SL.No	Folder & Description
1	**AndroidManifest.xml** 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.
2	**Java** 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.
3	**res/drawable** 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
4	**res/layout** This is a directory for files that define your app's user interface**.**
5	**res/values** This is a directory for other various XML files that contain a collection of resources, such as strings and colours definitions.
6	**Build.gradle** This is an auto generated file which contains compileSdkVersion, buildToolsVersion, applicationId, minSdkVersion, targetSdkVersion, versionCode and versionName

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"

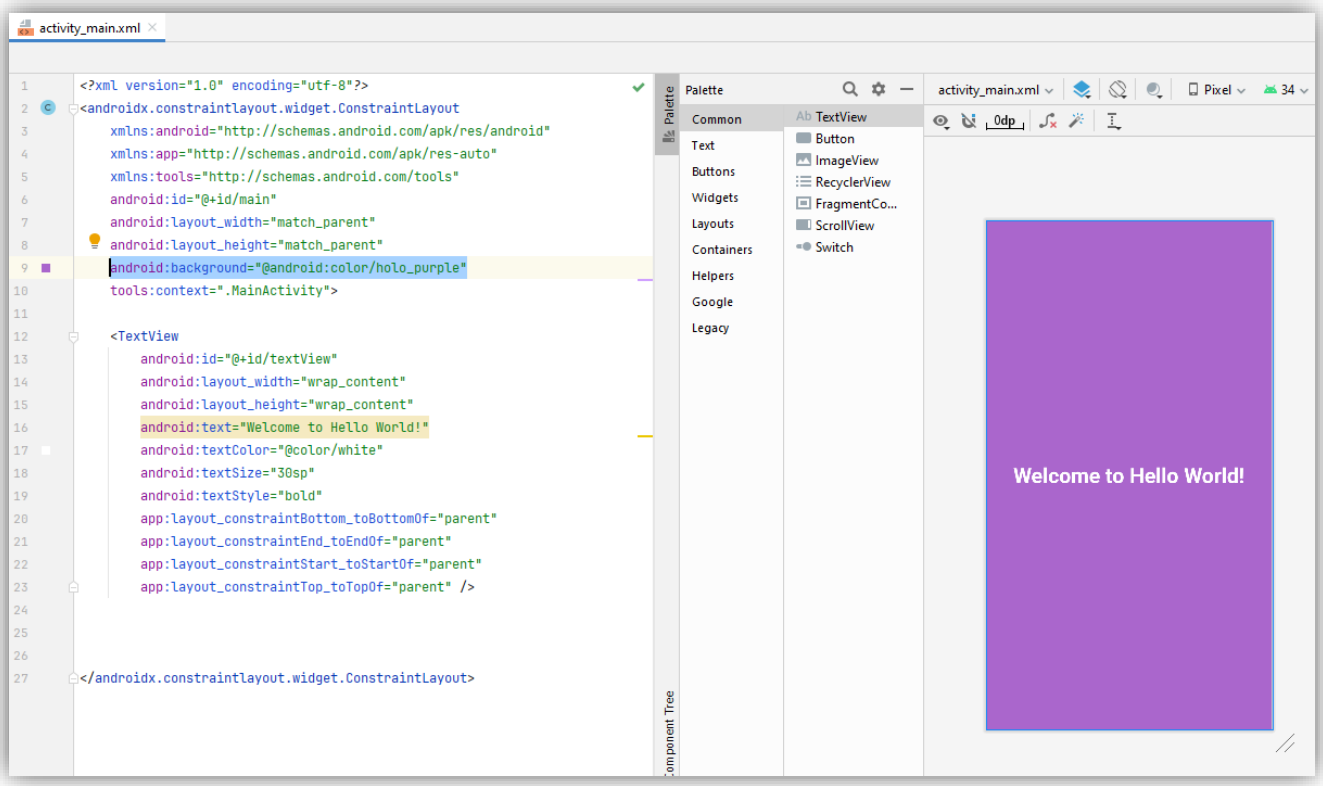
    android:layout_height="wrap_content"

    android:text="Welcome to Hello world!"

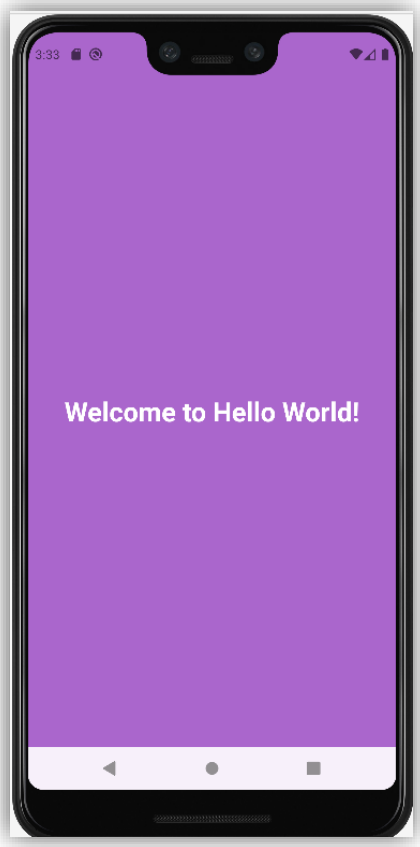
    android:textColor="@color/white"

    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:

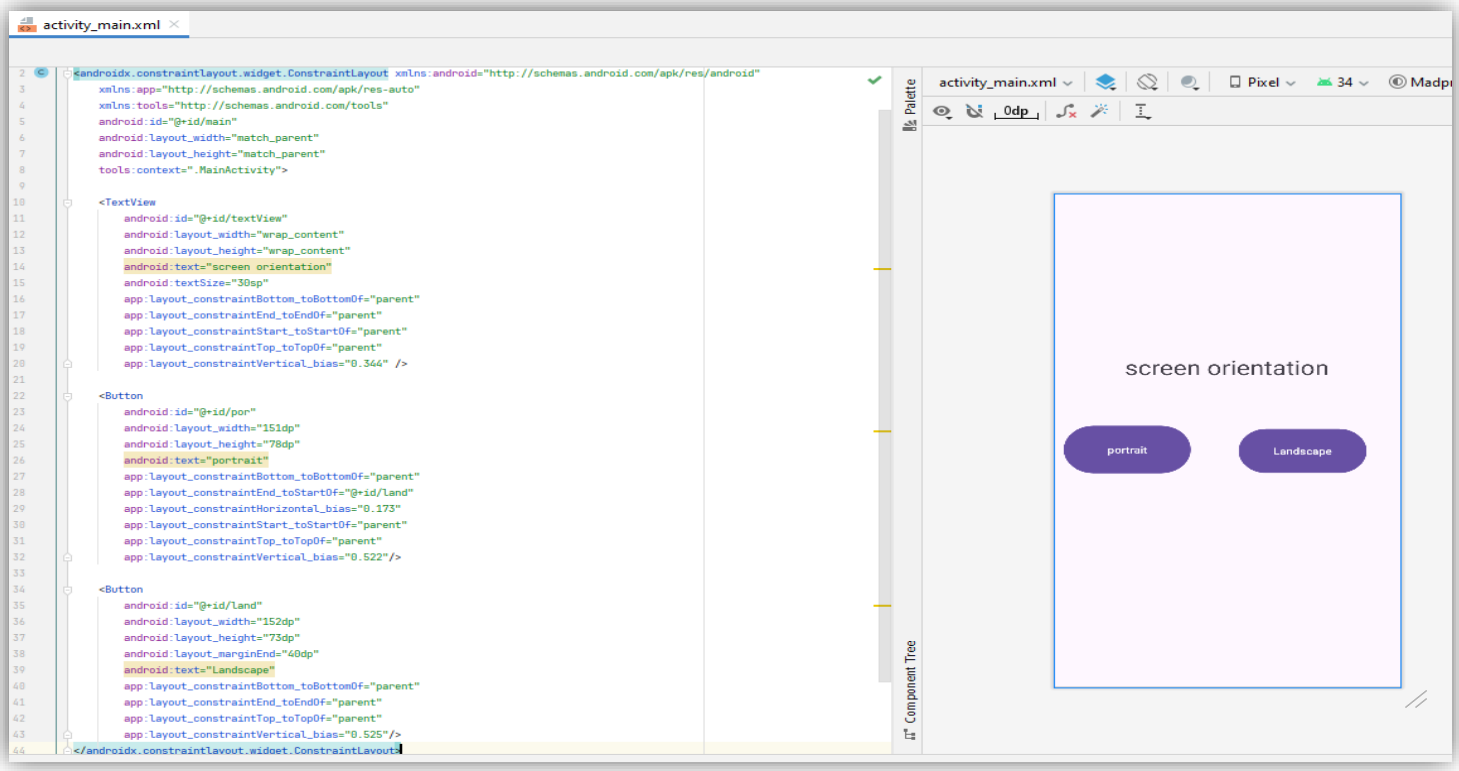
```
<TextView
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="screen orientation"
    android:textSize="30sp"/>
```

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:

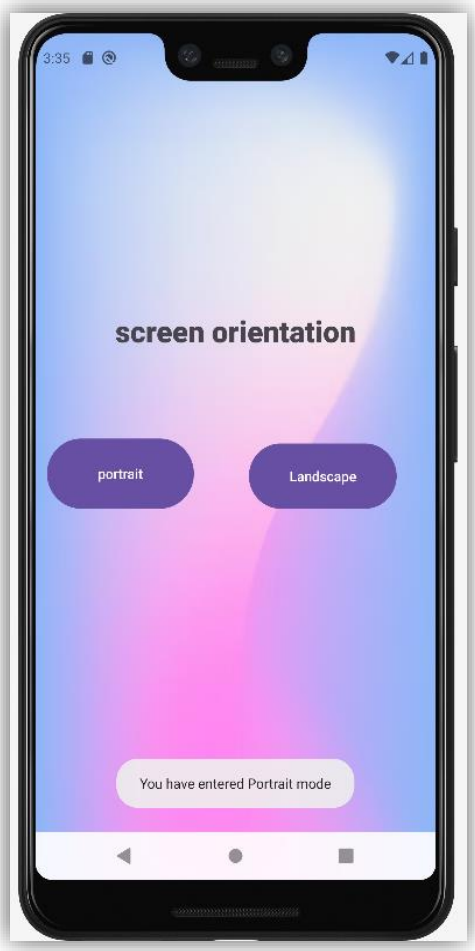
```
import android.view.View;
import android.widget.Button;
import android.widget.Toast;
```

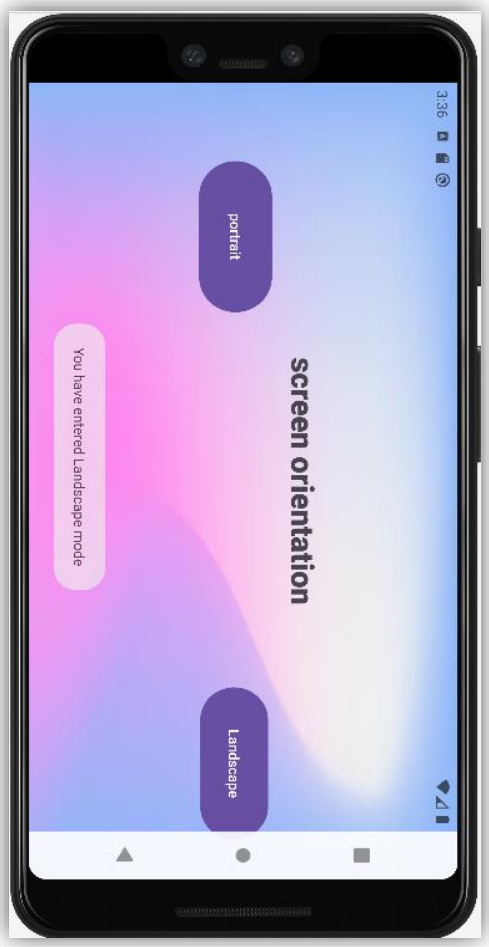
- Create two **Button** object, create **clickListener** event

```
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();
        }
    });
```

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

import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable( $this$enableEdgeToEdge: this);
        setContentView(R.layout.activity_main);

        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( context: MainActivity.this, text: "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( context: MainActivity.this, text: "You have entered Portrait mode",Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

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

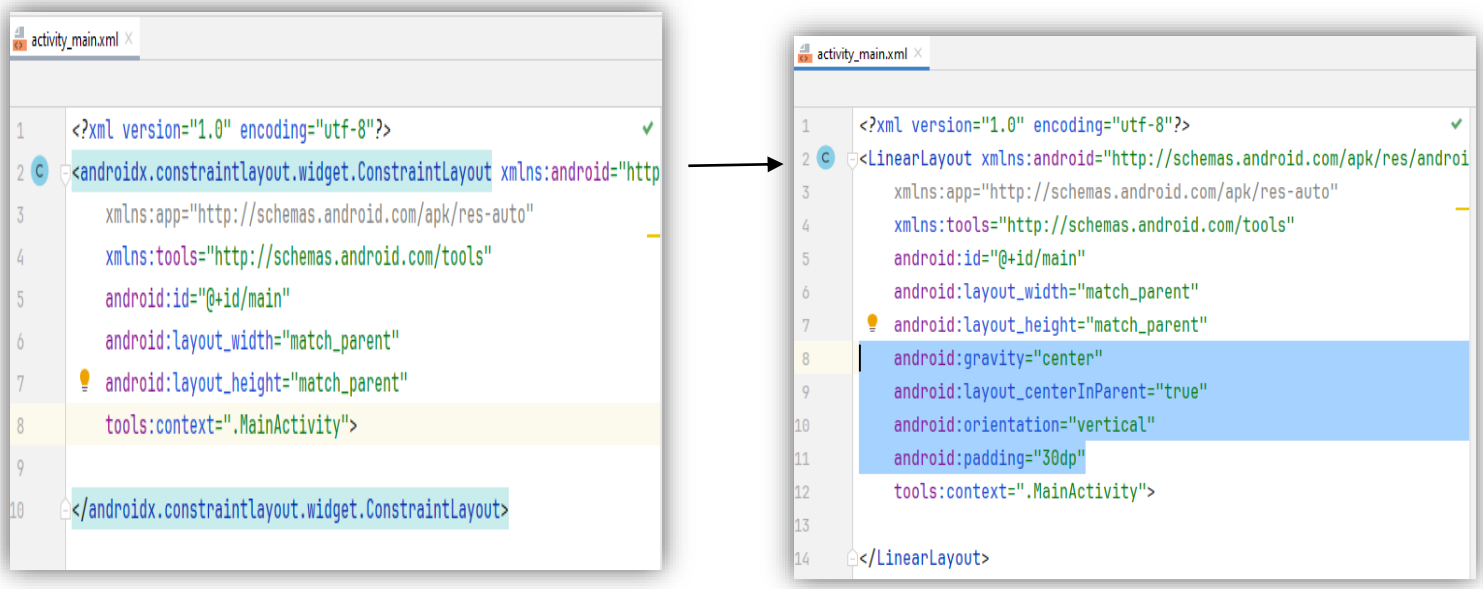




3. Create an application to develop Login window using UI controls.

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"*/>*

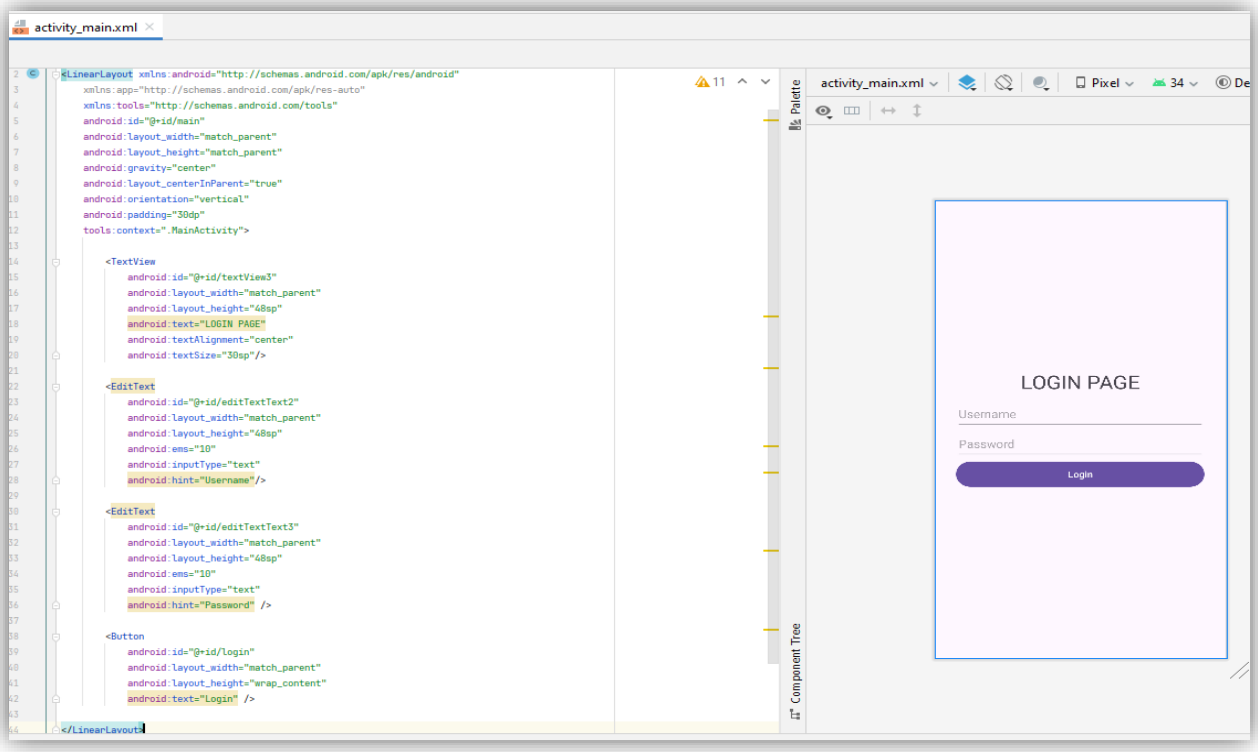
6. Add two edit text one after the other for usename and password
7. Go to palette option-> Text-> ***Plain Text***, Then Drag drop the Plain text into design window next to textview and update the following code
   <EditText
   android:id="@+id/editTextText2"
   android:layout_width="match_parent"
   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

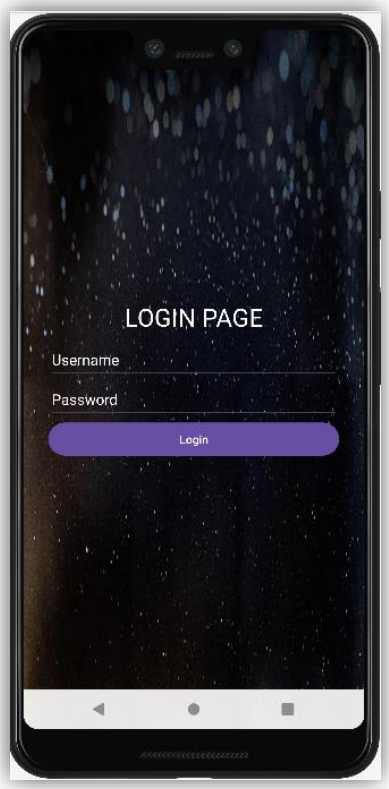
<EditText
android:id="@+id/editTextText2"
android:layout_width="match_parent"
android:layout_height="48sp"
android:ems="10"
android:inputType="text"
android:hint="Password"/>

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

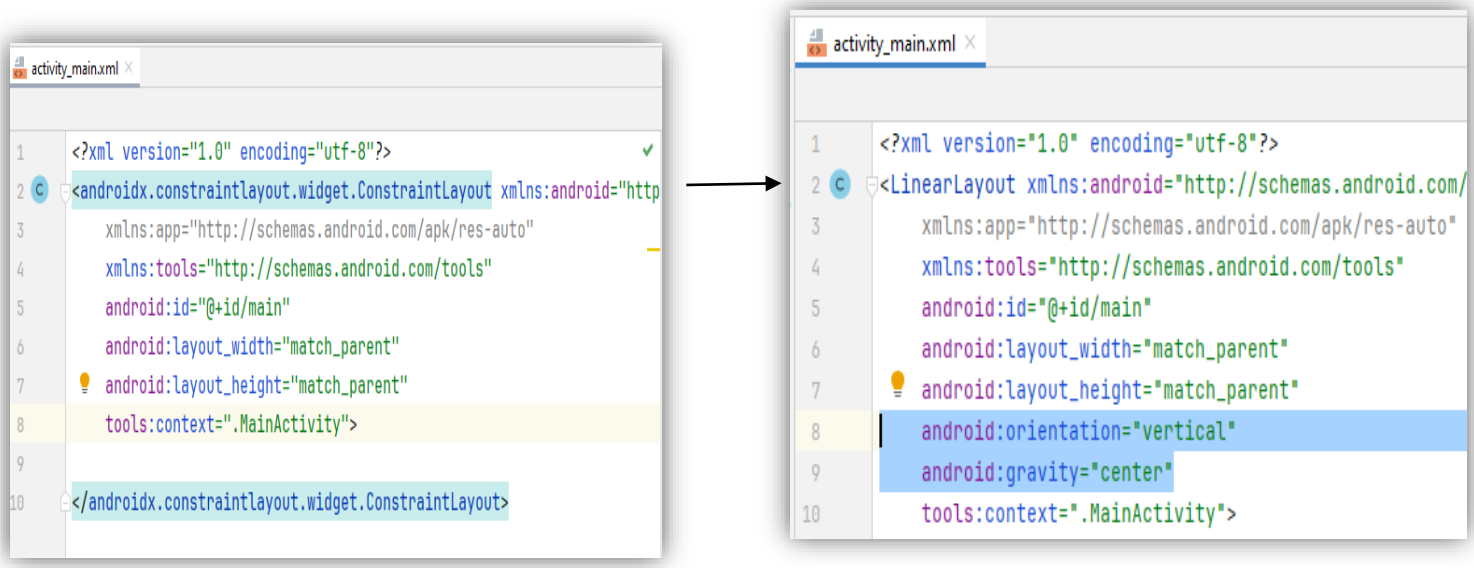


4. Create an application to implement new activity using explicit intent and implicit intent.

- ***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.
   <androidx.constraintlayout.widget.ConstraintLayout..
   To
   <LinearLayout..

3. update the following code in the linearlayout tag.
   android:orientation="vertical"
   android:gravity=”center”



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

<Button

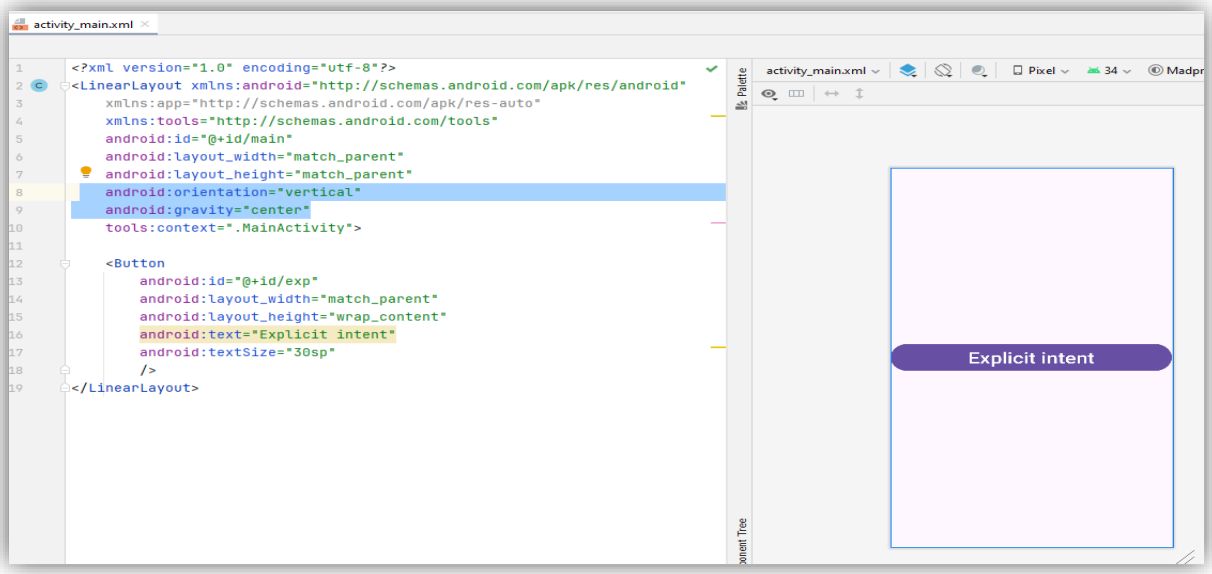
*android:id="@+id/**exp**"*

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);

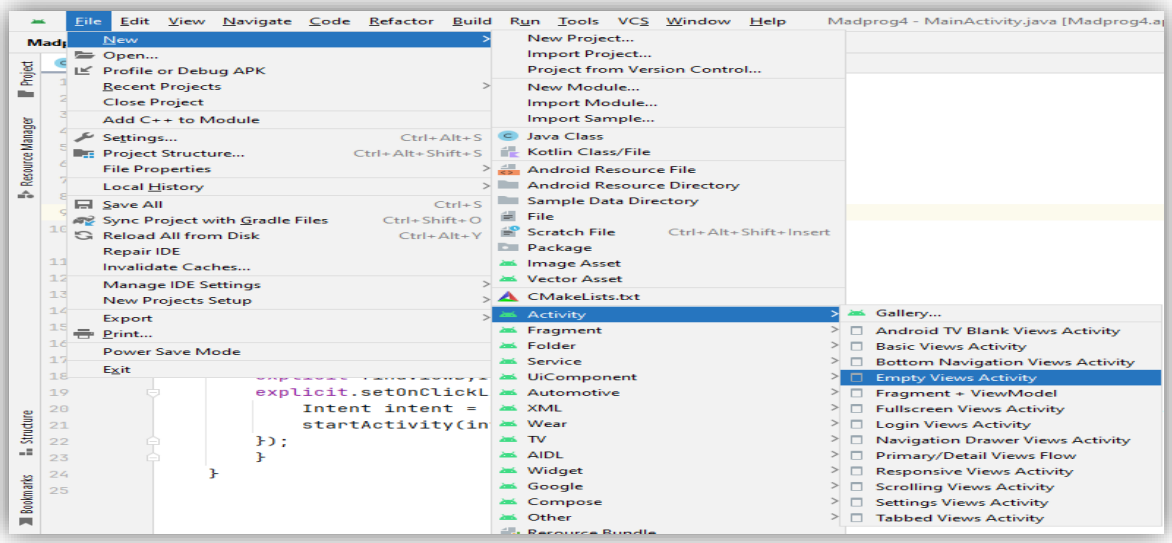
  explicit.setOnClickListener(logic -> {

  Intent intent = new Intent(MainActivity.this, SecondActivity.class);

  startActivity(intent);

  });

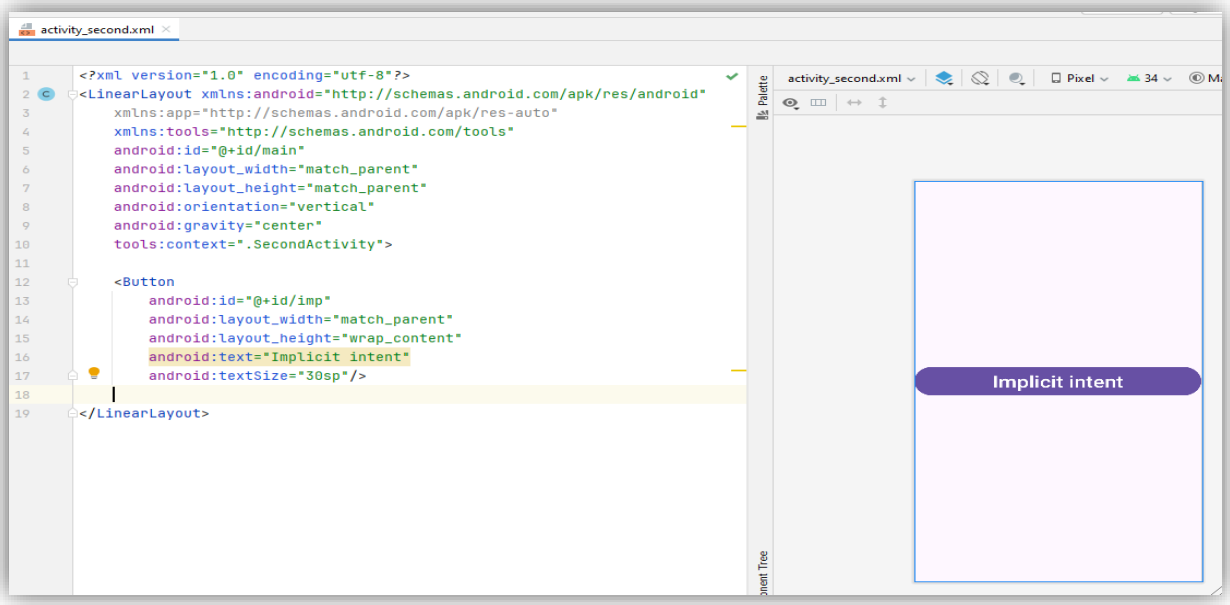
6. Add an another empty view activity and rename it as ***SecondActivity***
   Go to file -> New -> Activity -> Empty View Activity -> (*rename as* ***SecondActivity) -> Finish***
   (It will create ***Secondactivity.java*** *in* ***java*** *folder and* ***activity_second.xml*** *in* ***res*** *folder)*



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

8. 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"/>
```

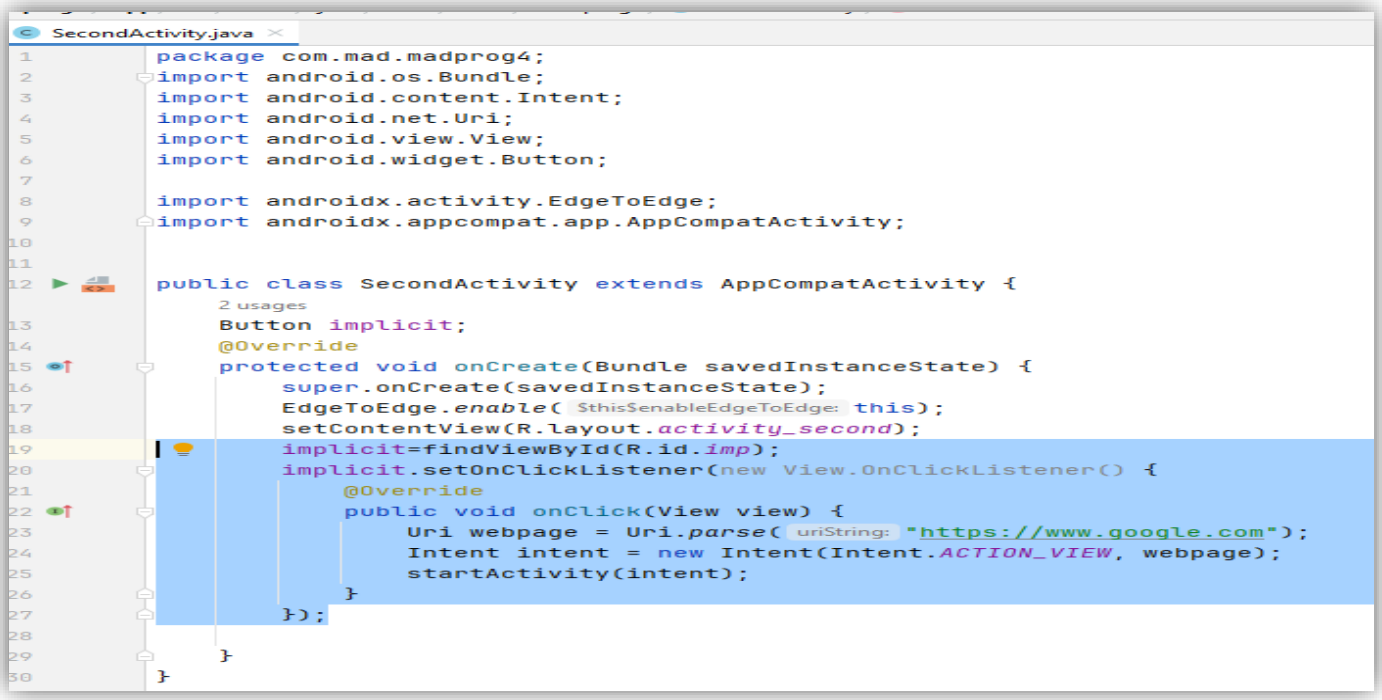


9. Perform a logic for implicit intent in ***SecondAvctivity.java*** file
   - Importing necessary packages

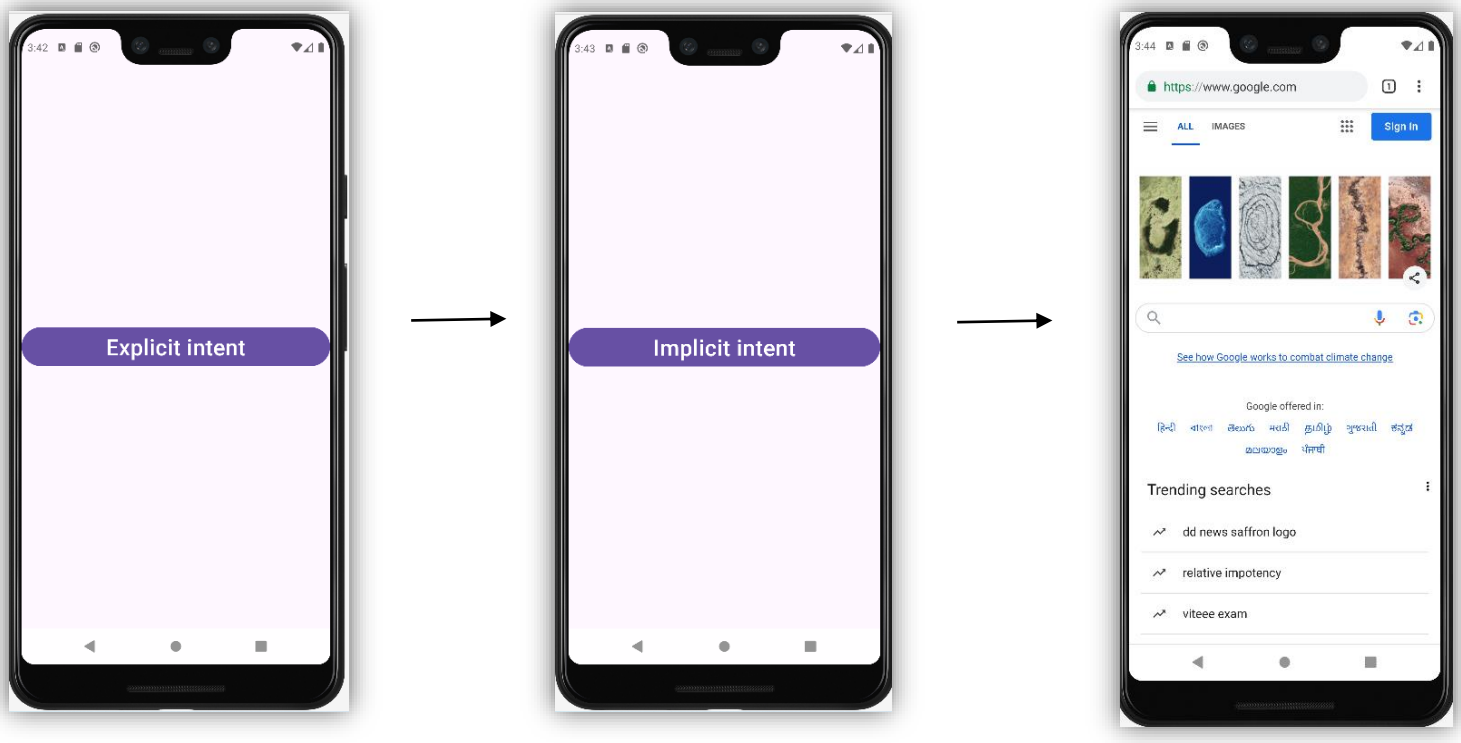
```
import android.content.Intent;
import android.net.Uri;
import android.view.View;
import android.widget.Button;
```

- 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);
  }
  });

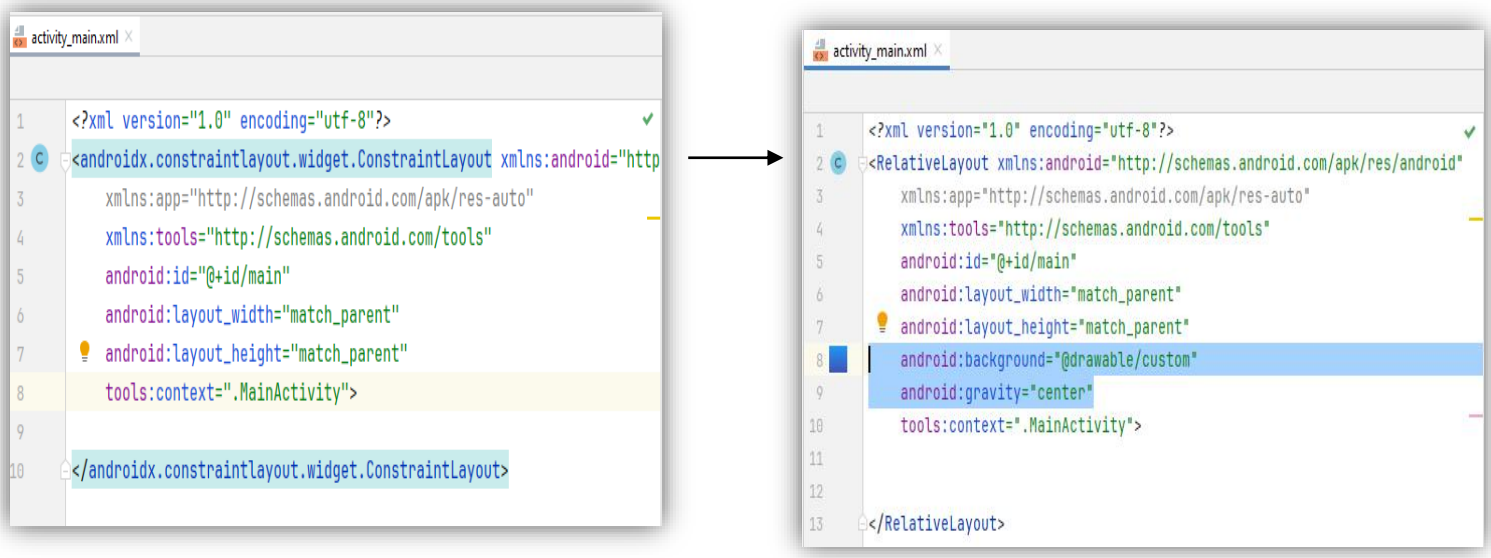


10. Press **Shift F10** to run the application Or, Press the run button on top of the interface



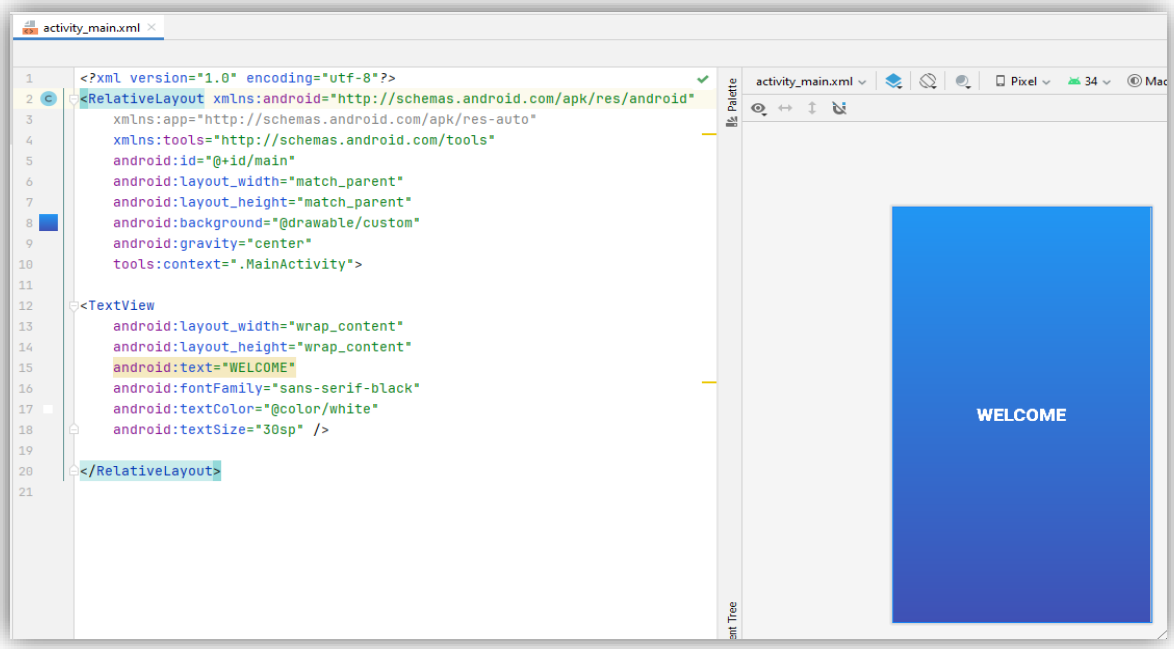
5. Create an application that displays custom designed Opening Screen

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:gravity=”center”



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" />
```

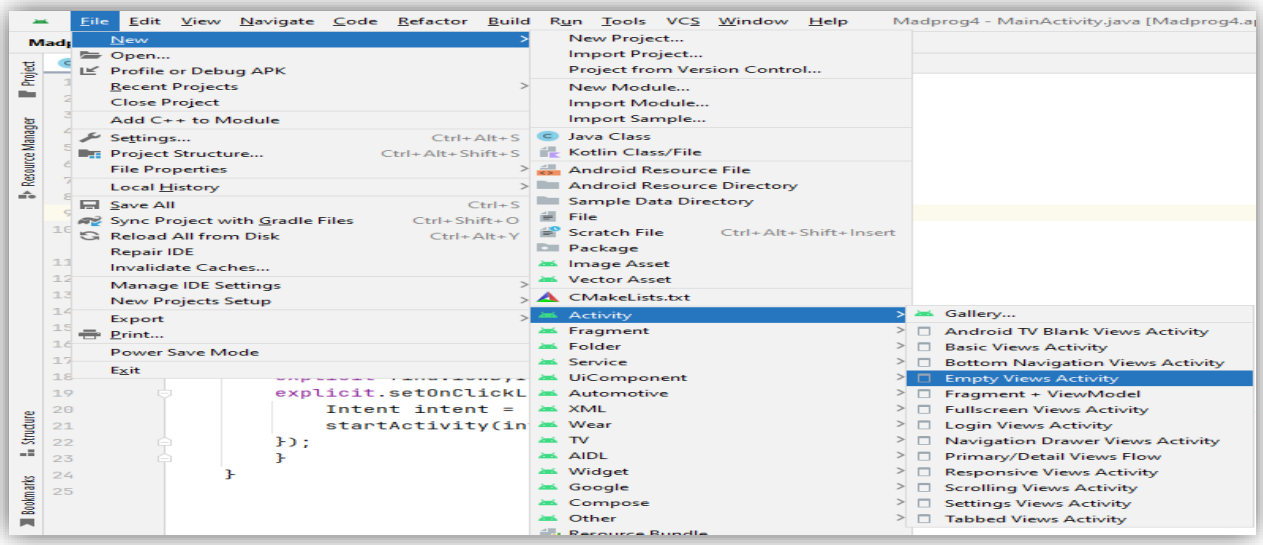


6. Update the following code in **MainActivity.java** file

- 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)*

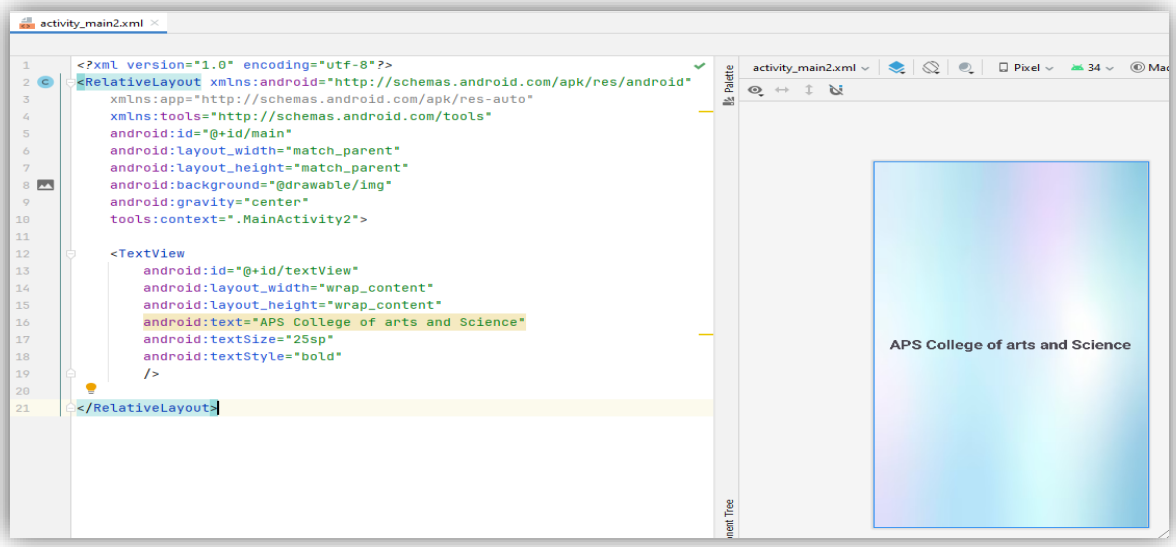


8. Change the layout from constraint Layout to Relative Layout in **activity_main2.xml** file.
   <androidx.constraintlayout.widget.ConstraintLayout..
   To
   <RelativeLayout...

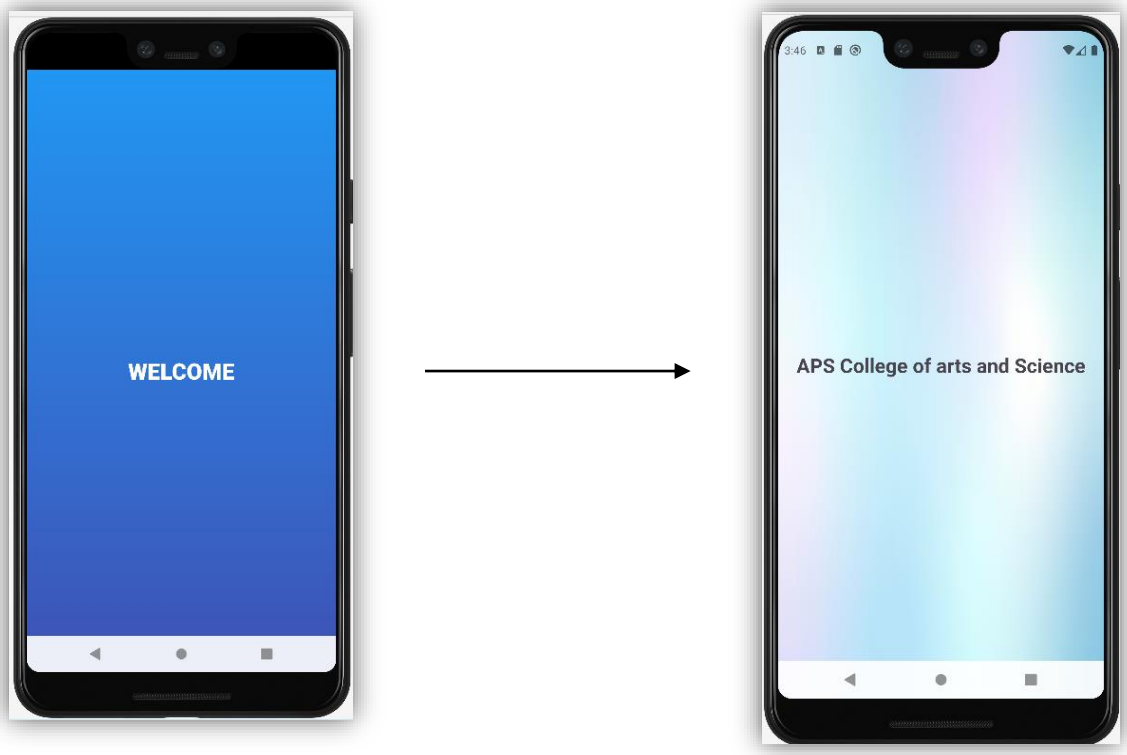
9. Change the background and update the following code in the Relativelayout tag.
   android:gravity=”center”

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” />



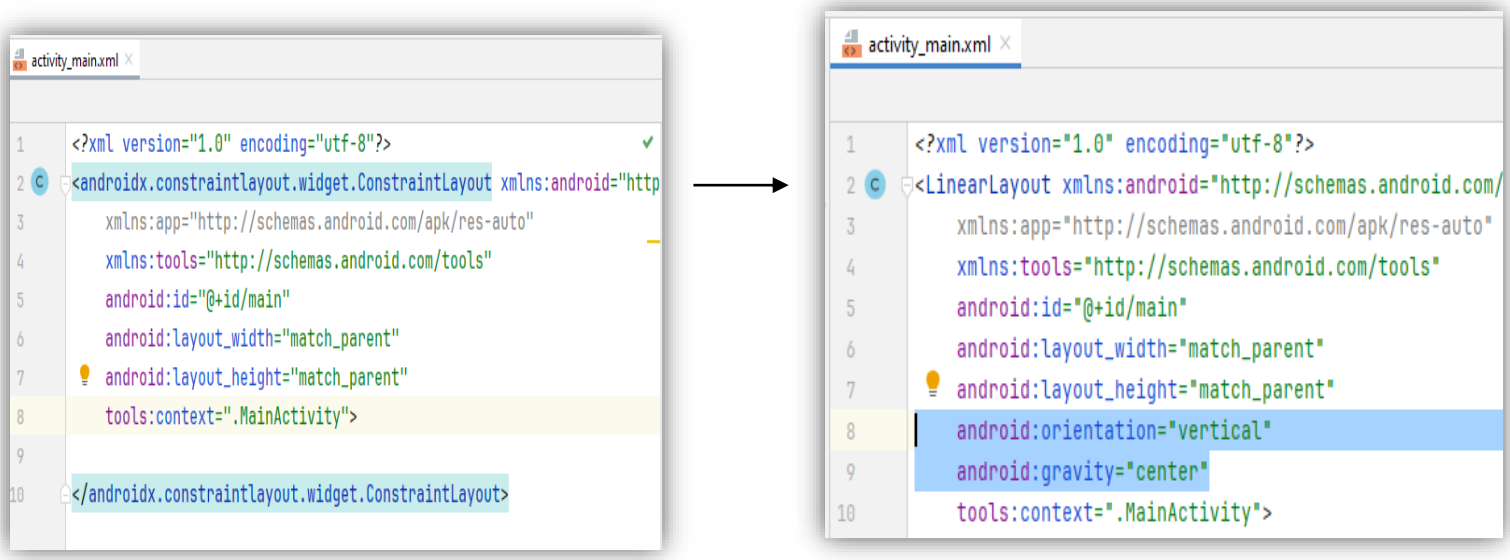
11. Press **Shift F10** to run the application Or, Press the run button on top of the interface



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"
    />
```

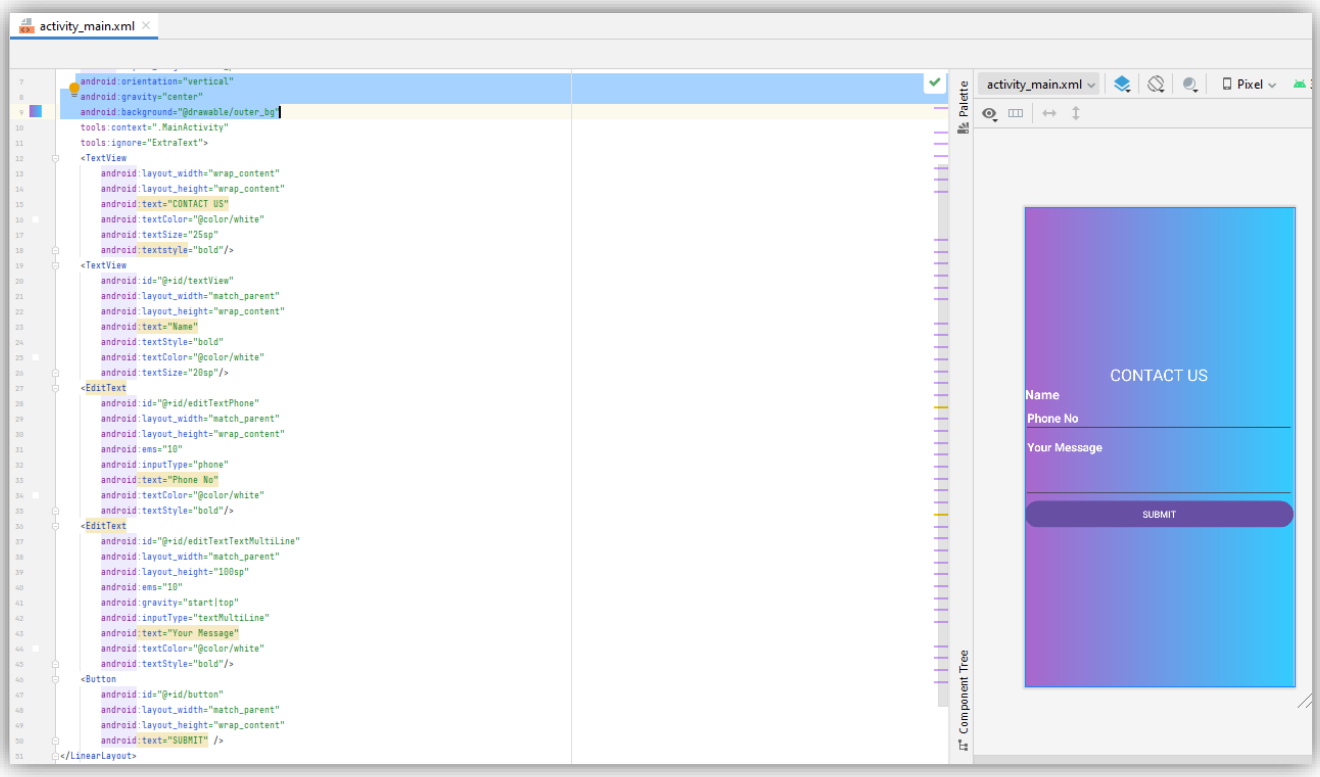
8. Add a multiline for message

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

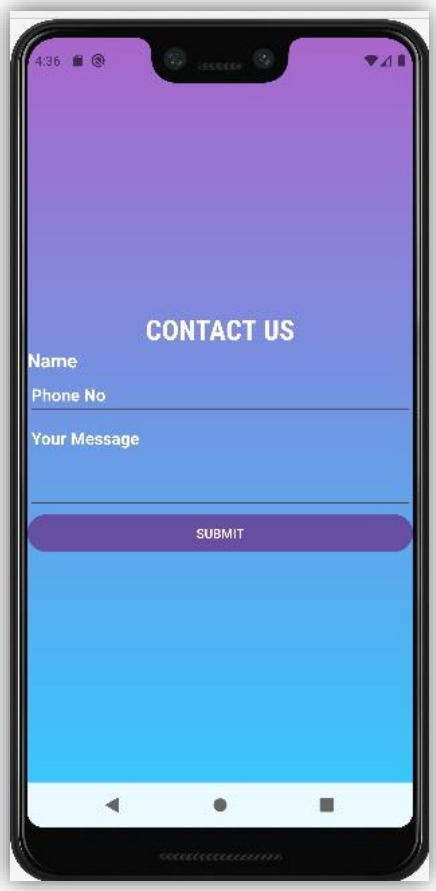
```
<EditText
    android:id="@+id/editTextTextMultiLine"
    android:layout_width="match_parent"
    android:layout_height="100sp"
    android:ems="10"
    android:gravity="start|top"
    android:inputType="textMultiLine"
    android:text="Your Message"
    />
```

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" />
```



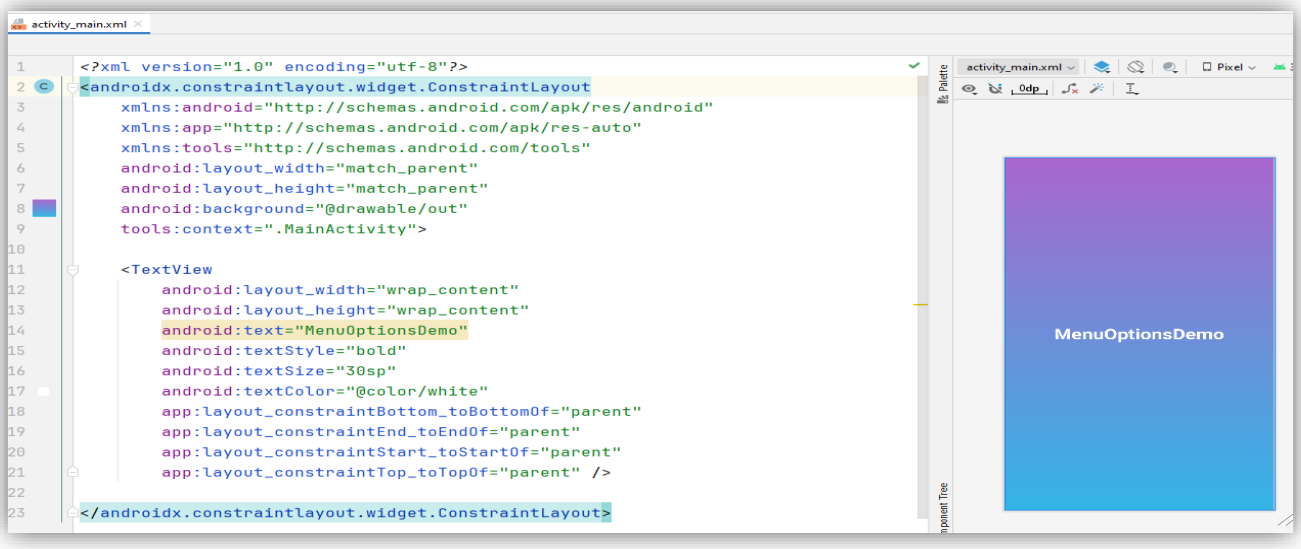
10. Press **Shift F10** to run the application Or, Press the run button on top of the interface



7. Create menu in application.

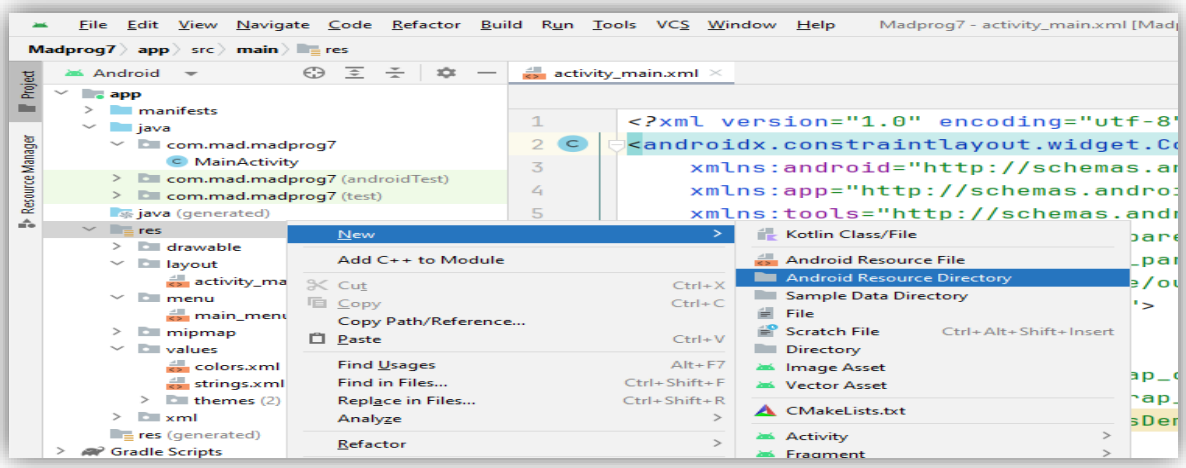
1. Create a **new project** and choose **Empty Views Activity** then click **Next**.
2. Change the Text View content in the code as follows:

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

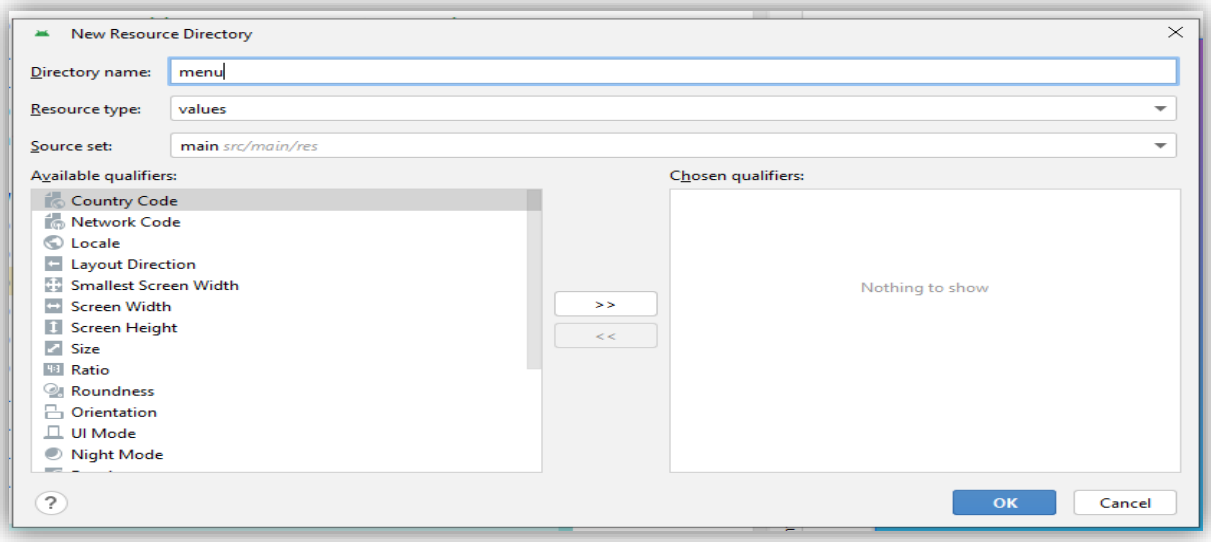


3. Create a new directory for menu

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

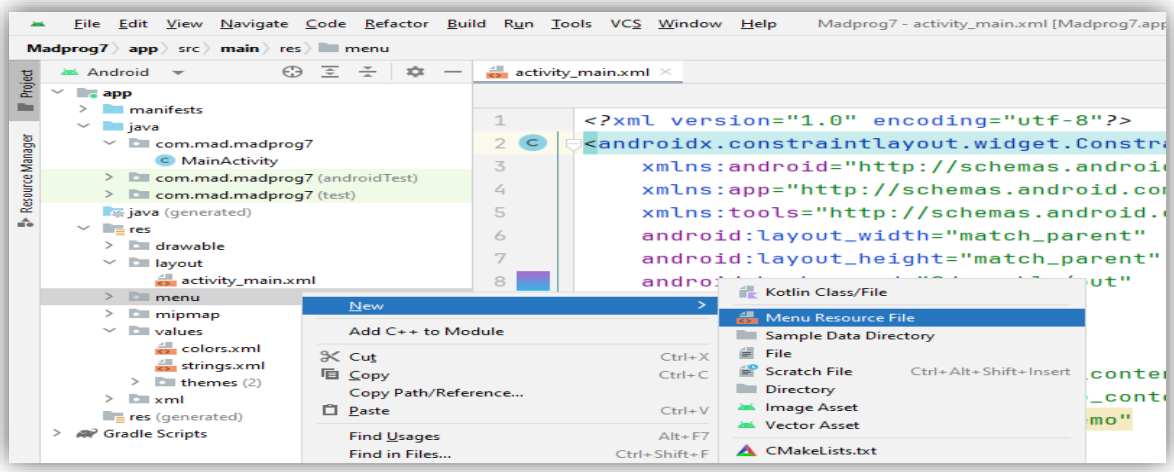


Change directory name to **menu**, Then Click **OK**. A menu folder will be created in res directory

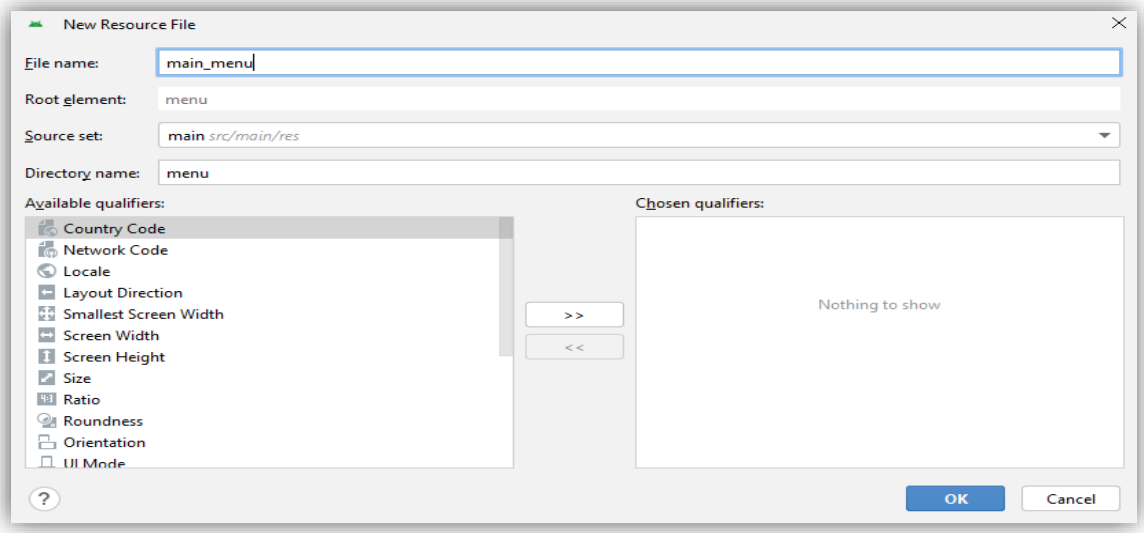


4. Create a main_menu.xml file under menu folder

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

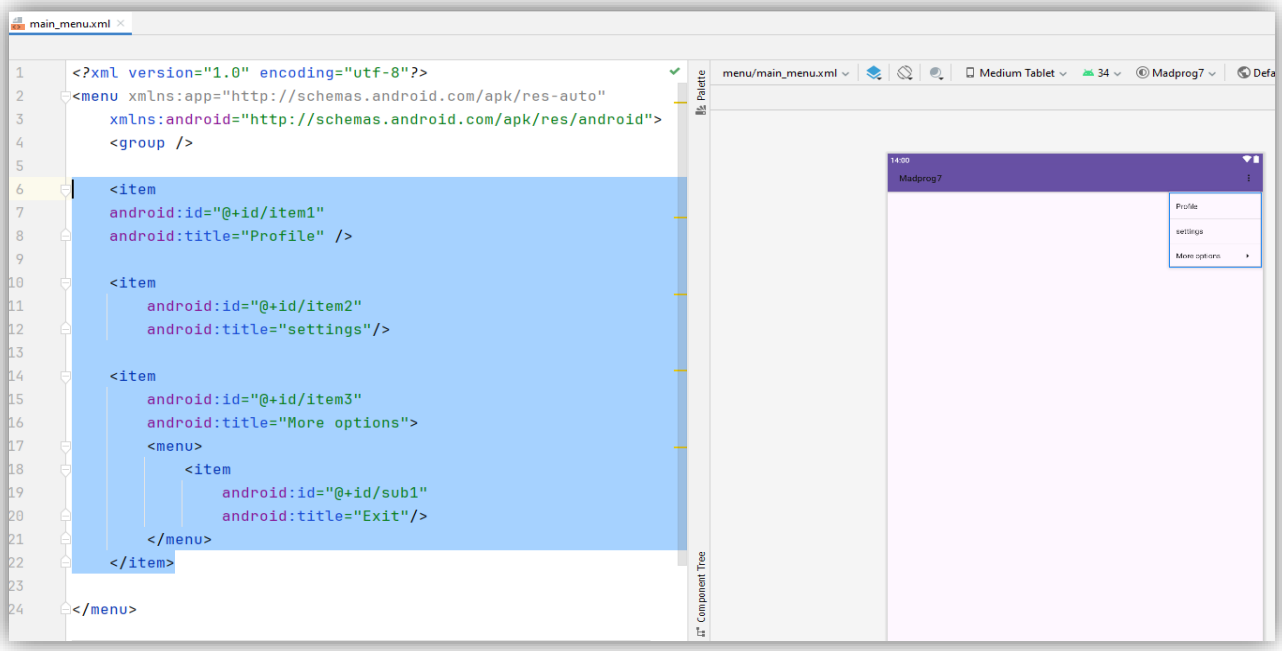


Change file name to **menu**, Then Click **OK**. A main_menu.xml file will be created in menu folder.



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>
```



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;
```

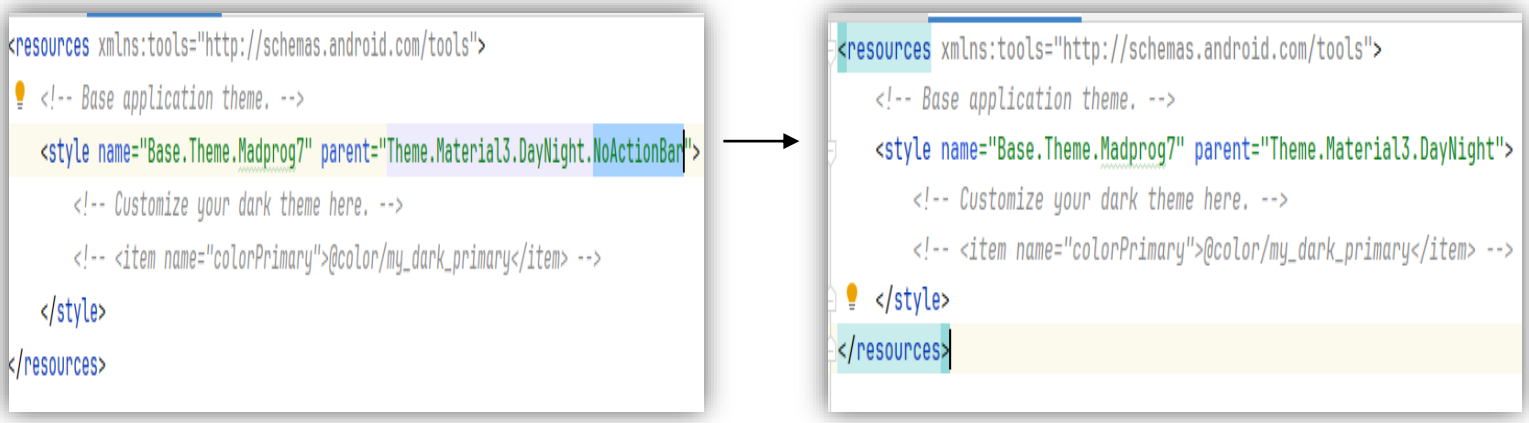
- create a method for menu options

```
@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;
}
```

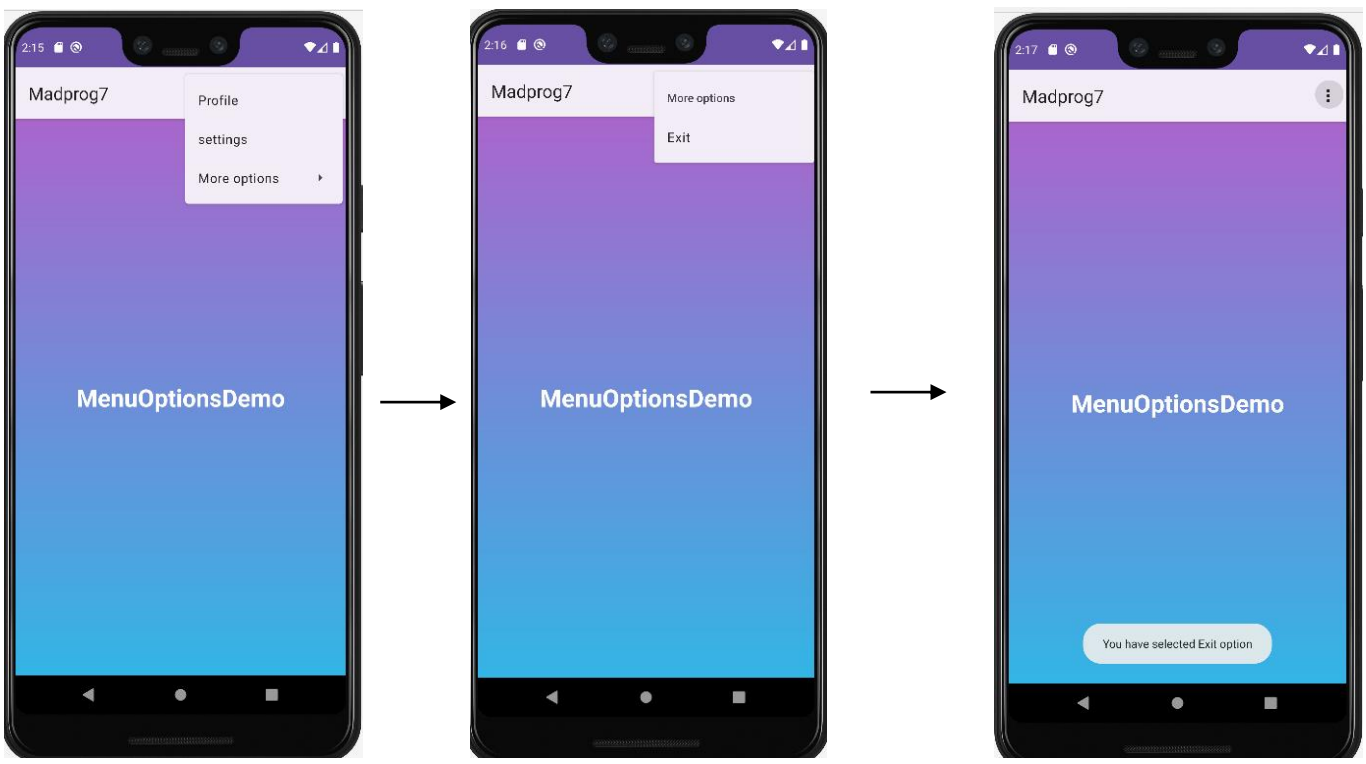
7. Go to themes and remove the NoActionBar attribute in both themes.xml and theme.xml(night)

- ***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

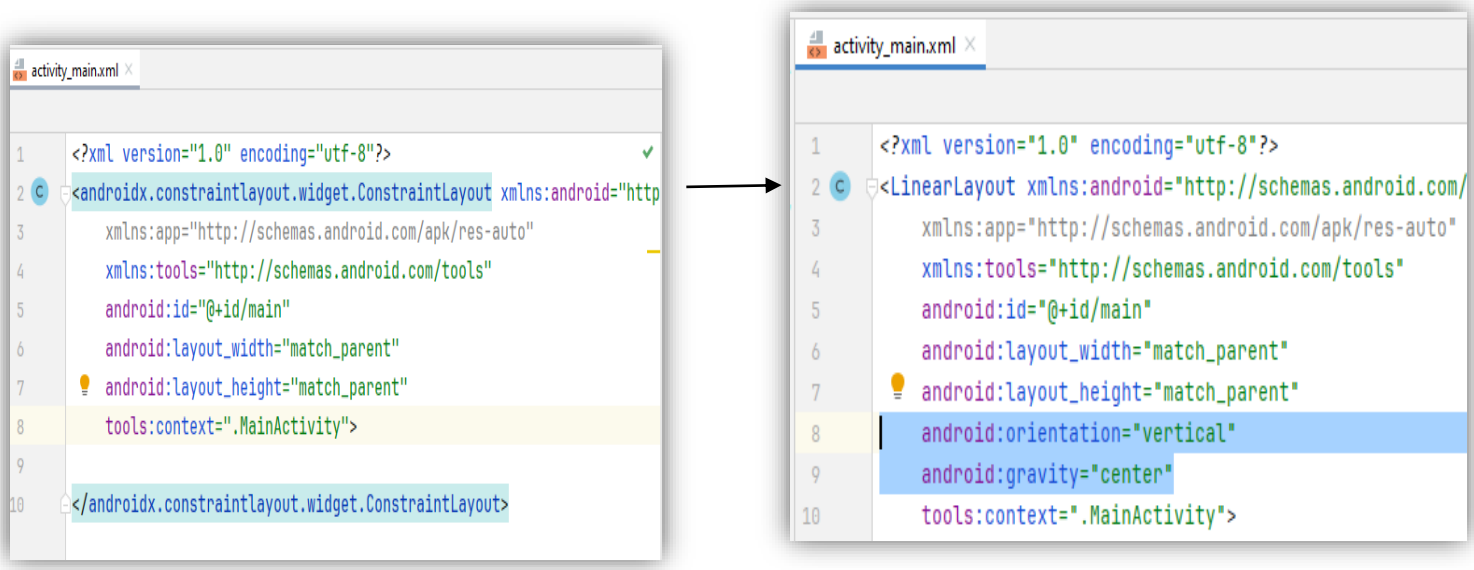


8. Read / Write the Local data.

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

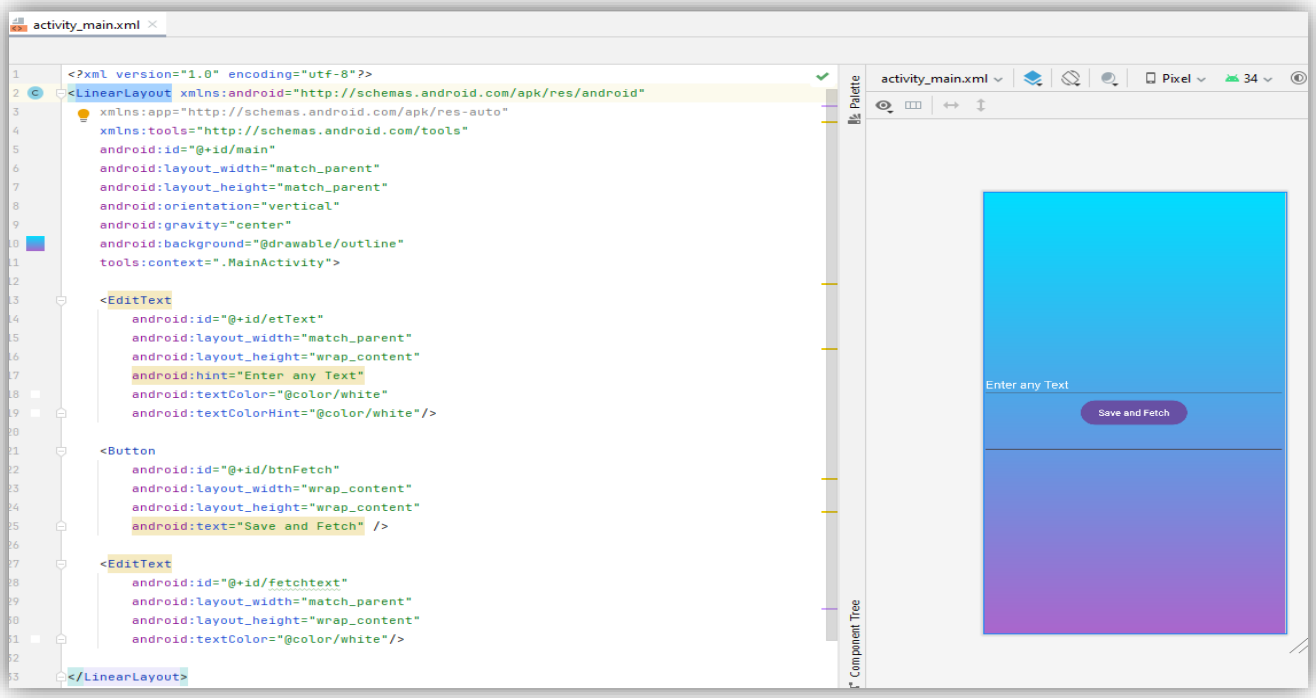
```
<EditText
    android:id="@+id/fetchtext"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>
```

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);

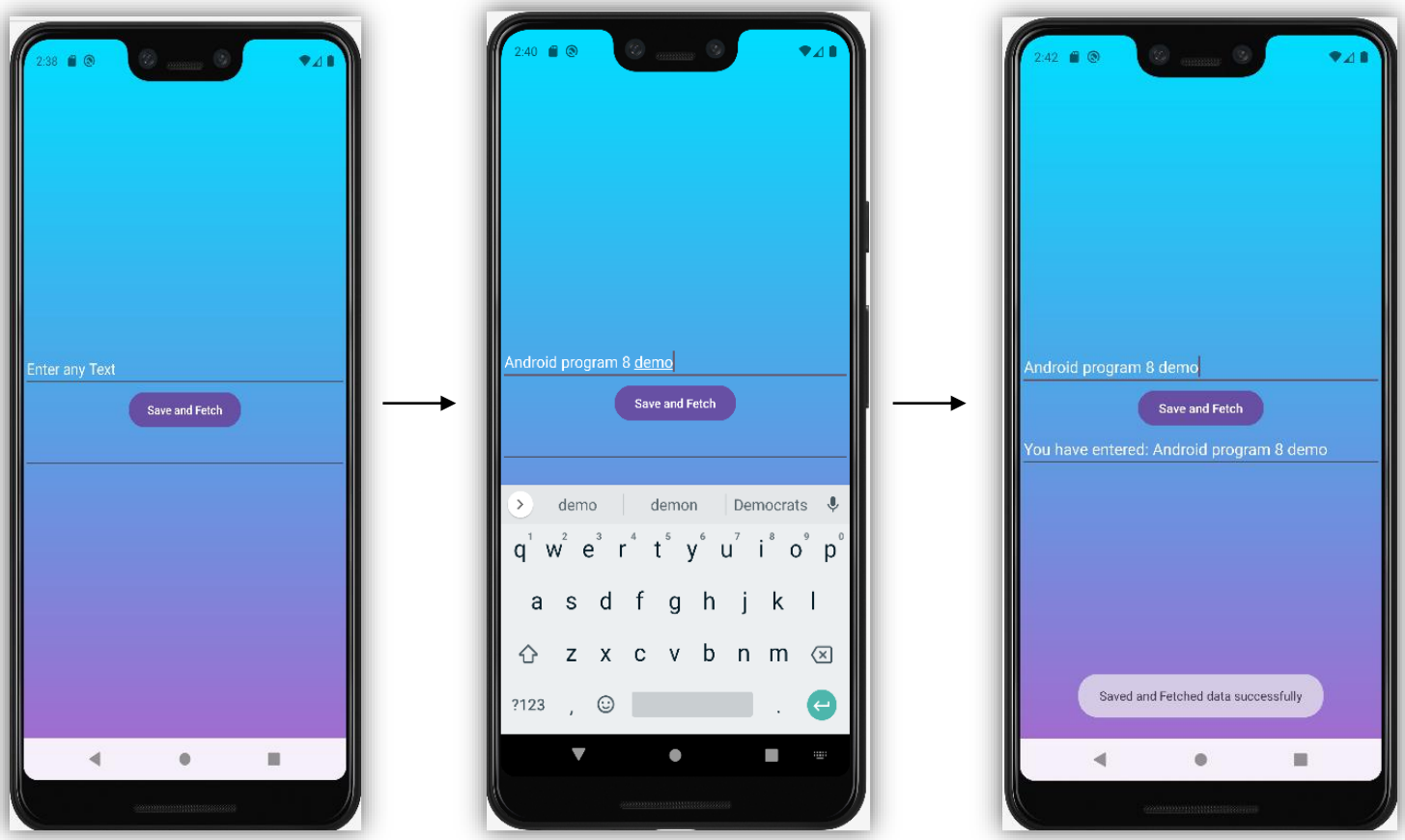
etText = (EditText)findViewById(R.id.etText);
fetchtext=(EditText) findViewById(R.id.fetchtext);
btnFetch = (Button) findViewById(R.id.btnFetch);

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 | MainActivity.java

```
import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    2 usages
    EditText etText,fetchtext;
    2 usages
    Button btnFetch;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable( $this$enableEdgeToEdge: 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(), text: "Saved and Fetched data successfully",Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

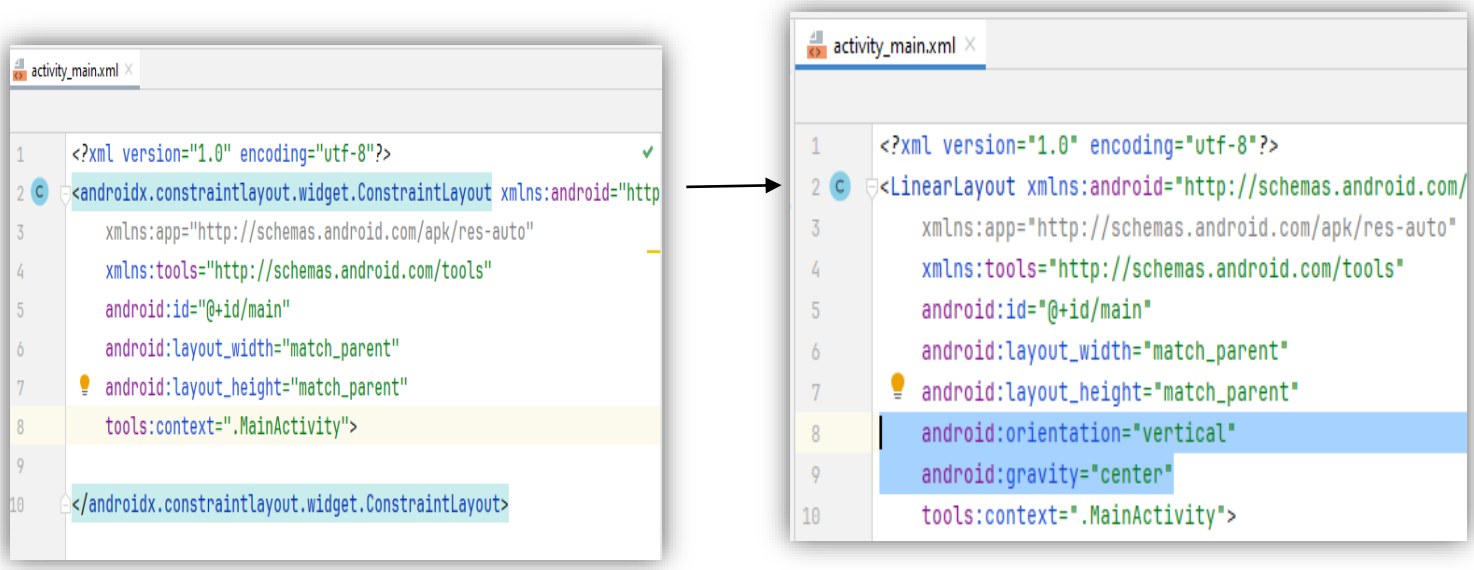
5. Press **Shift F10** to run the application Or, Press the run button on top of the interface



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:

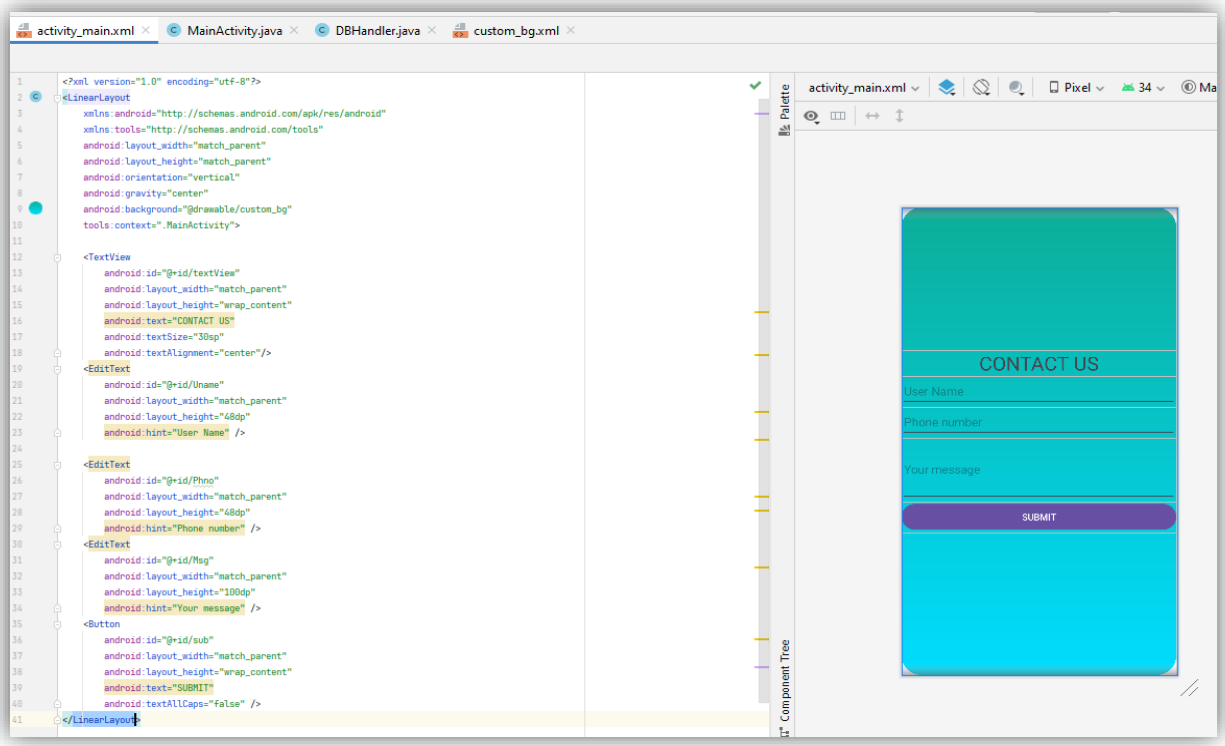
<TextView
android:id="@+id/textView"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="CONTACT US"/>

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" */>*

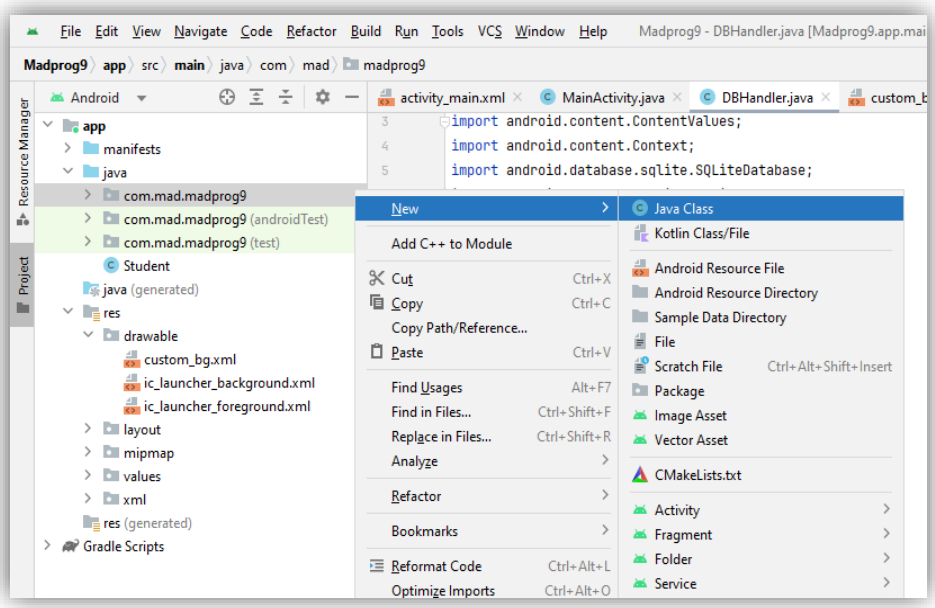
   <EditText
   android:id="@+id/Msg"
   android:layout_width="match_parent"
   android:layout_height="100dp"
   android:layout_margin="10dp"
   android:hint="Your message" */>*

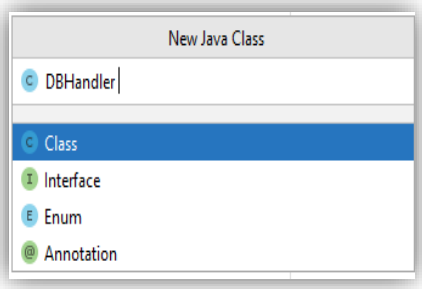
4. Add a button for submit
   <Button

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);
    }
}
```

DBHandler.java

```java
package com.mad.madprog9;
import android.content.ContentValues;
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
2 usages
public class DBHandler extends SQLiteOpenHelper {
    // below variable is for our database name.
    1 usage
    private static final String DB_NAME = "Feedbackdb";
    1 usage
    private static final int DB_VERSION = 1;
    3 usages
    private static final String TABLE_NAME = "contactus";
    1 usage
    private static final String ID_COL = "id";
    2 usages
    private static final String NAME_COL = "name";
    2 usages
    private static final String PHONE_COL = "phone_number";
    2 usages
    private static final String MESSAGE_COL = "message";
    1 usage
    public DBHandler(Context context) { super(context, DB_NAME, factory: 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.
    1 usage
    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, nullColumnHack: null, values);
        db.close(); }
    10 usages
    @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); } }
```

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);

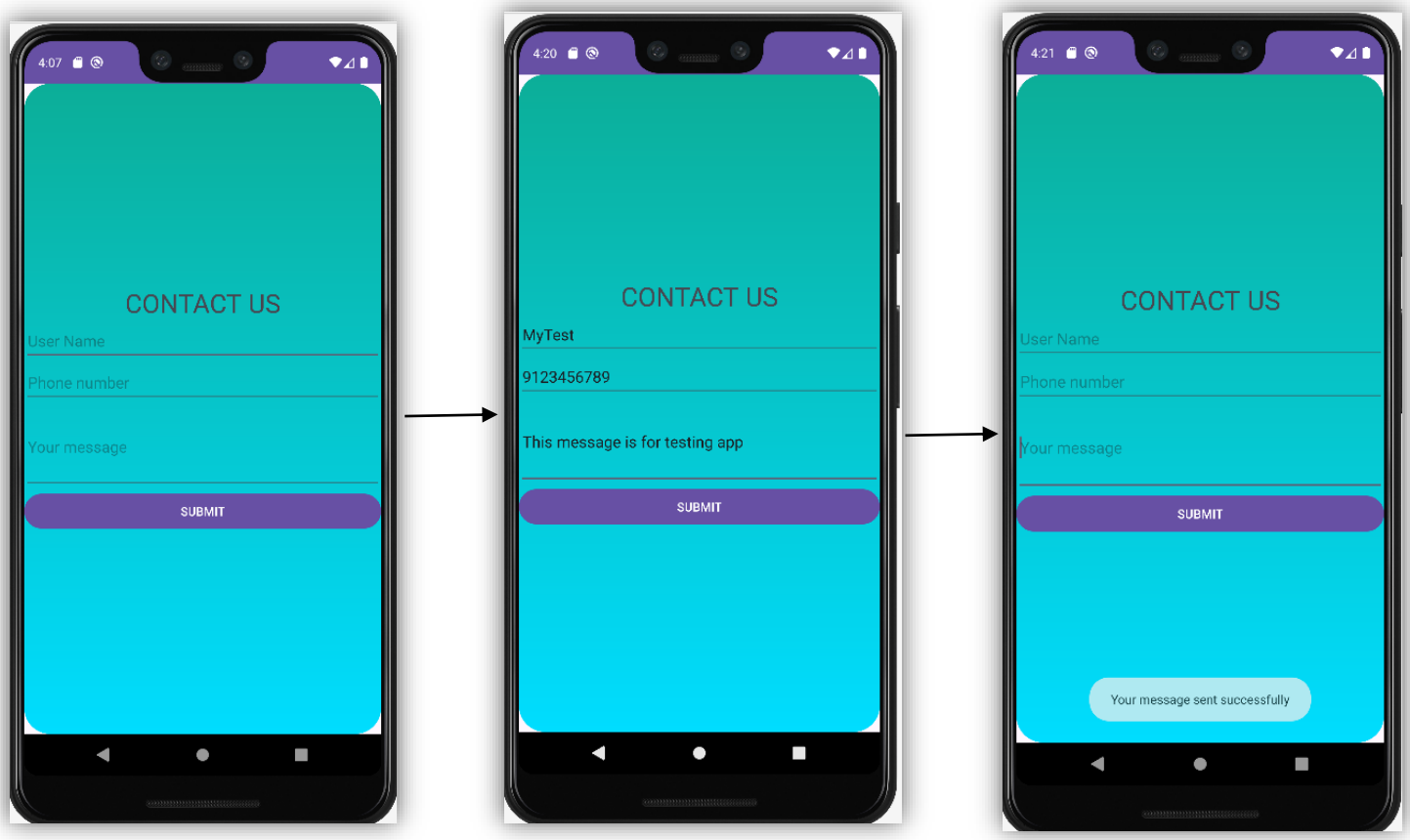
        submitBtn.setOnClickListener(new View.OnClickListener() {
          @Override

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

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

MainActivity.java

```
package com.mad.madprog9;
import ...
public class MainActivity extends AppCompatActivity {
    3 usages
    private EditText usernameEdt, phonenoEdt, messageEdt;
    2 usages
    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( context: MainActivity.this);

        submitBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String userName = usernameEdt.getText().toString();
                String phonenumber = phonenoEdt.getText().toString();
                String message = messageEdt.getText().toString();

                if (userName.isEmpty() && phonenumber.isEmpty() && message.isEmpty()) {
                    Toast.makeText( context: MainActivity.this,  text: "Please enter all the data..", Toast.LENGTH_SHORT).show();
                    return;
                }

                dbHandler.addNewCourse(userName, phonenumber, message);
                Toast.makeText( context: MainActivity.this,  text: "Your message sent successfully", Toast.LENGTH_SHORT).show();
                usernameEdt.setText("");
                phonenoEdt.setText("");
                messageEdt.setText("");
            }
        });
    }
}
```

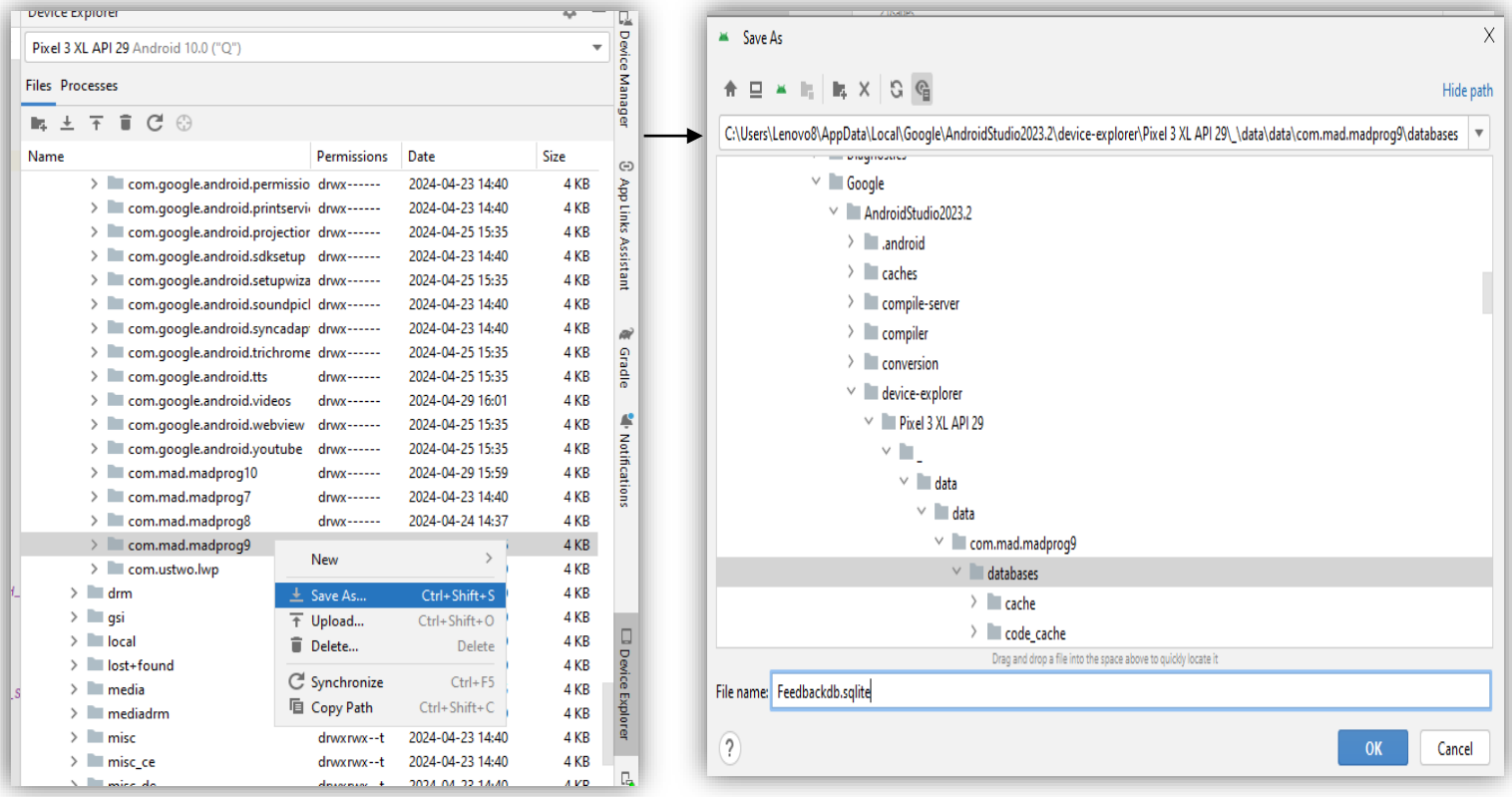
10. Press **Shift F10** to run the application Or, Press the run button on top of the interface



11. Save the database in android studio

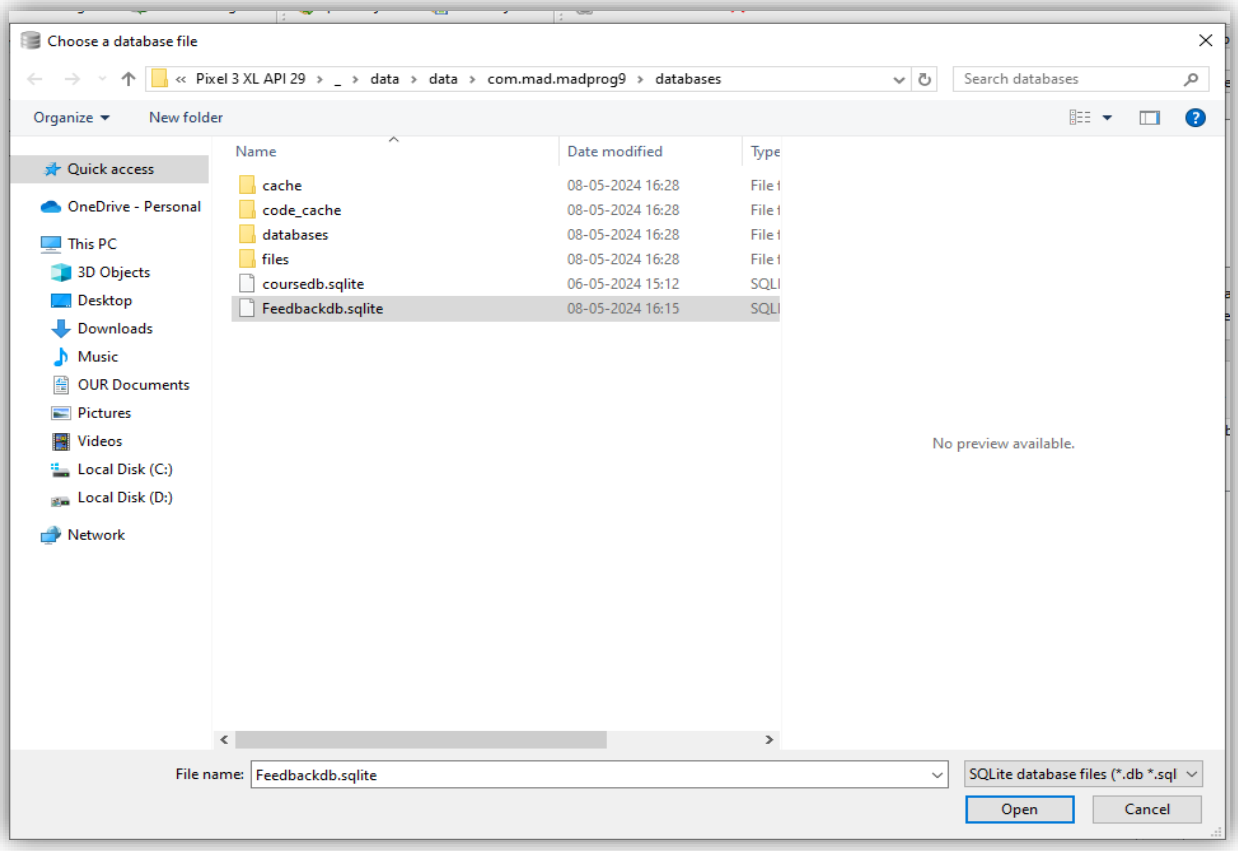
Go to device explorer- > data-> data-> Packagename(com.mad.madprog9)-> databases ->Feedbackdb (Right click) ->save As -> Feedbackdb.sqlite-> save

Copy the filepath location for further reference

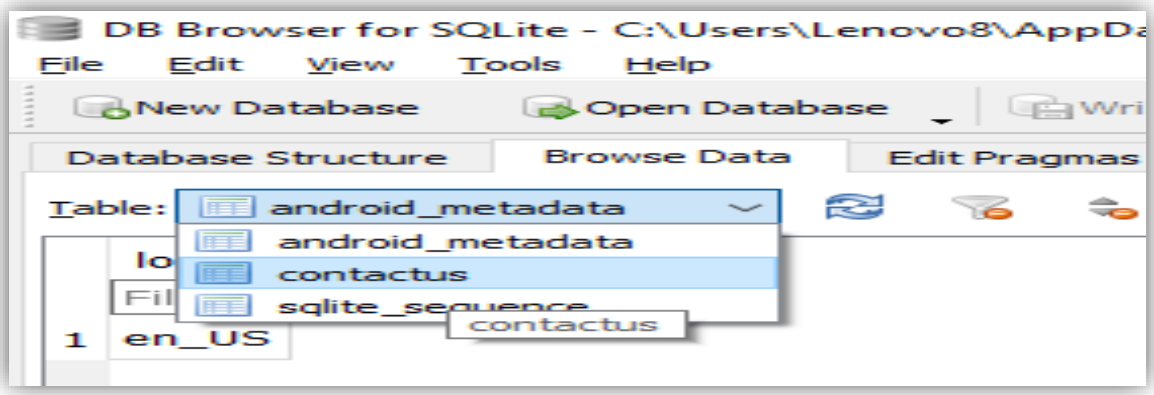


12. *Go to DBbrower for sqlite software then select* ***open database***

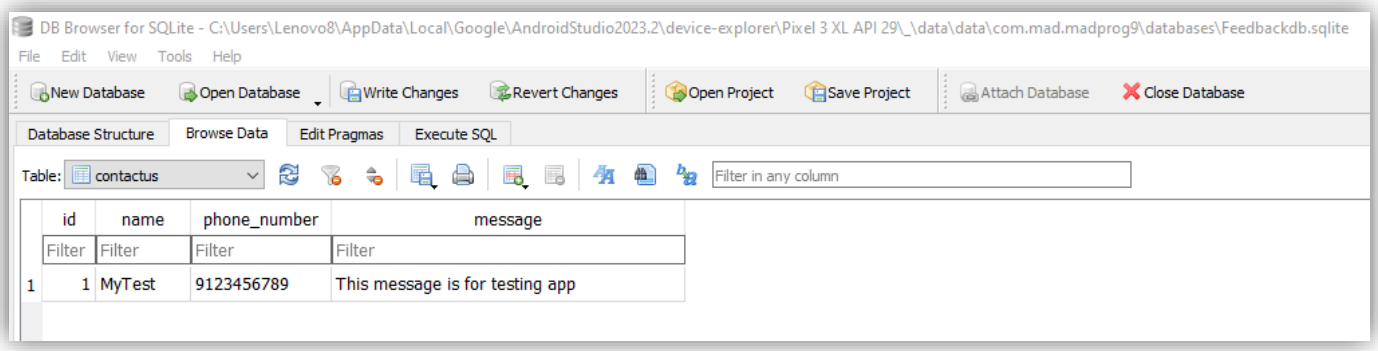
- *Search for the filepath you have saved your database*
- *Select* ***Feedbackdb.sqlite***



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



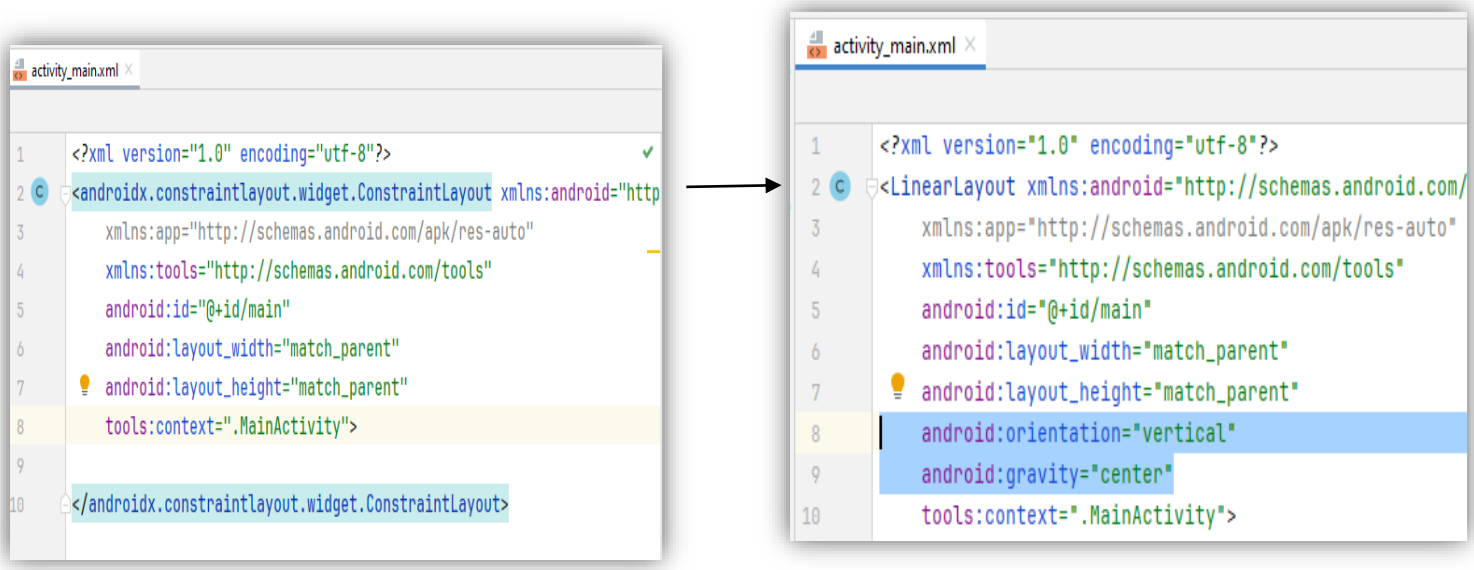
14. Now the database will be displayed as shown below



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

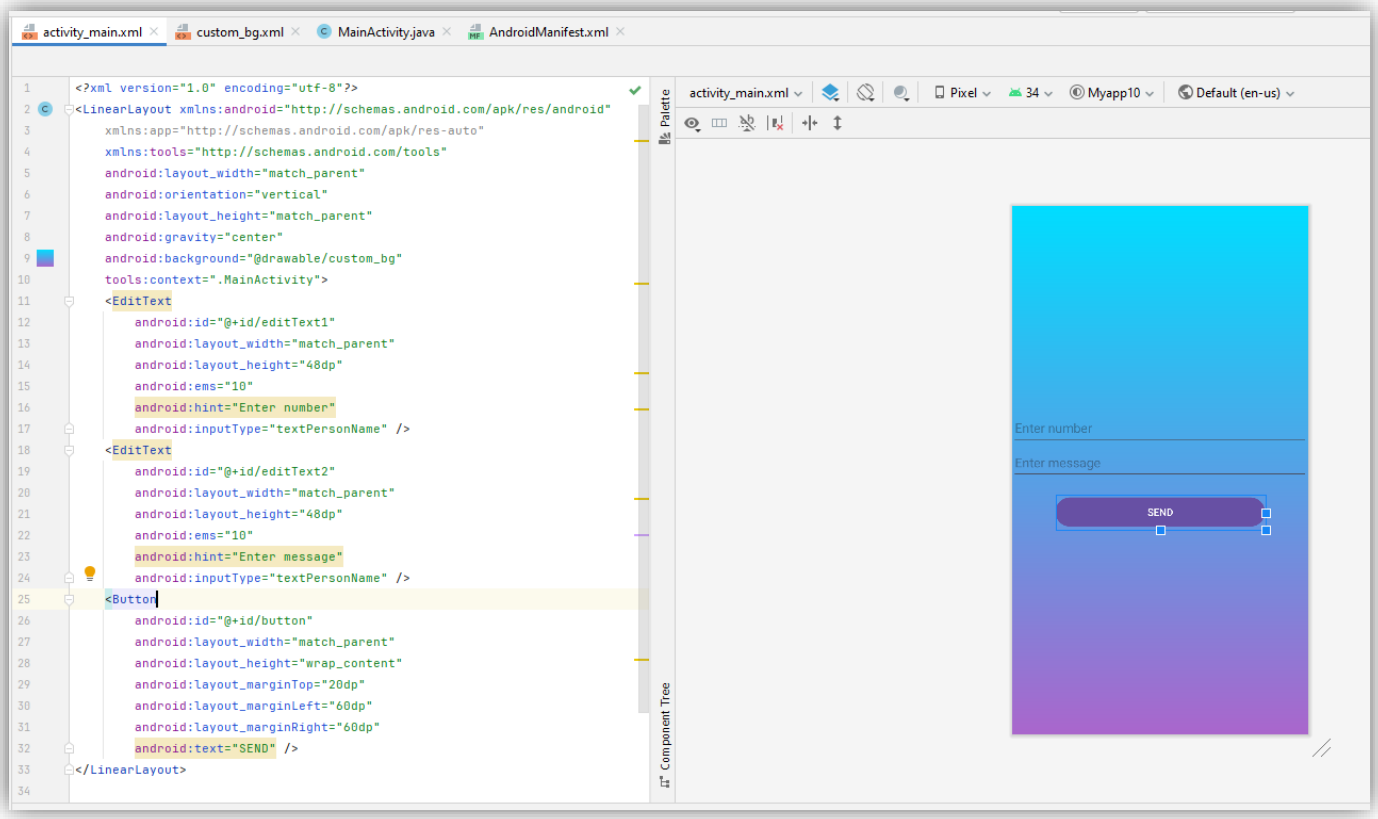
1. Create a **new project** and choose **Empty Views Activity** then click **Next**.
   Change the layout from constraint Layout to Relative Layout in **activity_main.xml** file.
   <androidx.constraintlayout.widget.ConstraintLayout..
   To
   <LinearLayout..

   update the following code in the linearlayout tag.
   android:orientation="vertical"
   android:gravity=”center”

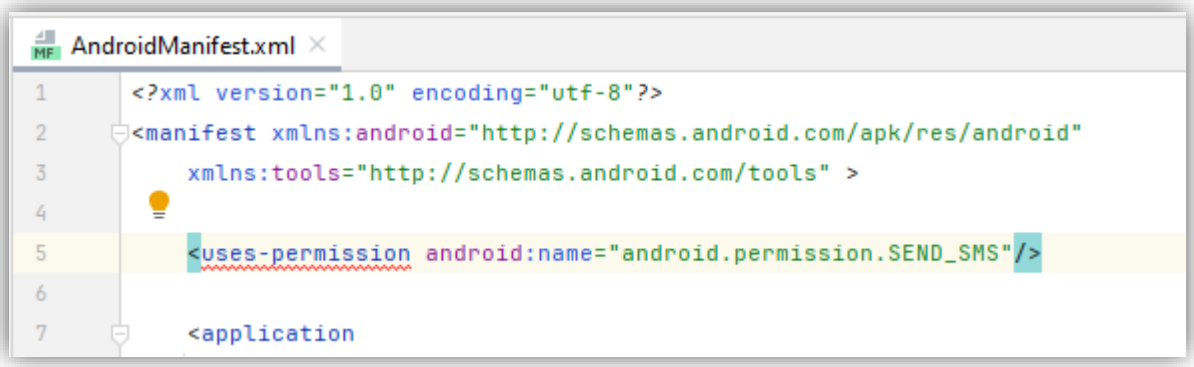


2. Add two Edit view, then edit the content in **activity_main.xml** code file as follows:

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" */>*



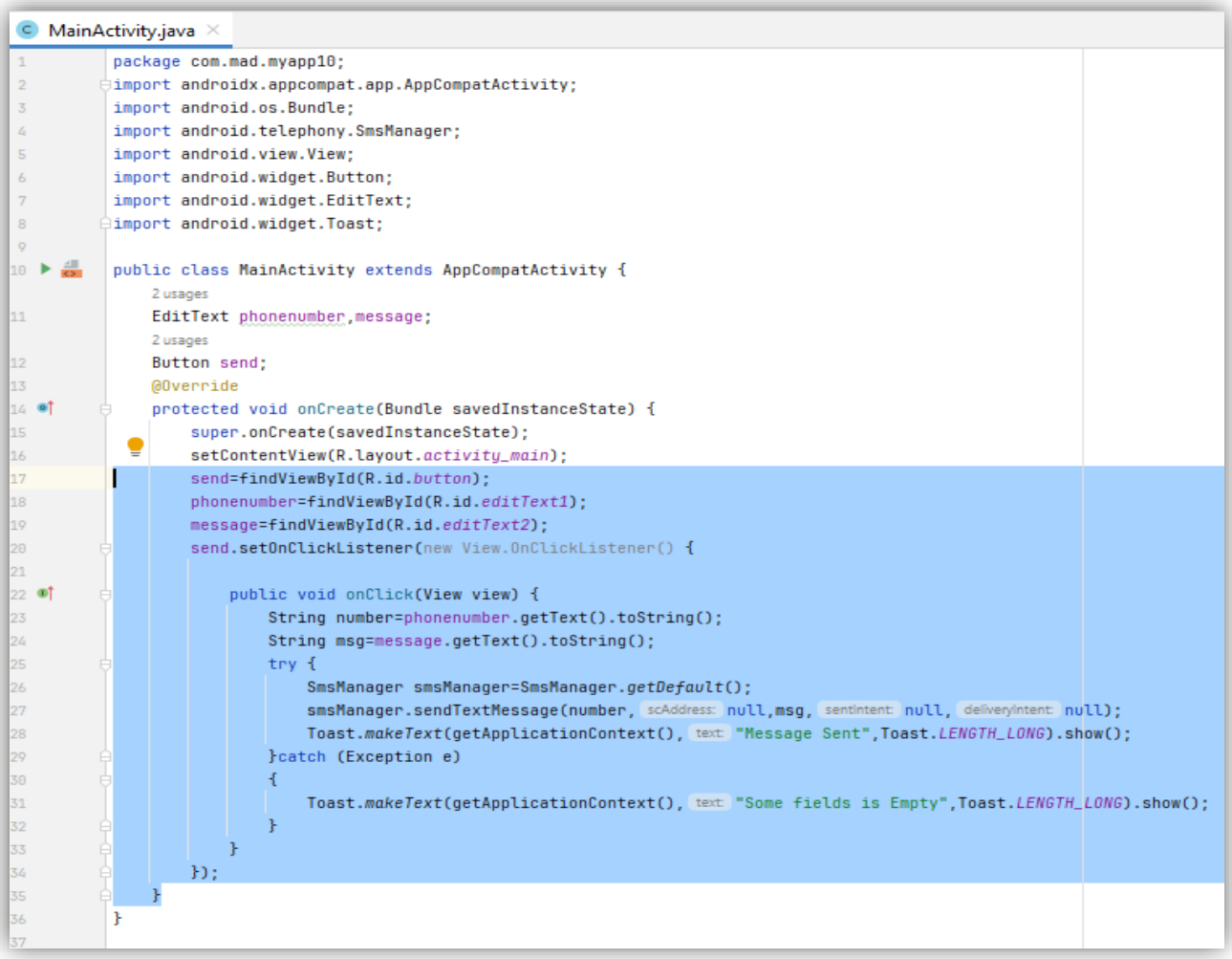
5. In ***AndroidManifest.xml*** add the permission to send SMS. It will permit an android application to send SMS.
   <uses-permission android:name=" android.permission.SEND_SMS " />



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

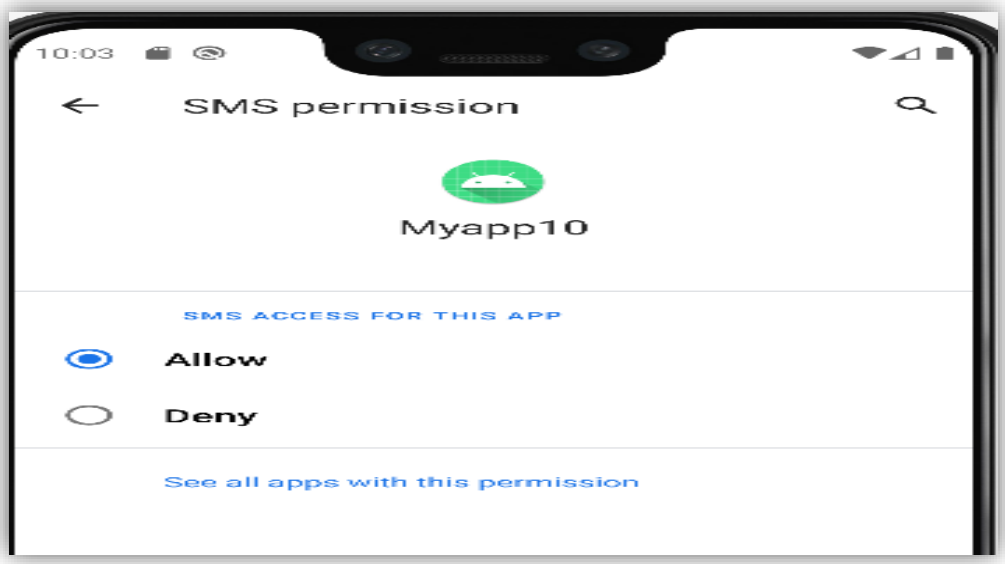
```
        send=findViewById(R.id.button);
        phonenumber=findViewById(R.id.editText1);
        message=findViewById(R.id.editText2);
        send.setOnClickListener(new View.OnClickListener() {

            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();
                }
            }
        });
    }
}
```

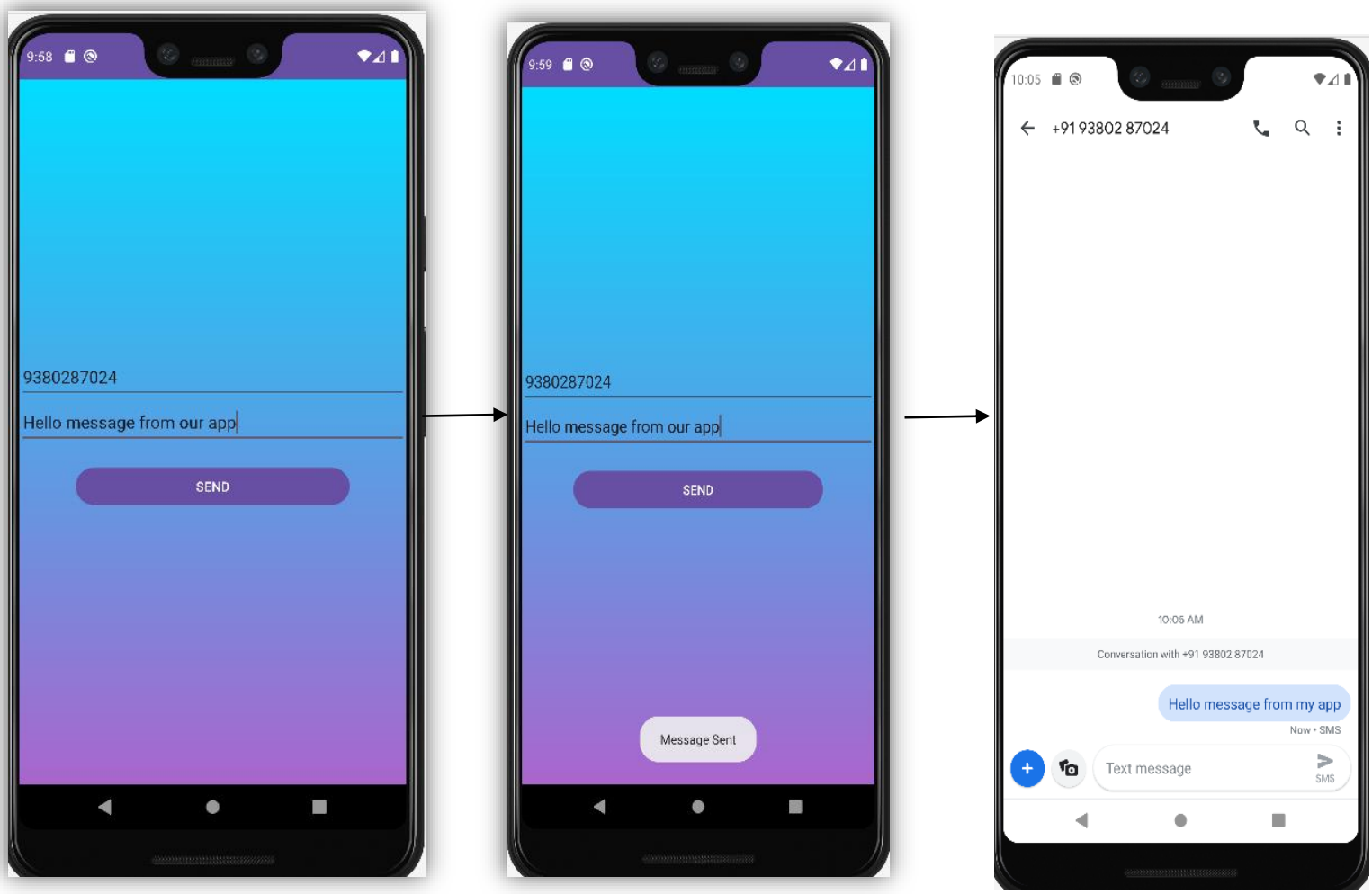


7. Connect USB to the physical mobile -> Go to developer option -> enable USB debugger , install via USB and USB debugger(security settings)

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..

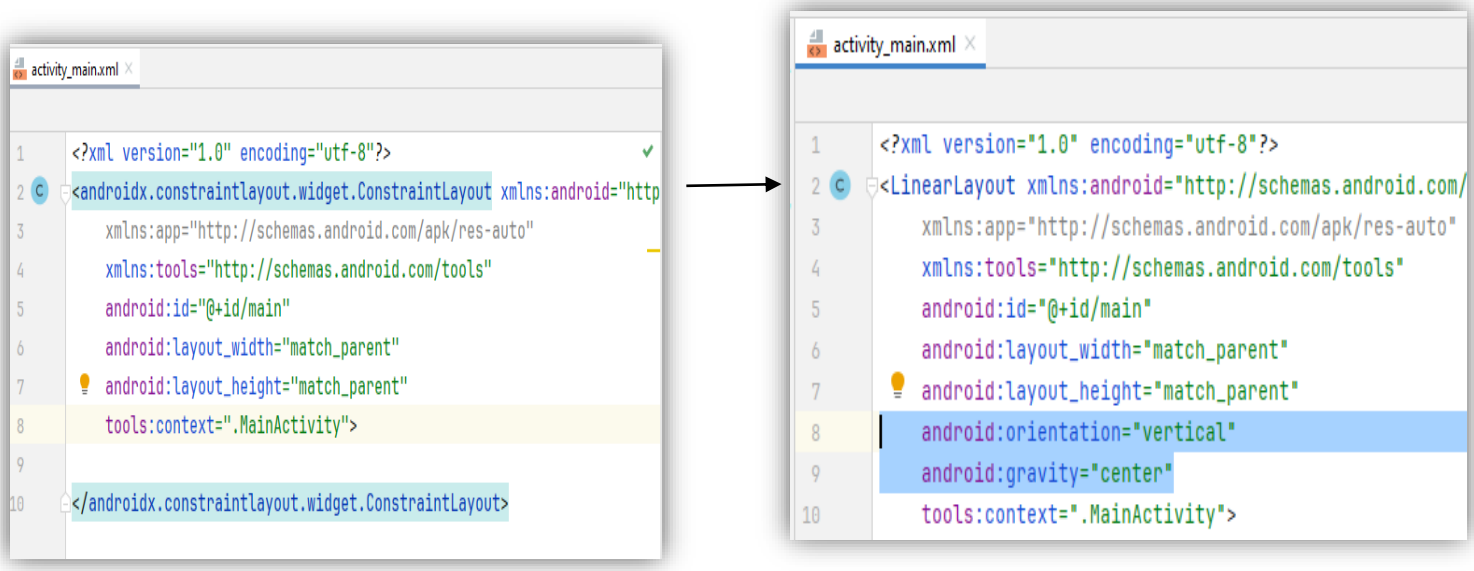
To

<LinearLayout..

update the following code in the linearlayout tag.

android:orientation="vertical"

android:gravity=”center”



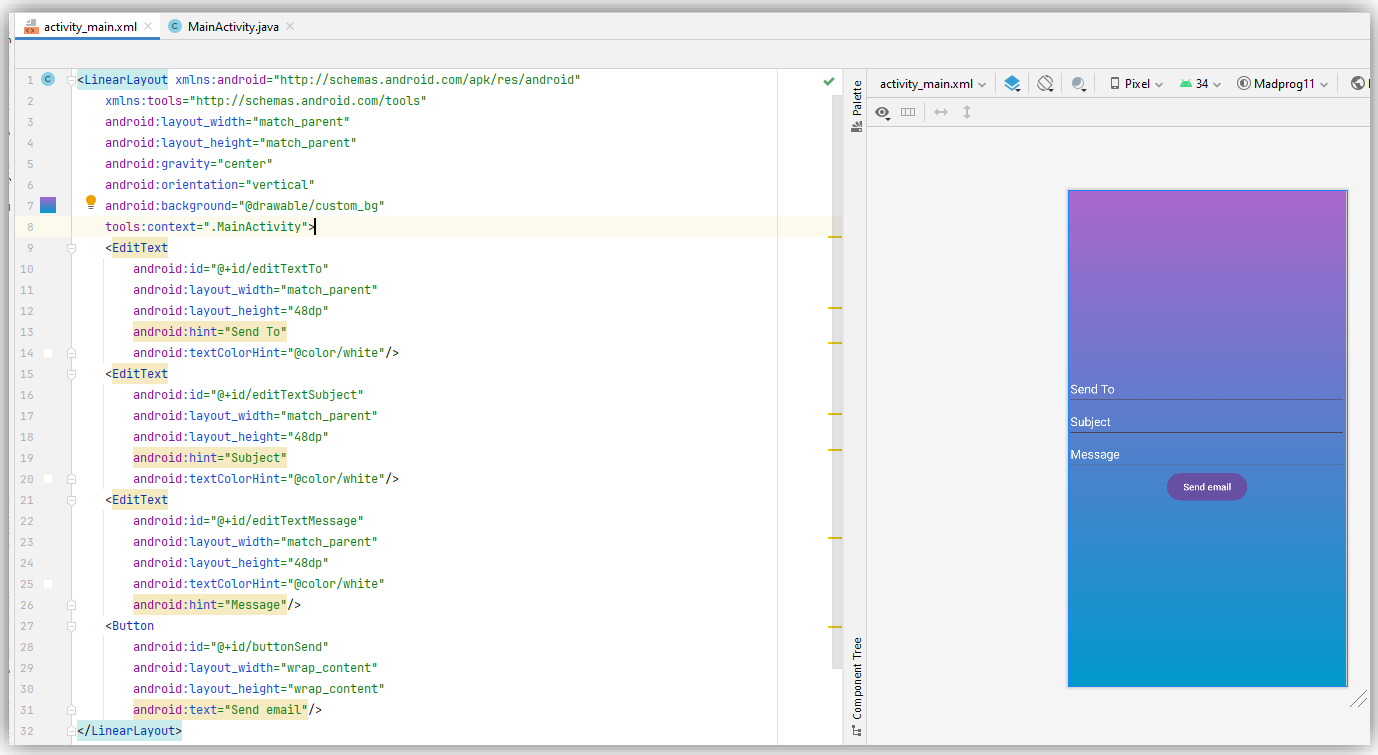
2. Add three Edit view, then edit the content in **activity_main.xml** code file as follows:

```
<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

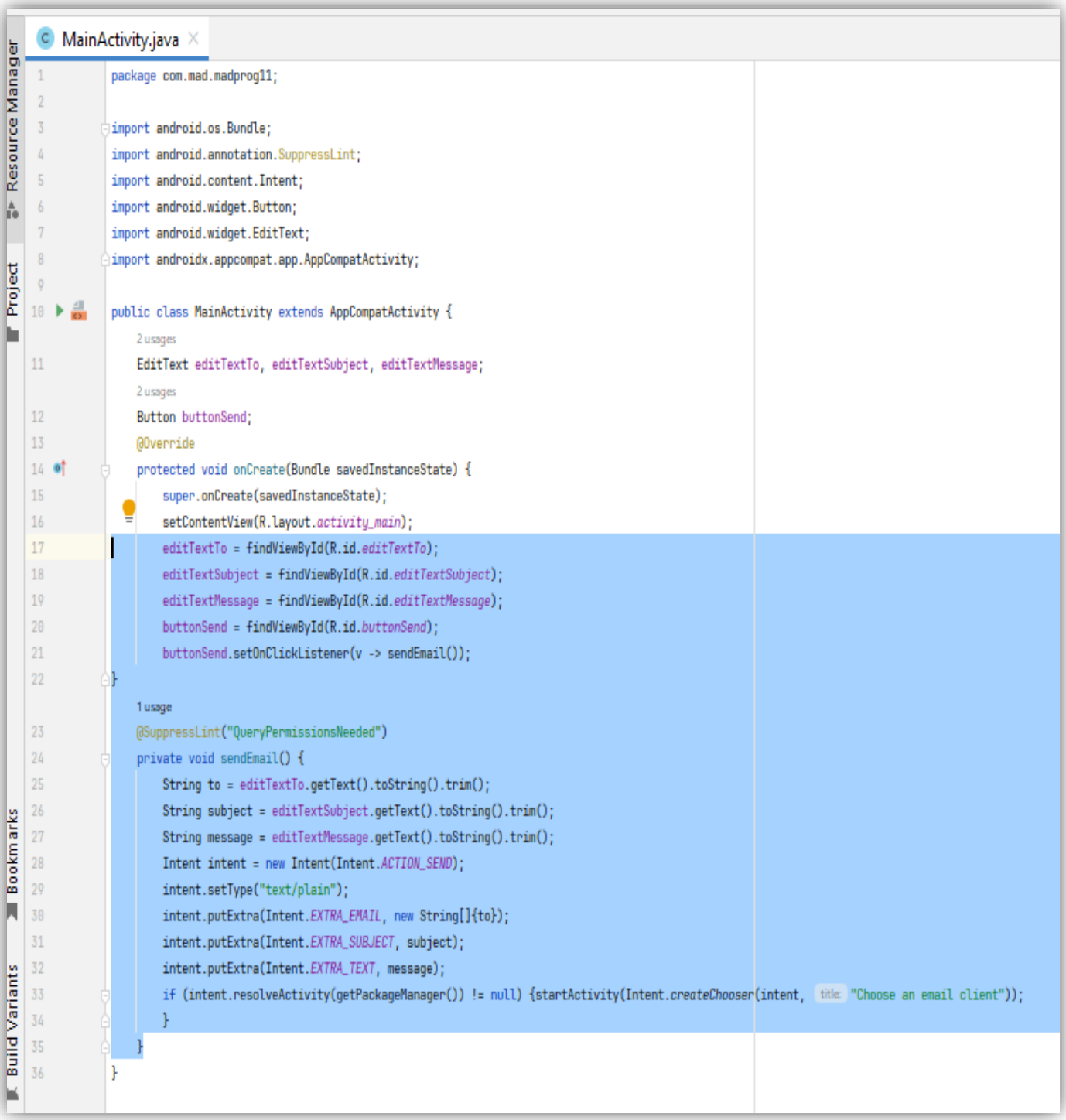
- Import necessary libraries

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

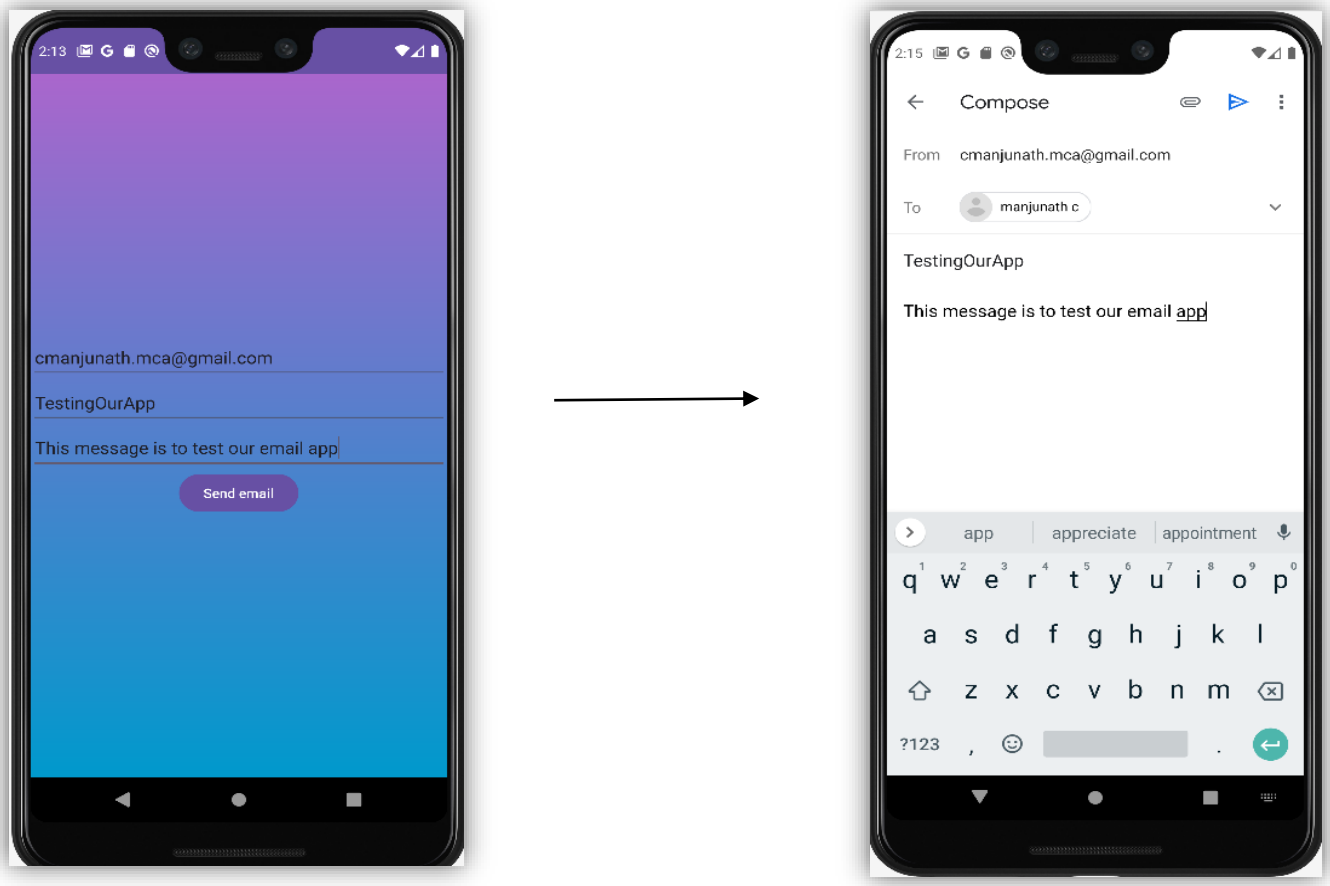
- Update the following code for creating method to send email

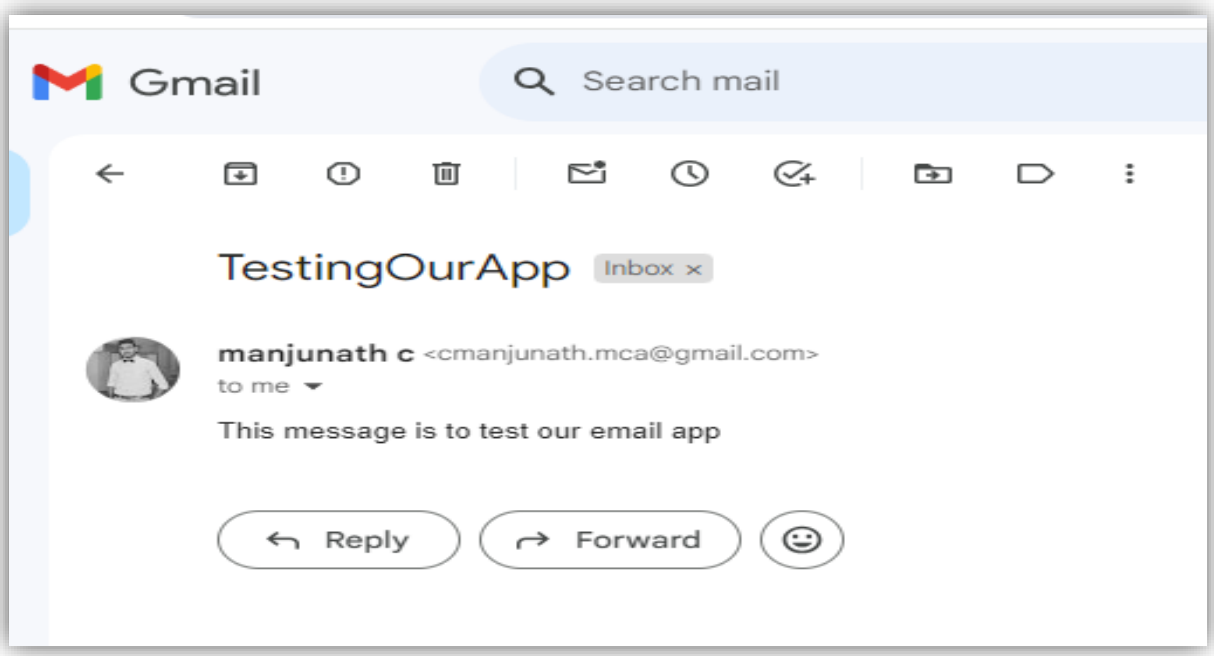
```
public class MainActivity extends AppCompatActivity {
    EditText editTextTo, editTextSubject, editTextMessage;
    Button buttonSend;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        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(getPackageManager()) != null)
{startActivity(Intent.createChooser(intent, "Choose an email client"));
        }
    }
}
```



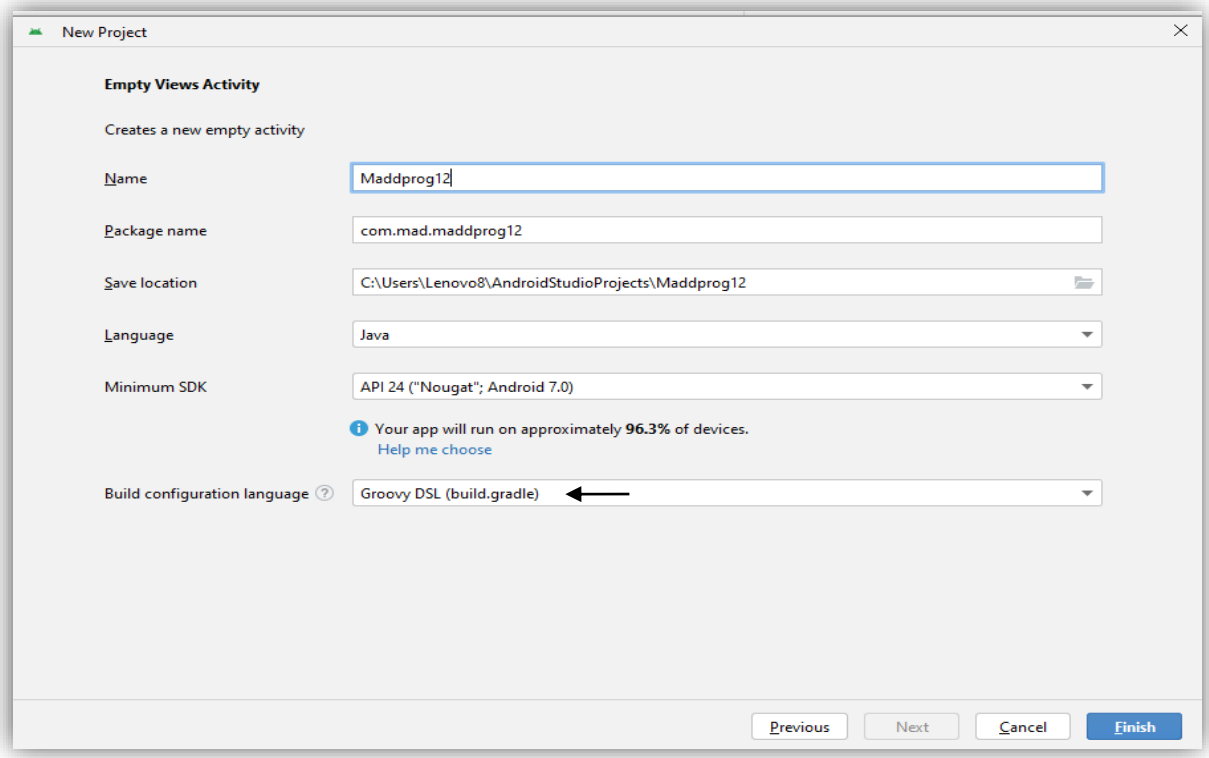
5. Press **Shift F10** to run the application Or, Press the run button on top of the interfac



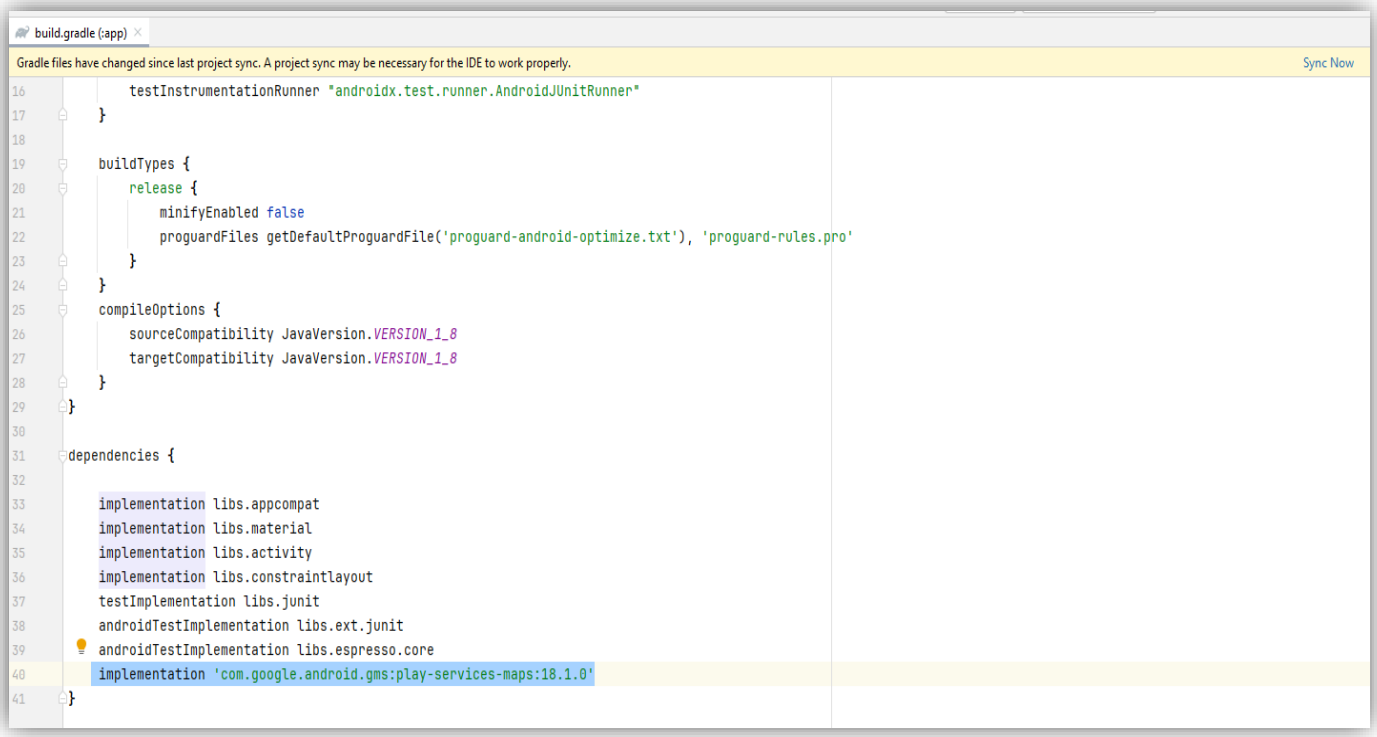


12. Display Map based on the Current/given location.

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

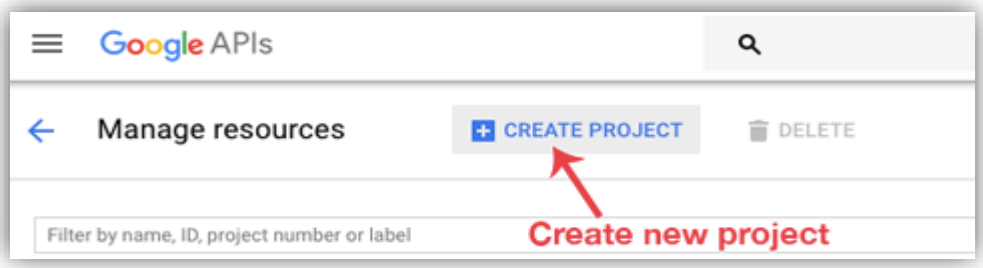


3. Get an API key from google cloud

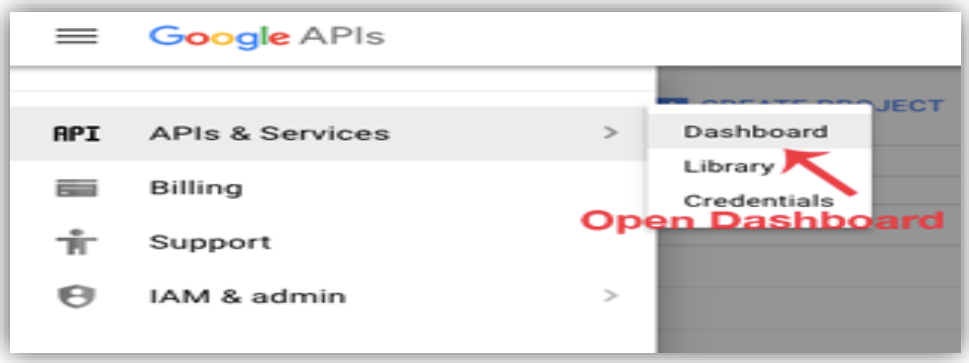
←----------***NOT TO BE WRITTEN IN RECORD***----------→

- Open Google developer console and signin with your gmail account: https://console.developers.google.com/project

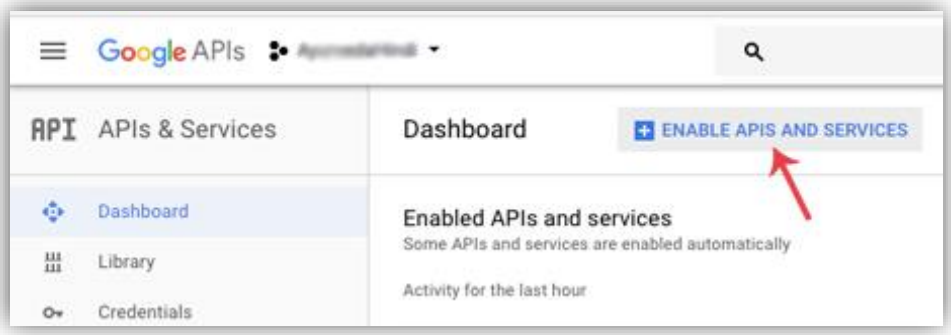
- Now create new project. You can create new project by clicking on the **Create Project** button and give name to your project.



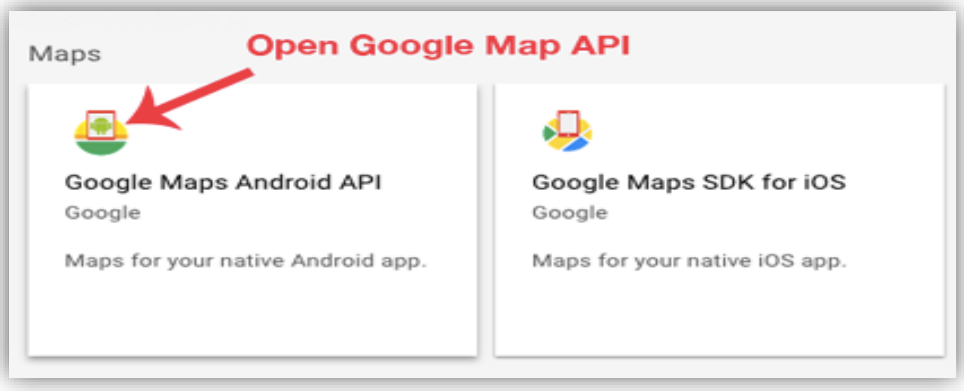
- Now click on APIs & Services and open **Dashboard** from it.



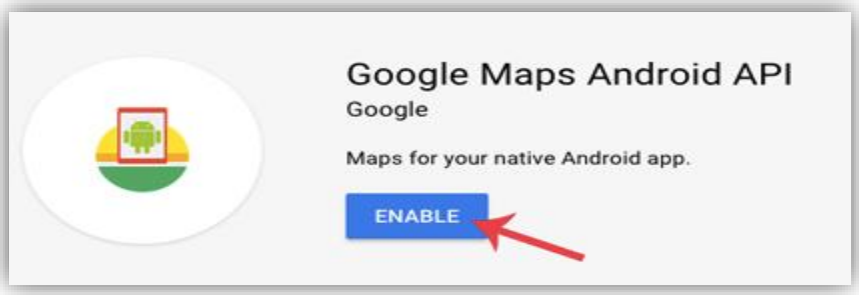
- In this open **Enable APIS AND SERICES**.



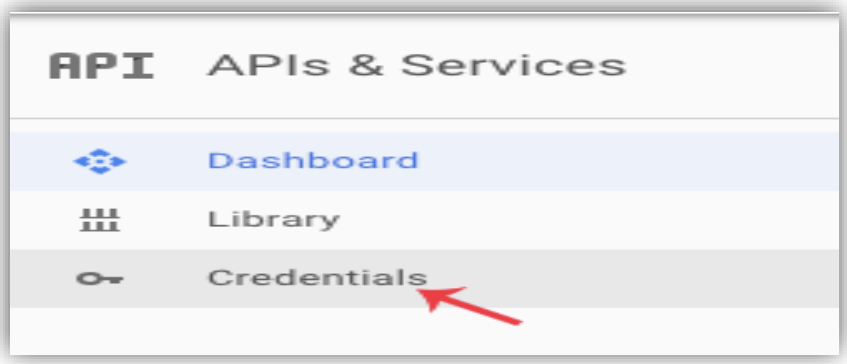
- Now open Google Map Android API



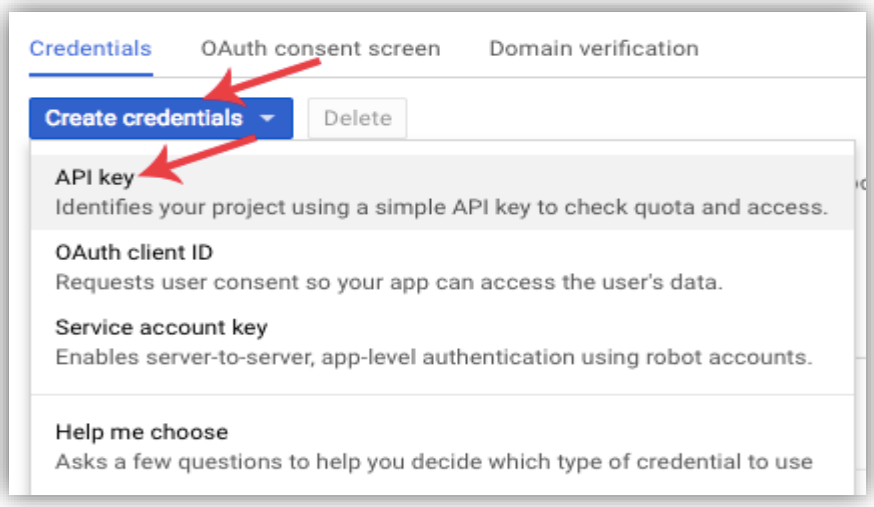
- Now enable the Google Maps Android API.



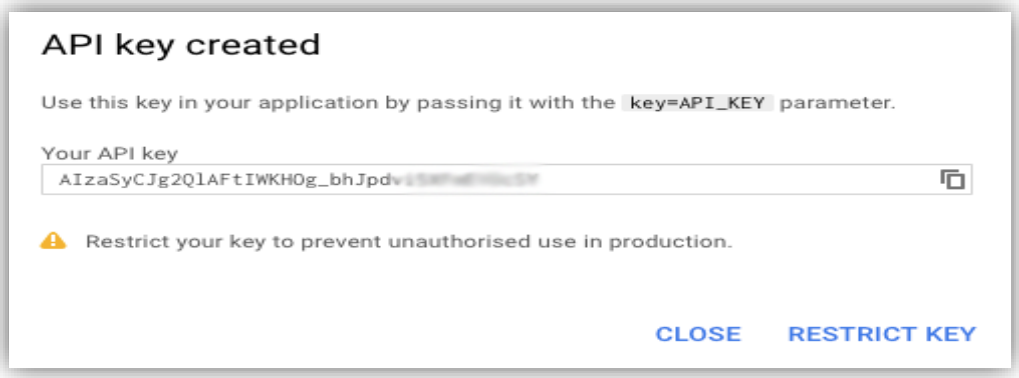
- Now go to **Credentials**



- 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.



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

<meta-data

android:name="com.google.android.geo.API_KEY"

android:value="AIzaSyCC3r3o_EzhXPj-mkMRxY2VFjokf4iE1R8" />

AndroidManifest.xml

```
<?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

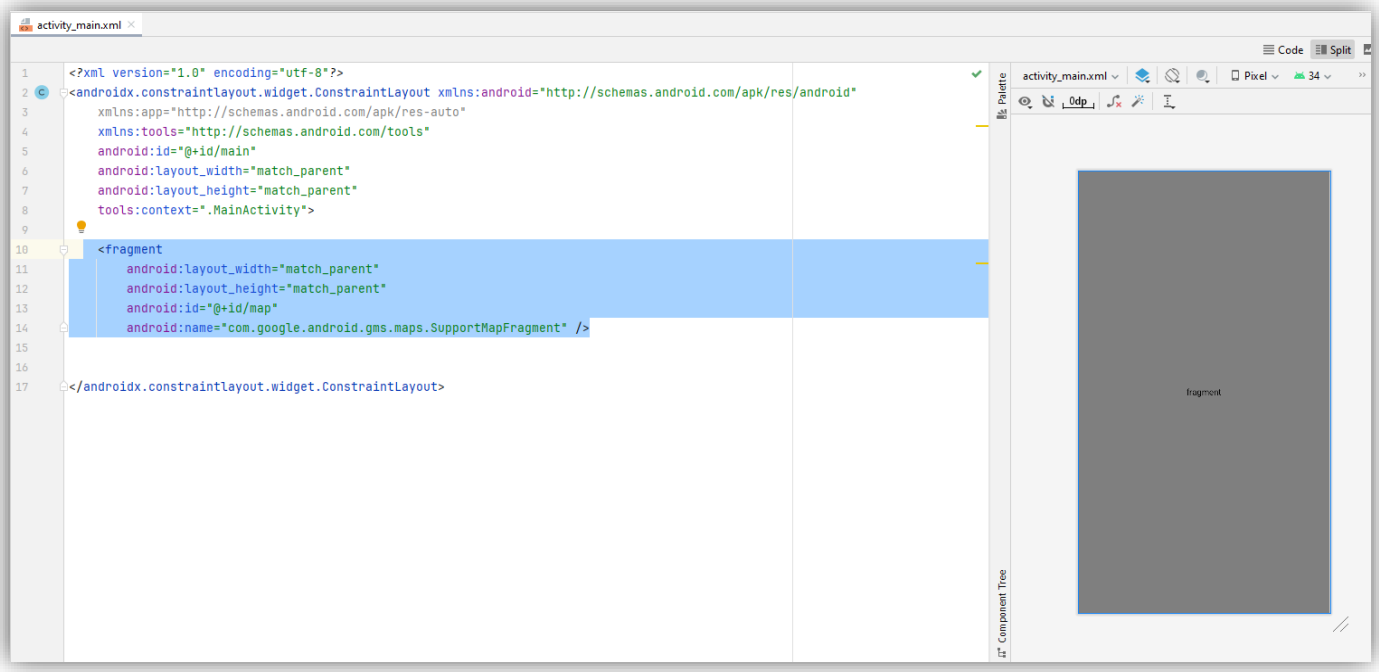
<fragment

android:layout_width="match_parent"

android:layout_height="match_parent"

android:id="@+id/map"

android:name="com.google.android.gms.maps.SupportMapFragment" />



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;
```

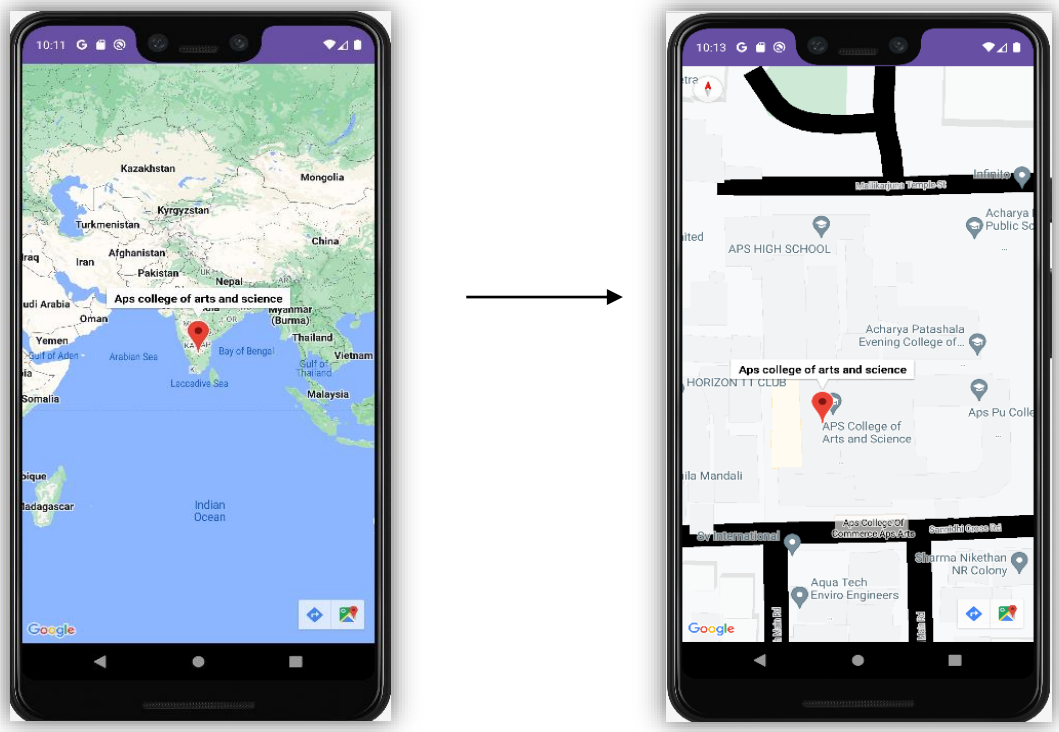
- Then update the method for current location

```
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));
   }
```

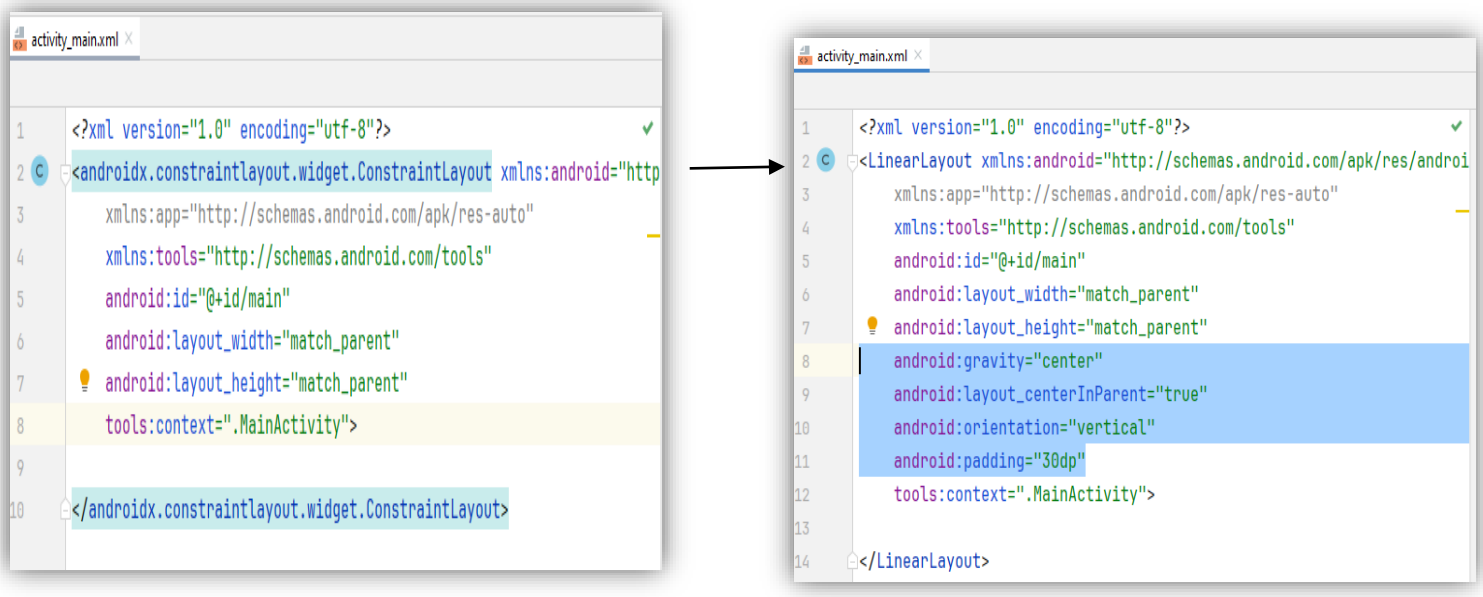
7. Press **Shift F10** to run the application Or, Press the run button on top of the interface



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"



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"**/>*

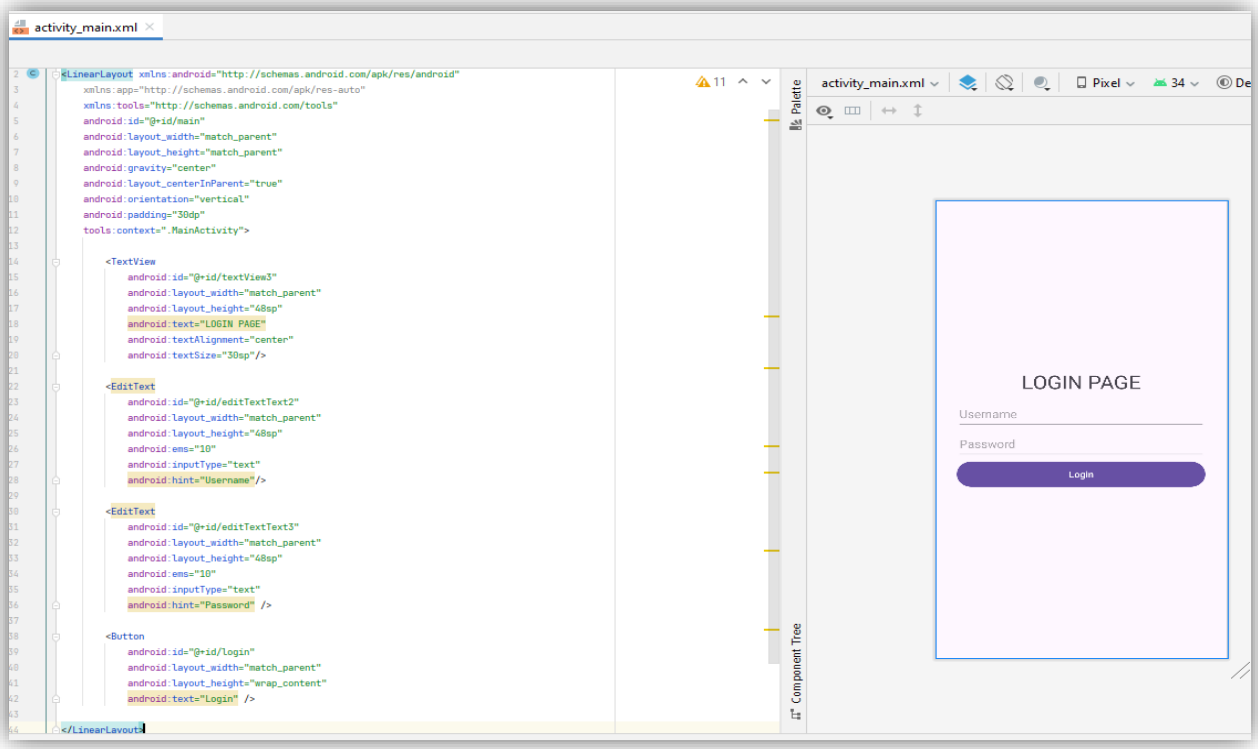
6. Add two edit text one after the other for username and password

<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;

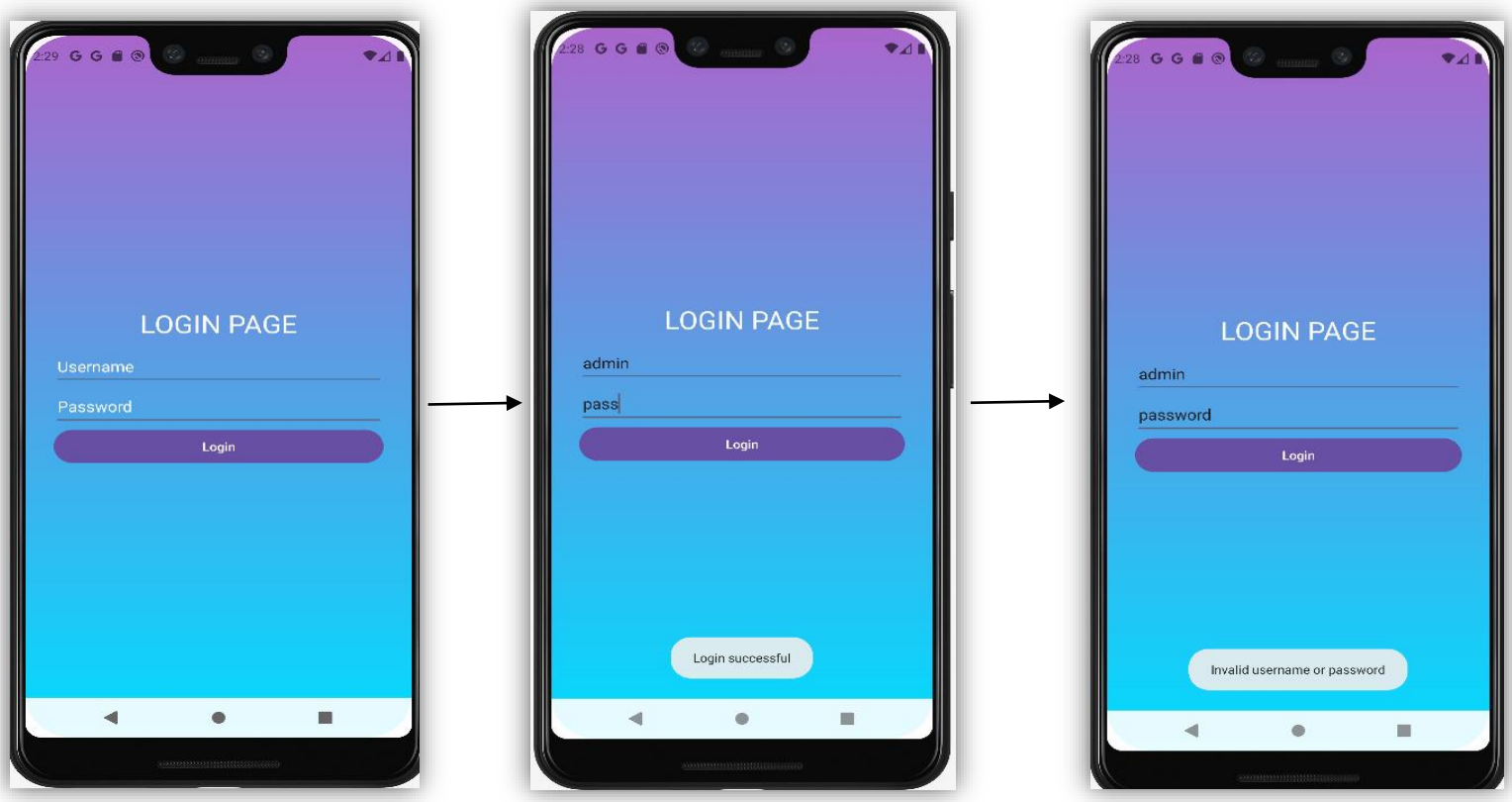
- Update the following code for creating method to validation

```
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

```java
package com.mad.madprog14;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    2 usages
    EditText etUsername,etPassword;
    2 usages
    Button btnLogin;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable( $this$enableEdgeToEdge: this);
        setContentView(R.layout.activity_main);
        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( context: MainActivity.this, text: "Login successful", Toast.LENGTH_SHORT).show();
                }
                else
                {
                    Toast.makeText( context: MainActivity.this, text: "Invalid username or password", Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}
```

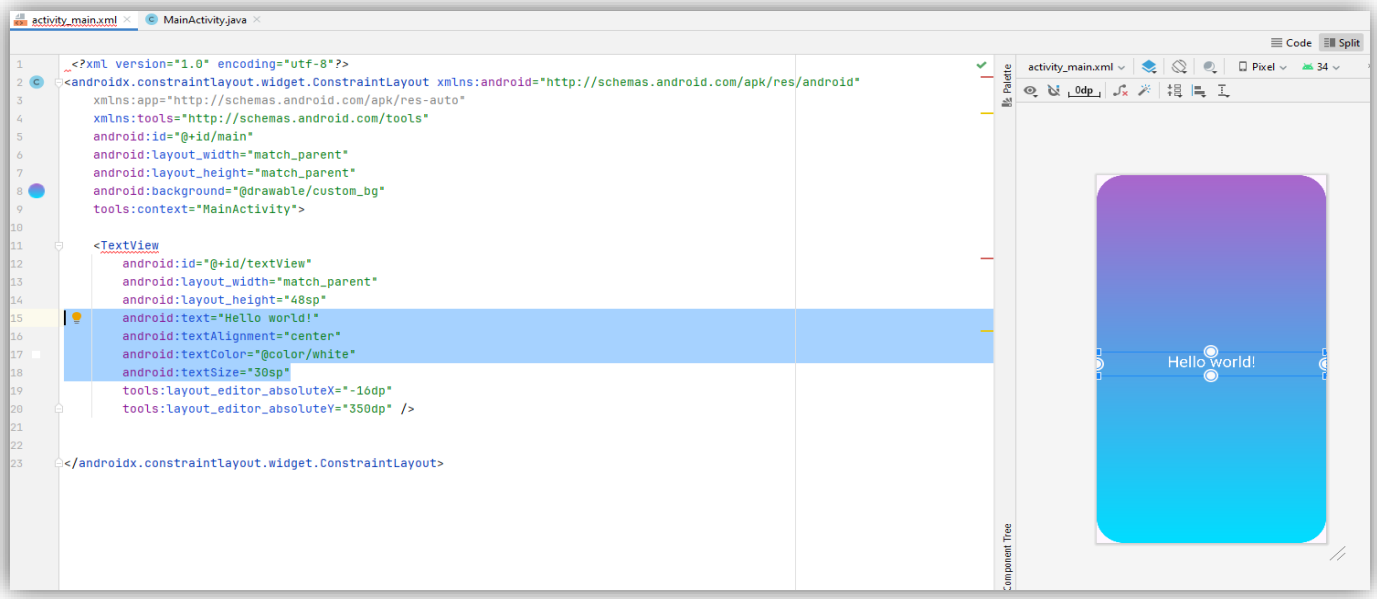
11. Press **Shift F10** to run the application Or, Press the run button on top of the interface



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