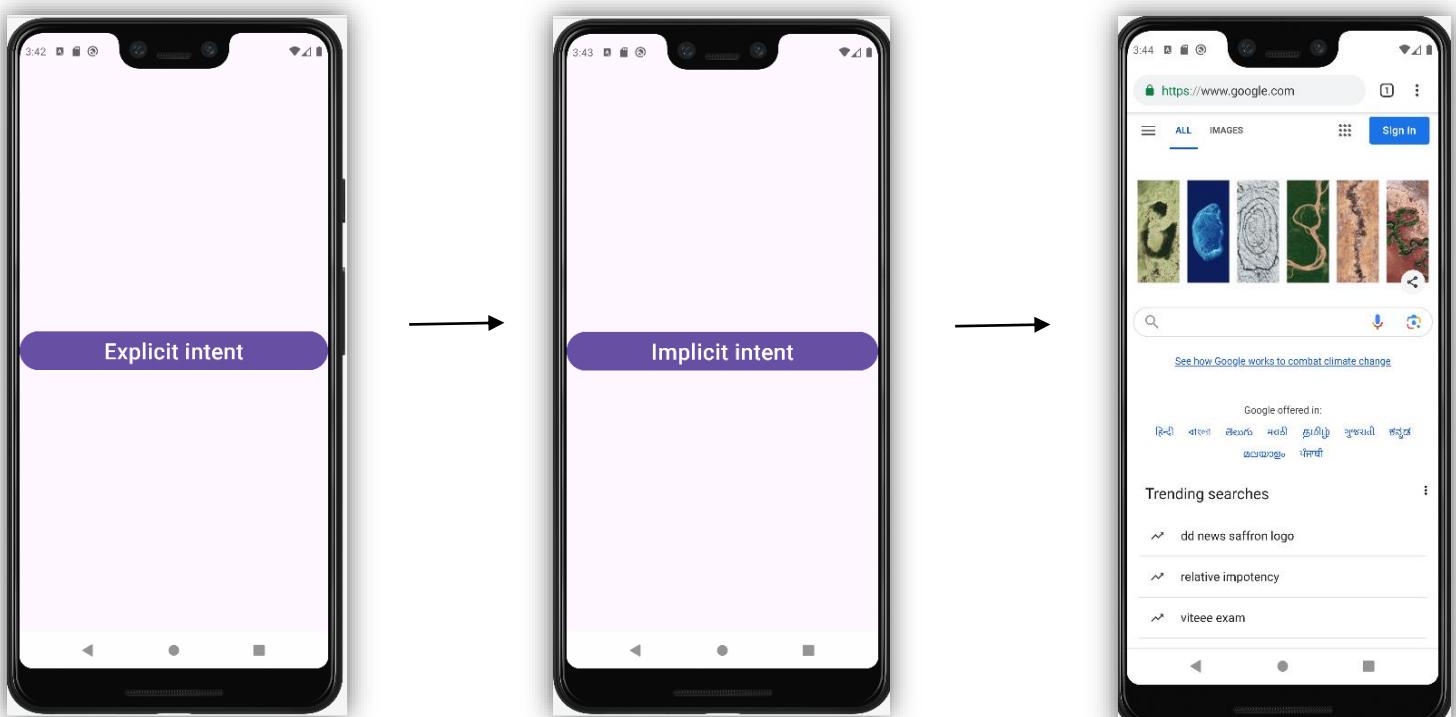
Button implicit;

- Add logic for button view using intent object

```
implicit=findViewById(R.id.imp);
implicit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Uri webpage = Uri.parse("https://www.google.com");
        Intent intent = new Intent(Intent.ACTION_VIEW, webpage);
        startActivity(intent);
    }
});
```

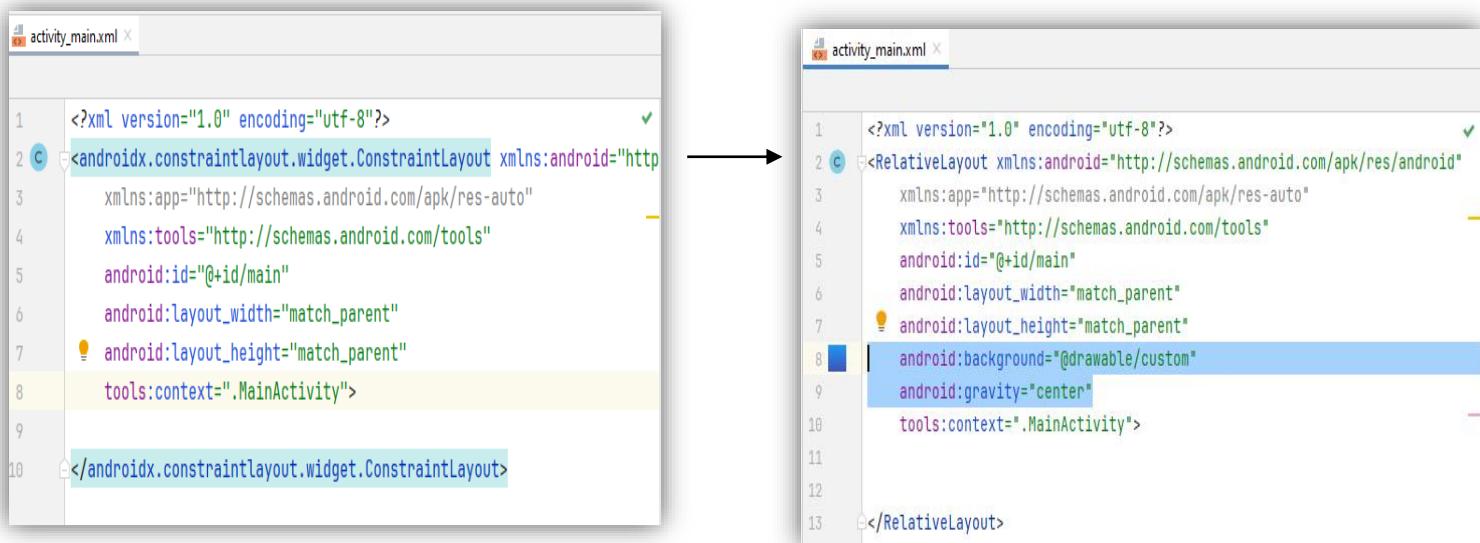
```
SecondActivity.java
1 package com.mad.madprog4;
2 import android.os.Bundle;
3 import android.content.Intent;
4 import android.net.Uri;
5 import android.view.View;
6 import android.widget.Button;
7
8 import androidx.activity.EdgeToEdge;
9 import androidx.appcompat.app.AppCompatActivity;
10
11
12 public class SecondActivity extends AppCompatActivity {
13     Button implicit;
14     @Override
15     protected void onCreate(Bundle savedInstanceState) {
16         super.onCreate(savedInstanceState);
17         EdgeToEdge.enable( SthisEnableEdgeToEdge: this );
18         setContentView(R.layout.activity_second);
19         implicit=findViewById(R.id.imp);
20         implicit.setOnClickListener(new View.OnClickListener() {
21             @Override
22             public void onClick(View view) {
23                 Uri webpage = Uri.parse(uriString: "https://www.google.com");
24                 Intent intent = new Intent(Intent.ACTION_VIEW, webpage);
25                 startActivity(intent);
26             }
27         });
28     }
29 }
30 }
```

10. Press **Shift F10** to run the application Or, Press the run button on top of the interface



5. Create an application that displays custom designed Opening Screen

1. Create a new project and choose **Empty Views Activity** then click **Next**.
2. Change the layout from constraint Layout to Relative Layout in **activity_main.xml** file.
`<androidx.constraintlayout.widget.ConstraintLayout..`
To
`<RelativeLayout...`
3. Change the background and update the following code in the Relativelayout tag.
 `android:background="@drawable/custom"
 android:gravity="center"`



The image shows two screenshots of the Android Studio code editor side-by-side, connected by a large black arrow pointing from left to right.

Left Screenshot: The XML code is for a `ConstraintLayout`. The entire code block is highlighted in light yellow. The code is as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

</androidx.constraintlayout.widget.ConstraintLayout>
```

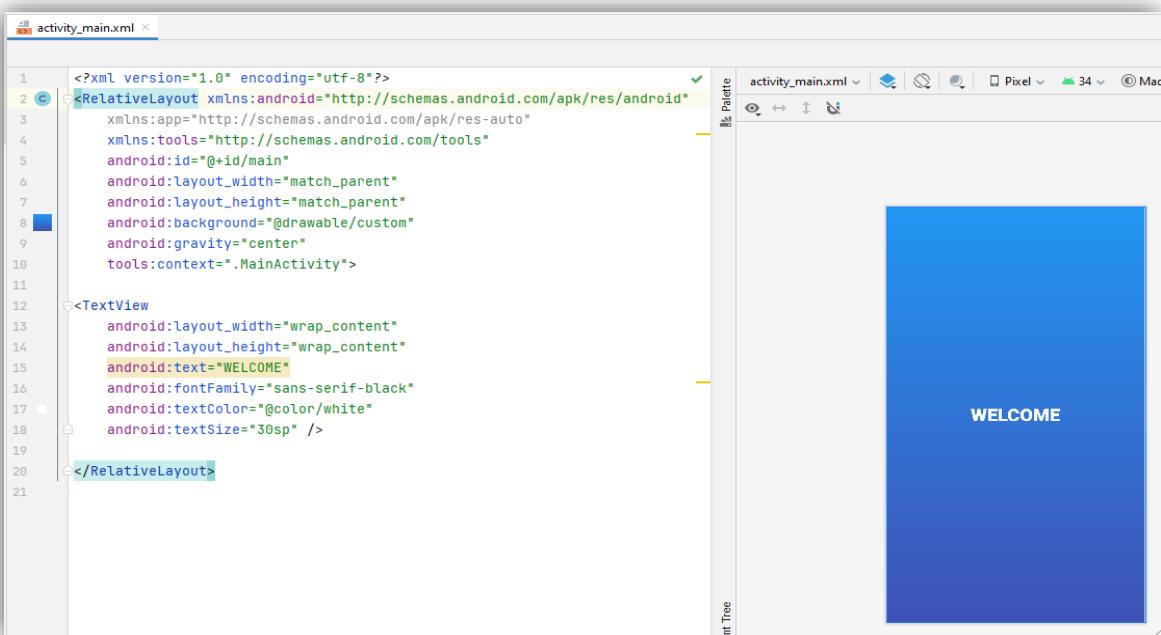
Right Screenshot: The XML code is now for a `RelativeLayout`. The entire code block is highlighted in light blue. The code has been modified at line 8:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/custom"
    android:gravity="center"
    tools:context=".MainActivity">

</RelativeLayout>
```

4. Go to palette option-> Common-> **TextView**, Then Drag drop the Button View into design window
5. Edit the text View content in **activity_main.xml** code file as follows:

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="WELCOME"
    android:textColor="@color/white"
    android:textSize="30sp" />
```



The image shows the Android Studio interface with the code editor and design preview side-by-side.

Code Editor: The XML code now includes a `TextView` element nested within the `RelativeLayout`:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/custom"
    android:gravity="center"
    tools:context=".MainActivity">

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="WELCOME"
    android:fontFamily="sans-serif-black"
    android:textColor="@color/white"
    android:textSize="30sp" />

</RelativeLayout>
```

Design Preview: The right pane shows a blue square with the word "WELCOME" centered in white text.

6. Update the following code in **MainActivity.java** file

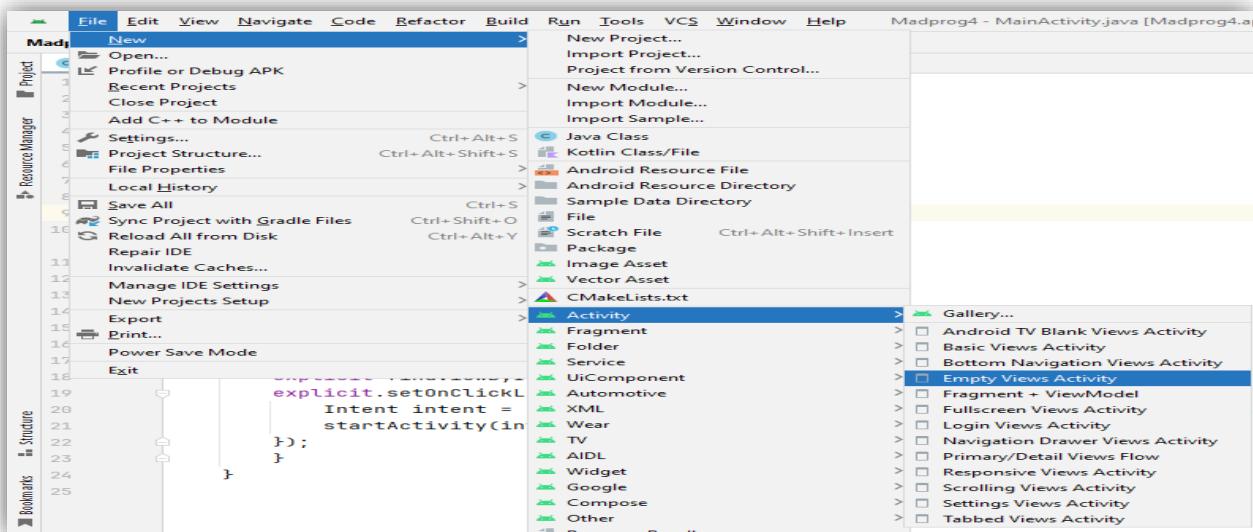
- Importing necessary packages:

```
import android.os.Handler;
import android.view.Window;
import android.content.Intent;
import android.view.WindowManager
```

```
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_NO_TITLE);
        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
                WindowManager.LayoutParams.FLAG_FULLSCREEN);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_main);
        Handler handler=new Handler();
        handler.postDelayed(new Runnable() {
            @Override
            public void run() {
                Intent intent=new Intent(MainActivity.this,MainActivity2.class);
                startActivity(intent);
                finish();
            }
        },3000);
    }
}
```

7. Add an another empty view activity and rename it as **MainActivity2**

Go to file -> New -> Activity -> Empty View Activity -> (rename as **MainActivity2) -> Finish**
*(It will create **mainactivity2.java** in **java** folder and **activity_main2.xml** in **res** folder)*



8. Change the layout from constraint Layout to Relative Layout in **activity_main2.xml** file.

`<androidx.constraintlayout.widget.ConstraintLayout..`

To

<RelativeLayout...

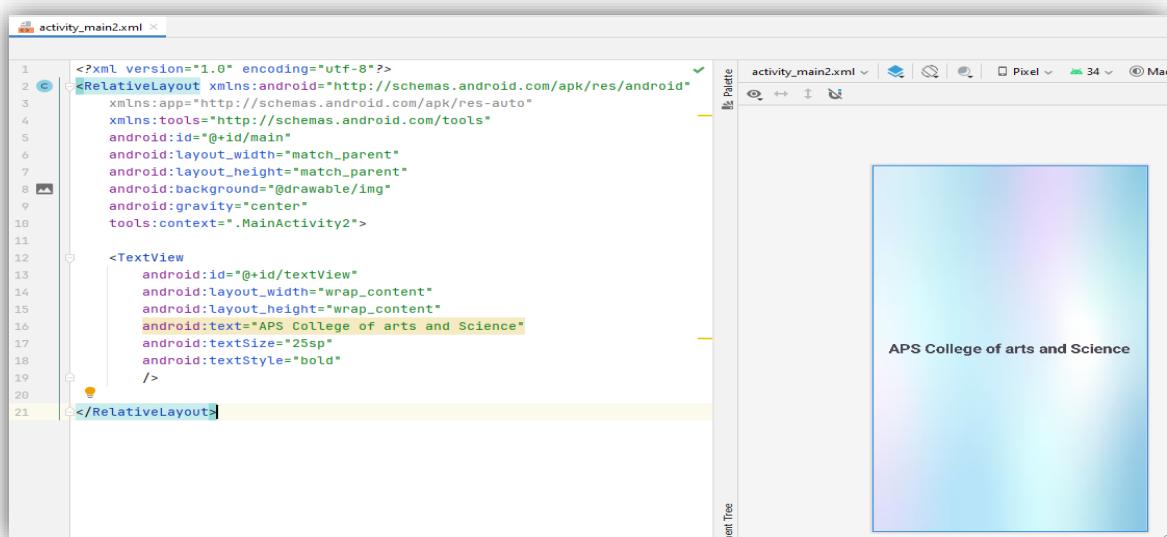
9. Change the background and update the following code in the Relativelayout tag.

android:gravity="center"

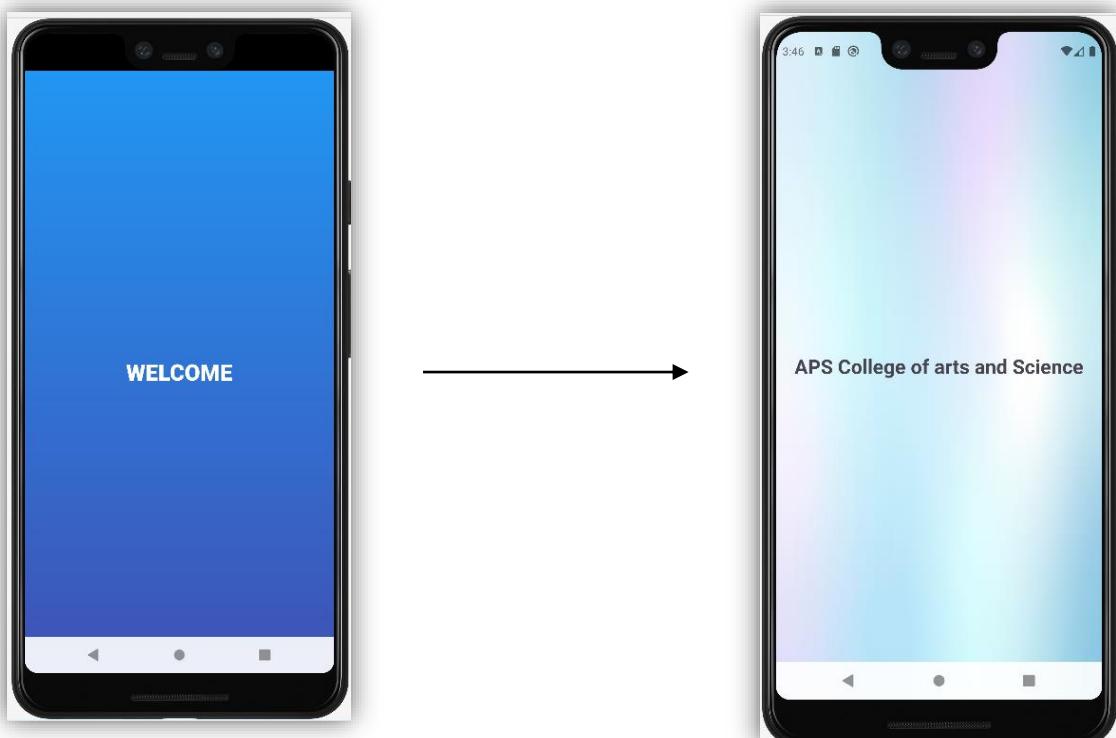
10. Add a TextView and edit the content in **activity_main2.xml** code file as follows:

<TextView

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="APS COLLEGE OF ARTS AND SCIENCE"
        android:textColor="@color/white"
        android:textSize="25sp"
        android:textstyle="bold" />
```

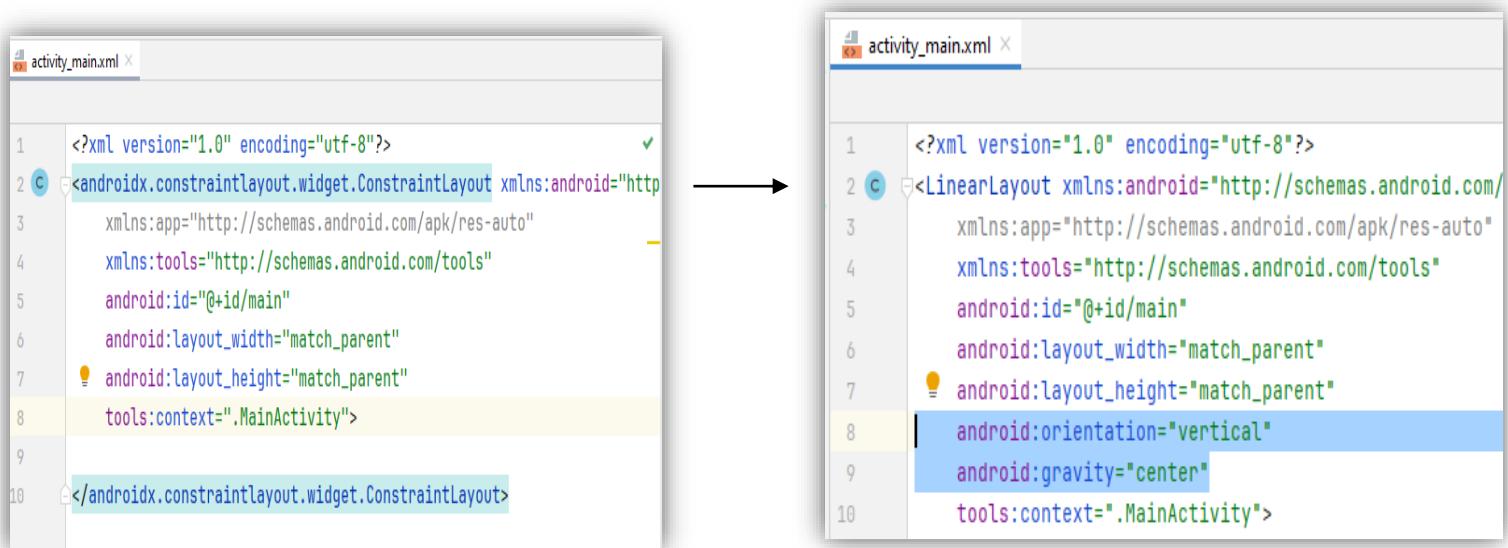


11. Press **Shift F10** to run the application Or, Press the run button on top of the interface



6. Create an UI with all views.

1. Create a **new project** and choose **Empty Views Activity** then click **Next**.
2. Change the layout from constraint Layout to Relative Layout in **activity_main.xml** file.
<androidx.constraintlayout.widget.ConstraintLayout..
To
<LinearLayout..
3. Change the background and update the following code in the linearlayout tag.
android:orientation="vertical"
android:gravity="center"



4. Add a Textview and edit the content in **activity_main.xml** code file as follows:

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="CONTACT US"
    android:textColor="@color/white"
    android:textSize="25sp"
    android:textstyle="bold" />
```

5. Add an edit text for Name
6. Go to palette option-> Text-> **Plain Text**, Then Drag drop the Plain text into design window next to textview and update the following code

```
<EditText
    android:id="@+id/editTextText2"
    android:layout_width="match_parent"
    android:layout_height="48sp"
    android:ems="10"
    android:inputType="text"
    android:hint="Name"/>
```

7. Add an Phone text for Phone No

Go to palette option-> Text-> **Phone**, Then Drag drop the Plain text into design window at bottom of previous Plain text and update the following code

```

<EditText
    android:id="@+id/editTextPhone"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ems="10"
    android:inputType="phone"
    android:text="Phone No"
/>

```

8. Add a multiline for message

Go to palette option-> Text-> **Multiline Text**, Then Drag drop the Plain text into design window at bottom of previous edit text and update the following code

```

<EditText
    android:id="@+id/editTextTextMultiLine"
    android:layout_width="match_parent"
    android:layout_height="100sp"
    android:ems="10"
    android:gravity="start|top"
    android:inputType="textMultiLine"
    android:text="Your Message"
/>

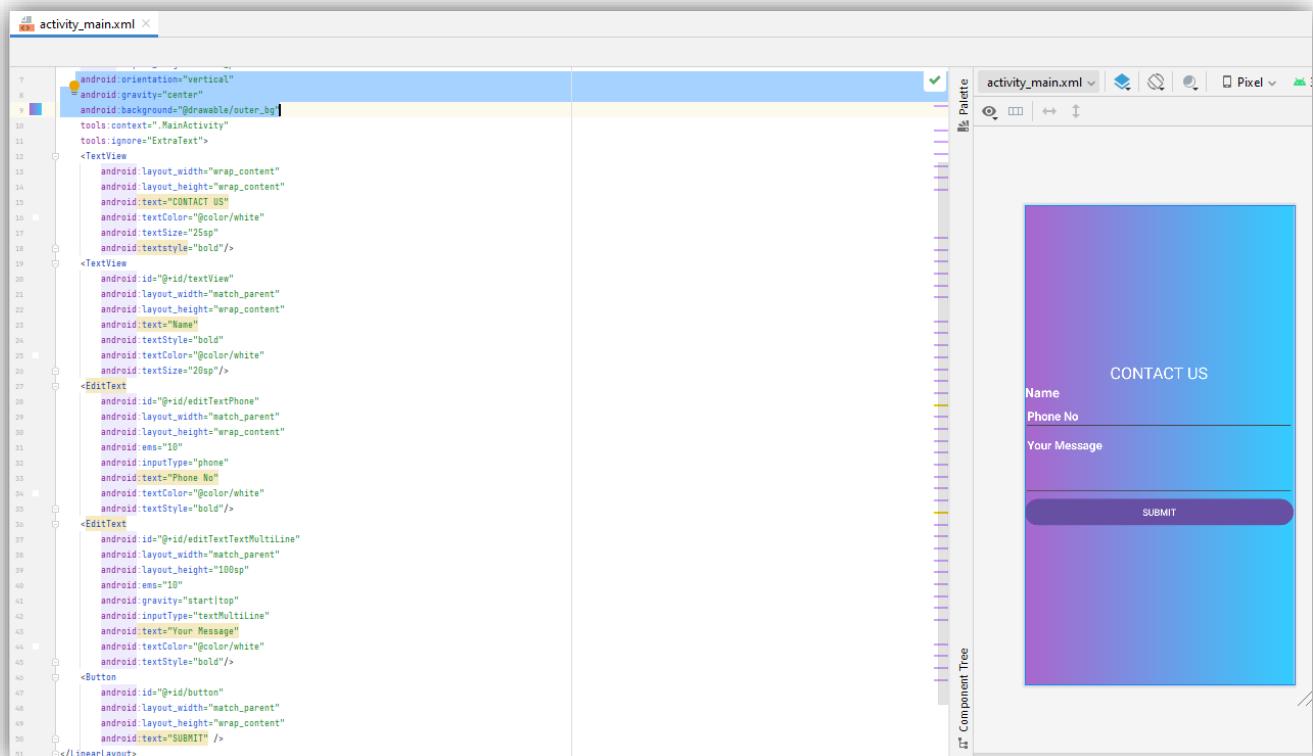
```

9. Add a **ButtonView** for submit and update the code.

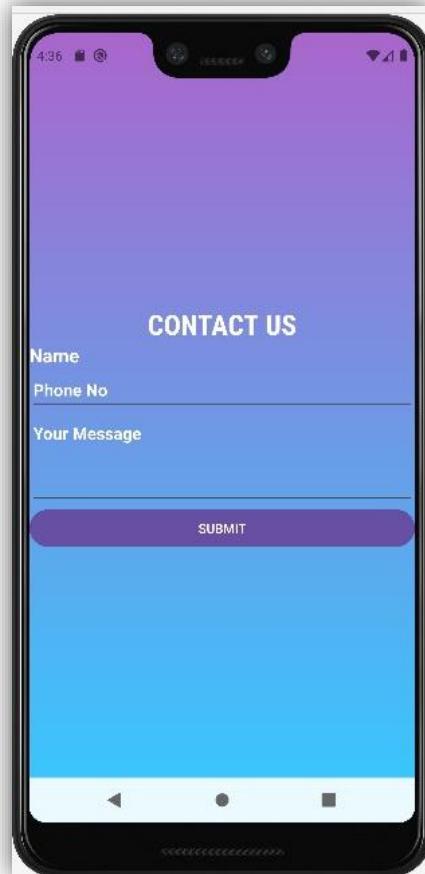
```

<Button
    android:id="@+id/login"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="SUBMIT" />

```



10. Press **Shift F10** to run the application Or, Press the run button on top of the interface



7. Create menu in application.

1. Create a **new project** and choose **Empty Views Activity** then click **Next**.
2. Change the Text View content in the code as follows:

<TextView

```
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="MenuOptionDemo"
    android:textSize="30sp"/>
```

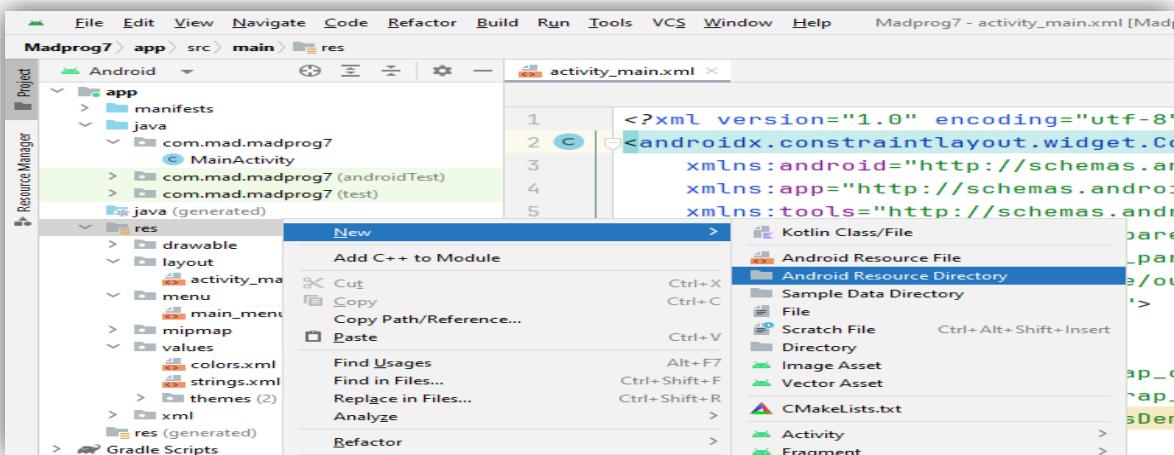
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/out"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="MenuOptionsDemo"
        android:textStyle="bold"
        android:textSize="30sp"
        android:textColor="@color/white"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

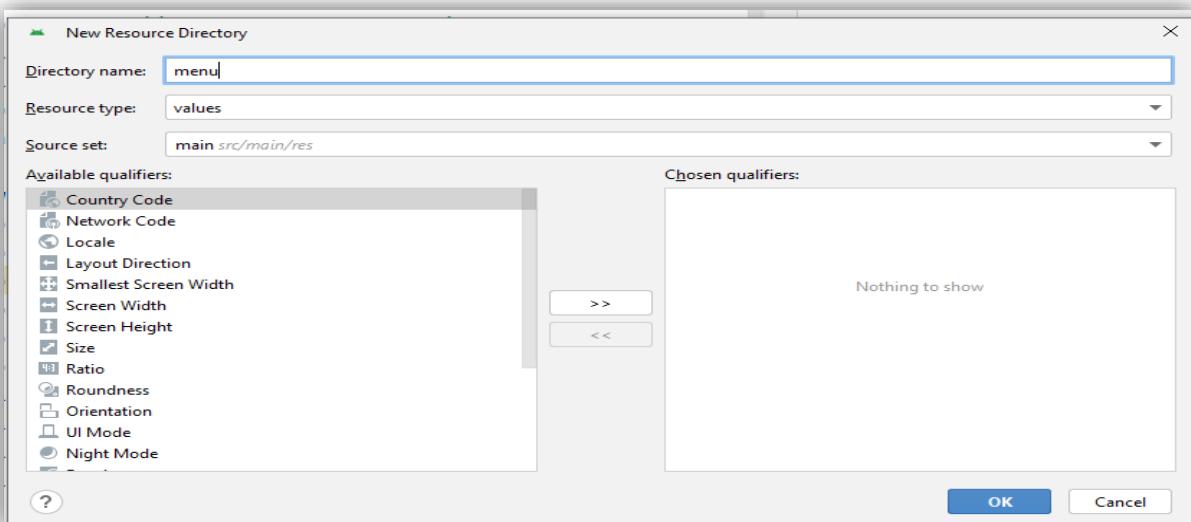
</androidx.constraintlayout.widget.ConstraintLayout>
```

3. Create a new directory for menu

Res (Right click)-> New- > Android Resource Directory

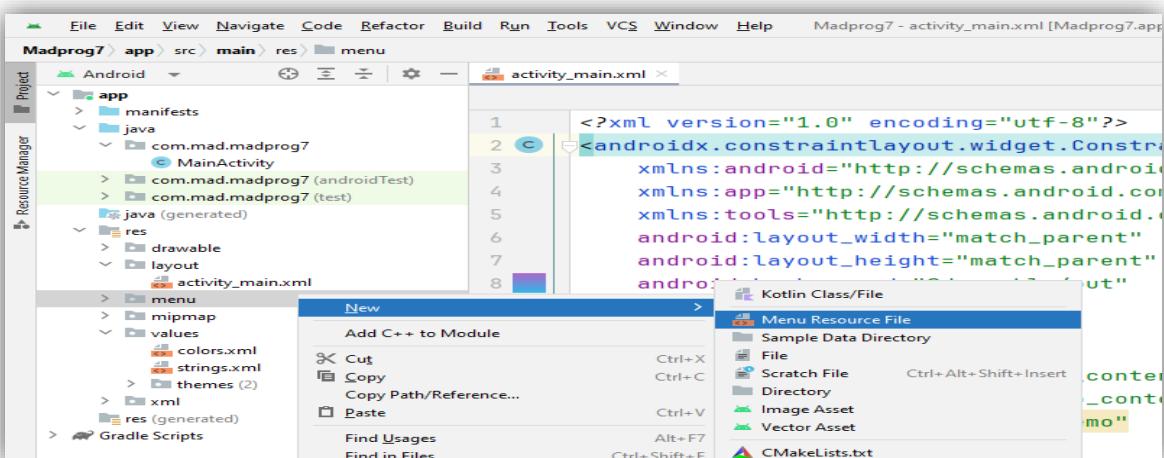


Change directory name to **menu**, Then Click **OK**. A menu folder will be created in res directory

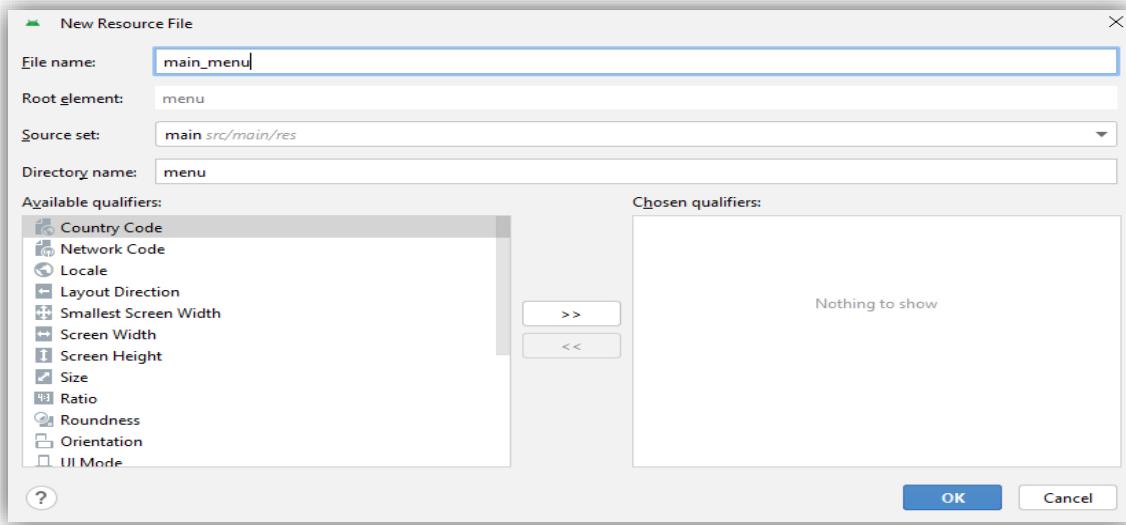


4. Create a main_menu.xml file under menu folder

Res-> menu(Right Click)-> New- > Menu resource file- > Filename: main_menu -> OK



Change file name to **menu**, Then Click **OK**. A main_menu.xml file will be created in menu folder.



5. Update the code in main_menu.xml file

```

<item
    android:id="@+id/item1"
    android:title="Profile" />

<item
    android:id="@+id/item2"
    android:title="settings"/>

<item
    android:id="@+id/item3"
    android:title="More options">
    <menu>
        <item
            android:id="@+id/sub1"
            android:title="Exit"/>
    </menu>
</item>

```

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <menu xmlns:app="http://schemas.android.com/apk/res-auto"
3      xmlns:android="http://schemas.android.com/apk/res/android">
4      <group />
5
6      <item
7          android:id="@+id/item1"
8          android:title="Profile" />
9
10     <item
11         android:id="@+id/item2"
12         android:title="settings"/>
13
14     <item
15         android:id="@+id/item3"
16         android:title="More options">
17         <menu>
18             <item
19                 android:id="@+id/sub1"
20                 android:title="Exit"/>
21         </menu>
22     </item>
23
24 </menu>

```

6. Now update the following code in **MainActivity.java** file

- Importing necessary packages:

```
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;
```

- create a method for menu options

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.main_menu, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    int id = item.getItemId();
    if (id == R.id.item1) {
        Toast.makeText(this, "You selected Profile option", Toast.LENGTH_SHORT).show();
    } else if (id == R.id.item2) {
        Toast.makeText(this, "You have selected Settings option",
        Toast.LENGTH_SHORT).show();
    } else if (id == R.id.sub1) {
        Toast.makeText(this, "You have selected Exit option", Toast.LENGTH_SHORT).show();
    } else {
        return super.onOptionsItemSelected(item);
    }

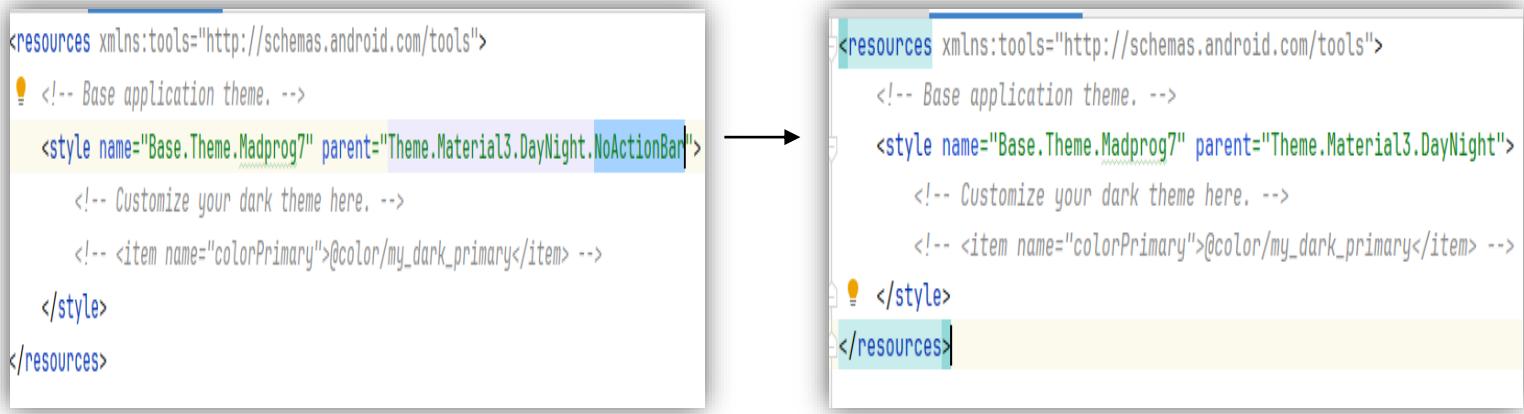
    return true;
}
```

The screenshot shows the Android Studio code editor with the file `MainActivity.java` open. The code is identical to the one provided in the previous text block. Several breakpoints are set in the code, indicated by small red dots with arrows on the left margin. The code is syntax-highlighted, with keywords in blue, comments in green, and strings in red. The editor interface includes toolbars at the top and a navigation bar at the bottom.

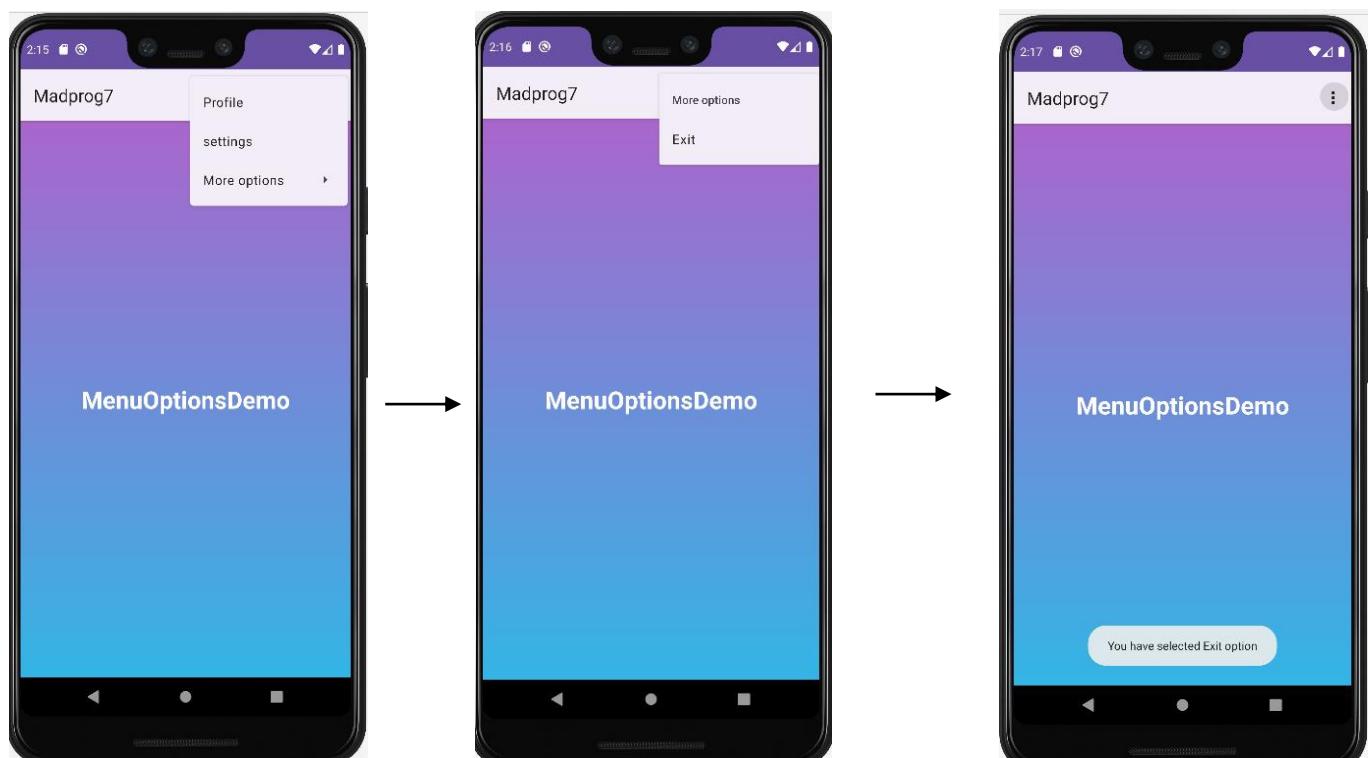
```
5     import android.view.Menu;
6     import android.view.MenuItem;
7     import android.widget.Toast;
8
9     public class MainActivity extends AppCompatActivity {
10         @Override
11         protected void onCreate(Bundle savedInstanceState) {
12             super.onCreate(savedInstanceState);
13             setContentView(R.layout.activity_main);
14         }
15
16         @Override
17         public boolean onCreateOptionsMenu(Menu menu) {
18             getMenuInflater().inflate(R.menu.main_menu, menu);
19             return true;
20         }
21
22         @Override
23         public boolean onOptionsItemSelected(@NonNull MenuItem item) {
24             int id = item.getItemId();
25             if (id == R.id.item1) {
26                 Toast.makeText(context, "You selected Profile option", Toast.LENGTH_SHORT).show();
27             } else if (id == R.id.item2) {
28                 Toast.makeText(context, "You have selected Settings option", Toast.LENGTH_SHORT).show();
29             } else if (id == R.id.sub1) {
30                 Toast.makeText(context, "You have selected Exit option", Toast.LENGTH_SHORT).show();
31             } else {
32                 return super.onOptionsItemSelected(item);
33             }
34
35         }
36     }
```

7. Go to themes and remove the NoActionBar attribute in both themes.xml and theme.xml(night)
- **Res-> Values-> themes**

```
<style name="Base.Theme.Madprog7" parent="Theme.Material3.DayNight.NoActionBar">  
To  
<style name="Base.Theme.Madprog7" parent="Theme.Material3.DayNight ">
```



8. Press **Shift F10** to run the application Or, Press the run button on top of the interface



8. Read / Write the Local data.

1. Create a **new project** and choose **Empty Views Activity** then click **Next**.

Change the layout from constraint Layout to Relative Layout in **activity_main.xml** file.

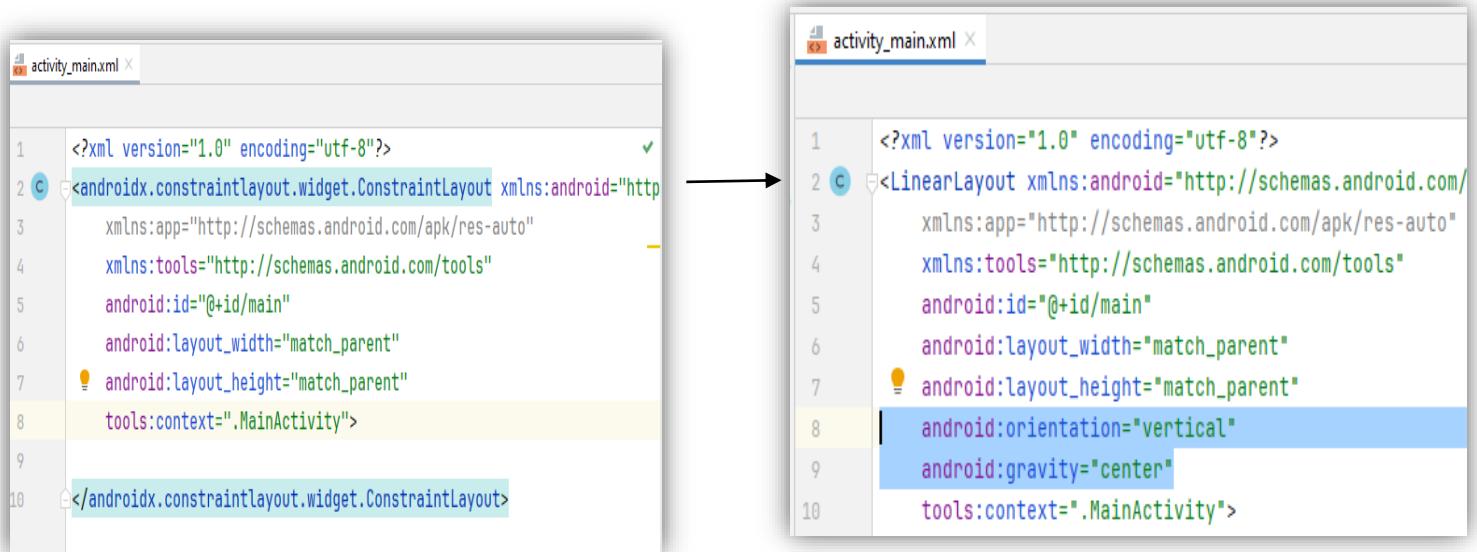
<androidx.constraintlayout.widget.ConstraintLayout..

To

<LinearLayout..

update the following code in the linearlayout tag.

**android:orientation="vertical"
 android:gravity="center"**



2. Add a Edit Text and edit the content in **activity_main.xml** code file as follows:

```
<EditText
    android:id="@+id/etText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Enter any Text"/>
```

Again add an EditText for fetching data

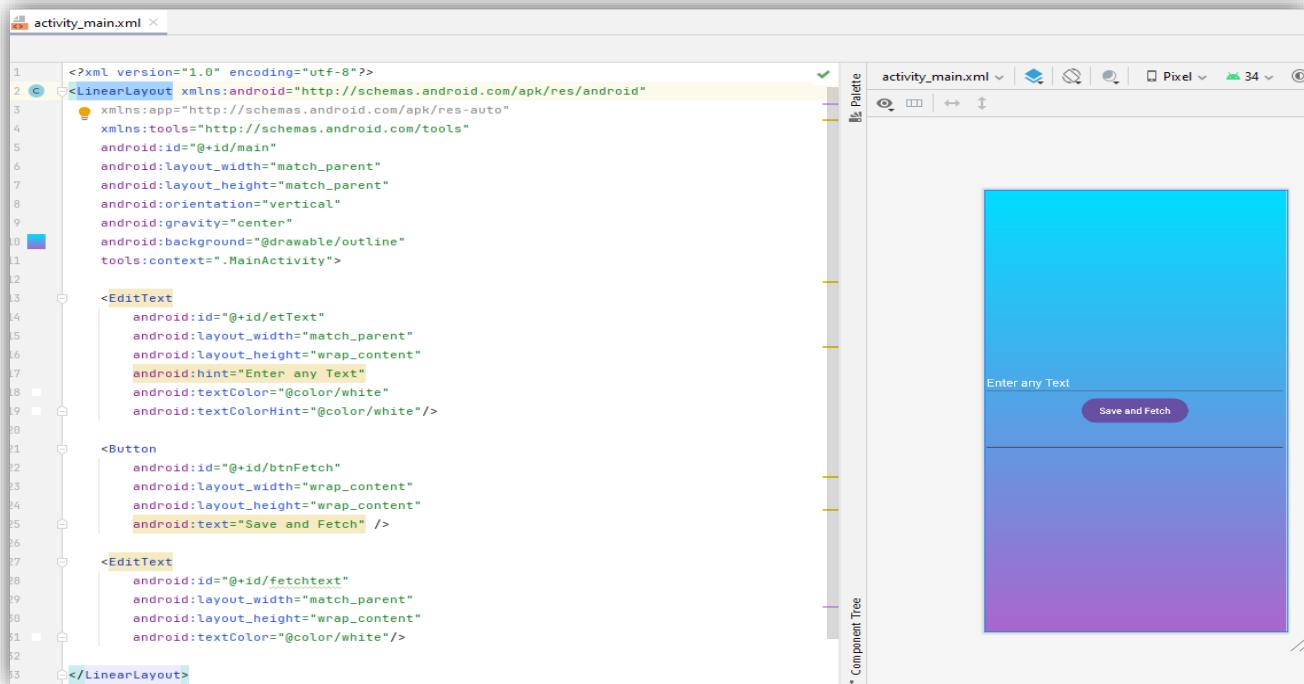
```
<EditText
    android:id="@+id/fetchtext"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>
```

3. Then add a button to read and fetch the data

```
<Button
    android:id="@+id/btnFetch"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Save and Fetch" />
```

Again add an EditText for fetching data

```
<EditText
    android:id="@+id/fetchtext"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>
```



4. Now update the following code in **MainActivity.java** file

- Importing necessary packages:

```
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
```

- Create a method for reading and writing

```
public class MainActivity extends AppCompatActivity {
```

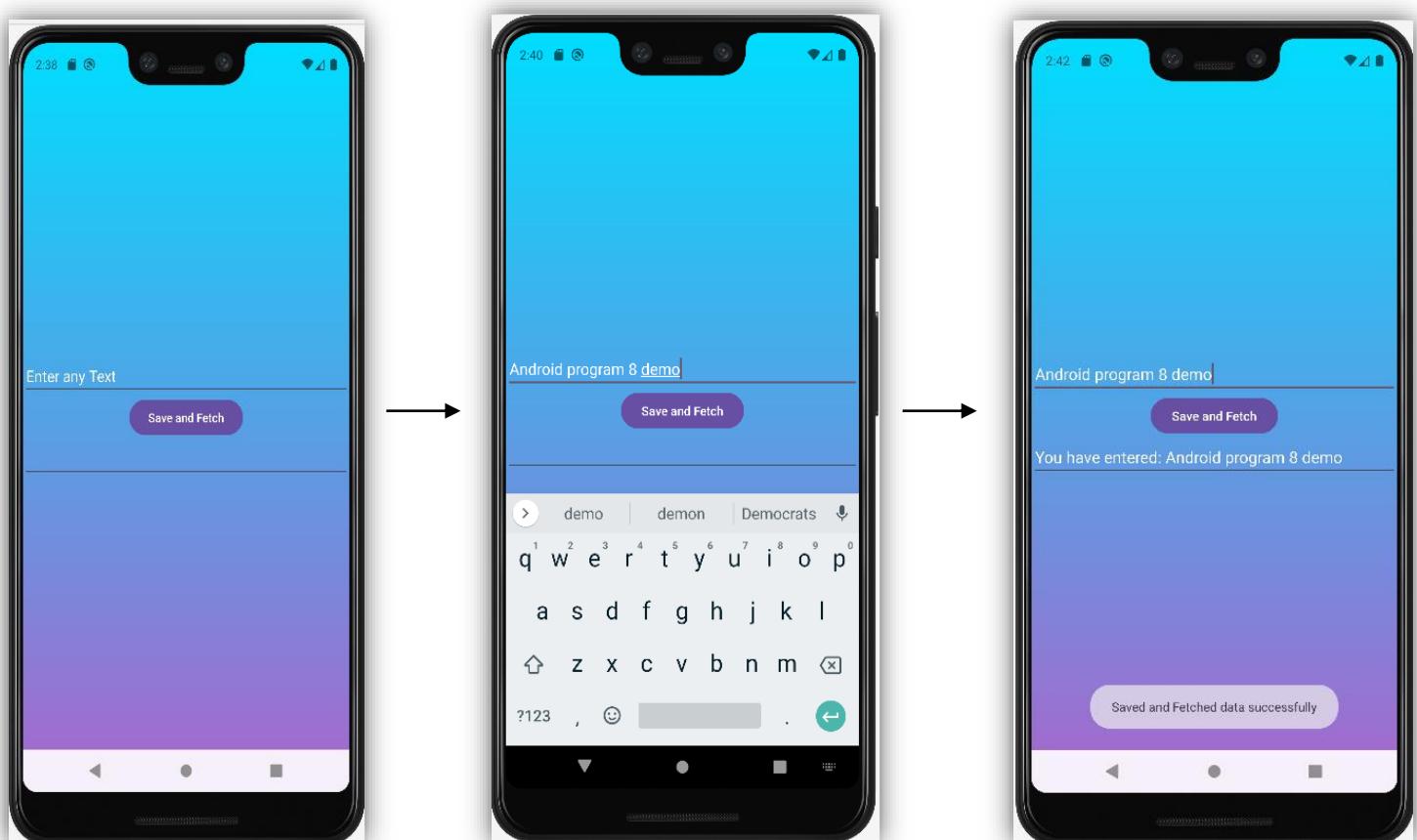
```
    EditText etText,fetchtext;
    Button btnFetch;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_main);

        etText = (EditText) findViewById(R.id.etText);
        fetchtext=(EditText) findViewById(R.id.fetchtext);
        btnFetch = (Button) findViewById(R.id.btnFetch);
```

```
        btnFetch.setOnClickListener(new View.OnClickListener() {
                        @Override
                        public void onClick(View v) {
                String outputText = etText.getText().toString();
                fetchtext.setText("You have entered: " + outputText);
                Toast.makeText(getApplicationContext(),"Saved and Fetched data
successfully",Toast.LENGTH_SHORT).show();
            }
        });
    }
```

```
activity_main.xml x MainActivity.java x
8
9     import androidx.activity.EdgeToEdge;
10    import androidx.appcompat.app.AppCompatActivity;
11
12    public class MainActivity extends AppCompatActivity {
13        EditText etText,fetchtext;
14        Button btnFetch;
15
16        @Override
17        protected void onCreate(Bundle savedInstanceState) {
18            super.onCreate(savedInstanceState);
19            EdgeToEdge.enable( this.getEdgeToEdge() );
20            setContentView(R.layout.activity_main);
21
22            etText = (EditText) findViewById(R.id.etText);
23            fetchtext=(EditText) findViewById(R.id.fetchtext);
24            btnFetch = (Button) findViewById(R.id.btnFetch);
25
26            btnFetch.setOnClickListener(new View.OnClickListener() {
27                @Override
28                public void onClick(View v) {
29                    String outputText = etText.getText().toString();
30                    fetchtext.setText("You have entered: " + outputText);
31                    Toast.makeText(getApplicationContext(), "Saved and Fetched data successfully", Toast.LENGTH_SHORT).show();
32                }
33            });
34        }
35    }
```

5. Press **Shift F10** to run the application Or, Press the run button on top of the interface



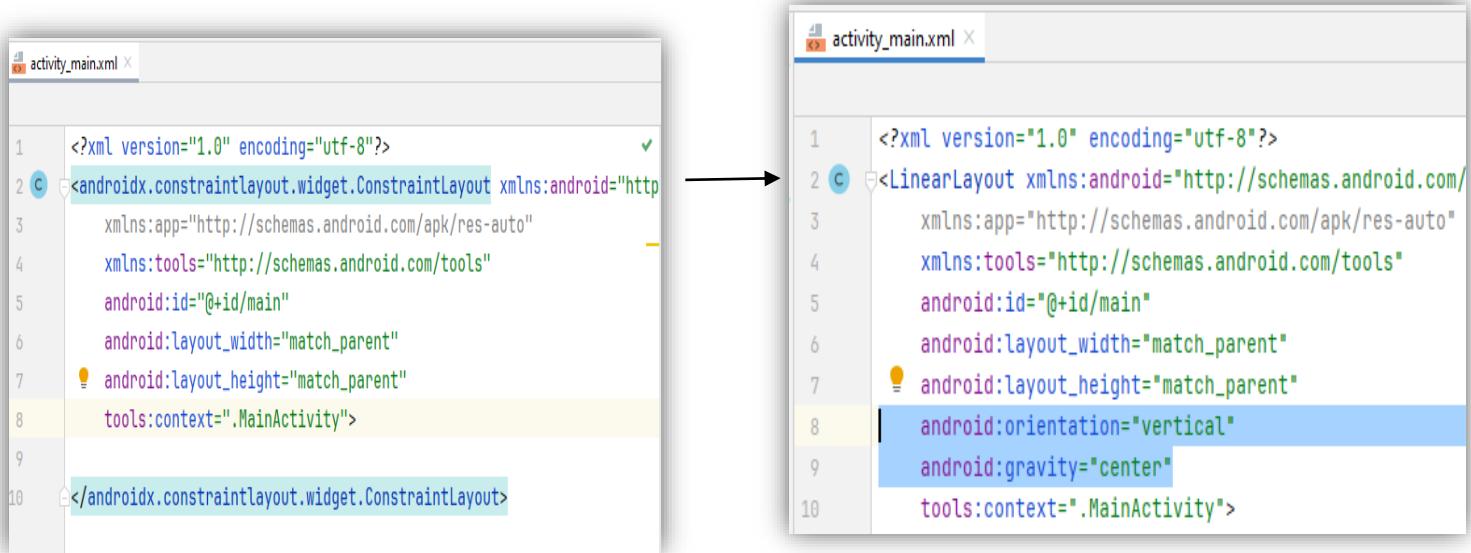
9. Create / Read / Write data with database (SQL Lite)

1. Create a **new project** and choose **Empty Views Activity** then click **Next**.

Change the layout from constraint Layout to Relative Layout in **activity_main.xml** file.
<androidx.constraintlayout.widget.ConstraintLayout..
To
<LinearLayout..

update the following code in the linearlayout tag.

**android:orientation="vertical"
 android:gravity="center"**



2. Add a Text and edit view, then edit the content in **activity_main.xml** code file as follows:

**<TextView
 android:id="@+id/textView"
 android:layout_width="match_parent"
 android:layout_height="wrap_content"
 android:text="CONTACT US"/>**

3. Add edit views for phone no and message

**<EditText
 android:id="@+id/Phno"
 android:layout_width="match_parent"
 android:layout_height="48dp"
 android:layout_margin="10dp"
 android:hint="Phone number" />**

**<EditText
 android:id="@+id/Msg"
 android:layout_width="match_parent"
 android:layout_height="100dp"
 android:layout_margin="10dp"
 android:hint="Your message" />**

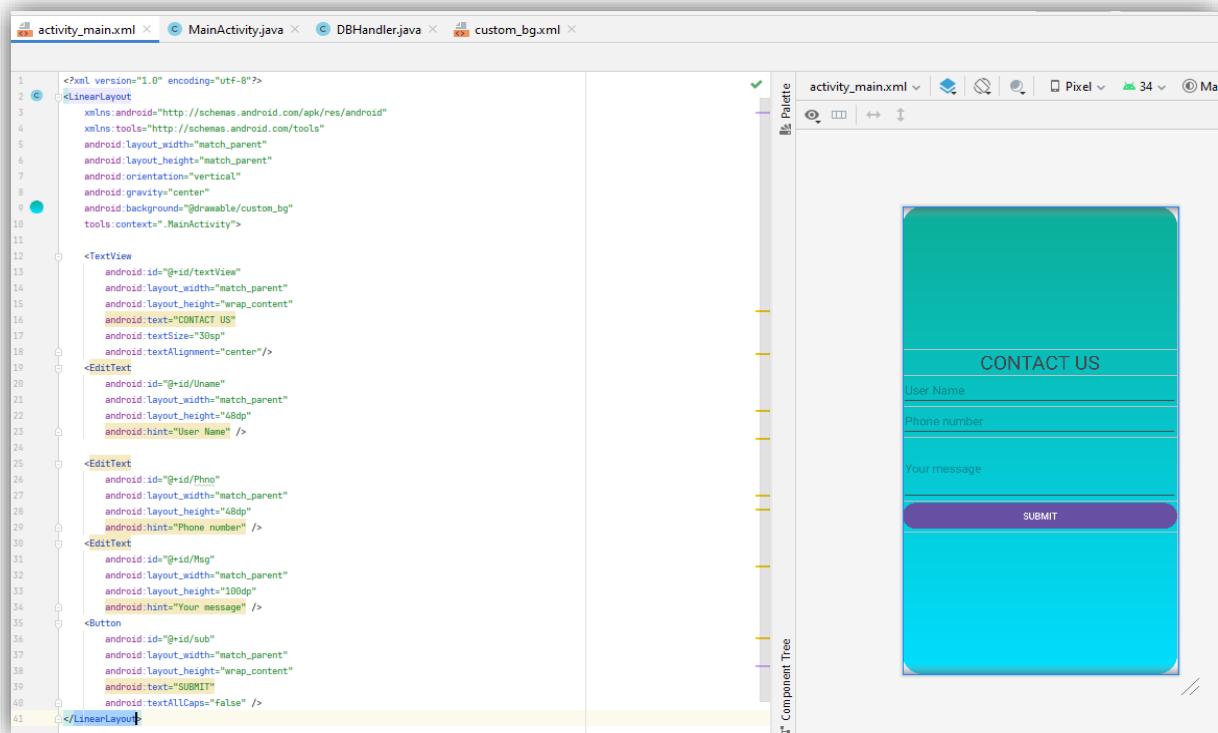
4. Add a button for submit

<Button

```

    android:id="@+id/sub"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:text="SUBMIT"/>

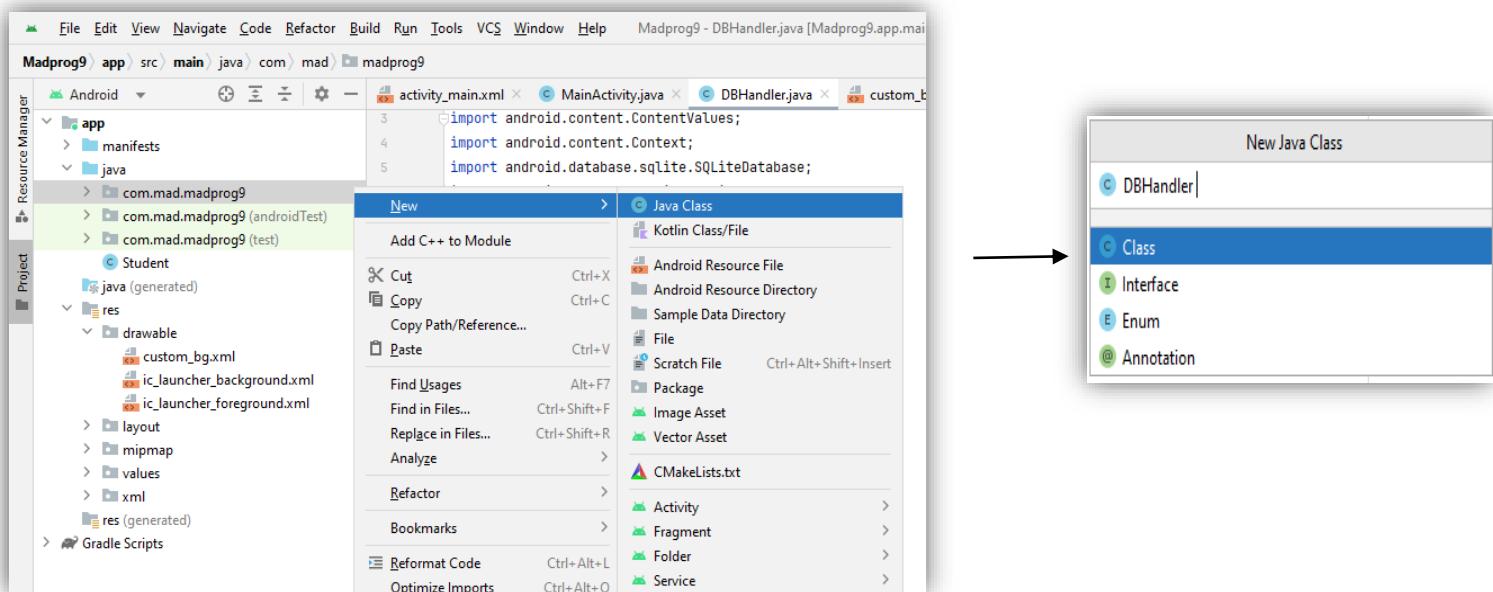
```



5. Creating a new Java class for performing SQLite operations

App -> java -> Package name(`com.mad.madprog9`) -> Right click->New -> java class

Give name as `DBHandler`



- Edit the content in `DBHandler`

`package com.mad.madprog9;`

```

import android.content.ContentValues;
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;

```

```

import android.database.sqlite.SQLiteOpenHelper;

public class DBHandler extends SQLiteOpenHelper {

    // below variable is for our database name.
    private static final String DB_NAME = "Feedbackdb";
    private static final int DB_VERSION = 1;
    private static final String TABLE_NAME = "contactus";
    private static final String ID_COL = "id";
    private static final String NAME_COL = "name";
    private static final String PHONE_COL = "phone_number";
    private static final String MESSAGE_COL = "message";

    public DBHandler(Context context) {
        super(context, DB_NAME, null, DB_VERSION);
    }

    // below method is for creating a database by running a sqlite query
    @Override
    public void onCreate(SQLiteDatabase db) {
        String query = "CREATE TABLE " + TABLE_NAME + "("
                + ID_COL + " INTEGER PRIMARY KEY AUTOINCREMENT,"
                + NAME_COL + " TEXT,"
                + PHONE_COL + " NUMBER,"
                + MESSAGE_COL + " TEXT)";
        db.execSQL(query);
    }

    // this method is use to add new course to our sqlite database.
    public void addNewCourse(String username, String phoneno, String message) {

        SQLiteDatabase db = this.getWritableDatabase();

        ContentValues values = new ContentValues();
        values.put(NAME_COL, username);
        values.put(PHONE_COL, phoneno);
        values.put(MESSAGE_COL, message);

        db.insert(TABLE_NAME, null, values);
        db.close();
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        // this method is called to check if the table exists already.
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
        onCreate(db);
    }
}

```

```

1 package com.mad.madprog9;
2 import android.content.ContentValues;
3 import android.content.Context;
4 import android.database.sqlite.SQLiteDatabase;
5 import android.database.sqlite.SQLiteOpenHelper;
6
7     2 usages
8     public class DBHandler extends SQLiteOpenHelper {
9         // below variable is for our database name.
10        1 usage
11        private static final String DB_NAME = "Feedbackdb";
12        1 usage
13        private static final int DB_VERSION = 1;
14        3 usages
15        private static final String TABLE_NAME = "contactus";
16        1 usage
17        private static final String ID_COL = "id";
18        2 usages
19        private static final String NAME_COL = "name";
20        2 usages
21        private static final String PHONE_COL = "phone_number";
22        2 usages
23        private static final String MESSAGE_COL = "message";
24        1 usage
25
26        public DBHandler(Context context) { super(context, DB_NAME, null, DB_VERSION); }
27        // below method is for creating a database by running a sqlite query
28        @Override
29        public void onCreate(SQLiteDatabase db) {
30            String query = "CREATE TABLE " + TABLE_NAME + " (" +
31                + ID_COL + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
32                + NAME_COL + " TEXT," +
33                + PHONE_COL + " NUMBER," +
34                + MESSAGE_COL + " TEXT)";
35            db.execSQL(query);
36        }
37        // this method is use to add new course to our sqlite database.
38        @Override
39        public void addNewCourse(String username, String phoneno, String message) {
40            SQLiteDatabase db = this.getWritableDatabase();
41            ContentValues values = new ContentValues();
42            values.put(NAME_COL, username);
43            values.put(PHONE_COL, phoneno);
44            values.put(MESSAGE_COL, message);
45            db.insert(TABLE_NAME, null, values);
46            db.close();
47        }
48
49        10 usages
50        @Override
51        public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
52            // this method is called to check if the table exists already.
53            db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
54            onCreate(db);
55        }

```

9. Go to **mainActivity.java** for managing database and layout

- Import necessary libraries.

```

import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

```

- create a method for database

```

private EditText usernameEdt, phonenoEdt, messageEdt;
private DBHandler dbHandler;

```

@Override

```

protected void onCreate(Bundle savedInstanceState) {

```

```

    super.onCreate(savedInstanceState);

```

```

    setContentView(R.layout.activity_main);

```

```

    usernameEdt = findViewById(R.id.Uname);

```

```

    phonenoEdt = findViewById(R.id.Phno);

```

```

    messageEdt = findViewById(R.id.Msg);

```

```

    Button submitBtn = findViewById(R.id.sub);

```

```

    dbHandler = new DBHandler(MainActivity.this);

```

```

    submitBtn.setOnClickListener(new View.OnClickListener() {

```

@Override

```

public void onClick(View v) {
    String userName = usernameEdt.getText().toString();
    String phoneno = phonenoEdt.getText().toString();
    String message = messageEdt.getText().toString();

    if (userName.isEmpty() && phoneno.isEmpty() && message.isEmpty()) {
        Toast.makeText(MainActivity.this, "Please enter all the data..",
        Toast.LENGTH_SHORT).show();
        return;
    }
    dbHandler.addNewCourse(userName, phoneno, message);
    Toast.makeText(MainActivity.this, "Your message sent successfully",
    Toast.LENGTH_SHORT).show();
    usernameEdt.setText("");
    phonenoEdt.setText("");
    messageEdt.setText("");
}
});

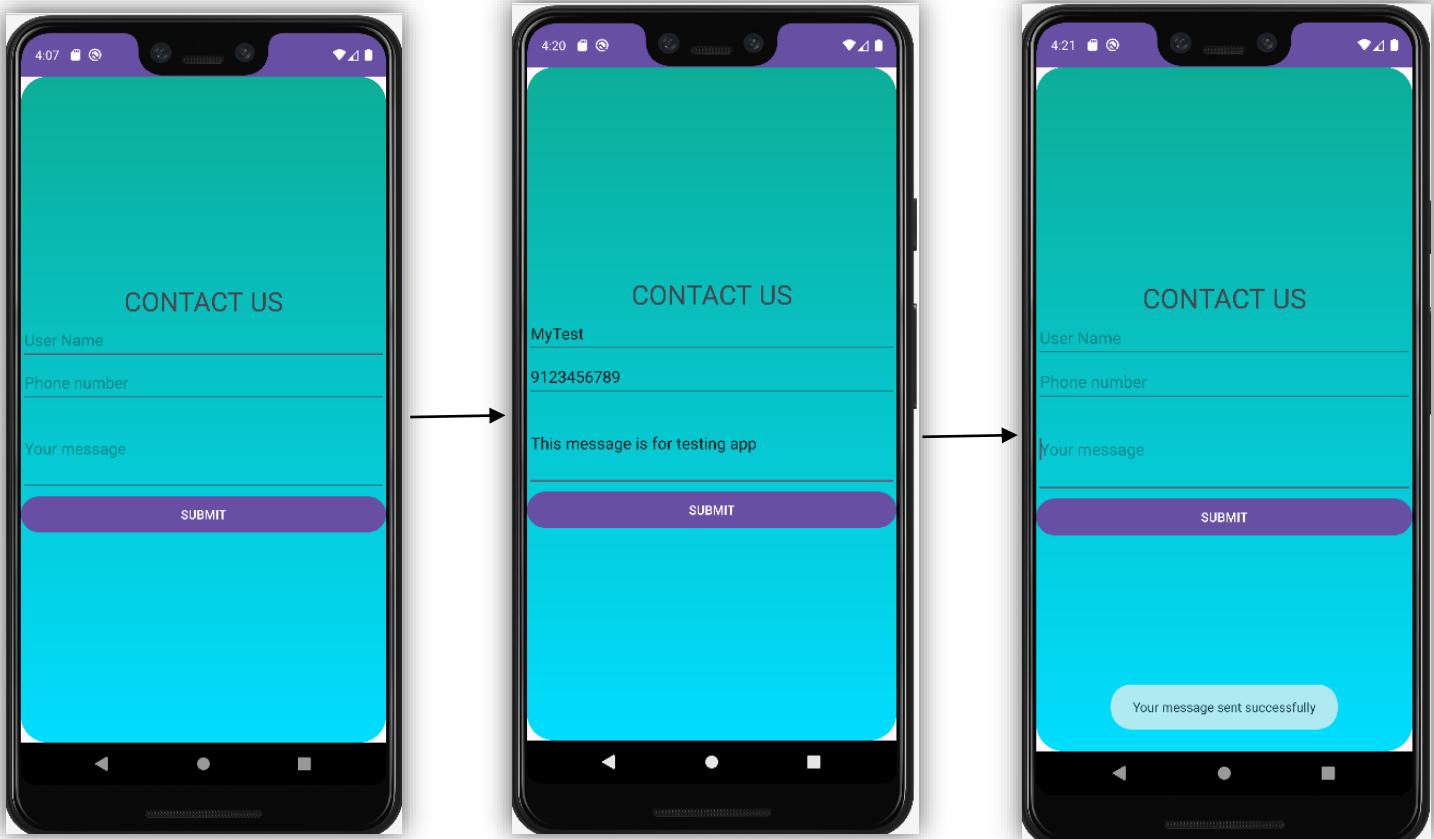
```

```

1 package com.mad.madprog9;
2 import ...
3 public class MainActivity extends AppCompatActivity {
4     private EditText usernameEdt, phonenoEdt, messageEdt;
5     private DBHandler dbHandler;
6     @Override
7     protected void onCreate(Bundle savedInstanceState) {
8         super.onCreate(savedInstanceState);
9         setContentView(R.layout.activity_main);
10
11         usernameEdt = findViewById(R.id.Uname);
12         phonenoEdt = findViewById(R.id.Phno);
13         messageEdt = findViewById(R.id.Msg);
14         Button submitBtn = findViewById(R.id.sub);
15
16         dbHandler = new DBHandler( context: MainActivity.this);
17
18         submitBtn.setOnClickListener(new View.OnClickListener() {
19             @Override
20             public void onClick(View v) {
21                 String userName = usernameEdt.getText().toString();
22                 String phoneno = phonenoEdt.getText().toString();
23                 String message = messageEdt.getText().toString();
24
25                 if (userName.isEmpty() && phoneno.isEmpty() && message.isEmpty()) {
26                     Toast.makeText( context: MainActivity.this, text: "Please enter all the data..", Toast.LENGTH_SHORT).show();
27                     return;
28                 }
29
30                 dbHandler.addNewCourse(userName, phoneno, message);
31                 Toast.makeText( context: MainActivity.this, text: "Your message sent successfully", Toast.LENGTH_SHORT).show();
32                 usernameEdt.setText("");
33                 phonenoEdt.setText("");
34                 messageEdt.setText("");
35             }
36         });
37     }
38 }

```

10. Press **Shift F10** to run the application Or, Press the run button on top of the interface

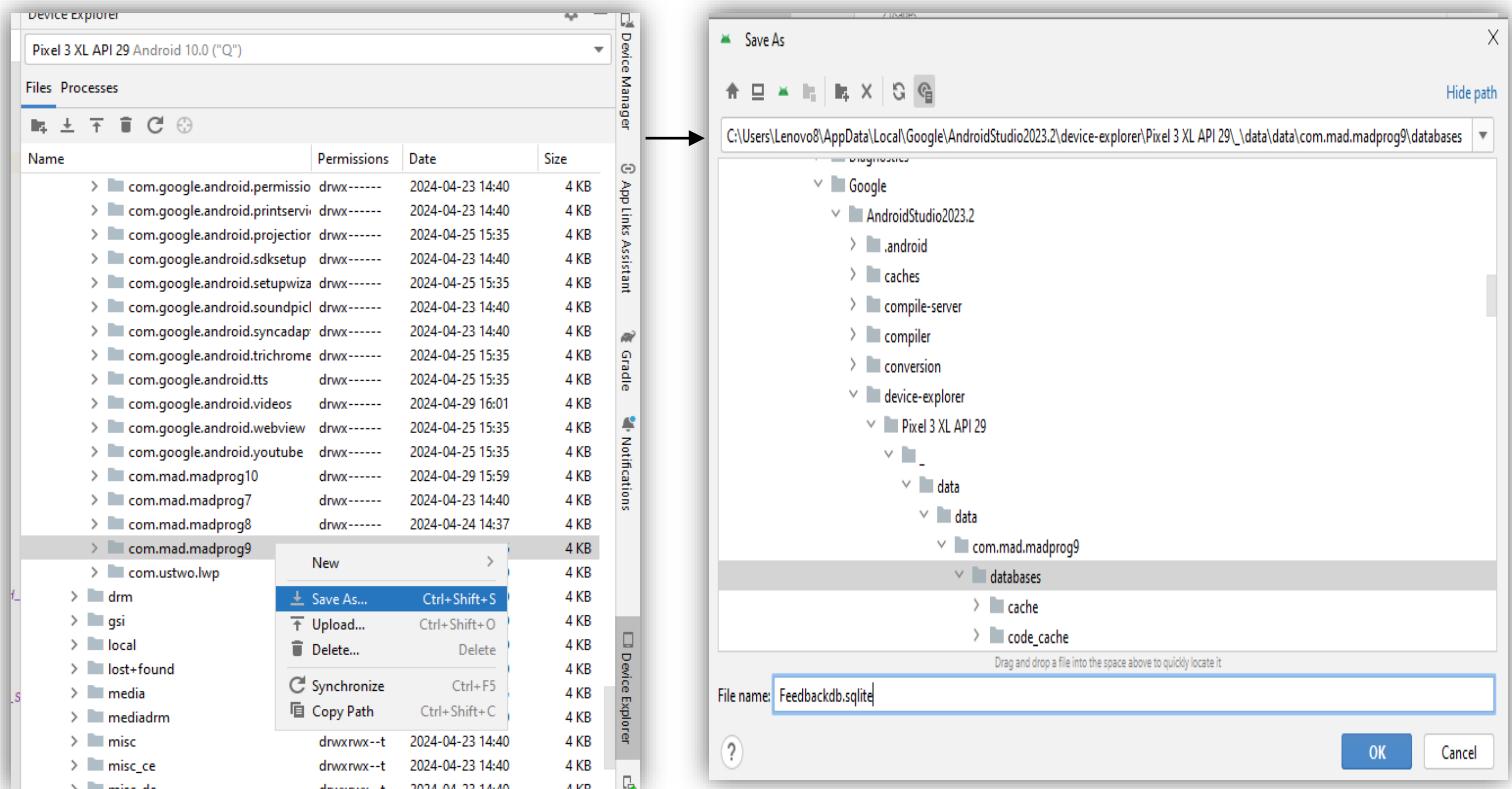


11. Save the database in android studio

Go to device explorer -> data-> data-> Packagename(com.mad.madprog9)-> databases -

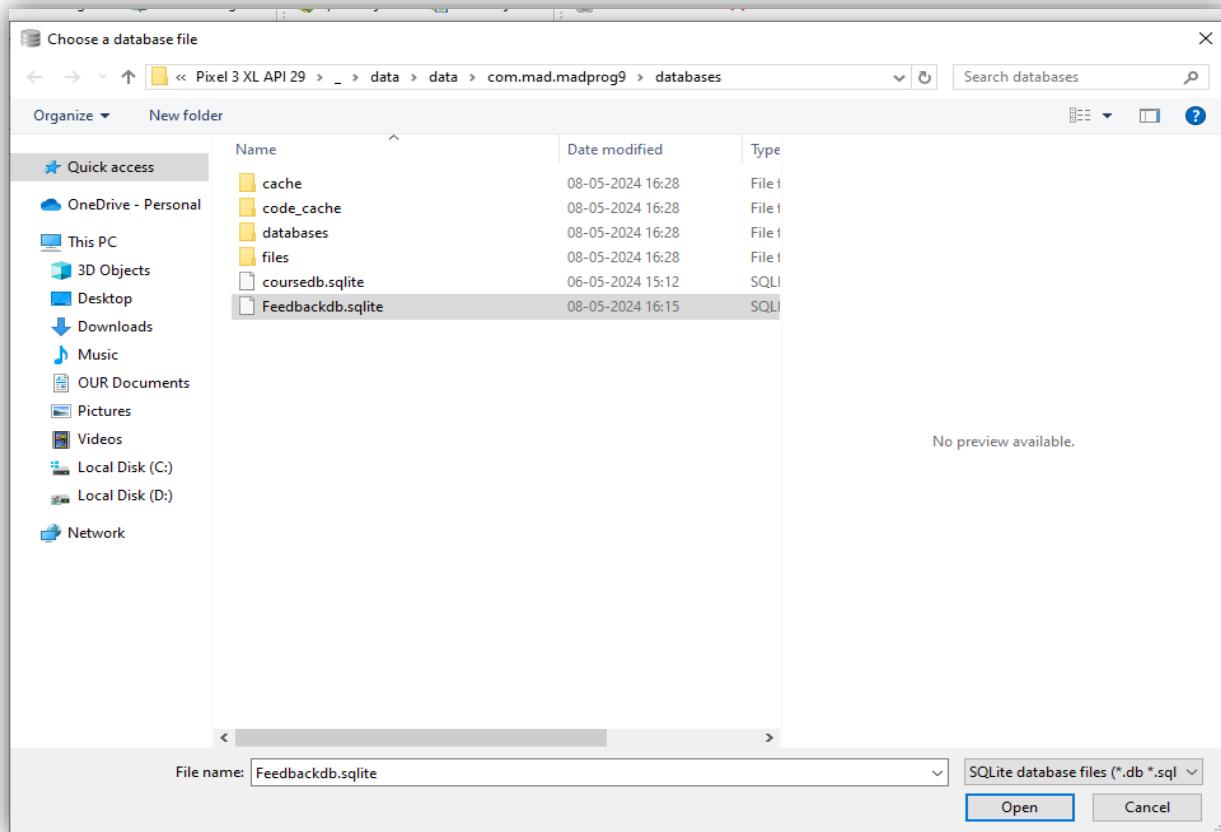
>Feedbackdb (Right click) ->save As -> Feedbackdb.sqlite-> save

Copy the filepath location for further reference

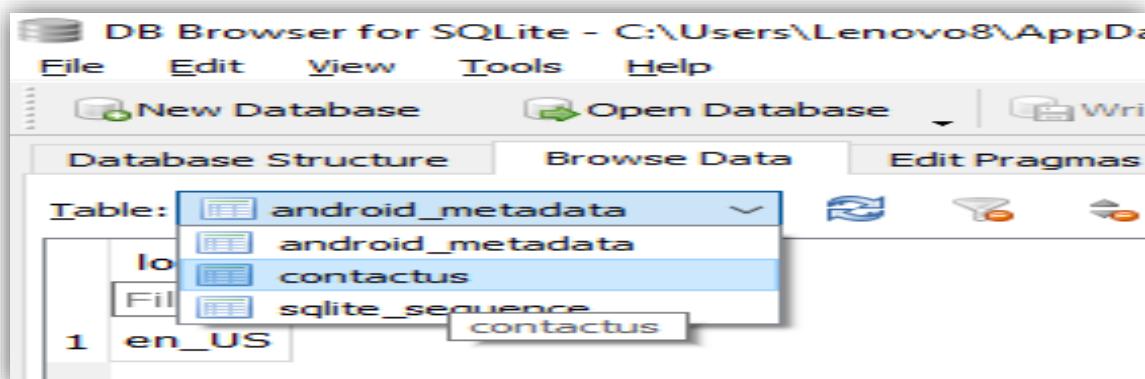


12. Go to DBbrowser for sqlite software then select open database

- Search for the filepath you have saved your database
- Select **Feedbackdb.sqlite**



13. Select the table name(contactus**) and check for stored data that have been entered.**



14. Now the database will be displayed as shown below

Table: contactus			
	id	name	phone_number
1	1	MyTest	9123456789

10. Create an application to send SMS and receive SMS.

1. Create a **new project** and choose **Empty Views Activity** then click **Next**.

Change the layout from constraint Layout to Relative Layout in **activity_main.xml** file.

<androidx.constraintlayout.widget.ConstraintLayout..

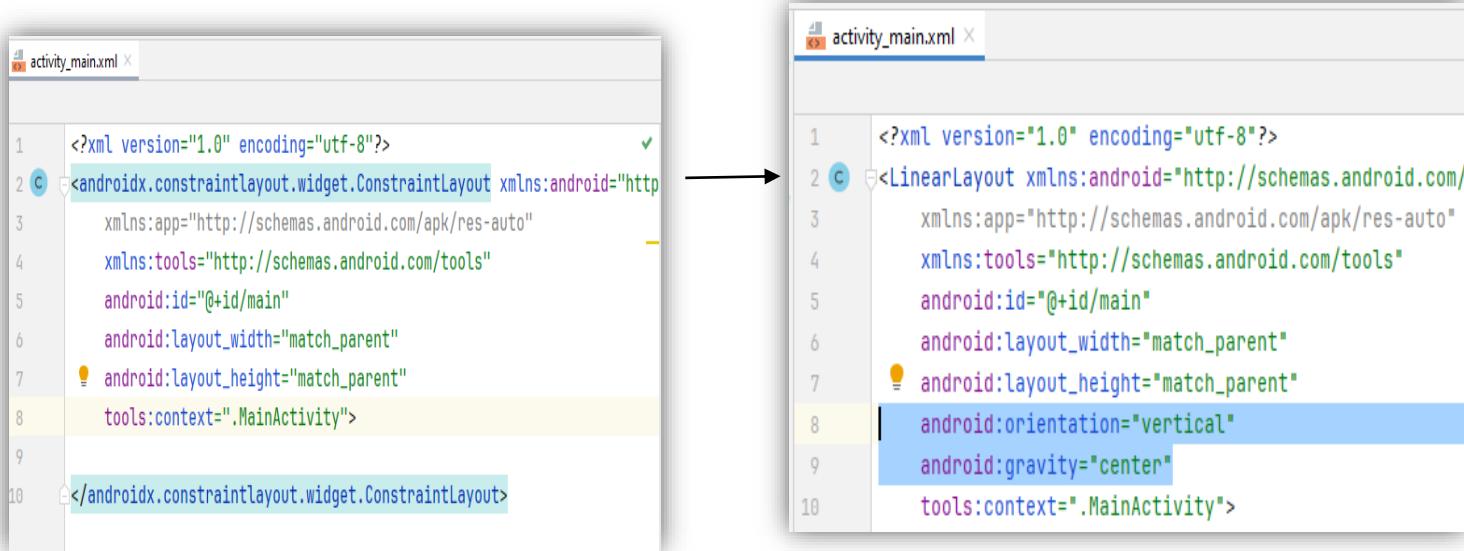
To

<LinearLayout..

update the following code in the linearlayout tag.

android:orientation="vertical"

android:gravity="center"



2. Add two Edit view, then edit the content in **activity_main.xml** code file as follows:

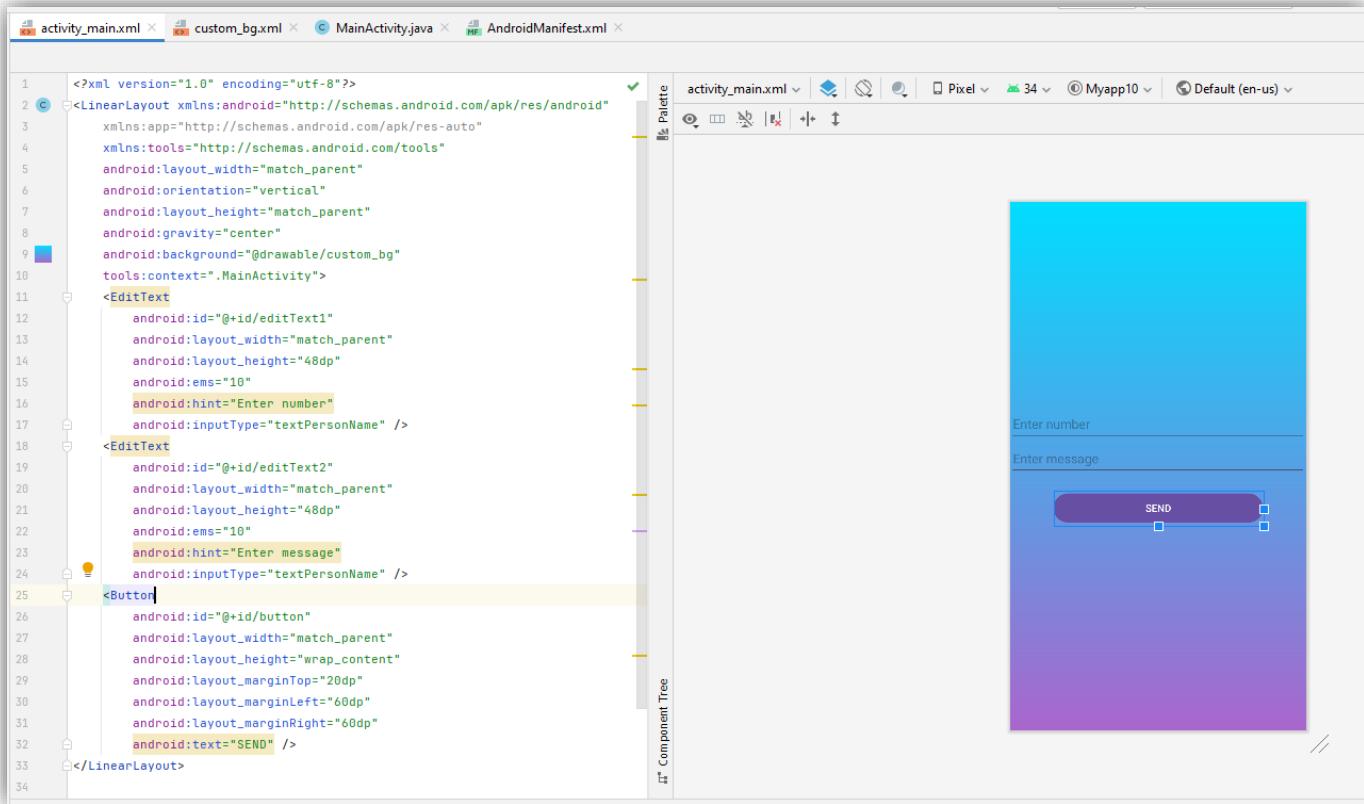
3. Add edit views for phoneno and message

```
<EditText
    android:id="@+id/editText1"
    android:layout_width="match_parent"
    android:layout_height="48dp"
    android:hint="Enter number"/>

<EditText
    android:id="@+id/editText2"
    android:layout_width="match_parent"
    android:layout_height="48dp"
    android:hint="Enter message"/>
```

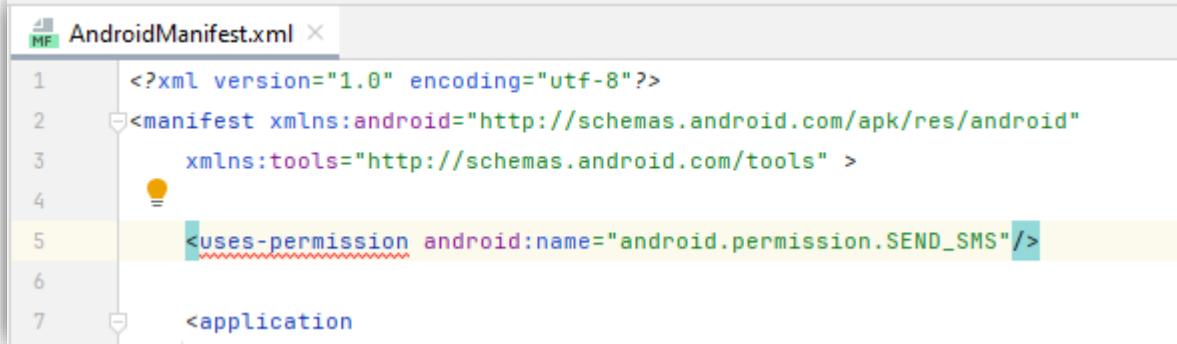
4. Add a button for submit

```
<Button
    android:id="@+id/button"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="SEND" />
```



5. In **AndroidManifest.xml** add the permission to send SMS. It will permit an android application to send SMS.

<uses-permission android:name=" android.permission.SEND_SMS " />



6. Go to **mainActivity.java** for sending sms

- Import necessary libraries
- ```

import android.telephony.SmsManager;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

```

Update the following code for creating method to send sms  
**public class MainActivity extends AppCompatActivity {**

```

 EditText phonenumbers,message;
 Button send;
 @Override
 protected void onCreate(Bundle savedInstanceState) {
 super.onCreate(savedInstanceState);
 setContentView(R.layout.activity_main);

```

```

send=findViewById(R.id.button);
phonenumber=findViewById(R.id.editText1);
message=findViewById(R.id.editText2);
send.setOnClickListener(new View.OnClickListener() {

 public void onClick(View view) {
 String number=phonenumber.getText().toString();
 String msg=message.getText().toString();
 try {
 SmsManager smsManager=SmsManager.getDefault();
 smsManager.sendTextMessage(number,null,msg,null,null);
 Toast.makeText(getApplicationContext(),"Message
Sent",Toast.LENGTH_LONG).show();
 }catch (Exception e)
 {
 Toast.makeText(getApplicationContext(),"Some fields is
Empty",Toast.LENGTH_LONG).show();
 }
 }
});
```

```

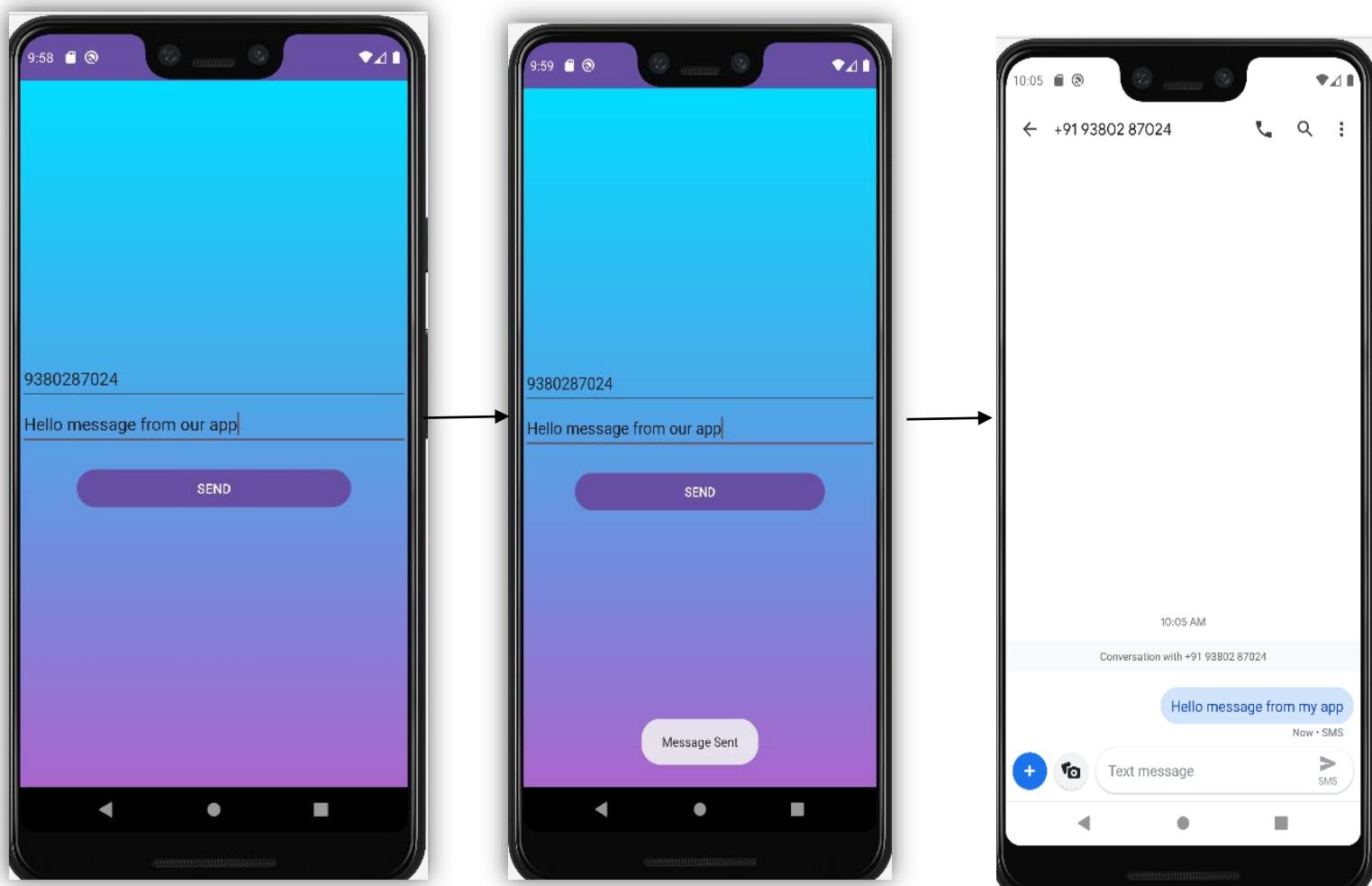
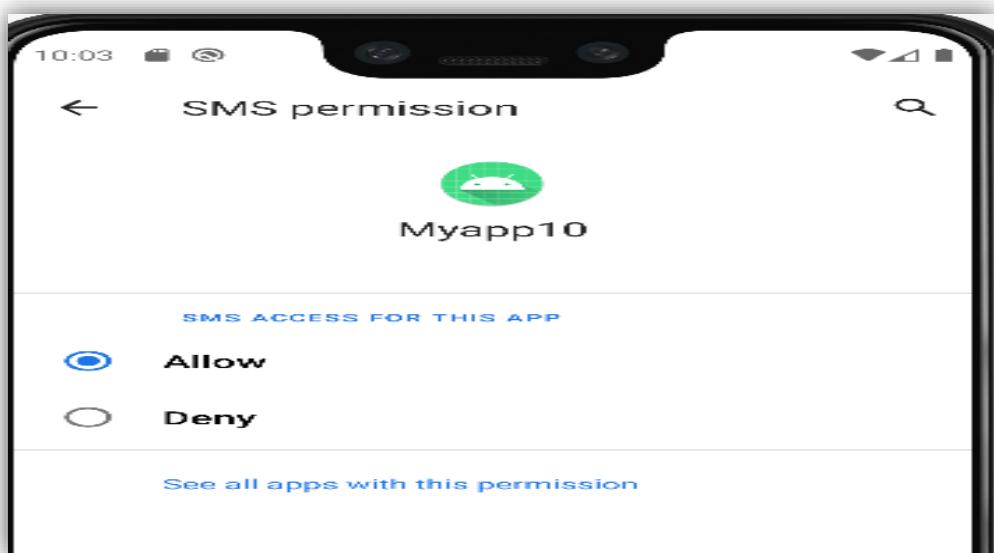
1 package com.mad.myapp10;
2 import androidx.appcompat.app.AppCompatActivity;
3 import android.os.Bundle;
4 import android.telephony.SmsManager;
5 import android.view.View;
6 import android.widget.Button;
7 import android.widget.EditText;
8 import android.widget.Toast;
9
10 public class MainActivity extends AppCompatActivity {
11 EditText phonenumber,message;
12 Button send;
13 @Override
14 protected void onCreate(Bundle savedInstanceState) {
15 super.onCreate(savedInstanceState);
16 setContentView(R.layout.activity_main);
17 send=findViewById(R.id.button);
18 phonenumber=findViewById(R.id.editText1);
19 message=findViewById(R.id.editText2);
20 send.setOnClickListener(new View.OnClickListener() {

21 public void onClick(View view) {
22 String number=phonenumber.getText().toString();
23 String msg=message.getText().toString();
24 try {
25 SmsManager smsManager=SmsManager.getDefault();
26 smsManager.sendTextMessage(number, scAddress: null, msg, sentIntent: null, deliveryIntent: null);
27 Toast.makeText(getApplicationContext(), text: "Message Sent",Toast.LENGTH_LONG).show();
28 }catch (Exception e)
29 {
30 Toast.makeText(getApplicationContext(), text: "Some fields is Empty",Toast.LENGTH_LONG).show();
31 }
32 }
33 });
34 }
35 }
36
37 }
```

**7. Connect USB to the physical mobile -> Go to developer option -> enable USB debugger , install via USB and USB debugger(security settings)**

8. Press **Shift F10** to run the application Or, Press the run button on top of the interface

- **Allow permission for installing app In mobile.**
- **Got o app info-> permission- >allow sms**



## 11. Create an application to send an E-mail.

1. Create a **new project** and choose **Empty Views Activity** then click **Next**.

Change the layout from constraint Layout to Relative Layout in **activity\_main.xml** file.

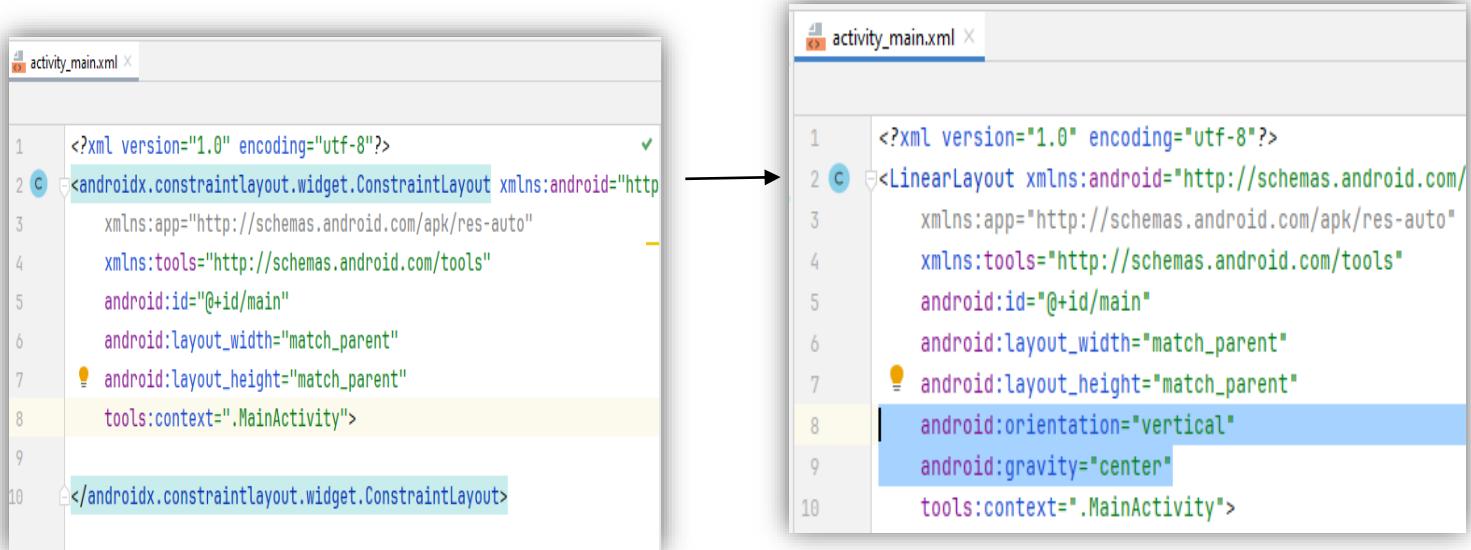
<**androidx.constraintlayout.widget.ConstraintLayout..**

To

<**LinearLayout..**

update the following code in the linearlayout tag.

```
 android:orientation="vertical"
 android:gravity="center"
```



2. Add three Edit view, then edit the content in **activity\_main.xml** code file as follows:

```
<EditText
 android:id="@+id/editTextTo"
 android:layout_width="match_parent"
 android:layout_height="48dp"
 android:hint="Send To"/>

<EditText
 android:id="@+id/editTextSubject"
 android:layout_width="match_parent"
 android:layout_height="48dp"
 android:hint="Subject"/>

<EditText
 android:id="@+id/editTextMessage"
 android:layout_width="match_parent"
 android:layout_height="48dp"
 android:hint="Message"/>

<Button
 android:id="@+id/buttonSend"
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
 android:text="Send email"/>
```

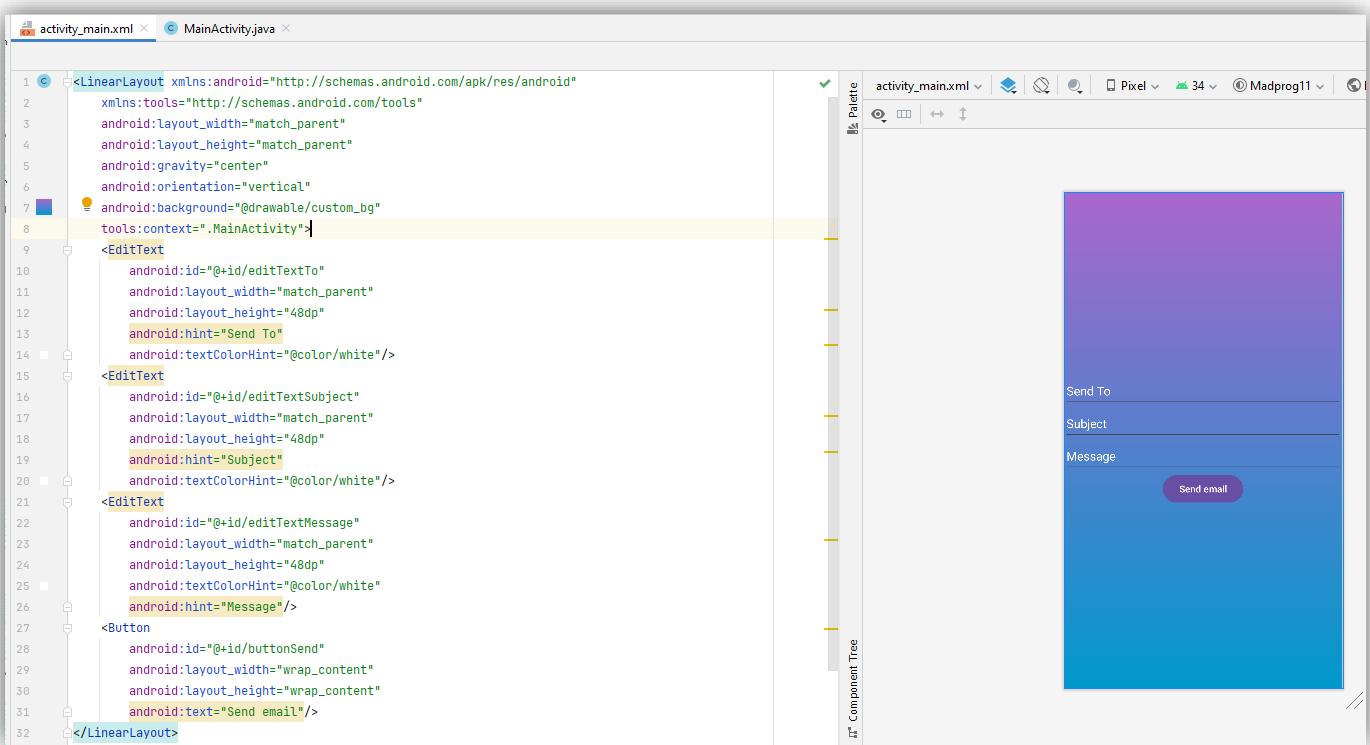
3. Add a button for submit

```
<Button
 android:id="@+id/buttonSend"
```

```

 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
 android:text="Send email"/>

```



#### 4. Go to **mainActivity.java** for sending email

- Import necessary libraries

```

import android.annotation.SuppressLint;
import android.content.Intent;
import android.widget.Button;
import android.widget.EditText;

```

- Update the following code for creating method to send email

```

public class MainActivity extends AppCompatActivity {
 EditText editTextTo, editTextSubject, editTextMessage;
 Button buttonSend;
 @Override
 protected void onCreate(Bundle savedInstanceState) {
 super.onCreate(savedInstanceState);
 setContentView(R.layout.activity_main);
 editTextTo = findViewById(R.id.editTextTo);
 editTextSubject = findViewById(R.id.editTextSubject);
 editTextMessage = findViewById(R.id.editTextMessage);
 buttonSend = findViewById(R.id.buttonSend);
 buttonSend.setOnClickListener(v -> sendEmail());
 }
 @SuppressLint("QueryPermissionsNeeded")
 private void sendEmail() {
 String to = editTextTo.getText().toString().trim();
 String subject = editTextSubject.getText().toString().trim();
 String message = editTextMessage.getText().toString().trim();
 }
}

```

```

Intent intent = new Intent(Intent.ACTION_SEND);
intent.setType("text/plain");
intent.putExtra(Intent.EXTRA_EMAIL, new String[]{to});
intent.putExtra(Intent.EXTRA_SUBJECT, subject);
intent.putExtra(Intent.EXTRA_TEXT, message);
if (intent.resolveActivity(getApplicationContext()) != null)
 startActivityForResult(Intent.createChooser(intent, "Choose an email client"));
}
}
}

```

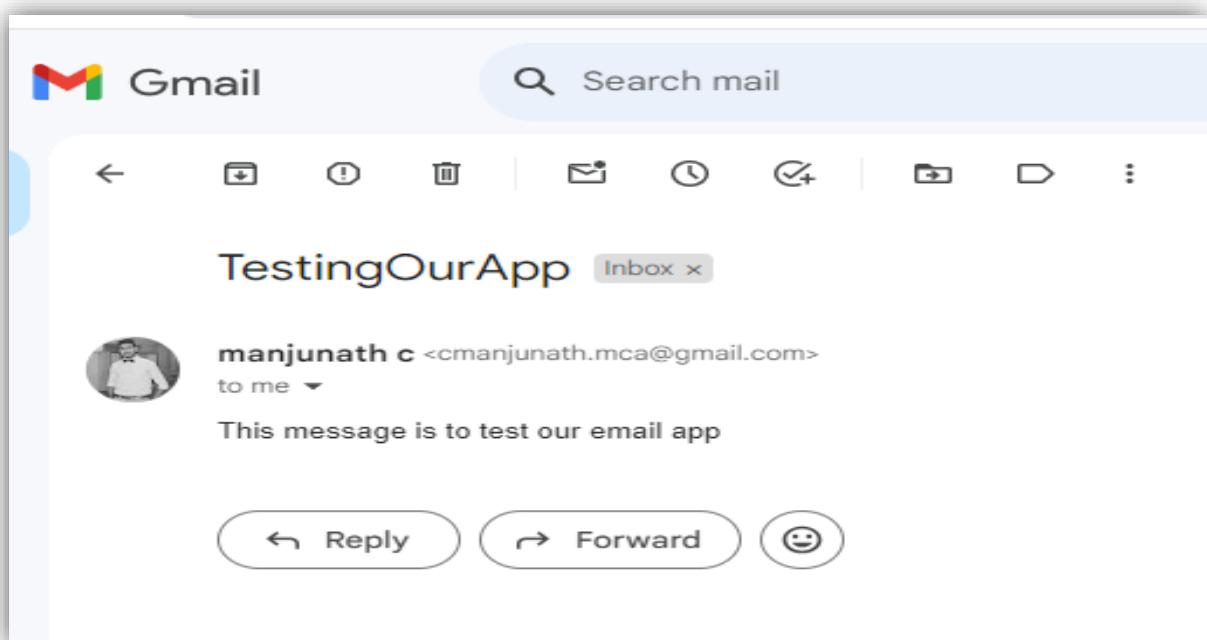
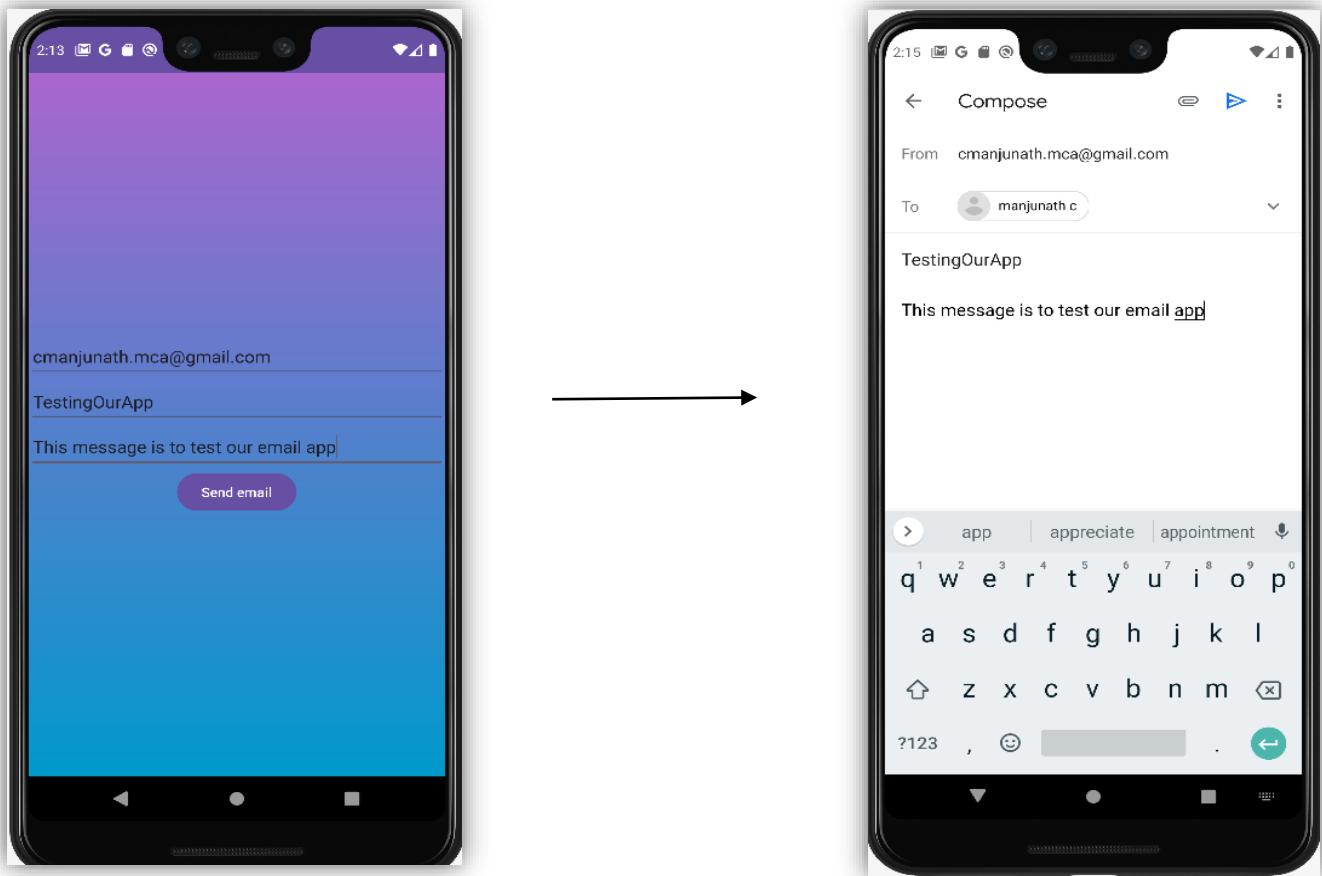
MainActivity.java

```

1 package com.mad.madprog11;
2
3 import android.os.Bundle;
4 import android.annotation.SuppressLint;
5 import android.content.Intent;
6 import android.widget.Button;
7 import android.widget.EditText;
8 import androidx.appcompat.app.AppCompatActivity;
9
10 public class MainActivity extends AppCompatActivity {
11 EditText editTextTo, editTextSubject, editTextMessage;
12 Button buttonSend;
13
14 @Override
15 protected void onCreate(Bundle savedInstanceState) {
16 super.onCreate(savedInstanceState);
17 setContentView(R.layout.activity_main);
18 editTextTo = findViewById(R.id.editTextTo);
19 editTextSubject = findViewById(R.id.editTextSubject);
20 editTextMessage = findViewById(R.id.editTextMessage);
21 buttonSend = findViewById(R.id.buttonSend);
22 buttonSend.setOnClickListener(v -> sendEmail());
23
24 }
25 @SuppressLint("QueryPermissionsNeeded")
26 private void sendEmail() {
27 String to = editTextTo.getText().toString().trim();
28 String subject = editTextSubject.getText().toString().trim();
29 String message = editTextMessage.getText().toString().trim();
30 Intent intent = new Intent(Intent.ACTION_SEND);
31 intent.setType("text/plain");
32 intent.putExtra(Intent.EXTRA_EMAIL, new String[]{to});
33 intent.putExtra(Intent.EXTRA_SUBJECT, subject);
34 intent.putExtra(Intent.EXTRA_TEXT, message);
35 if (intent.resolveActivity(getApplicationContext()) != null) startActivityForResult(Intent.createChooser(intent, "Choose an email client"));
36 }
}

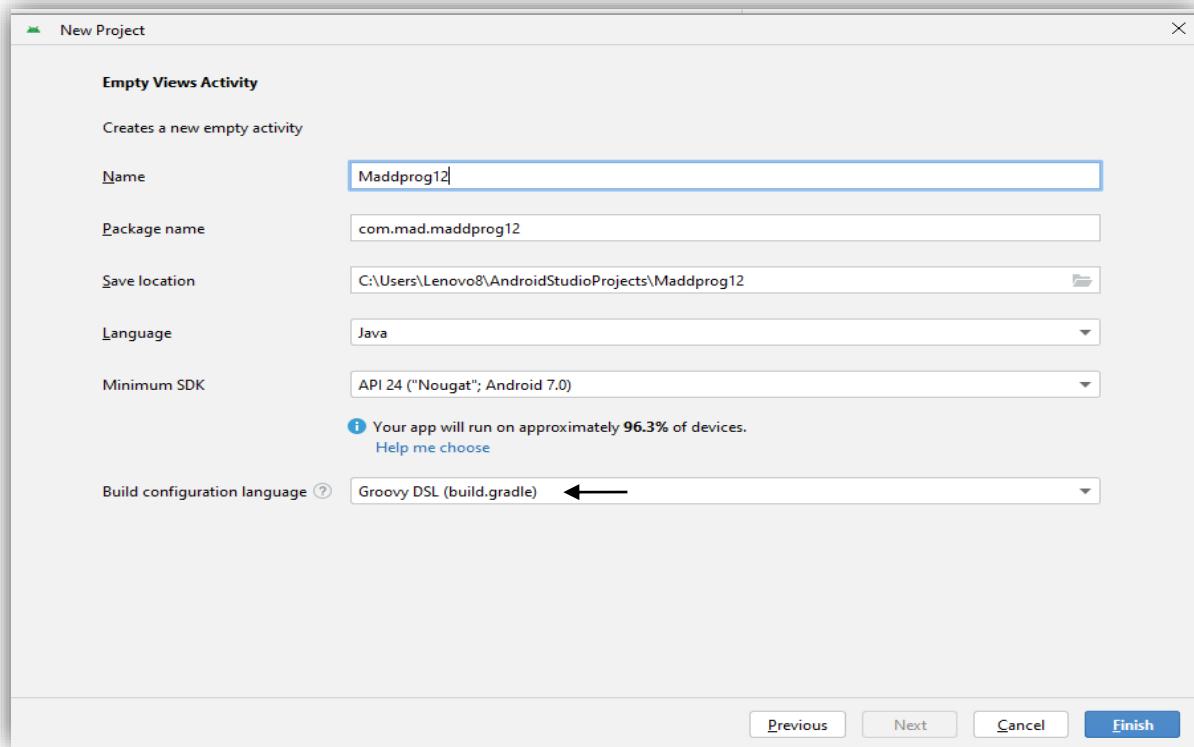
```

5. Press **Shift F10** to run the application Or, Press the run button on top of the interface



## **12. Display Map based on the Current/given location.**

1. Create a new project and choose ***Empty Views Activity*** then select Build configuration language as ***Groovy DSL (build.gradle)*** click finish.



2. Go to app->***Gradle scripts -> build.gradle(Module:app)*** then add dependencies
  - ***implementation 'com.google.android.gms:play-services-maps:18.1.0'***
  - Then click on sync Now

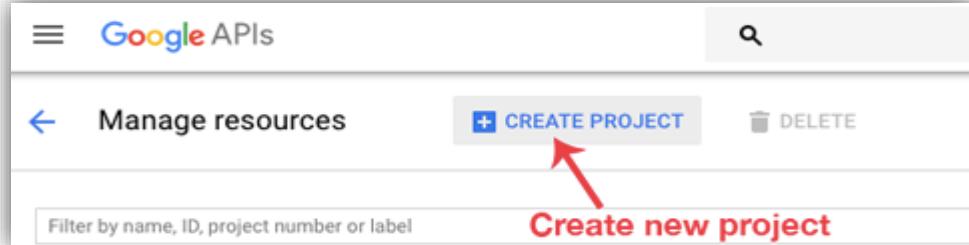
```
build.gradle (:app) < Sync Now>
Gradle files have changed since last project sync. A project sync may be necessary for the IDE to work properly.

16 testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
17 }
18
19 buildTypes {
20 release {
21 minifyEnabled false
22 proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
23 }
24 }
25 compileOptions {
26 sourceCompatibility JavaVersion.VERSION_1_8
27 targetCompatibility JavaVersion.VERSION_1_8
28 }
29 }
30
31 dependencies {
32
33 implementation libs.appcompat
34 implementation libs.material
35 implementation libs.activity
36 implementation libs.constraintlayout
37 testImplementation libs.junit
38 androidTestImplementation libs.ext.junit
39 androidTestImplementation libs.espresso.core
40 implementation 'com.google.android.gms:play-services-maps:18.1.0'
41 }
```

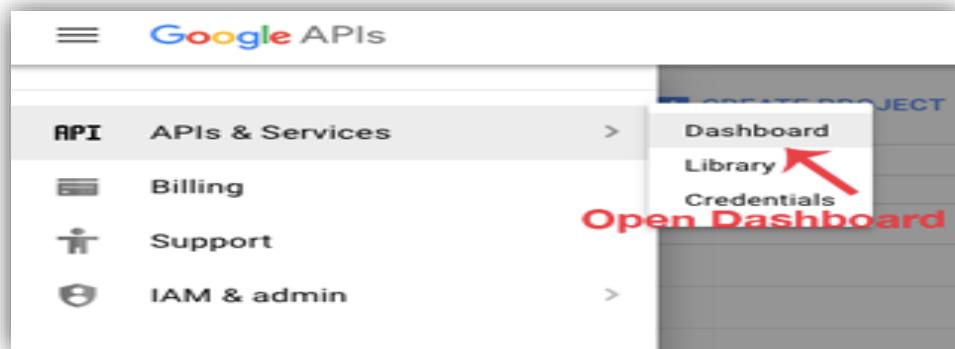
3. Get an API key from google cloud

←-----NOT TO BE WRITTEN IN RECORD-----→

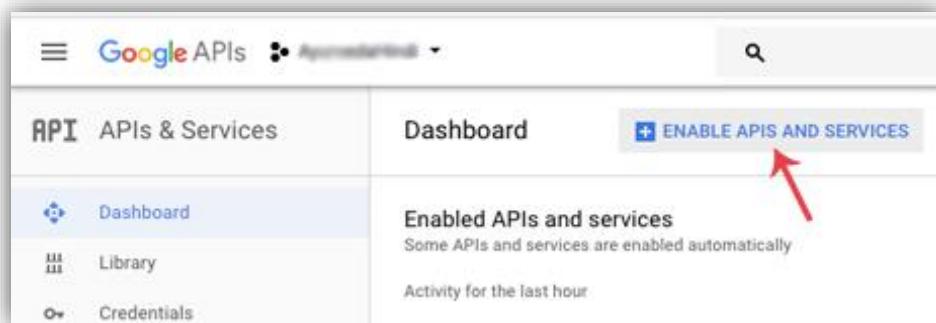
- Open Google developer console and signin with your gmail account: <https://console.developers.google.com/project>
- Now create new project. You can create new project by clicking on the **Create Project** button and give name to your project.



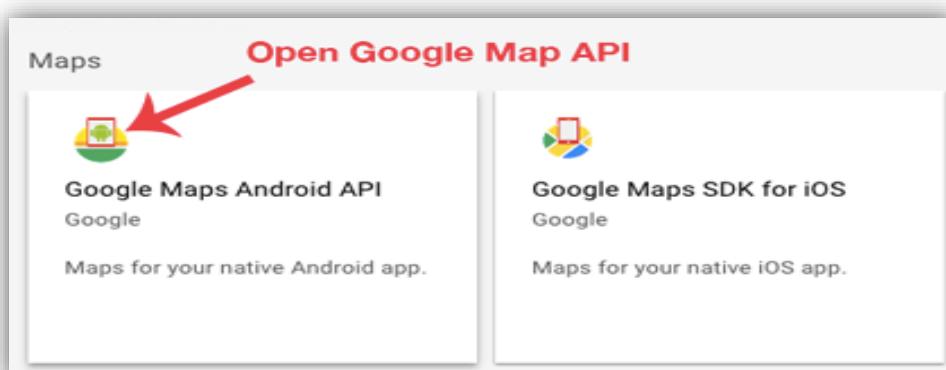
- Now click on APIs & Services and open Dashboard from it.



- In this open **Enable APIS AND SERVICES**.



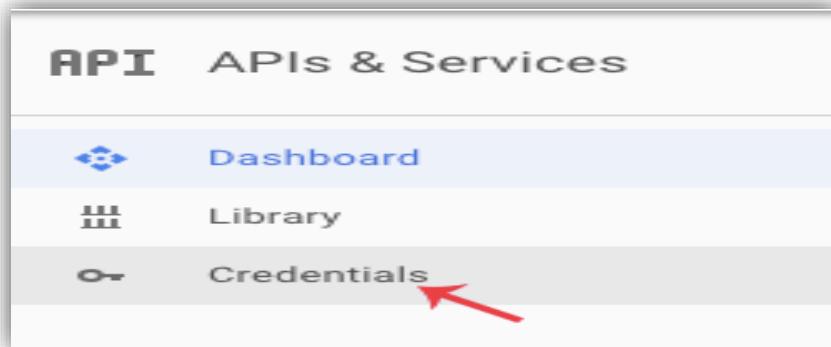
- Now open Google Map Android API



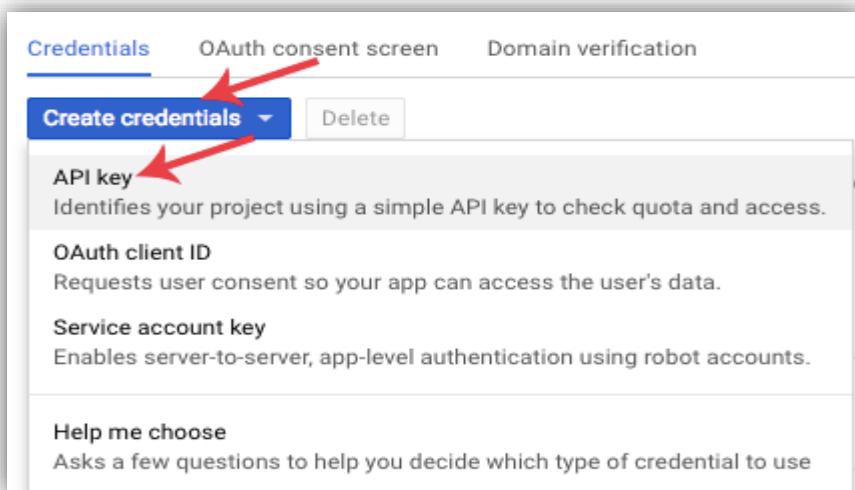
- Now enable the Google Maps Android API.



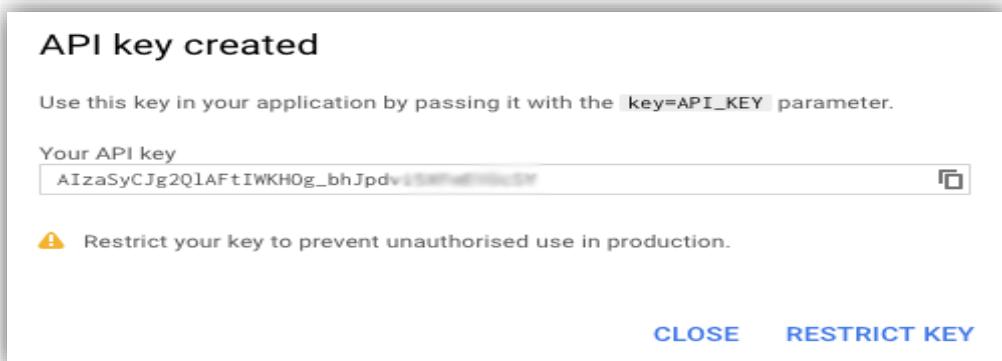
- Now go to **Credentials**



- Here click on **Create credentials** and choose **API key**



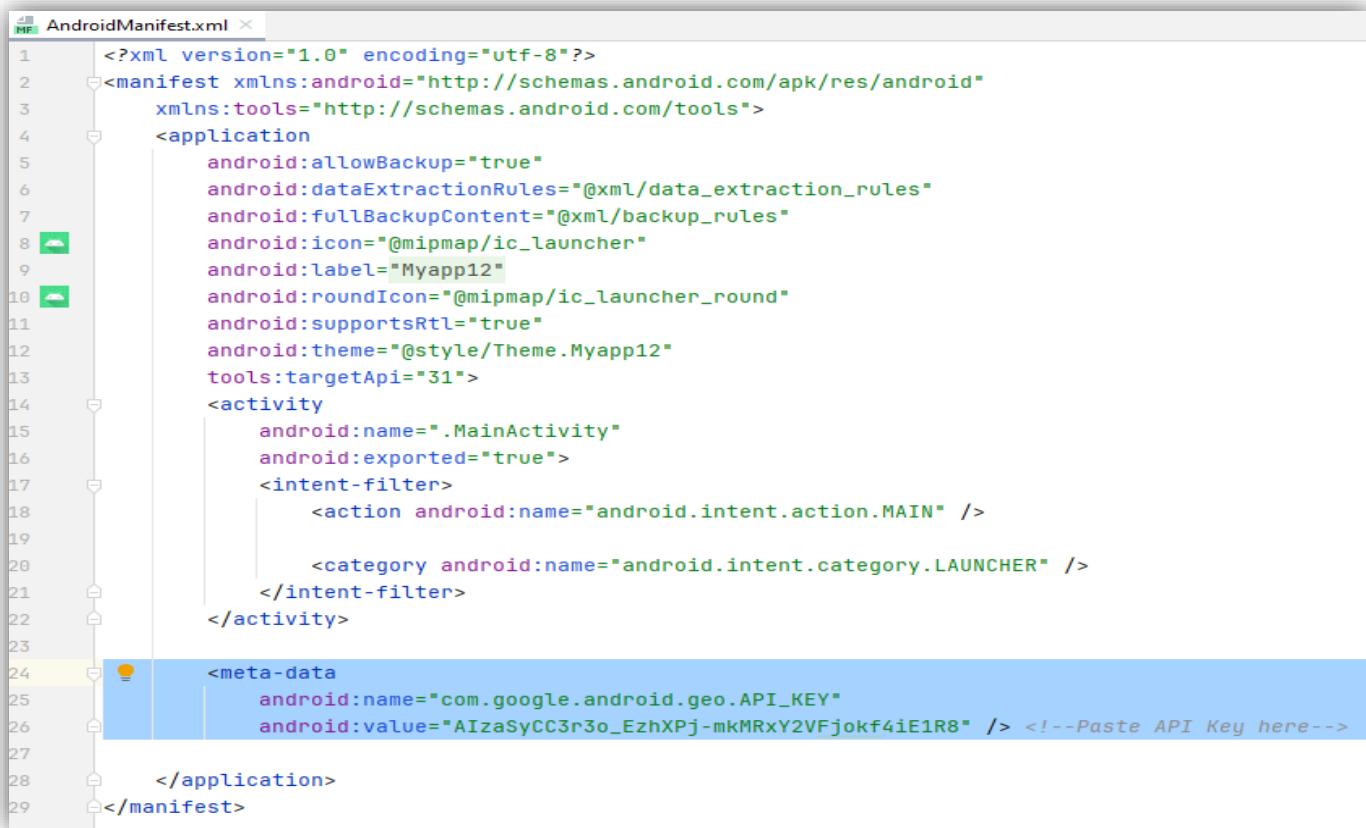
Now API your API key will be generated. Copy it and save it somewhere as we will need it when implementing Google Map in our Android project.



**<-----JUST FOR REERENCE----->**

4. Add API key to the ***AndroidManifest.xml*** file

```
<meta-data
 android:name="com.google.android.geo.API_KEY"
 android:value="AIzaSyCC3r3o_EzhXPj-mkMRxY2VFjokf4iE1R8" />
```

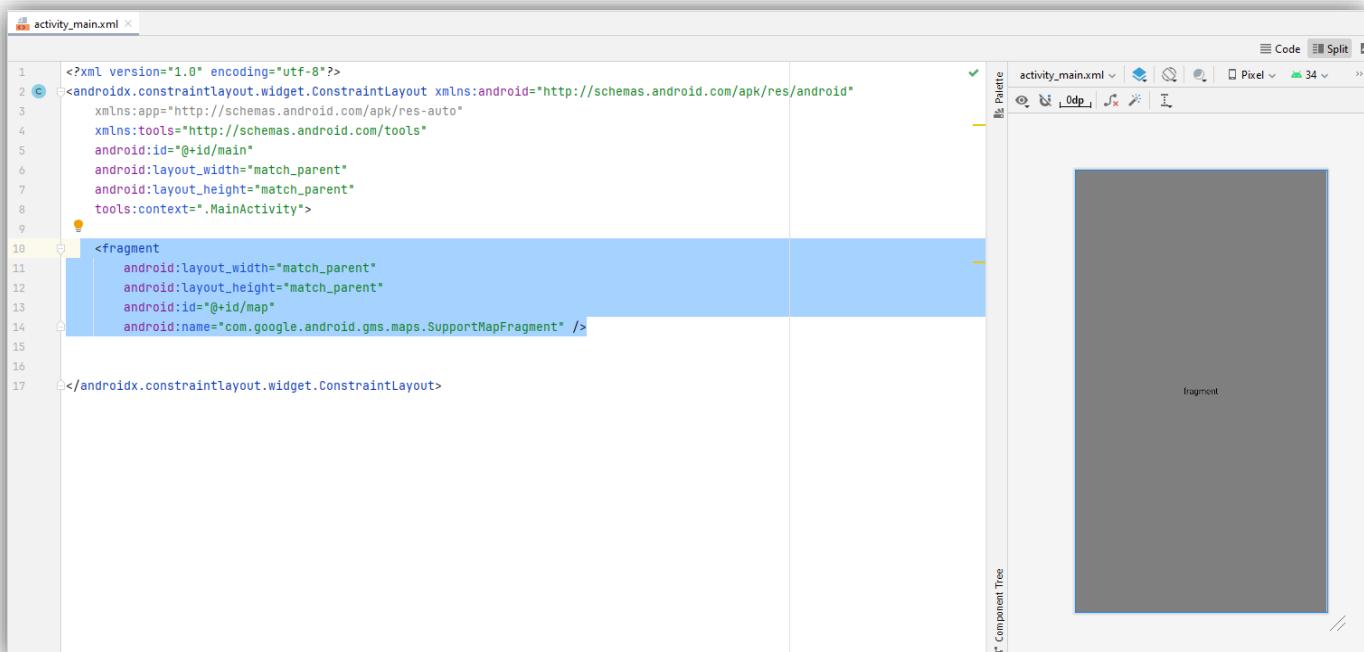


The screenshot shows the `AndroidManifest.xml` file in an IDE. The code is as follows:

```
<?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="Myapp12"
 android:roundIcon="@mipmap/ic_launcher_round"
 android:supportsRtl="true"
 android:theme="@style/Theme.Myapp12"
 tools:targetApi="31">
 <activity
 android:name=".MainActivity"
 android:exported="true">
 <intent-filter>
 <action android:name="android.intent.action.MAIN" />
 <category android:name="android.intent.category.LAUNCHER" />
 </intent-filter>
 </activity>
 <meta-data
 android:name="com.google.android.geo.API_KEY"
 android:value="AIzaSyCC3r3o_EzhXPj-mkMRxY2VFjokf4iE1R8" /> <!-- Paste API Key here -->
 </application>
</manifest>
```

5. Now got to ***activity\_main.xml*** file and create a fragment

```
<fragment
 android:layout_width="match_parent"
 android:layout_height="match_parent"
 android:id="@+id/map"
 android:name="com.google.android.gms.maps.SupportMapFragment" />
```



The screenshot shows the `activity_main.xml` file in an IDE. The code is as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
 xmlns:app="http://schemas.android.com/apk/res-auto"
 xmlns:tools="http://schemas.android.com/tools"
 android:id="@+id/main"
 android:layout_width="match_parent"
 android:layout_height="match_parent"
 tools:context=".MainActivity">
 <fragment
 android:layout_width="match_parent"
 android:layout_height="match_parent"
 android:id="@+id/map"
 android:name="com.google.android.gms.maps.SupportMapFragment" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

6. Go to **MainActivity.java** and update the code

- Import necessary libraries

```
import androidx.annotation.NonNull;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
```

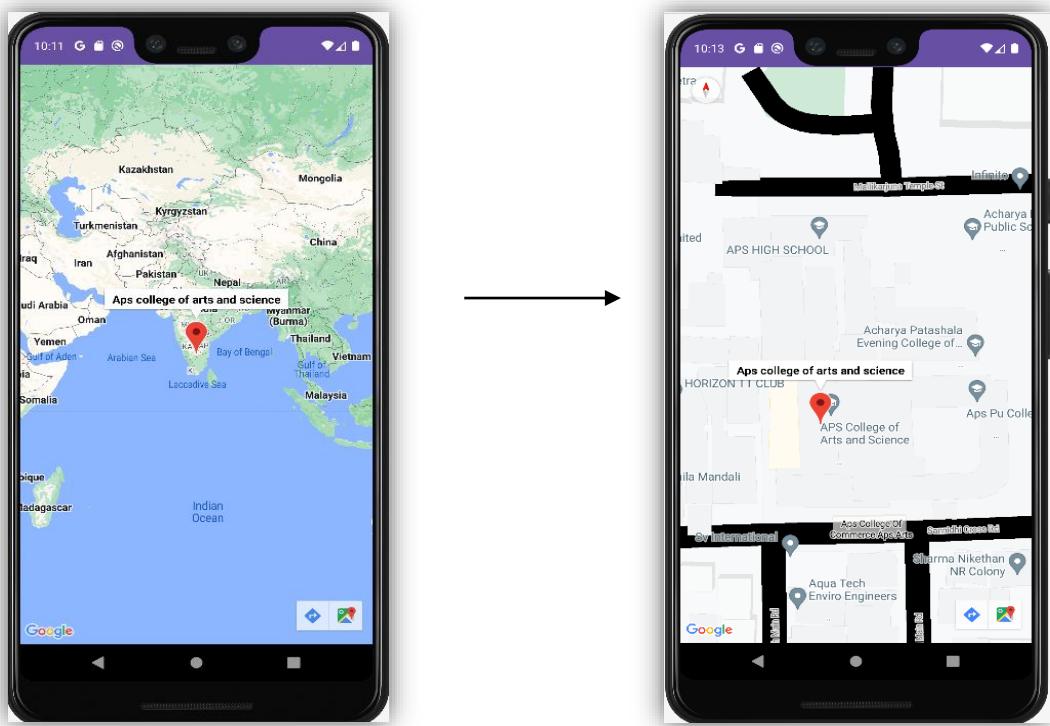
- Then update the method for current location

```
public class MainActivity extends AppCompatActivity implements OnMapReadyCallback {
 private GoogleMap myMap;
 @Override
 protected void onCreate(Bundle savedInstanceState) {
 super.onCreate(savedInstanceState);
 setContentView(R.layout.activity_main);
 SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager().findFragmentById(R.id.map);
 mapFragment.getMapAsync(MainActivity.this);
 }
}
```

- Get the current location latitude and longitude

```
@Override
public void onMapReady(@NonNull GoogleMap googleMap) {
 myMap = googleMap;
 LatLng aps = new LatLng(12.9398, 77.5676); //Paste current location latitude and
longitude
 myMap.addMarker(new MarkerOptions().position(aps).title("Aps college of arts and
science"));
 myMap.moveCamera(CameraUpdateFactory.newLatLng(aps));
}
```

7. Press **Shift F10** to run the application Or, Press the run button on top of the interface



**13. Create a sample application with login module(check user name and password) On successful login change TextView “Login Successful”. On login fail alert using Toast “login fail”**

1. Create a new project and choose Empty Views Activity then click Next.
2. Change the layout from constraint Layout to Relative Layout in **activity\_main.xml** file.

**<androidx.constraintlayout.widget.ConstraintLayout..**

To

**<LinearLayout..**

3. update the following code in the linearlayout tag.

**android:gravity="center"**

**android:layout\_centerInParent="true"**

**android:orientation="vertical"**

**android:padding="30dp"**

```
activity_main.xml X
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3 xmlns:app="http://schemas.android.com/apk/res-auto"
4 xmlns:tools="http://schemas.android.com/tools"
5 android:id="@+id/main"
6 android:layout_width="match_parent"
7 android:layout_height="match_parent"
8 tools:context=".MainActivity">
9
10 </androidx.constraintlayout.widget.ConstraintLayout>
```

```
activity_main.xml X
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3 xmlns:app="http://schemas.android.com/apk/res-auto"
4 xmlns:tools="http://schemas.android.com/tools"
5 android:id="@+id/main"
6 android:layout_width="match_parent"
7 android:layout_height="match_parent"
8 android:gravity="center"
9 android:layout_centerInParent="true"
10 android:orientation="vertical"
11 android:padding="30dp"
12 tools:context=".MainActivity">
13
14 </LinearLayout>
```

4. Go to palette option-> Common-> **TextView**, Then Drag drop the Button View into design window
5. Edit the text View content in **activity\_main.xml** code file as follows:

```
<TextView
 android:id="@+id/textView3"
 android:layout_width="match_parent"
 android:layout_height="48sp"
 android:text="LOGIN PAGE"
 android:textAlignment="center"
 android:textSize="30sp"/>
```

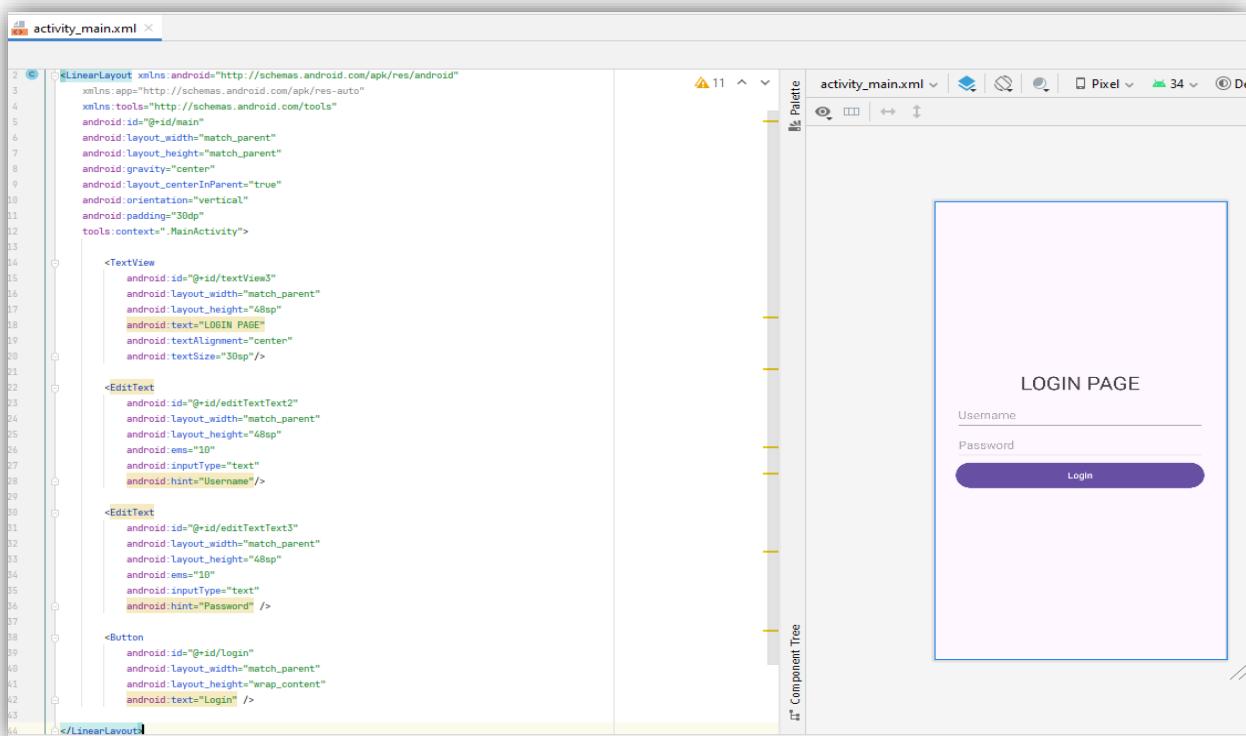
6. Add two edit text one after the other for username and password

```
<EditText
 android:id="@+id/editTextText2"
 android:layout_width="match_parent"
 android:layout_height="48sp"
 android:ems="10"
 android:inputType="text"
 android:hint="Username"/>
```

```
<EditText
 android:id="@+id/editTextText2"
 android:layout_width="match_parent"
 android:layout_height="48sp"
 android:ems="10"
 android:inputType="text"
 android:hint="Password"/>
```

9. Add a **ButtonView** for login and update the code.

```
<Button
 android:id="@+id/login"
 android:layout_width="match_parent"
 android:layout_height="wrap_content"
 android:text="Login" />
```



10.

Go to **mainActivity.java** for sending email

- Import necessary libraries

```
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
```

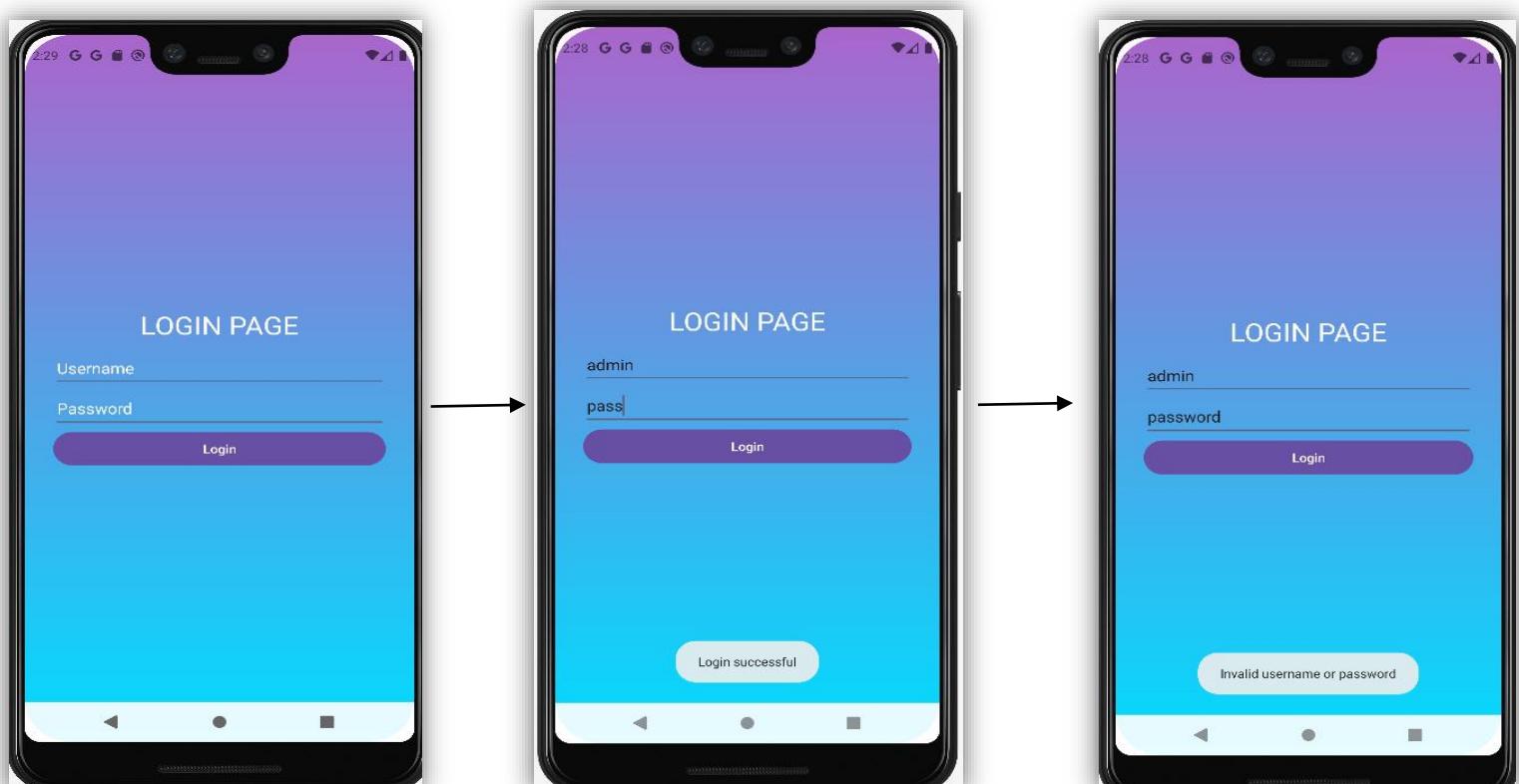
- Update the following code for creating method to validation

```
etUsername=findViewById(R.id.Uname);
etPassword=findViewById(R.id.password);
btnLogin=findViewById(R.id.login);
btnLogin.setOnClickListener(new View.OnClickListener() {
 @Override
 public void onClick(View v) {
 String username = etUsername.getText().toString().trim();
 String password = etPassword.getText().toString().trim();
 if(username.equals("admin") && password.equals("pass"))
 {
 Toast.makeText(MainActivity.this, "Login successful",
Toast.LENGTH_SHORT).show();
 }
 else
 {
 Toast.makeText(MainActivity.this, "Invalid username or password",
Toast.LENGTH_SHORT).show();
 }
 }
});
```

```
activity_main.xml MainActivity.java
```

```
1 package com.mad.madprog14;
2
3 import android.os.Bundle;
4 import android.view.View;
5 import android.widget.Button;
6 import android.widget.EditText;
7 import android.widget.Toast;
8 import androidx.activity.EdgeToEdge;
9 import androidx.appcompat.app.AppCompatActivity;
10
11 public class MainActivity extends AppCompatActivity {
12 EditText etUsername,etPassword;
13 Button btnLogin;
14 @Override
15 protected void onCreate(Bundle savedInstanceState) {
16 super.onCreate(savedInstanceState);
17 EdgeToEdge.enable($this$enableEdgeToEdge: this);
18 setContentView(R.layout.activity_main);
19 etUsername=findViewById(R.id.Uname);
20 etPassword=findViewById(R.id.password);
21 btnLogin=findViewById(R.id.login);
22 btnLogin.setOnClickListener(new View.OnClickListener() {
23 @Override
24 public void onClick(View v) {
25 String username = etUsername.getText().toString().trim();
26 String password = etPassword.getText().toString().trim();
27 if(username.equals("admin") && password.equals("pass"))
28 {
29 Toast.makeText(context: MainActivity.this, text: "Login successful", Toast.LENGTH_SHORT).show();
30 }
31 else
32 {
33 Toast.makeText(context: MainActivity.this, text: "Invalid username or password", Toast.LENGTH_SHORT).show();
34 }
35 }
36 });
37 }
38 }
```

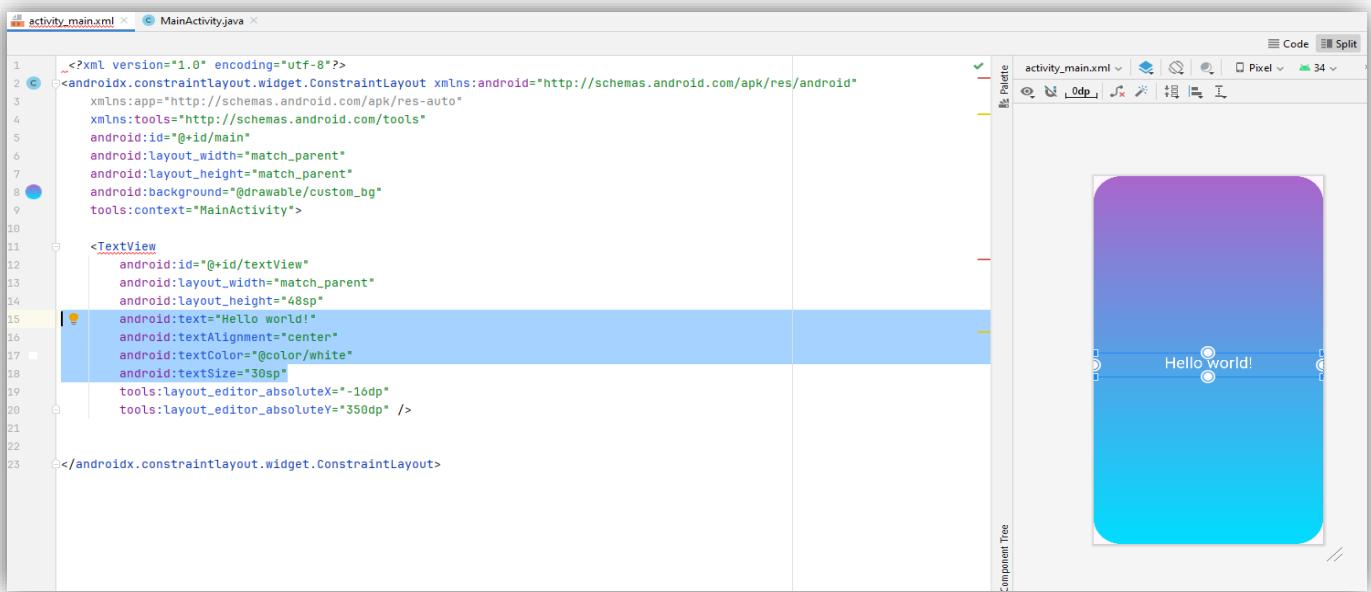
11. Press **Shift F10** to run the application Or, Press the run button on top of the interface



## 14.Learn to deploy Android applications

1. Create a **new project** and choose **Empty Views Activity** then click **Next**.
2. Change the Text View content in the code as follows:

```
<TextView
 android:id="@+id/textView"
 android:layout_width="wrap_content"
 android:layout_height="wrap_content"
 android:text="Hello world!"
 android:textSize="30sp"/>
```



3. **Connect USB to the physical mobile -> Go to developer option -> enable USB debugger , install via USB and USB debugger(security settings)**
4. Press **Shift F10** to run the application Or, Press the run button on top of the interface
5. Allow access to install app in physical mobile

