SIDDHI S. KOTRE
ROLL NO: 50

MOBILE COMPUTING
ASSIGNMENT 1

1. **Aim :** Write a program to demonstrate activity life cycle.

Objective :

The activity life cycle in Android refers to the series of states and transitions that an Activity undergoes from creation to destruction. Understanding these states is crucial for managing resources, handling user interactions, and ensuring a smooth user experience. Key lifecycle methods include onCreate(), onStart(), onResume(), onPause(), onStop(), and onDestroy(), each representing different stages of the Activity's lifecycle.

Theory:

- To illustrate the different stages of the Activity life cycle in an Android application.
- To demonstrate how lifecycle methods can be used to manage resources and maintain application state.
- To provide practical examples of how to handle transitions between lifecycle states effectively.

Code :

Activity.xml

```
<?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"
  tools:context=".MainActivity">

  <LinearLayout
     android:layout_width="match_parent"
     android:layout_height="match_parent">

     <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Siddhi" />
  </LinearLayout>
</androidx.constraintlayout.widget.ConstraintLayout>
```

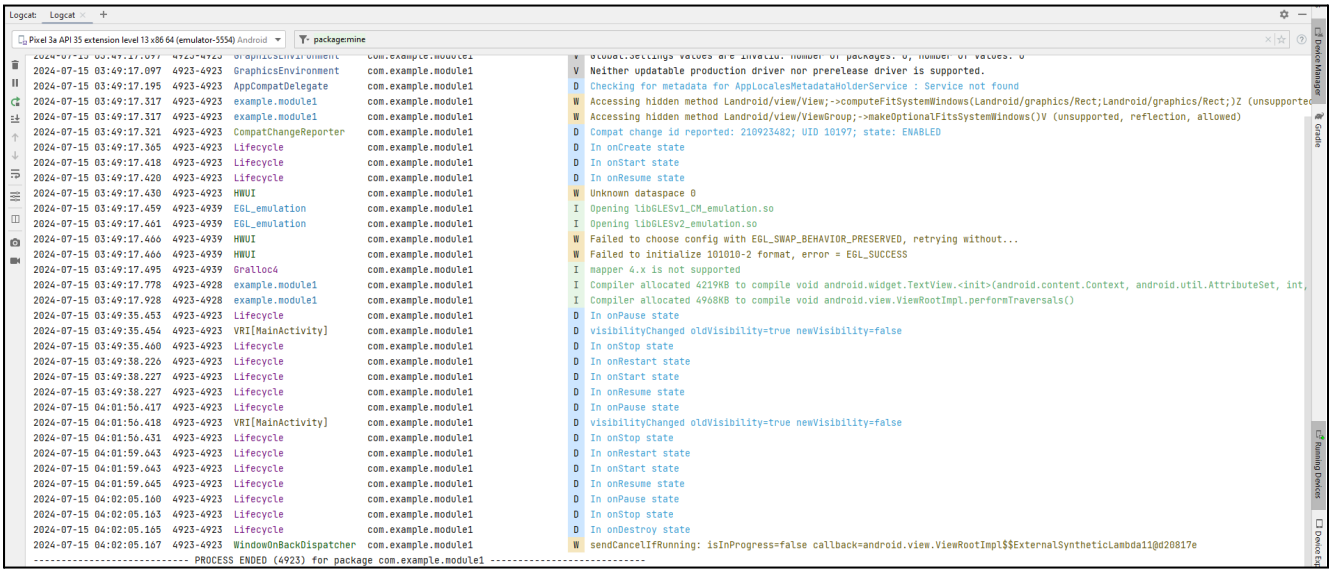
MainActivity.java

```
package com.example.module1;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
public class MainActivity extends AppCompatActivity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);
     Log.d("Lifecycle","In onCreate state");
  }
  @Override
  protected void onStart(){
     super.onStart();
     Log.d("Lifecycle","In onStart state");
  }
  @Override
  protected void onResume(){
     super.onResume();
     Log.d("Lifecycle","In onResume state");
  }
  @Override
  protected void onPause(){
     super.onPause();
     Log.d("Lifecycle","In onPause state");
  }
  @Override
  protected void onRestart(){
     super.onRestart();
     Log.d("Lifecycle","In onRestart state");
  }
  @Override
  protected void onStop(){
     super.onStop();
     Log.d("Lifecycle","In onStop state");
  }
  @Override
  protected void onDestroy(){
     super.onDestroy();
     Log.d("Lifecycle","In onDestroy state");
  }

}
```

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="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/Theme.Module1"
    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>
  </application>
</manifest>
```

Output :



2. **Aim :** Design the following User Interface using Linear Layout and Explicit Intent.

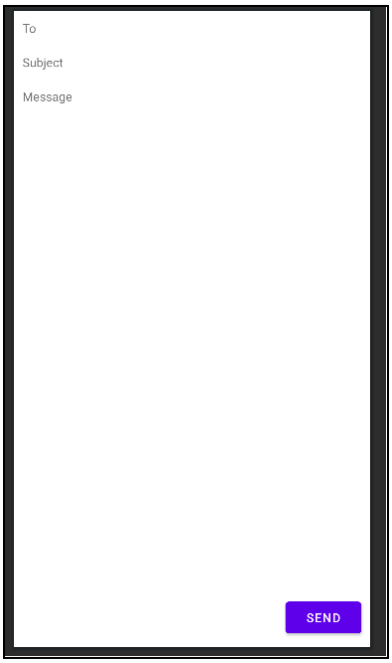
Objective :
Linear Layout is a view group in Android that arranges its child views in a single horizontal or vertical row. It is particularly useful for designing simple and straightforward UIs where elements need to be aligned in a linear fashion. Explicit Intent, on the other hand, is used to start a specific Activity within the same application or in another application, allowing for clear navigation between different components. Using these tools together allows developers to create organized and navigable interfaces.

Theory:
- To design a user interface using Linear Layout to arrange UI components in a linear sequence.
- To implement Explicit Intent to navigate between different Activities within the application.
- To demonstrate the integration of Linear Layout and Explicit Intent to create a functional and user-friendly application flow.

Code :
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
  tools:context=".MainActivity">


  <TextView
    android:id="@+id/To"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:text="To" />

  <TextView
    android:id="@+id/Subject"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:text="Subject" />
```

```
  <TextView
     android:id="@+id/Message"
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:layout_margin="10dp"
     android:text="Message" />
  <TextView
     android:id="@+id/Data"
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:layout_margin="10dp"
     android:gravity="bottom"
     android:layout_weight="1"
     android:text="" />
  <Button
     android:id="@+id/send"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:layout_margin="10dp"
     android:layout_gravity="right"
     android:text="Send" />
</LinearLayout>
```

Output :



3. **Aim :** Design the following User Interface using Relative Layout.

Objective :
Relative Layout is a versatile view group in Android that enables you to position and size child views relative to each other or to the parent container. This layout allows for more complex and flexible arrangements compared to Linear Layout, as it lets you specify rules for how views should align, such as aligning with the top, bottom, start, or end of other views or the parent. By using Relative Layout, developers can create dynamic and responsive UIs that adapt to different screen sizes and orientations.

Theory:

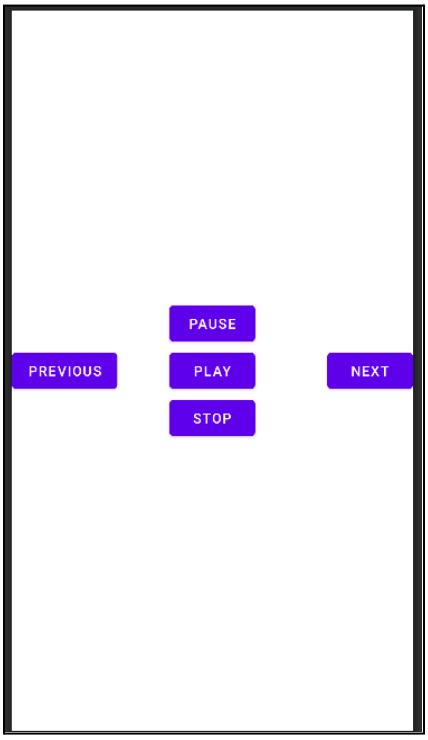
- To design a user interface using Relative Layout to position and align UI components relative to each other or to the parent view.
- To leverage the flexibility of Relative Layout to create a more adaptable and responsive UI compared to linear arrangements.
- To demonstrate how to use Relative Layout attributes and rules to achieve a complex and visually appealing design.

Code :

```
<?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:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".Relative_Activity">

  <Button
    android:id="@+id/Play"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerInParent="true"
    android:text="Play"
    />

  <Button
    android:id="@+id/Pause"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_above="@id/Play"


    android:text="Pause"
```

```
    />
  <Button
     android:id="@+id/Stop"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:layout_below="@id/Play"
     android:layout_centerHorizontal="true"
     android:text="Stop"
     />

  <Button
     android:id="@+id/Previous"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:text="Previous"
     android:layout_centerVertical="true"
     android:layout_alignParentLeft="true"

     />

  <Button
     android:id="@+id/Next"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:layout_centerVertical="true"
     android:layout_alignParentRight="true"
     android:text="Next"
    />


</RelativeLayout>
```

Output :



4. **Aim :** Design the following User Interface using TableLayout.

Objective :
TableLayout is a view group in Android that arranges child views in a grid-like format of rows and columns. It provides an easy way to create structured layouts similar to tables, where you can align elements horizontally and vertically. This layout is ideal for organizing UI components that need to be aligned in a tabular format.

Theory:

- To design a user interface using TableLayout for structured alignment of UI elements.
- To utilize rows and columns to organize components in a grid format.
- To demonstrate how TableLayout can simplify the design of complex, grid-based UIs.

Code :

```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout 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"
  tools:context=".Table_Activity">
  <!--Row 1-->
<TableRow
  android:orientation="vertical"
  android:padding="10dp">
<TextView
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:layout_weight="1"
  android:text="ICC RANKING OF PLAYERS"
  android:gravity="center"
  ></TextView>
</TableRow>

  <!--Row 2-->
  <TableRow
     android:orientation="horizontal"
     android:padding="10dp">
     <TextView
        android:text="Rank"></TextView>
     <TextView
        android:text="Name"></TextView>
     <TextView
        android:text="Team"></TextView>
```

```
      <TextView
         android:text="Points"></TextView>
   </TableRow>

   <!--Row 3-->
   <TableRow
      android:orientation="horizontal"
      android:padding="10dp">
<TextView
   android:text="1"></TextView>
<TextView
   android:text="Kane Williamson"></TextView>
<TextView
   android:text="NZ"></TextView>
<TextView
   android:text=" 102"></TextView>


   </TableRow>



   <!--Row 4-->
   <TableRow
      android:orientation="horizontal"
      android:padding="10dp">
      <TextView
         android:text="2"></TextView>
      <TextView
         android:text="Ben Stokes"></TextView>
      <TextView
         android:text="ENG"></TextView>
      <TextView
         android:text=" 134"></TextView>
   </TableRow>

   <!--Row 5-->
   <TableRow
      android:orientation="horizontal"
      android:padding="10dp">

         <TextView
            android:text="3"></TextView>
         <TextView
```

```
            android:text="AB de villers"></TextView>
        <TextView
            android:text="SA"></TextView>
        <TextView
            android:text=" 1234"></TextView>
    </TableRow>

</TableLayout>
```

Output :

5. **Aim :** Write a program to take user input from one activity and display it in another activity

Objective :
In Android, data can be passed between activities using Intents, which are messaging objects used to request an action from another component. The Intent object can carry data in the form of key-value pairs using a Bundle. By leveraging this mechanism, one activity can send data to another activity, which can then retrieve and display this data. This approach facilitates communication within the app and helps in building multi-screen user interfaces.

Theory:
To capture user input from an EditText in the first activity.
To use Intent and Bundle to pass the input data to a second activity.
To display the received data in a TextView in the second activity.

Code :

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
  tools:context=".MainActivity">

  <EditText
    android:id="@+id/Name"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ems="10"
    android:inputType="text"
    android:layout_marginTop="10dp"
    android:layout_marginHorizontal="10dp"
    android:hint="Full Name" />

  <EditText
    android:id="@+id/Email"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ems="10"
    android:inputType="text"
    android:layout_marginHorizontal="10dp"
    android:hint="E-mail Address" />
```

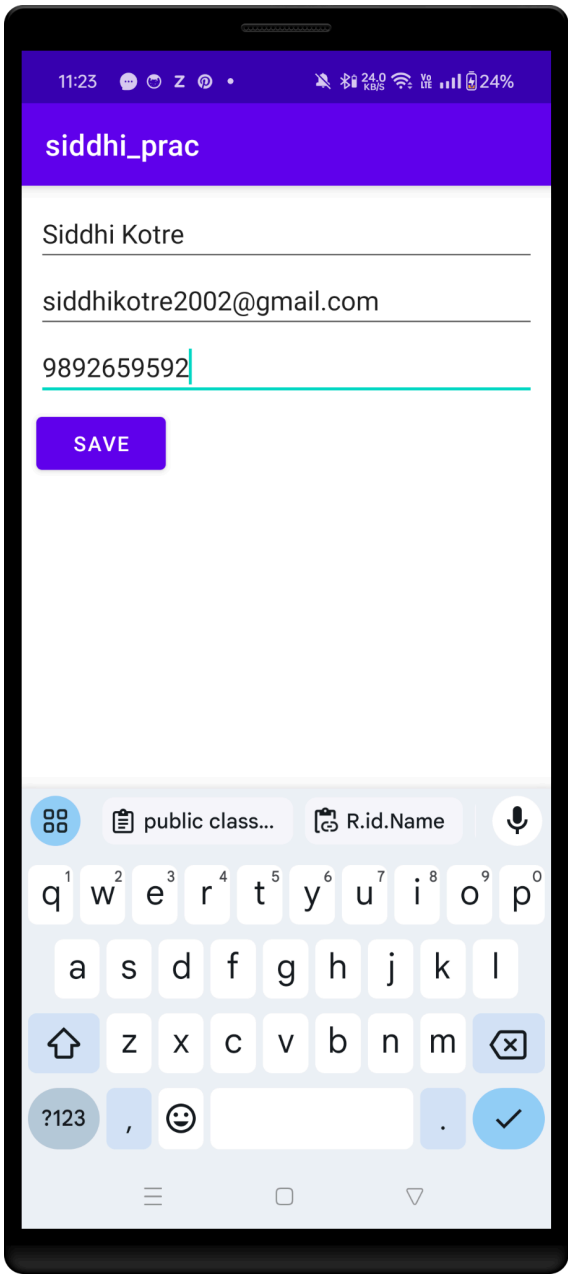
```
  <EditText
     android:id="@+id/Contact"
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:ems="10"
     android:inputType="text"
     android:layout_marginHorizontal="10dp"
     android:hint="Phone No." />

  <Button
     android:id="@+id/Save"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:layout_marginTop="5dp"
     android:layout_marginHorizontal="10dp"
     android:layout_gravity="left"
     android:text="Save" />
</LinearLayout>

package com.example.module1e;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

public class MainActivity extends AppCompatActivity {

  EditText Name, Email, Contact;
  Button Save;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);

     Name = findViewById(R.id.Name);
     Email = findViewById(R.id.Email);
     Contact = findViewById(R.id.Contact);

     Save = findViewById(R.id.Save);
```

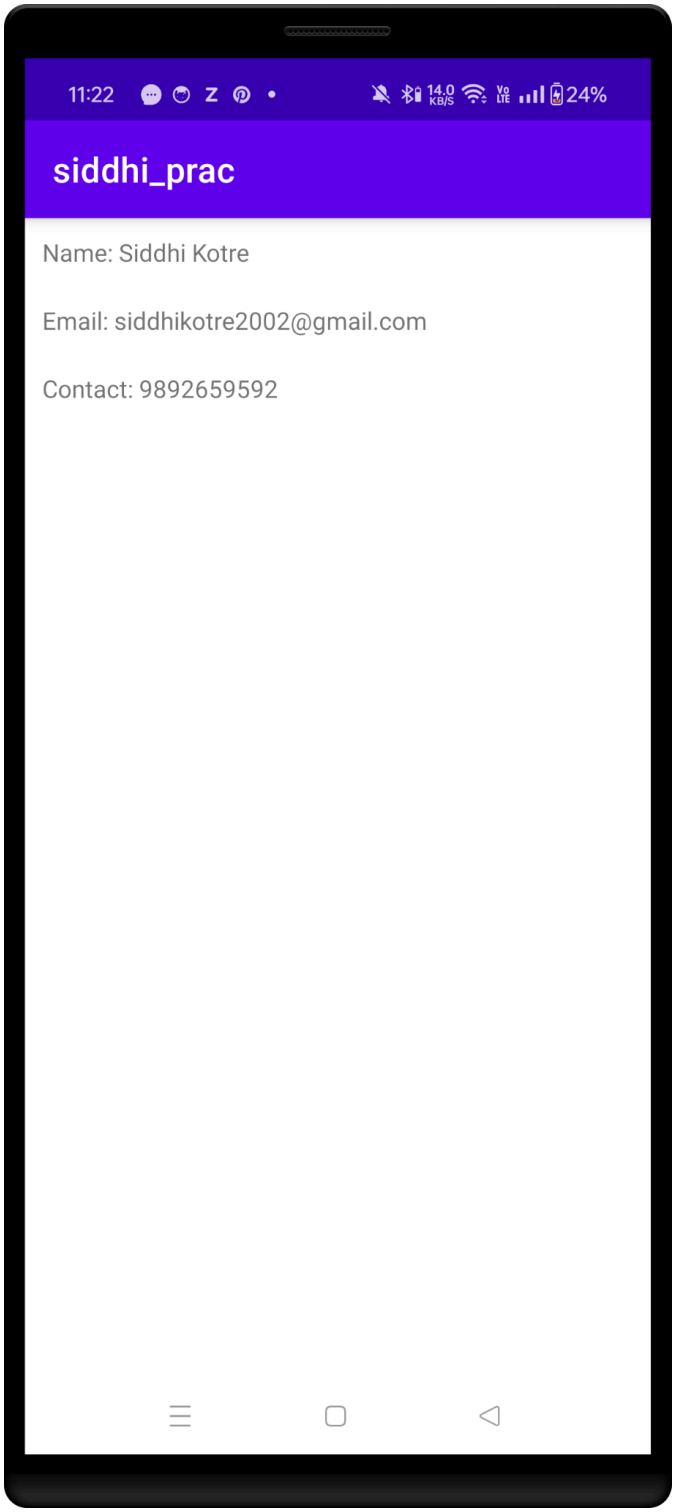
```
        Save.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String name = Name.getText().toString();
                String email = Email.getText().toString();
                String contact = Contact.getText().toString();
                Intent intent = new Intent(MainActivity.this, Display.class);
                intent.putExtra("Name", name);
                intent.putExtra("Email", email);
                intent.putExtra("Contact", contact);
                startActivity(intent);
            }
        });
    }
}
```

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    tools:context=".Display">

    <TextView
        android:id="@+id/Name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:padding="10dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/Email"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:padding="10dp"
        android:text="TextView" />

    <TextView
        android:id="@+id/Contact"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:padding="10dp"
```

```
        android:text="TextView" />
</LinearLayout>

package com.example.module1e;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.widget.TextView;

public class Display extends AppCompatActivity {

  TextView Name, Email, Contact;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_display);

      Name = findViewById(R.id.Name);
      Email = findViewById(R.id.Email);
      Contact = findViewById(R.id.Contact);

      String name = getIntent().getStringExtra("Name");
      String email = getIntent().getStringExtra("Email");
      String contact = getIntent().getStringExtra("Contact");

      Name.setText("Name: " + name);
      Email.setText("Email: " + email);
      Contact.setText("Contact: " + contact);
  }
}
```

Output:


11:23
24%
siddhi_prac
Siddhi Kotre
siddhikotre2002@gmail.com
9892659592
SAVE
public class...
R.id.Name


11:22
24%
siddhi_prac
Name: Siddhi Kotre
Email: siddhikotre2002@gmail.com
Contact: 9892659592

6. **Aim :** Write a program to demonstrate implicit intent.

Objective :
Implicit intents in Android are used to request an action from another application component without specifying the exact component to handle it. Instead, an implicit intent declares a general action to be performed, and the Android system determines the appropriate app component that can handle that action based on the intent's data and type. This mechanism facilitates interaction between different applications and allows users to choose how they want to perform an action, such as viewing a webpage or sending an email.

Theory:

- To demonstrate how to use implicit intents to request actions without specifying a particular component.
- To illustrate how Android's intent resolution mechanism allows interaction with different applications for tasks like opening web pages, sending emails, or viewing images.
- To provide practical examples of using implicit intents for common actions and handling user choices through intent resolution.

Code :

```
package com.example.demoapp;

import androidx.appcompat.app.AppCompatActivity;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);

     TextView relative =findViewById(R.id.Relative);
     TextView table = findViewById(R.id.Table);

     relative.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
           Intent relative = new Intent(MainActivity.this,Relative_Activity.class);
           startActivity(relative);
```

```
        }
    });
    relative.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Intent table = new Intent(MainActivity.this,Table_Activity.class);
            startActivity(table);
        }
    });
  }
}
```

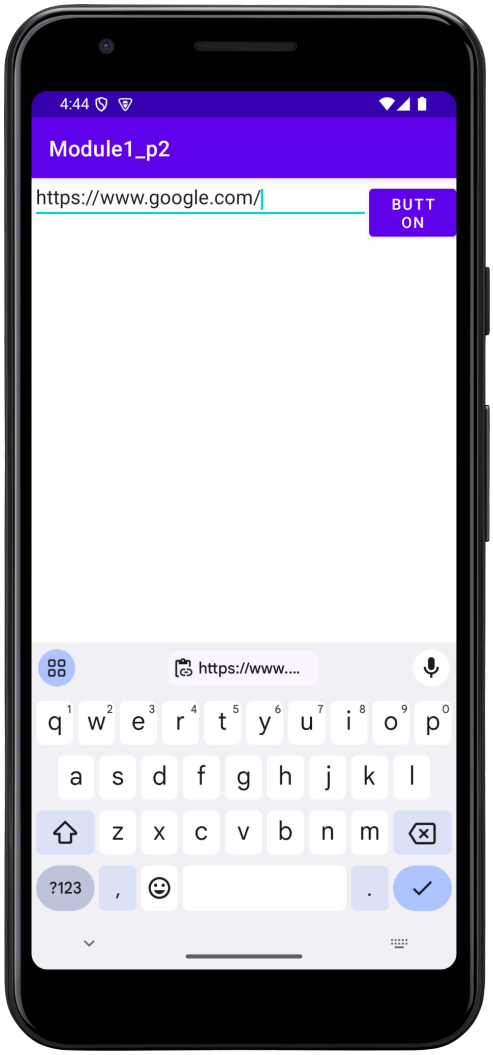
MainActivity.java

```
package com.example.module1_p2;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

public class MainActivity extends AppCompatActivity {
  EditText url;
  Button btn_search;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    url =findViewById(R.id.url_string);
    btn_search = findViewById(R.id.search);

    btn_search.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Intent i = new Intent(Intent.ACTION_VIEW, Uri.parse(url.getText().toString()));
            startActivity(i);
        }
    });
  }
}
```

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
   tools:context=".MainActivity">

   <EditText
      android:id="@+id/url_string"
      android:layout_width="312dp"
      android:layout_height="40dp"
      android:ems="10"
      android:inputType="text"
      android:text="Name"
      tools:layout_editor_absoluteX="34dp"
      tools:layout_editor_absoluteY="330dp"
      tools:ignore="MissingConstraints" />

   <Button
      android:id="@+id/search"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Button"
      tools:layout_editor_absoluteX="158dp"
      tools:layout_editor_absoluteY="462dp"
      tools:ignore="MissingConstraints" />
</LinearLayout>
```

Output :


4:44
Module1_p2
https://www.google.com/
BUTT
ON
https://www....


4:44
google.com

7. **Aim :** Write a program to add two numbers and display the result in a toast message.

Objective :
A Toast message in Android provides a small pop-up notification that appears briefly on the screen to convey information to the user. It is often used for displaying simple messages or feedback without interrupting the user experience. By capturing user input from EditText fields, performing arithmetic operations, and using Toast to display the result, developers can create interactive and user-friendly applications.

Theory:

- To implement functionality for capturing two numbers from user input.
- To perform addition on the captured numbers.
- To display the result of the addition in a Toast message.

Code :

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
  tools:context=".MainActivity2">

  <EditText
    android:id="@+id/Number1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ems="10"
    android:inputType="text"
    android:layout_marginTop="10dp"
    android:layout_marginHorizontal="10dp"
    android:hint="Enter First Number" />

  <EditText
    android:id="@+id/Number2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ems="10"
    android:inputType="text"
    android:layout_marginHorizontal="10dp"
    android:hint="Enter Second Number" />

  <Button
```

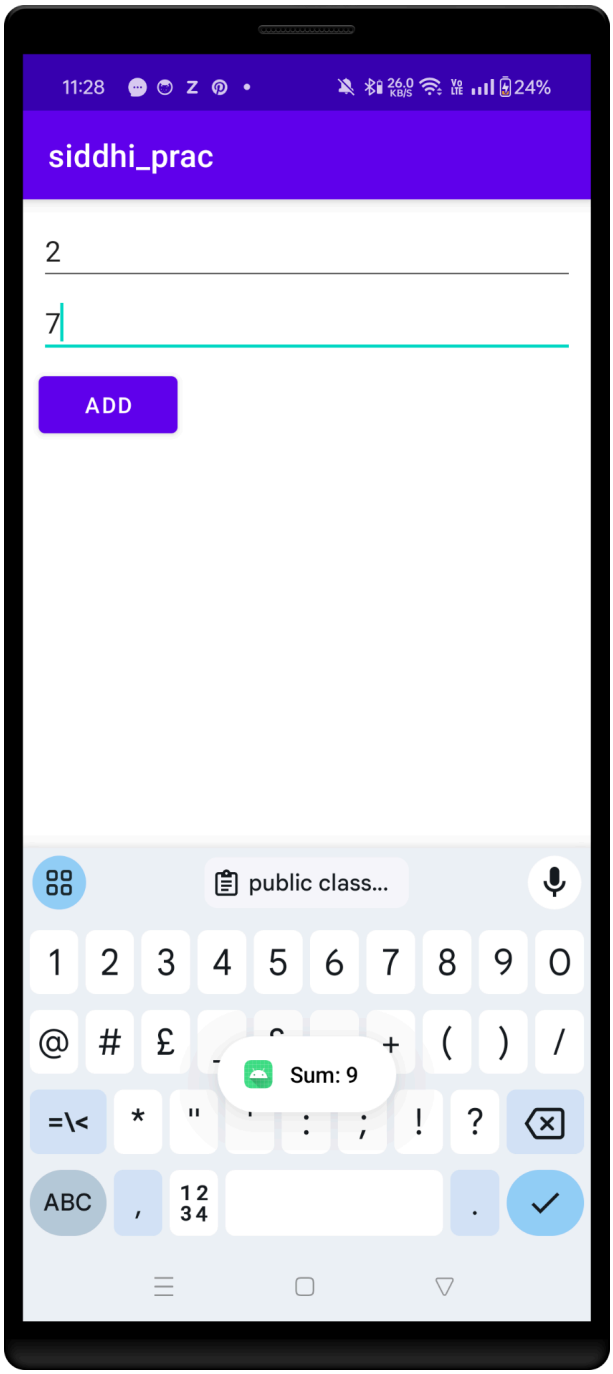
```
    android:id="@+id/Add"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="left"
    android:layout_marginHorizontal="10dp"
    android:layout_marginTop="5dp"
    android:text="Add" />
</LinearLayout>
```

MainActivity.java

```
package com.example.module1e;

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

public class MainActivity2 extends AppCompatActivity {

  EditText Num1, Num2;
  Button Add;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main2);

    Num1 = findViewById(R.id.Number1);
    Num2 = findViewById(R.id.Number2);

    Add = findViewById(R.id.Add);
    Add.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View view) {
        String num1 = Num1.getText().toString();
        String num2 = Num2.getText().toString();
        int sum = Integer.parseInt(num1) + Integer.parseInt(num2);
        Toast.makeText(getApplicationContext(), "Sum: " + sum,
Toast.LENGTH_SHORT).show();
      }
    });
  }
}
```

SIDDHI S. KOTRE
ROLL NO: 50

Output :



8. **Aim :** Design a screen that displays the frame image and write a quote on that.

Objective :
Designing a screen with an image and overlaying text involves combining visual elements in a way that maintains clarity and aesthetics. By using layouts such as FrameLayout or RelativeLayout, developers can stack a ImageView and a TextView to achieve the desired effect. The image serves as the background while the text is displayed on top, allowing for a visually appealing presentation of content. This approach is often used in applications to create engaging and dynamic interfaces.

Theory:

- To create a layout that positions an image and a text view for displaying a quote.
- To use appropriate layout attributes and styles to ensure the image and quote are displayed clearly and attractively.
- To demonstrate how to combine image and text in a single view using Android's layout system.

Code :

MainActivity.xml

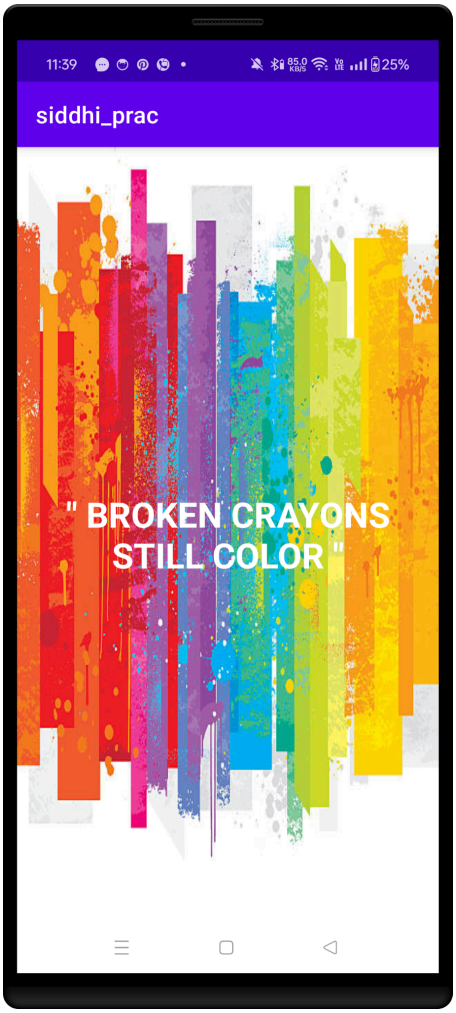
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
<TextView
android:id="@+id/textView"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="sunset"
android:gravity="top|center">
</TextView>

<ImageView
android:id="@+id/imageView"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
app:srcCompat="@drawable/image" />
<TextView
android:id="@+id/textView2"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="beach"
```

```
android:gravity="bottom|center">
</TextView>
</LinearLayout>
```

MainActivity.java

```
package com.example.layouts;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity10 extends AppCompatActivity {

@Override
protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState);
setContentView(R.layout.mainactivity10);
}
}
```

Output :



9. **Aim :** Write a program to demonstrate radio buttons and checkboxes.

Objective :
Radio buttons and checkboxes are common UI elements in Android used for selecting options from a set of choices. Radio buttons are used when only one option can be selected from a group, ensuring that only one item is chosen at a time. They are typically used for mutually exclusive choices. Checkboxes, on the other hand, allow multiple selections from a group of options, where each option can be independently selected or deselected. Understanding the use and behavior of these components is essential for creating interactive forms and settings in an application.

Theory:

- To demonstrate how to use radio buttons for selecting a single option from a predefined set of choices.
- To illustrate how to use checkboxes for allowing multiple independent selections from a set of options.
- To show how to handle user interactions and retrieve selected values from radio buttons and checkboxes in an Android application.

Code :

MainActivity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
  tools:context=".MainActivity">

  <TextView
    android:id="@+id/genders"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Select Gender"
    android:gravity="center"
    android:textSize="25dp"
    tools:ignore="HardcodedText,SpUsage" />
  <RadioGroup
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/gender_group">
```

```
    <RadioButton
        android:id="@+id/male"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Male"
        android:checked="false"
        tools:ignore="HardcodedText" />

    <RadioButton
        android:id="@+id/female"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Female"
        android:checked="false"
        tools:ignore="HardcodedText" />
</RadioGroup>
<TextView
    android:id="@+id/hobbies"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Select Hobbies"
    android:gravity="center"
    android:textSize="25dp"

    tools:ignore="HardcodedText" />

<CheckBox
    android:id="@+id/cricket"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Cricket"
    tools:ignore="HardcodedText" />
<CheckBox
    android:id="@+id/vollyball"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Volley ball"
    tools:ignore="HardcodedText" />

<CheckBox
    android:id="@+id/football"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Baseket Ball"
```

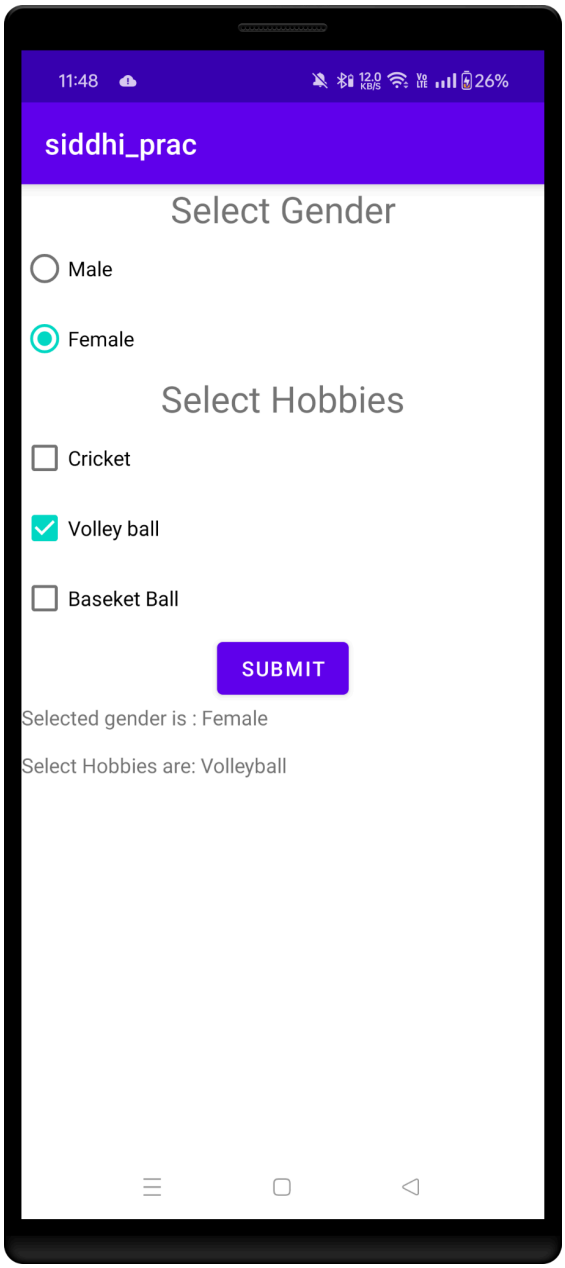
```
    tools:ignore="HardcodedText" />

  <Button
    android:id="@+id/showdata"
    android:text="Submit"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"/>
  <TextView
    android:id="@+id/data"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    />

</LinearLayout>
```

MainActivity.java

```
package com.example.siddhi_prac;

import android.annotation.SuppressLint;
import android.content.Intent;
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;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
  RadioGroup gender_group;
  CheckBox cricket,vollyball,football;
  Button showdata;
```

```
  TextView data;
  RadioButton radio;

  @Override
  protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

    gender_group=findViewById(R.id.gender_group); cricket=findViewById(R.id.cricket);
vollyball=findViewById(R.id.vollyball); football=findViewById(R.id.football);
showdata=findViewById(R.id.showdata); data=findViewById(R.id.data);

    showdata.setOnClickListener(new View.OnClickListener()
    {
      @SuppressLint("SetTextI18n")
      @Override
      public void onClick(View view) {
        String result="\n\nSelect Hobbies are: ";

        radio=findViewById(gender_group.getCheckedRadioButtonId());
if(cricket.isChecked()){
            result+="Cricket ";
        }
        if(football.isChecked()){ result+="Football ";
        }
        if(vollyball.isChecked()){ result+="Volleyball ";

        }
        data.setText("Selected gender is : "+ radio.getText().toString() +result + "\n\n");
      }
    });
  }
}
```

Output :


11:48
26%
siddhi_prac
Select Gender
Male
Female
Select Hobbies
Cricket
Volley ball
Baseket Ball
SUBMIT
Selected gender is : Female
Select Hobbies are: Volleyball

10. **Aim :** Design a option menu (use whatsapp option menu as reference)

Objective :
An options menu in Android provides a standard interface for presenting actions that users can perform within an activity. Typically accessed via the overflow button (three vertical dots) in the app bar, this menu offers additional functionality without cluttering the main interface.
Theory:

- To create a menu with multiple options that users can access by interacting with a menu button or overflow menu.
- To implement different menu items that perform various actions or navigate to different parts of the application.
- To demonstrate how to handle menu item selections and provide appropriate responses based on user choices.

Code :

menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
   <item android:id="@+id/download"
      android:title="Download" />
   <item android:id="@+id/search"
      android:title="Search" />
   <item android:id="@+id/help"
      android:title="Help" />
   <item android:id="@+id/settings"
      android:title="Settings" />
</menu>
```

MainActivity.java

```
package com.example.module1;

import android.os.Bundle;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

   @Override
   protected void onCreate(Bundle savedInstanceState) {
```

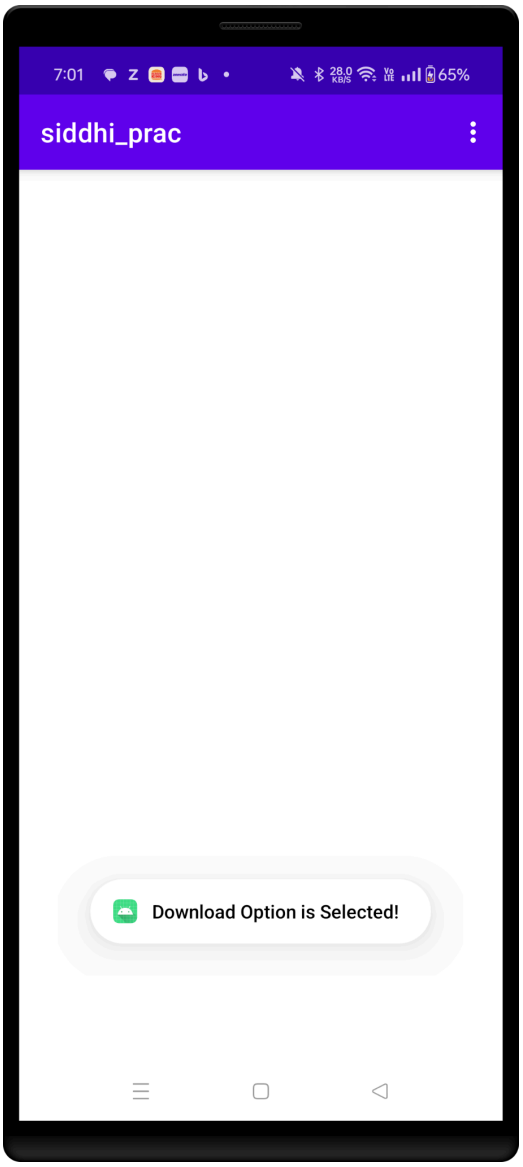
```
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater menuInflater = getMenuInflater();
        menuInflater.inflate(R.menu.menu, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    public boolean onOptionsItemSelected(@NonNull MenuItem item) {
        int id = item.getItemId();
        if (id == R.id.download) {
                                Toast.makeText(MainActivity.this, "Download Option is Selected!",
Toast.LENGTH_SHORT).show();
            return true;
        }
        else if (id == R.id.search) {
                                  Toast.makeText(MainActivity.this, "Search Option is Selected!",
Toast.LENGTH_SHORT).show();
            return true;
        }
        else if (id == R.id.help) {
                                    Toast.makeText(MainActivity.this, "Help Option is Selected!",
Toast.LENGTH_SHORT).show();
            return true;
        }
        else if (id == R.id.settings) {
                                Toast.makeText(MainActivity.this, "Settings Option is Selected!",
Toast.LENGTH_SHORT).show();
            return true;
        }
        return super.onOptionsItemSelected(item);
    }
}
```

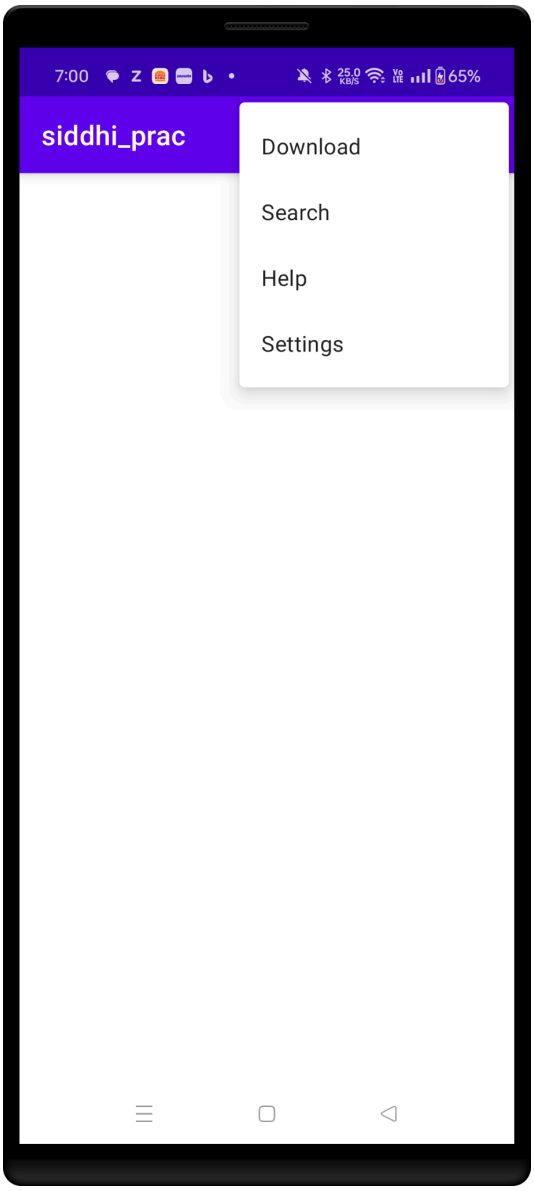
activity_main.xml

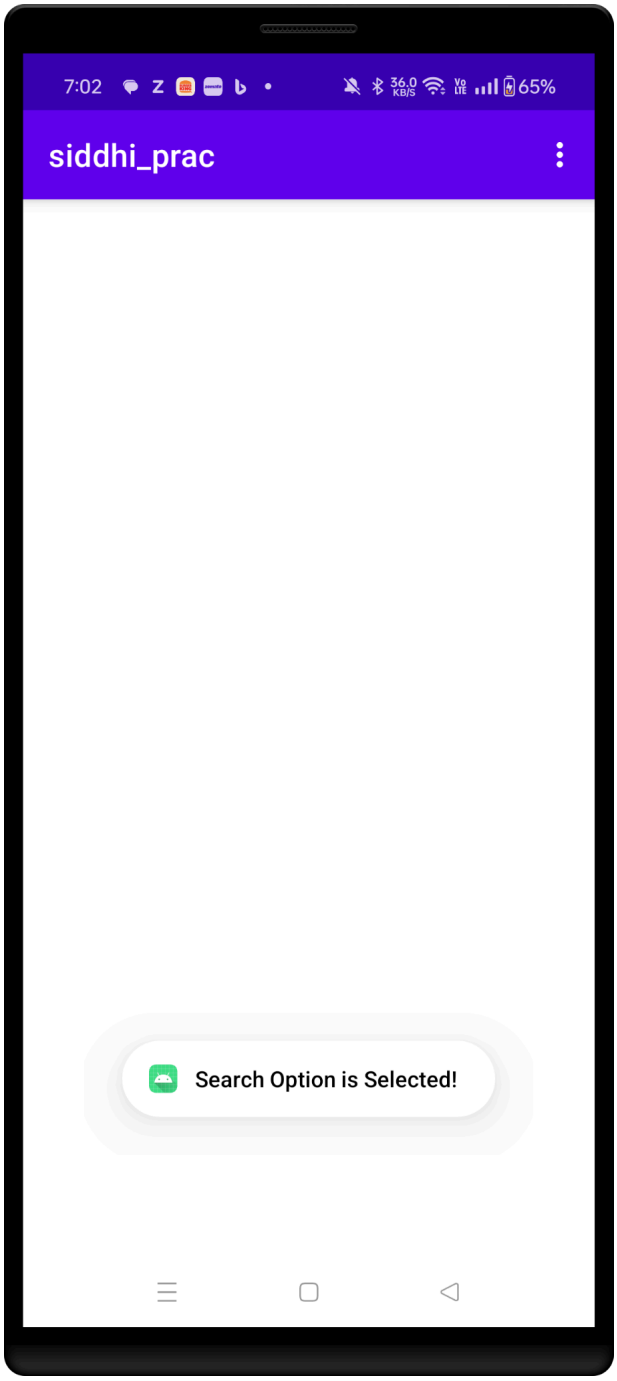
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"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

```
    android:orientation="vertical"
    tools:context=".MainActivity10">

</LinearLayout>
```

Output :






7:02
65%
siddhi_prac
Search Option is Selected!

11. **Aim :** Design an application which is having one Image and display a context menu on that image and also create and redirect to different activities

Objective :

A context menu in Android is a type of floating menu that appears when a user performs a long-press action on a UI element, such as an image. It provides additional options relevant to the context of the item. By defining context menu items and handling their selections, developers can offer users a range of actions without cluttering the main interface.

Theory:

1. To implement a context menu that appears when the user long-presses on an image, providing various options.
2. To create multiple activities within the application that can be navigated to from the context menu.
3. To handle menu item selections and perform appropriate actions, such as launching different activities.

Code :

menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
   <item android:id="@+id/download"
      android:title="Download" />
   <item android:id="@+id/search"
      android:title="Search" />
   <item android:id="@+id/help"
      android:title="Help" />
   <item android:id="@+id/settings"
      android:title="Settings" />
</menu>
```

MainActivity.java

```
package com.example.module1;

import android.content.Intent;
import android.os.Bundle;
import android.view.ContextMenu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.ImageView;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
```

```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        ImageView imageView = findViewById(R.id.imageView2);
        registerForContextMenu(imageView);
    }

    @Override
                public void onCreateContextMenu(ContextMenu menu, View v,
ContextMenu.ContextMenuInfo menuInfo) {
        MenuInflater menuInflater = getMenuInflater();
        menuInflater.inflate(R.menu.menu, menu);
        super.onCreateContextMenu(menu, v, menuInfo);
    }

    @Override
    public boolean onContextItemSelected(@NonNull MenuItem item) {
        int id = item.getItemId();
        if (id == R.id.download) {
            Intent intent = new Intent(MainActivity.this, DownloadActivity.class);
            startActivity(intent);
            return true;
        } else if (id == R.id.search) {
            Intent intent = new Intent(MainActivity.this, SearchActivity.class);
            startActivity(intent);
            return true;
        } else if (id == R.id.help) {
            Intent intent = new Intent(MainActivity.this, HelpActivity.class);
            startActivity(intent);
            return true;
        } else if (id == R.id.settings) {
            Intent intent = new Intent(MainActivity.this, SettingsActivity.class);
            startActivity(intent);
            return true;
        }
        return super.onContextItemSelected(item);
    }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    tools:context=".MainActivity11">

    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        app:srcCompat="@drawable/astronaut" />
</LinearLayout>
```

DownloadActivity.java

```
package com.example.module1;

import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;

public class DownloadActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_download);
    }
}
```

activity_download.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
```

```
    android:gravity="center"
    tools:context=".DownloadActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="You're redirected to the download page!" />
</LinearLayout>
```

SearchActivity.java

```
package com.example.module1;

import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;

public class SearchActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_search);
    }
}
```

activity_search.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:gravity="center"
    tools:context=".SearchActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="You're redirected to the search page!" />
</LinearLayout>
```

HelpActivity.java

```
package com.example.module1;

import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;

public class HelpActivity extends AppCompatActivity {

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_help);
   }
}
```

activity_help.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
   android:gravity="center"
   tools:context=".HelpActivity">

   <TextView
      android:id="@+id/textView"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:gravity="center"
      android:text="You're redirected to the help page!" />
</LinearLayout>
```

SettingsActivity.java

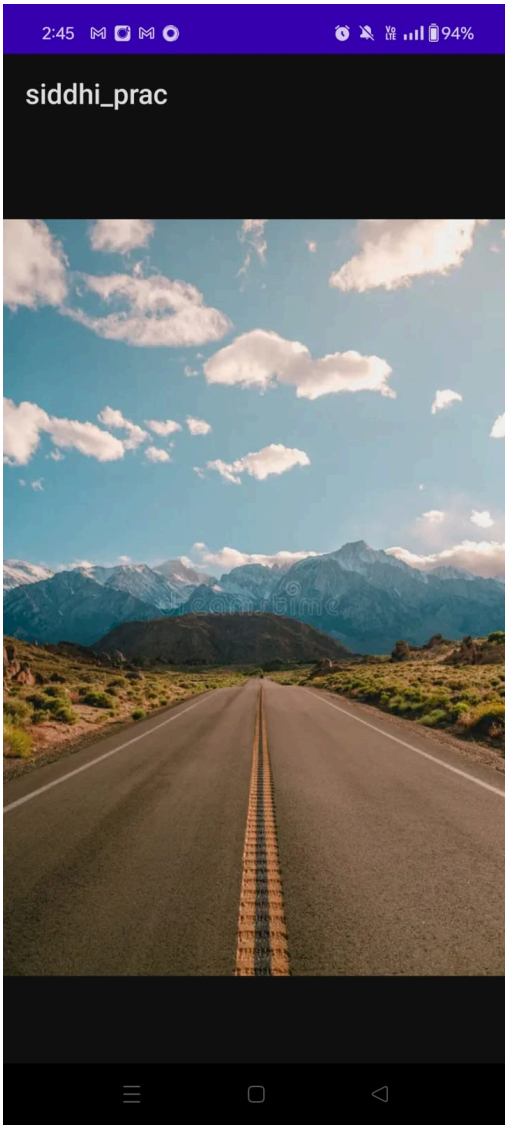
```
package com.example.module1;

import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;

public class SettingsActivity extends AppCompatActivity {

   @Override
```

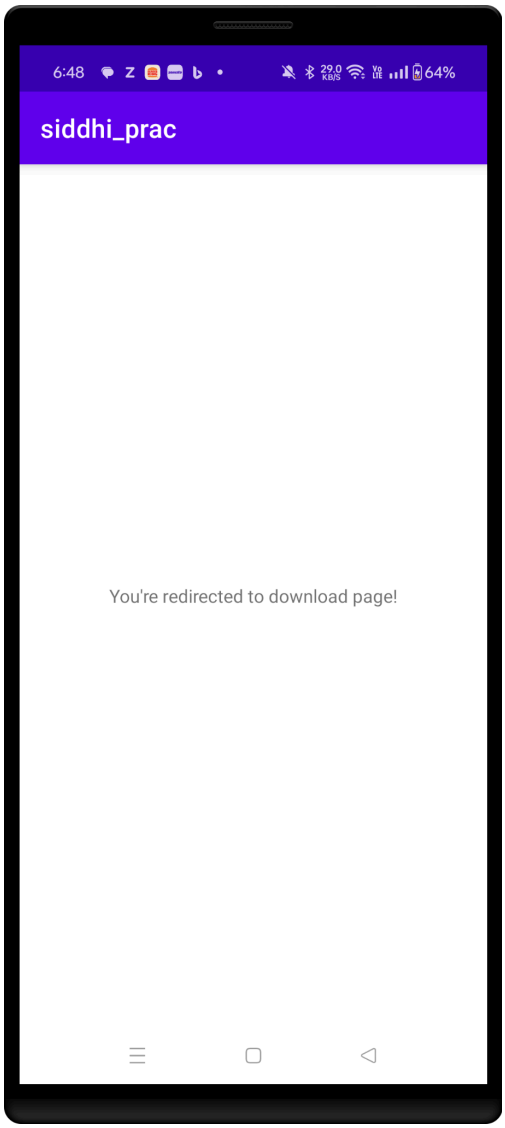
```
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_settings);
    }
}
```

activity_settings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:gravity="center"
    tools:context=".SettingsActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="You're redirected to the settings page!" />
</LinearLayout>
```

Output :


2:45
94%
siddhi_prac


6:48
64%
siddhi_prac
You're redirected to download page!

12. **Aim :** Demonstrate different shapes of control.

Objective :
In Android, UI controls can be styled and customized using drawable resources, which allows developers to create various shapes for elements such as buttons, text fields, and backgrounds. By defining shapes in XML drawable files, developers can create controls with specific visual styles, including rounded corners, circles, and custom borders. This customization enhances the aesthetics of the application and provides a more engaging and intuitive user interface.

Theory:

- To illustrate the implementation of different shapes for UI controls, such as buttons, text fields, and containers.
- To show how to customize and style UI elements to use different shapes, including rectangles, circles, and rounded corners.
- To demonstrate how shape customization can improve user experience and interface design.

Code :

MainActivity.java

```
package com.example.module1;

import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity12 extends AppCompatActivity {

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main12);
   }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
   android:padding="10dp"
   tools:context=".MainActivity12">
```

```
    <TextView
        android:id="@+id/cylindrical"
        android:background="@drawable/cylindrical"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="Cylindrical" />

    <TextView
        android:id="@+id/oval"
        android:background="@drawable/oval"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:gravity="center"
        android:text="Oval" />

    <TextView
        android:id="@+id/rectangle"
        android:background="@drawable/rectangle"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:gravity="center"
        android:text="Rectangle" />

    <TextView
        android:id="@+id/ring"
        android:background="@drawable/ring"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:gravity="center"
        android:text="Ring" />
</LinearLayout>
```

cylindrical.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape                                xmlns:android="http://schemas.android.com/apk/res/android"
android:shape="rectangle">
    <stroke android:color="@color/black"
        android:width="2dp" />
    <corners android:radius="50dp" />
    <solid android:color="#EB455F" />
    <size android:width="120dp" android:height="40dp" />
```

```
</shape>
```

oval.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android" android:shape="oval">
   <stroke android:color="@color/black"
      android:width="2dp" />
   <solid android:color="#52DE97" />
   <size android:width="75dp" android:height="30dp" />
</shape>
```

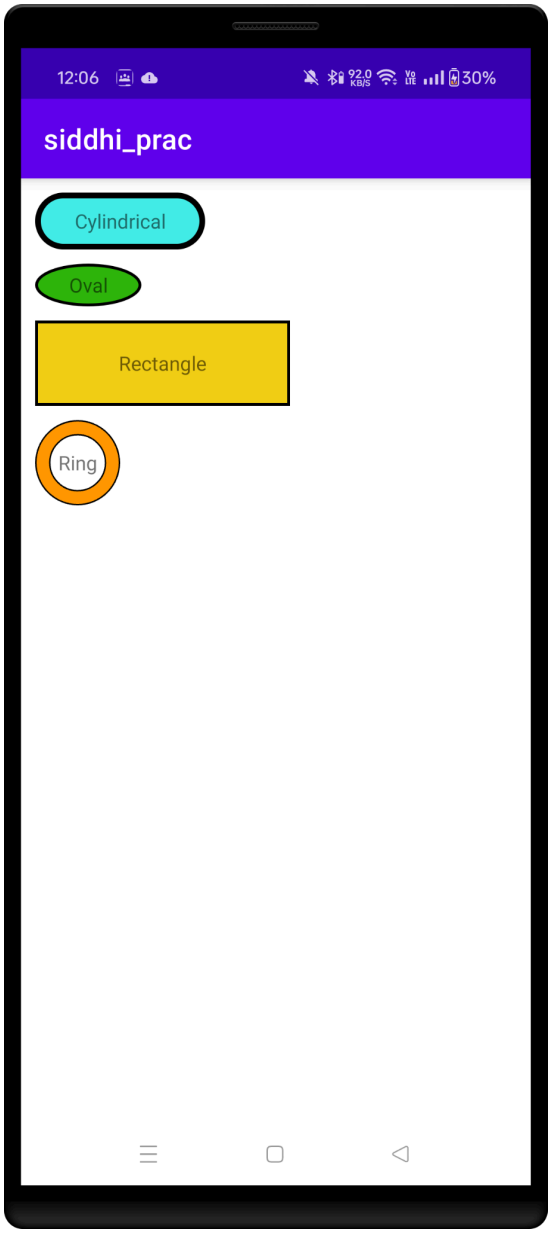
rectangle.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape                                    xmlns:android="http://schemas.android.com/apk/res/android"
android:shape="rectangle">
   <stroke android:color="@color/black"
      android:width="2dp" />
   <solid android:color="#4CAF50" />
   <size android:width="180dp" android:height="60dp" />
</shape>
```

ring.xml

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
   android:shape="ring"
   android:thicknessRatio="6"
   android:useLevel="false">
   <stroke android:width="1dp" android:color="@color/black"/>
   <size android:width="60dp" android:height="60dp" />
   <solid android:color="#FF9800" />
</shape>
```

Output :



13. **Aim :** Write an application to increase font size using seekbar.

Objective :
A SeekBar in Android is a user interface element that allows users to select a value from a continuous range by sliding a thumb along a horizontal track. By linking a SeekBar to a TextView, developers can provide a dynamic way to adjust the text size in real-time. This interaction is achieved by listening to SeekBar's progress changes and applying the corresponding text size to the TextView.

Theory:

- To implement a SeekBar that allows users to adjust the font size of a text view in real-time.
- To demonstrate how to programmatically change the text size based on the SeekBar's progress.
- To provide a user-friendly interface that updates the font size interactively as the SeekBar is adjusted.

Code :

```
package com.example.prac13;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.widget.SeekBar;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

  TextView data;
  SeekBar range;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);

     data = findViewById(R.id.textView);
     range = findViewById(R.id.seekBar);

     range.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
        @Override
        public void onProgressChanged(SeekBar seekBar, int i, boolean b) {
           data.setTextSize(i);
```

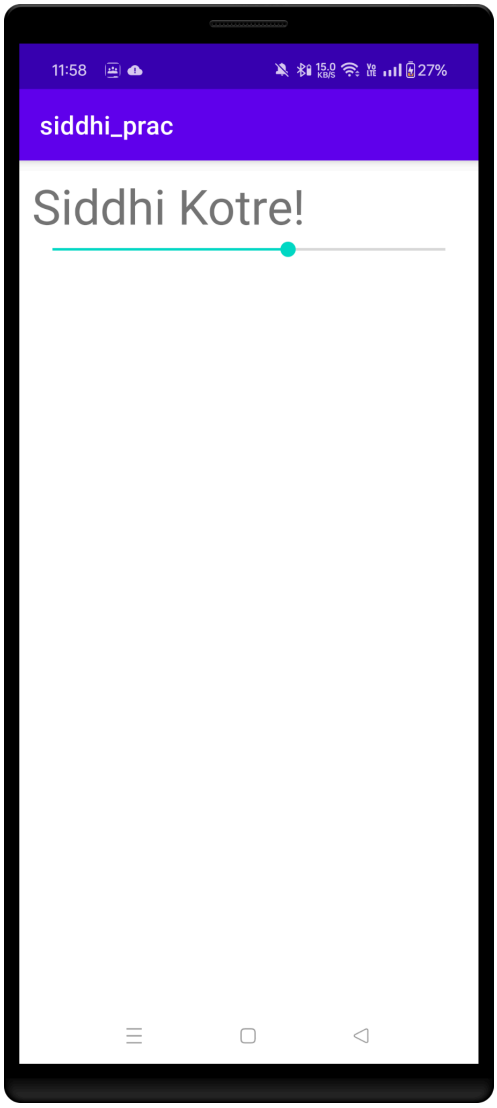
```
        }

        @Override
        public void onStartTrackingTouch(SeekBar seekBar) {

        }

        @Override
        public void onStopTrackingTouch(SeekBar seekBar) {

        }
      });
  }
}
```

MainActivity.java

```
package com.example.prac13;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.widget.SeekBar;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

  TextView data;
  SeekBar range;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);

     data = findViewById(R.id.textView);
     range = findViewById(R.id.seekBar);

     range.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
        @Override
        public void onProgressChanged(SeekBar seekBar, int i, boolean b) {
           data.setTextSize(i);
        }

        @Override
        public void onStartTrackingTouch(SeekBar seekBar) {
```

```
        }

        @Override
        public void onStopTrackingTouch(SeekBar seekBar) {

        }
    });
  }
}
```

Output :



14. **Aim :** Write a program to demonstrate Background Service in Android Application.

Objective :

A background service in Android is a component that runs independently of the user interface and can perform long-running operations or tasks even when the app is not in the foreground. Services are used for tasks such as network operations, file I/O, or periodic updates. By using a background service, developers can ensure that these tasks continue to run smoothly, even when the user interacts with other parts of the application or the app is closed.

Theory:

- To create a background service that runs even when the application is not in the foreground.
- To illustrate how to start, stop, and interact with a background service from different components of the application.
- To show how the service performs tasks such as logging, data fetching, or periodic updates without interfering with the user interface.

Code :

Activity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="horizontal"
  android:gravity="center"
  tools:context=".MainActivity">

  <Button
    android:id="@+id/start"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Start"
    android:onClick="startMusic" />
  <Button
    android:id="@+id/stop"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:onClick="stopMusic"
    android:layout_marginStart="50dp"
    android:text="Stop" />

</LinearLayout>
```

MainActivity.java

```
package com.example.practical1_module2;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity {
Button start,stop;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main2);
     start = findViewById(R.id.start);
     stop = findViewById(R.id.stop);
  }

  public void startMusic(View view){
     Intent intent = new Intent(this,musicService.class);
     startService(intent);
  }
  public void stopMusic(View view){
     Intent intent = new Intent(this,musicService.class);
     stopService(intent);
  }
}
```

musicService.java

```
package com.example.practical1_module2;

import android.app.Service;
import android.content.Intent;
import android.media.MediaPlayer;
import android.os.IBinder;
import android.widget.Toast;

public class musicService extends Service {
MediaPlayer player;

  @Override
  public int onStartCommand(Intent intent, int flags, int startId) {
     player = MediaPlayer.create(this,R.raw.music);
```

```
        player.setLooping(true);
        player.start();
        Toast.makeText(this, "Music is Started", Toast.LENGTH_SHORT).show();
       return START_STICKY;
    }
public void onDestroy(){
        super.onDestroy();
        player.stop();
    Toast.makeText(this, "Stopped", Toast.LENGTH_SHORT).show();
}
    public musicService() {

    }

    @Override
    public IBinder onBind(Intent intent) {
        // TODO: Return the communication channel to the service.
        throw new UnsupportedOperationException("Not yet implemented");
    }
}
```

Output :


4:03
Practical1_Module2
START
STOP
Music is Started


4:02
Practical1_Module2
START
STOP
Stopped

15. **Aim :** Write a program to demonstrate Fragment in Android.

Objective :
Fragments are a modular section of an activity that represent a portion of the user interface or behavior. They allow for more flexible and dynamic UI designs by enabling the combination of multiple fragments within a single activity. Fragments can be used to create reusable UI components and are managed by the hosting activity.

Theory:

- To create and display a Fragment within an Activity, illustrating the basic lifecycle and interaction of Fragments.
- To show how to dynamically add, replace, or remove Fragments in response to user actions or application logic.
- To demonstrate communication between an Activity and its Fragments, enabling interaction and data exchange

Code :

Fragment_1.xml

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".Fragment1">

  <!-- TODO: Update blank fragment layout -->
  <TextView
     android:layout_width="match_parent"
     android:layout_height="match_parent"
     android:text="This is Fragment 1" />

</FrameLayout>
```

Fragment_2.xml

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  tools:context=".Fragment2">

  <!-- TODO: Update blank fragment layout -->
  <TextView
     android:layout_width="match_parent"
```

```
    android:layout_height="match_parent"
    android:text="This is Fragment 2" />

</FrameLayout>
```

Activity_main.xml

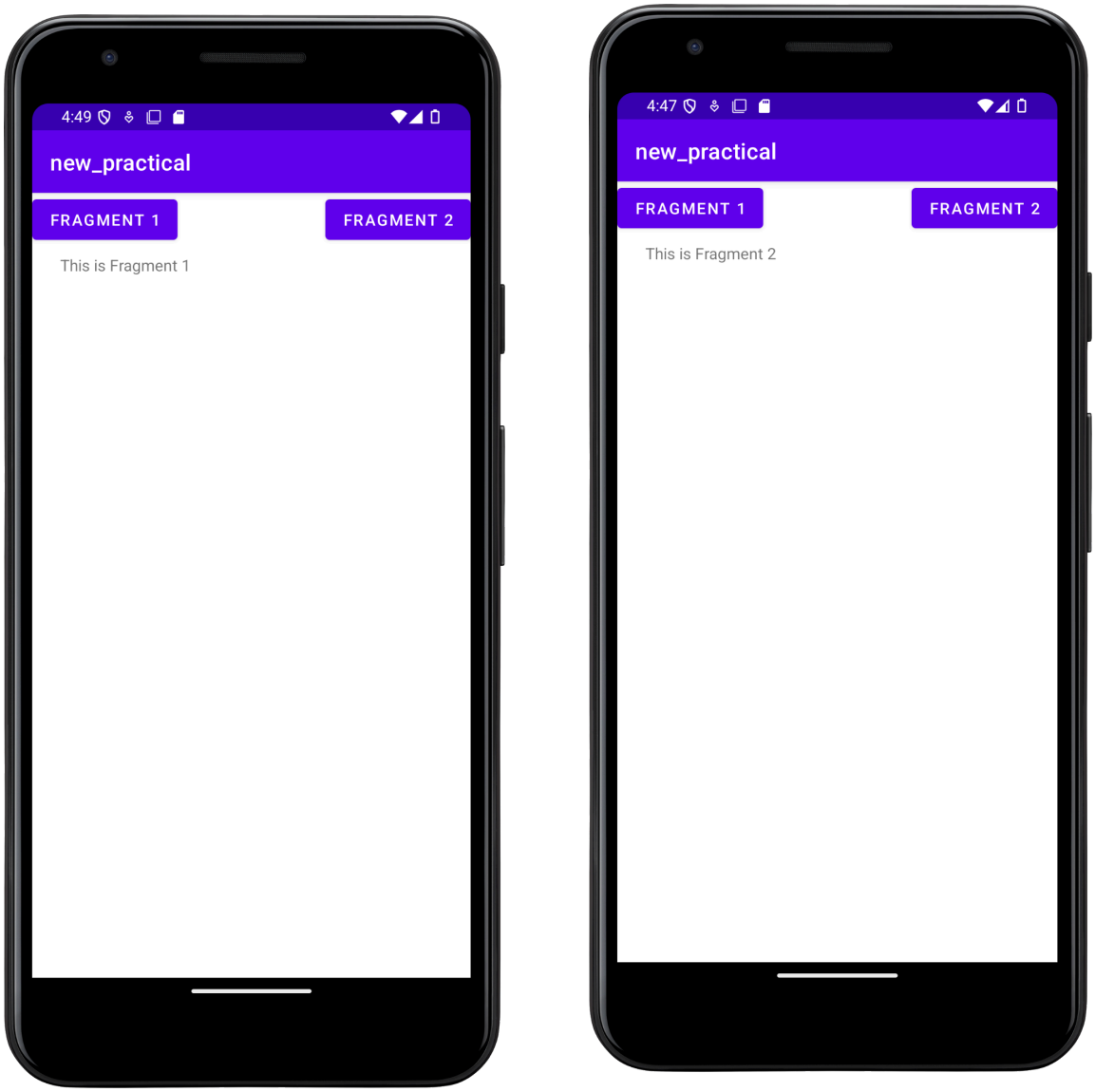
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"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

  <Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Fragment 1"
    android:onClick="One_Click"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
     />

  <Button
    android:id="@+id/button2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Fragment 2"
    android:onClick="Two_Click"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    />

  <FrameLayout
    android:id="@+id/frag_switch"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginLeft="25dp"
    android:layout_marginRight="25dp"
    android:layout_marginTop="55dp"
    android:layout_marginBottom="25dp"
    ></FrameLayout>
```

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

MainActivity.java

```
package com.example.new_practical;

import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.FragmentTransaction;

import android.os.Bundle;
import android.view.View;

public class MainActivity extends AppCompatActivity {

  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);

  }
  public void One_Click(View view){
     Fragment1  myFragment = new Fragment1();
     FragmentTransaction transaction = getSupportFragmentManager().beginTransaction();
     transaction.replace(R.id.frag_switch,myFragment);
     transaction.commit();
  }

  public void Two_Click(View view){
     Fragment2  myFragment = new Fragment2();
     FragmentTransaction transaction = getSupportFragmentManager().beginTransaction();
     transaction.replace(R.id.frag_switch,myFragment);
     transaction.commit();
  }
}
```

Output :


4:49
new_practical
FRAGMENT 1
FRAGMENT 2
This is Fragment 1
4:47
new_practical
FRAGMENT 1
FRAGMENT 2
This is Fragment 2

16. **Aim :** Create a mobile application for currency converter. Use a spinner for selecting the currency (min 5 currencies).

Objective :
A currency converter application enables users to convert amounts from one currency to another using conversion rates. In Android, a Spinner is used as a dropdown menu to allow users to select an option from a list, such as different currencies. By integrating a Spinner with a currency conversion logic, users can choose their desired currencies and view the conversion results. The application typically involves handling user input, performing calculations based on conversion rates, and updating the user interface with the results.
Theory:

- To implement a Spinner for selecting from a list of currencies, allowing users to choose the currency they want to convert from or to.
- To design and implement a user interface that allows users to input an amount and display the converted value based on the selected currency.
- To integrate currency conversion logic and display real-time or static conversion rates to perform accurate currency conversions.

Code :

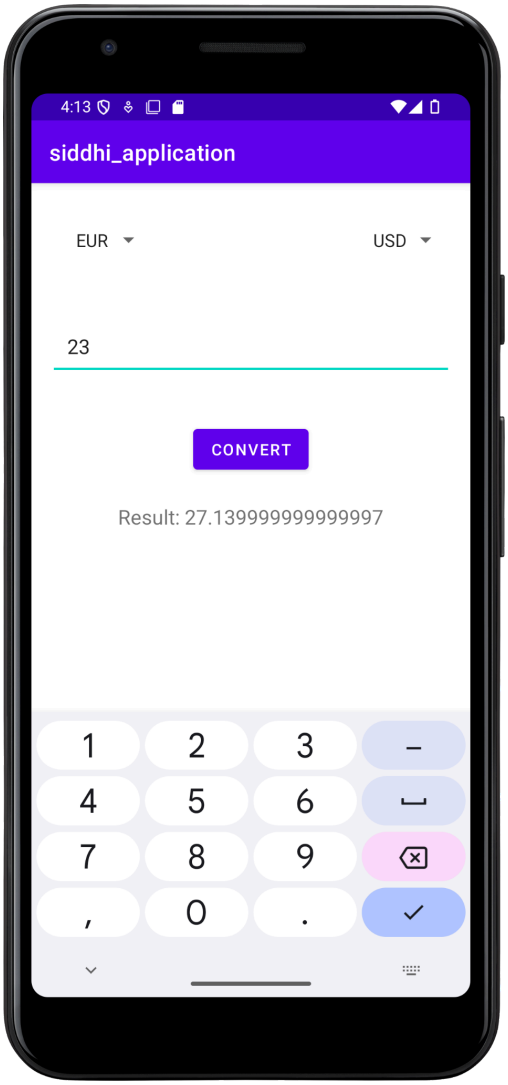
```
package com.example.prac16;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

  Spinner To, From;
  EditText Amount;
  Button Convert;
  TextView Display;

  private String[] currencies = {"USD", "EUR", "INR", "GBP", "JPY"};

  private double[][] rates = {
        {1.0, 0.85, 74.34, 0.75, 110.62},
        {1.18, 1.0, 87.43, 0.88, 130.06},
```

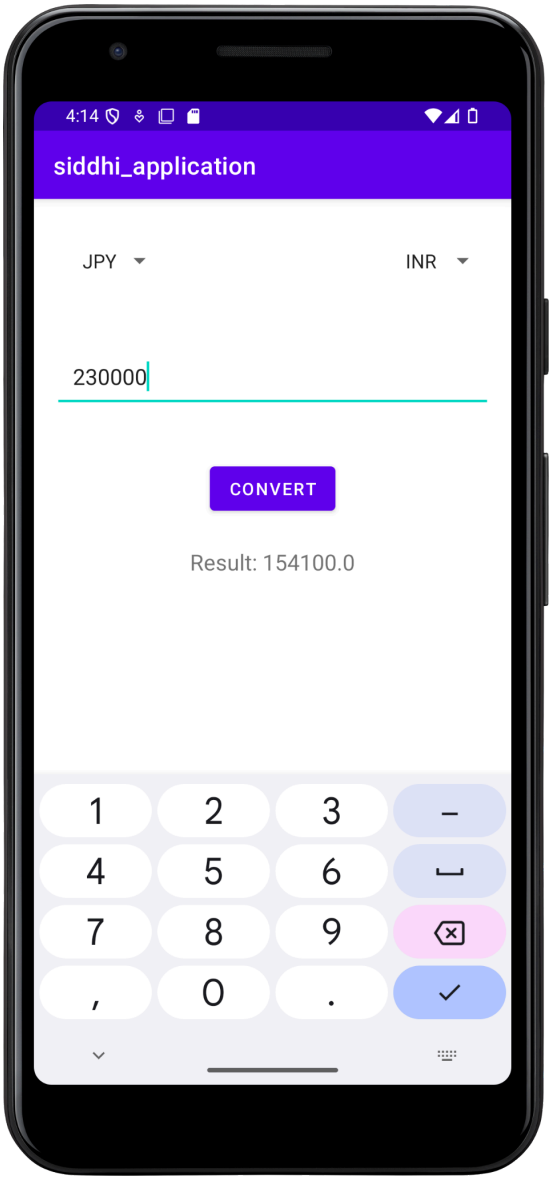
```
        {0.013, 0.011, 1.0, 0.010, 1.49},
        {1.33, 1.14, 100.84, 1.0, 148.36},
        {0.009, 0.0077, 0.67, 0.0067, 1.0}
  };

  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);

     To = findViewById(R.id.to);
     From = findViewById(R.id.from);
     Amount = findViewById(R.id.amount);
     Convert = findViewById(R.id.convert);
     Display = findViewById(R.id.display);

     ArrayAdapter<String> arrayAdapter = new ArrayAdapter<>(MainActivity.this,
android.R.layout.simple_spinner_item, currencies);

arrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
     To.setAdapter(arrayAdapter);
     From.setAdapter(arrayAdapter);

     Convert.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
           convertCurrency();
        }
     });
  }

  private void convertCurrency() {
     int from = From.getSelectedItemPosition();
     int to = To.getSelectedItemPosition();
     double dataAmount = Double.parseDouble(Amount.getText().toString());
     double result = dataAmount * rates[from][to];
     Display.setText(dataAmount + " = " + result);
  }
}

<?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:layout_width="match_parent"
  android:layout_height="match_parent"
  android:padding="10dp"
  tools:context=".MainActivity">

  <Spinner
     android:id="@+id/to"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:layout_alignParentLeft="true"/>

  <Spinner
     android:id="@+id/from"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:layout_alignParentRight="true"/>

  <EditText
     android:id="@+id/amount"
     android:layout_width="match_parent"
     android:layout_height="wrap_content"
     android:inputType="numberDecimal"
     android:layout_marginTop="20dp"
     android:layout_below="@+id/to"
     android:hint="Enter amount" />

  <Button
     android:id="@+id/convert"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:layout_below="@+id/amount"
     android:layout_centerHorizontal="true"
     android:layout_marginTop="10dp"
     android:text="Convert" />

  <TextView
     android:id="@+id/display"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:layout_below="@+id/convert"
     android:layout_centerHorizontal="true"
     android:layout_marginTop="10dp"
     android:text="" />
```

</RelativeLayout>

Output :




4:14
siddhi_application
JPY
INR
230000
CONVERT
Result: 154100.0
1
2
3
4
5
6
7
8
9
,
0
.