

# **Practical File**

**Mobile Application Development Laboratory**

**Subject Code – DEIT-14712**



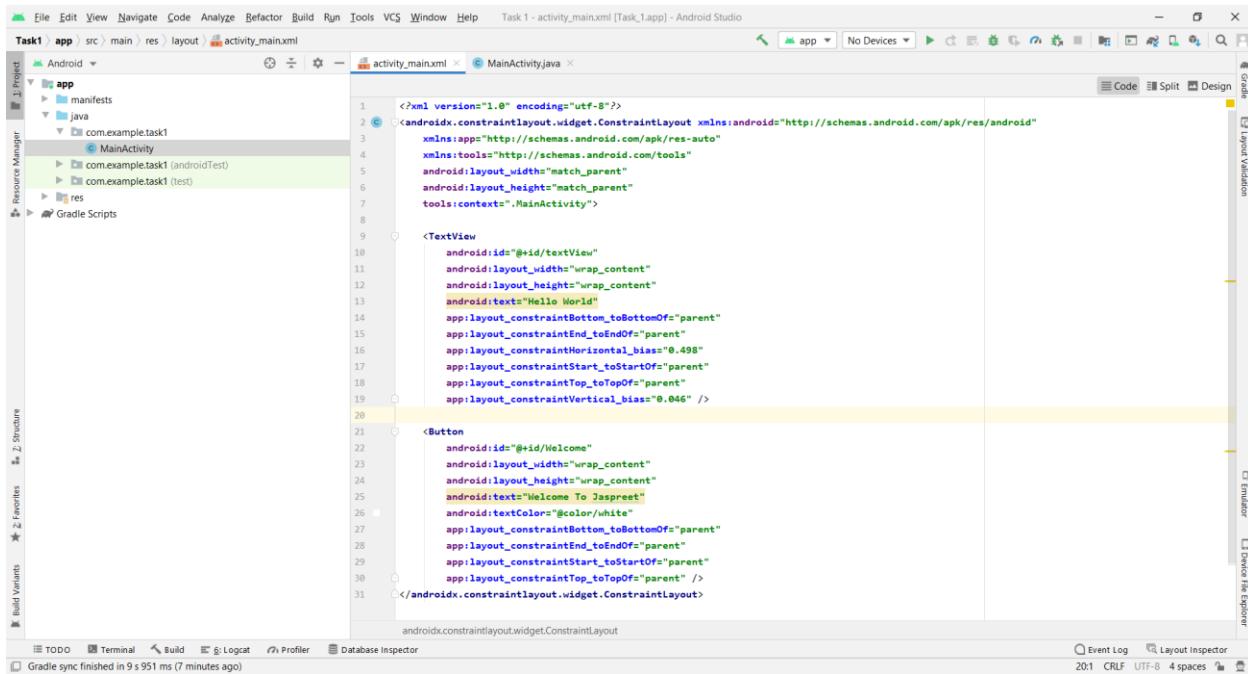
**Submitted By**

**Jaspreet Kaur (1706861)**

**Section: D4 I.T.( A2)**

# Practical 1:- Create a first app on Android

## Xml File

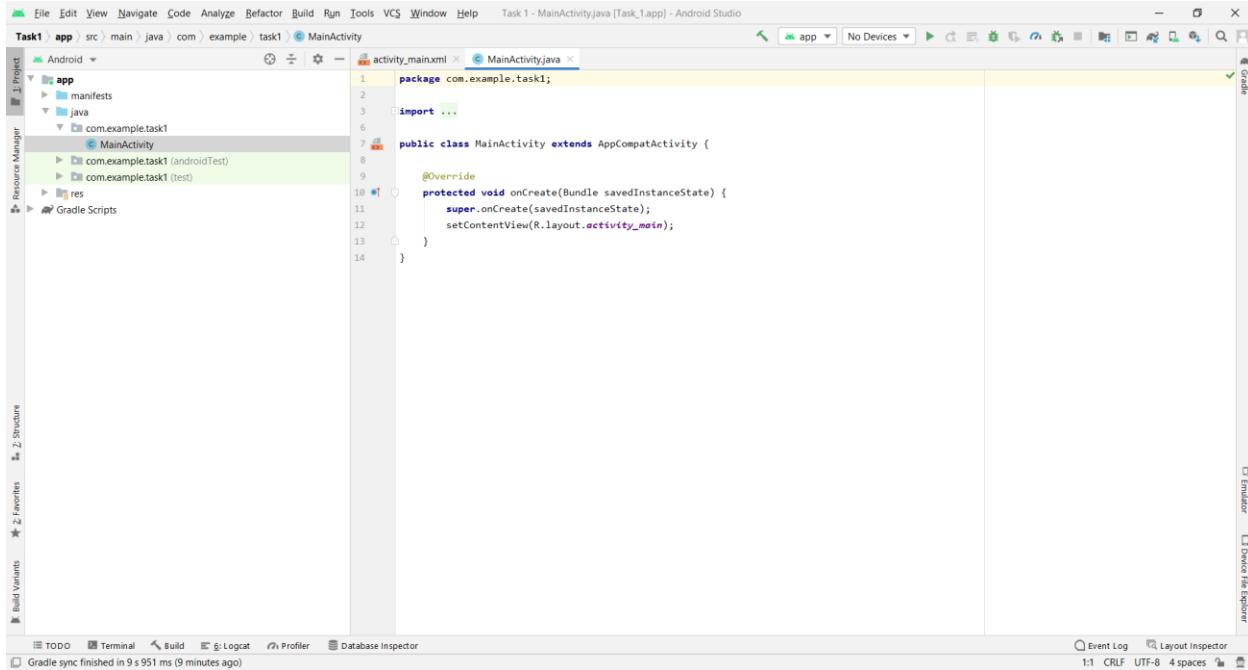


```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
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

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.046" />

    <Button
        android:id="@+id/Welcome"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Welcome To Jaspree"
        android:textColor="#color/white"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

## Java File



```
package com.example.task1;

import ...

public class MainActivity extends AppCompatActivity {

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

# OUTPUT:-

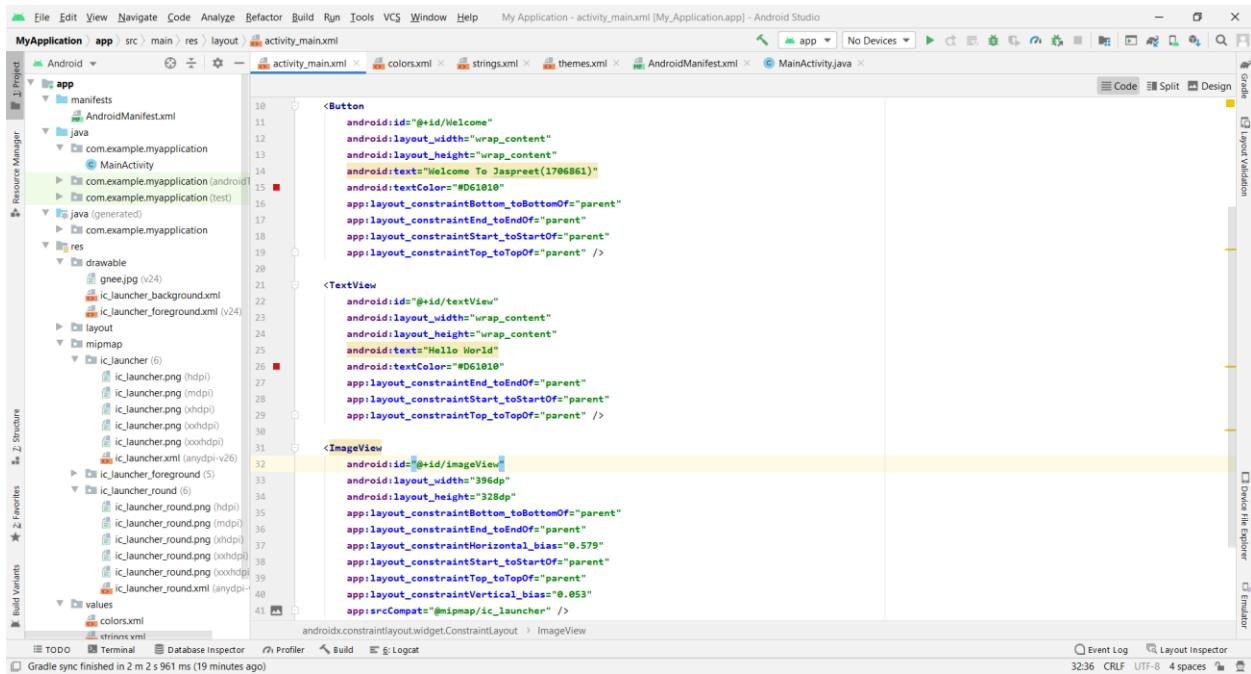


WELCOME TO JASPREET

---

## Practical 2:- Create icon of app, change the label Background

### Xml File

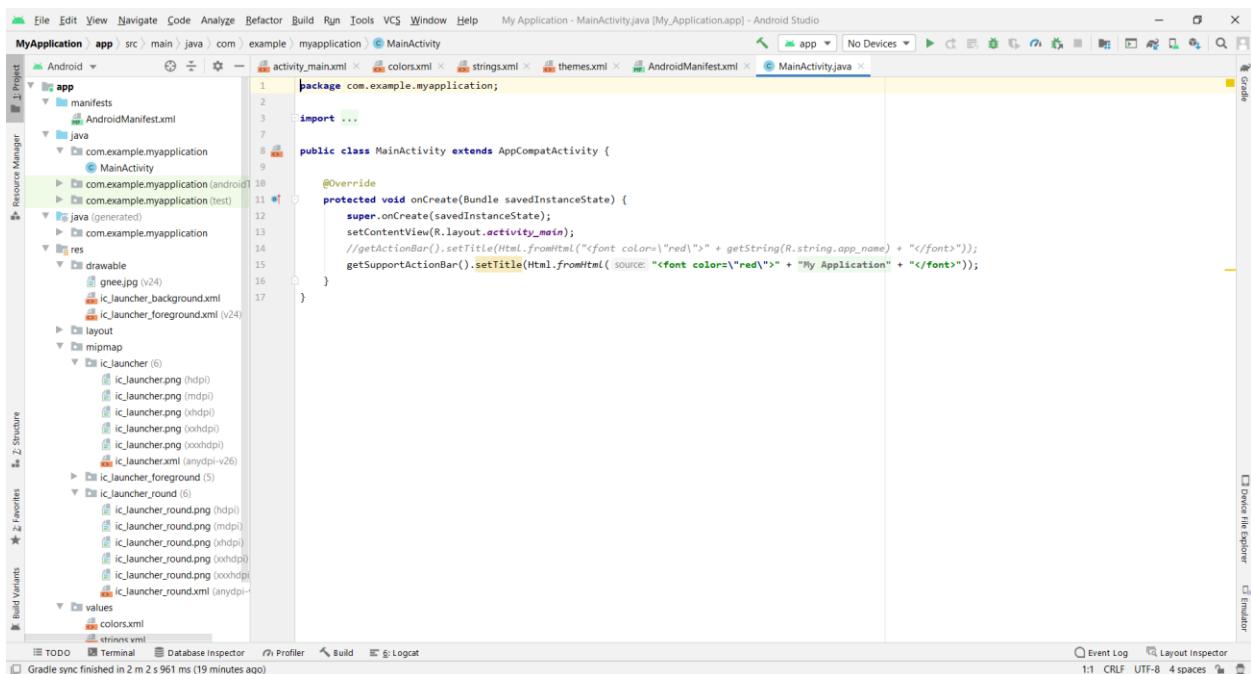


```
<Button
    android:id="@+id>Welcome"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Welcome To Jaspreet(1706861)"
    android:textColor="#000000"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<TextView
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Hello World"
    android:textColor="#000000"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<ImageView
    android:id="@+id/imageView"
    android:layout_width="394dp"
    android:layout_height="328dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.579"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.053"
    app:srcCompat="@mipmap/ic_launcher" />
```

### Java File:-



```
package com.example.myapplication;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        //getActionBar().setTitle(Html.fromHtml("<font color=\"red\\\">" + getString(R.string.app_name) + "</font>"));
        getSupportActionBar().setTitle(Html.fromHtml("source: <font color=\"red\\\">" + "My Application" + "</font>"));
    }
}
```

# OUTPUT:-



---

## Practical 3:- Create the menu, Buttons and ScrollView in an Application

### Xml File:-

The screenshot shows the Android Studio interface with the XML file `activity_main.xml` open. The code defines a `ScrollView` containing a `LinearLayout` with a `TextView`. A yellow callout box highlights the `TextView` element, providing information about creating compound controls.

```
<ScrollView  
    android:layout_width="329dp"  
    android:layout_height="617dp"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent">  
  
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:orientation="vertical" >  
  
<TextView  
    android:id="@+id/textView"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:text="Compound Controls:-"  
  
If you don't want to create a completely customized component, but instead are looking to put together a reusable component that  
In Android, there are actually two other Views readily available to do this: Spinner and AutoCompleteTextView, but regardless, t  
To create a compound component:  
The usual starting point is a Layout of some kind, so create a class that extends a Layout. Perhaps in the case of a Combo box +  
In the constructor for the new class, take whatever parameters the superclass expects, and pass them through to the superclass +  
You can also create listeners for events that your contained views might generate, for example, a listener method for the List It  
You might also create your own properties with accessors and modifiers, for example, allow the EditText value to be set initially  
In the case of extending a Layout, you don't need to override the onDraw() and onMeasure() methods since the layout will have def:  
You might override other on... methods, like onKeyDown(), to perhaps choose certain default values from the popup list of a combo  
To summarize, the use of a Layout as the basis for a Custom Control has a number of advantages, including:  
  
1:1 CRLF UTF-8 4 spaces
```

### Java File:-

The screenshot shows the Android Studio interface with the Java file `MainActivity.java` open. The code defines a `MainActivity` class that extends `AppCompatActivity`. It overrides the `onCreate` and `onCreateOptionsMenu` methods.

```
package com.example.scrollview;  
  
import ...  
  
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 inflater = getMenuInflater();  
        inflater.inflate(R.menu.game_menu, menu);  
        return true;  
    }  
}
```

## Menu File:

The screenshot shows the Android Studio interface with the game\_menu.xml file open in the code editor. The code defines a menu with three items: Green, Yellow, and Pink.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:app="http://schemas.android.com/apk/res-auto"
      xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:title="Green" />
    <item android:title="Yellow" />
    <item android:title="Pink" />
</menu>
```

## OUTPUT:-



## Practical 4:- Create the form using all the UI i.e Check Button, Radio Button, TextView

### Xml File

```
<?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">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="this is my app"
        android:textStyle="bold"
        android:textColor="#990EE90"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintHorizontal_bias="0.05"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.049" />

    <EditText
        android:id="@+id/editTextTextPersonName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPersonName"
        android:text="Jaspreet (1706861)"
        android:textStyle="bold" />

```

```
<CheckBox
    android:id="@+id/checkBox3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Option1"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.044"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.232" />

<CheckBox
    android:id="@+id/checkBox4"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Option2"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.05"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.303" />

<RadioButton
    android:id="@+id/radioButton3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginBottom="540dp"
    android:text="radio button" />
```

The screenshot shows the Android Studio interface with the code editor open to the `activity_main.xml` file. The XML code defines a layout with three main components: an `ImageButton`, an `ImageView`, and a `Button`. The `ImageButton` has an ID of `@+id/imageButton2` and is positioned at the bottom center of the screen. The `ImageView` has an ID of `@+id/imageView2` and is positioned above the `ImageButton`. The `Button` has an ID of `@+id/button2` and is positioned below the `ImageView`. All components are constrained to the parent layout using `app:layout_constraint*` attributes.

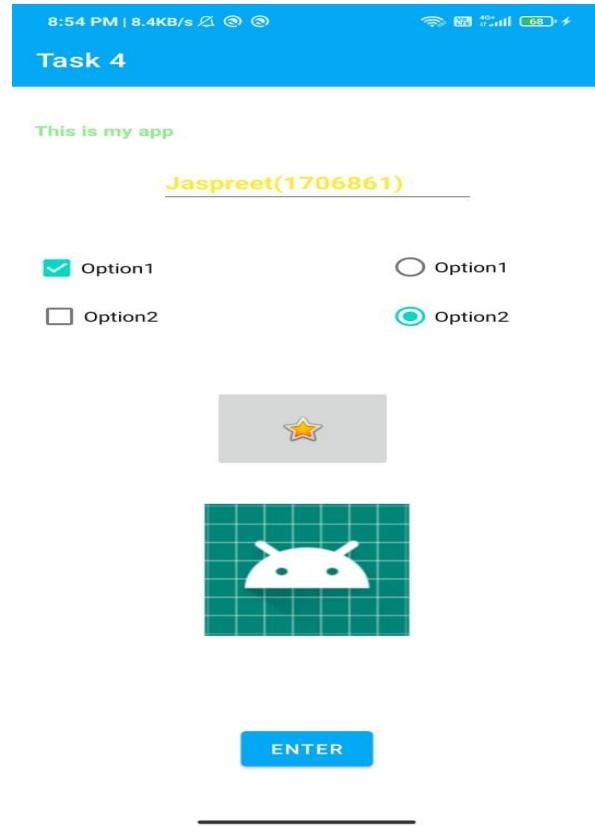
```
<ImageButton  
    android:id="@+id/imageButton2"  
    android:layout_width="120dp"  
    android:layout_height="82dp"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.459"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.466"  
    app:srcCompat="@android:drawable/btn_star_big_on" />  
  
<ImageView  
    android:id="@+id/imageView2"  
    android:layout_width="357dp"  
    android:layout_height="136dp"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.459"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.7"  
    app:srcCompat="@android:drawable/sym_def_app_icon" />  
  
<Button  
    android:id="@+id/button2"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Enter"
```

## Java File:-

The screenshot shows the Android Studio interface with the code editor open to the `MainActivity.java` file. The Java code defines a new activity named `MainActivity` that extends `AppCompatActivity`. The `onCreate` method is overridden to set the content view to the `R.layout.activity_main` layout.

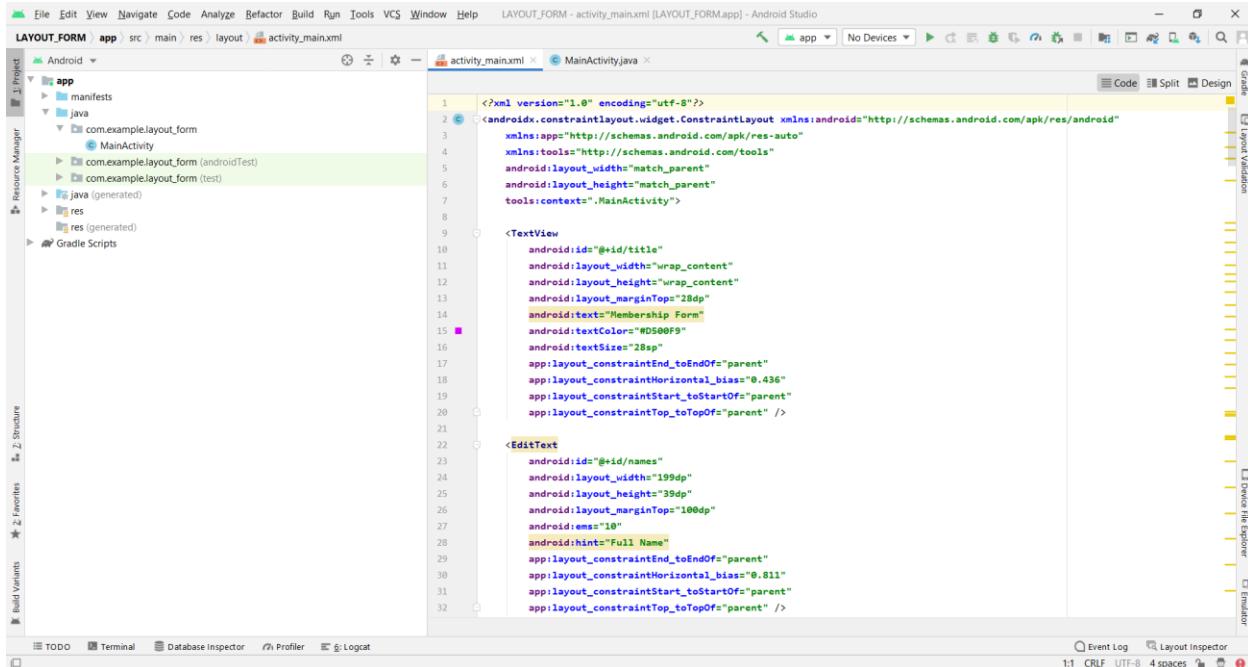
```
package com.example.myapplication;  
  
import ...  
  
public class MainActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
}
```

# OUTPUT:-



# Practical 5:- By using all Layouts design the form

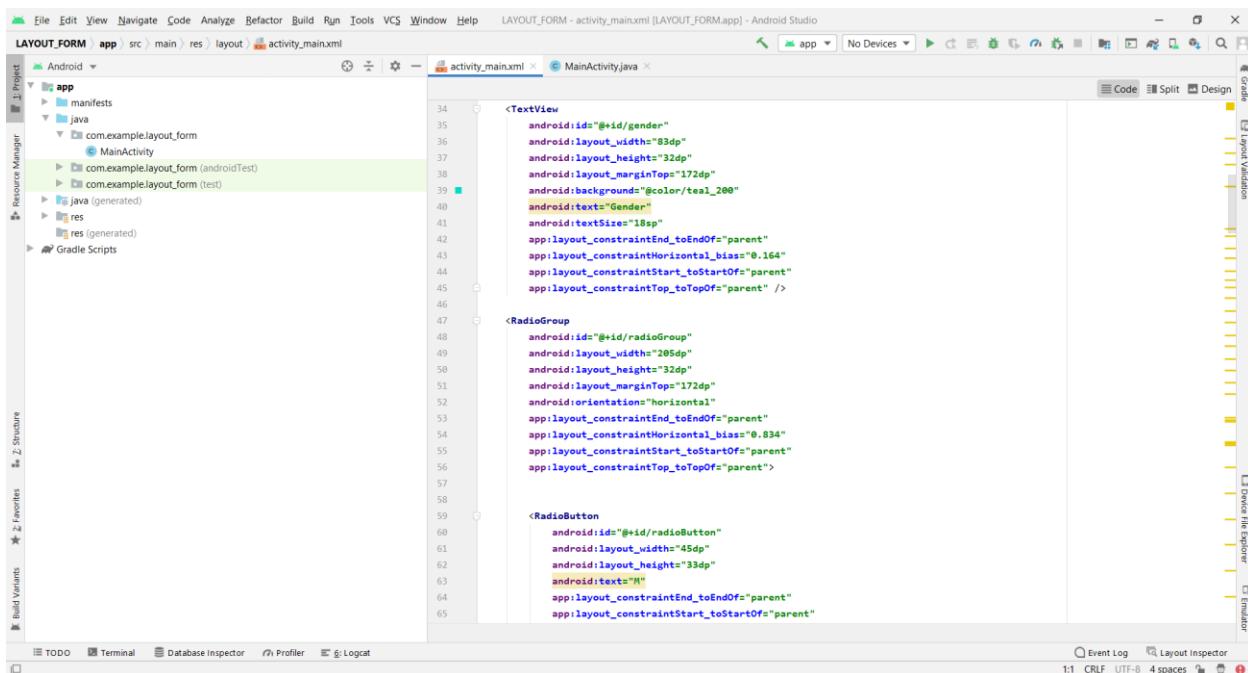
## Xml File:-



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

    <TextView
        android:id="@+id/title"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="28dp"
        android:text="Membership Form"
        android:textColor="#0500F9"
        android:textSize="28sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.436"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/names"
        android:layout_width="199dp"
        android:layout_height="39dp"
        android:layout_marginTop="100dp"
        android:ems="10"
        android:hint="Full Name"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.811"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```



```
<Textview
    android:id="@+id/gender"
    android:layout_width="83dp"
    android:layout_height="32dp"
    android:layout_marginTop="172dp"
    android:background="@color/teal_200"
    android:text="Gender"
    android:textSize="18sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.164"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<RadioGroup
    android:id="@+id/radioGroup"
    android:layout_width="205dp"
    android:layout_height="32dp"
    android:layout_marginTop="172dp"
    android:orientation="horizontal"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.834"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<RadioButton
    android:id="@+id/radioButton"
    android:layout_width="45dp"
    android:layout_height="33dp"
    android:text="M"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
```

The screenshot shows the Android Studio interface with the project 'LAYOUT\_FORM' open. The left sidebar displays the project structure under 'app'. The main editor window shows the XML code for 'activity\_main.xml'. The code defines a RadioGroup containing two RadioButtons ('female' and 'other') and an EditText ('current\_weight'). The XML includes various attributes like android:id, android:layout\_width, android:layout\_height, and android:layout\_marginTop. The code is color-coded for syntax highlighting.

```
67     android:onClick="radioButtonhandler"
68     tools:ignore="OnClick" />
69
70 <RadioButton
71     android:id="@+id/female"
72     android:layout_width="45dp"
73     android:layout_height="33dp"
74     android:text="F"
75     app:layout_constraintEnd_toEndOf="parent"
76     app:layout_constraintHorizontal_bias="0.655"
77     app:layout_constraintStart_toStartOf="parent"
78     app:layout_constraintTop_toTopOf="parent"
79     android:onClick="radioButtonhandler"
80     tools:ignore="OnClick" />
81
82 <RadioButton
83     android:id="@+id/other"
84     android:layout_width="96dp"
85     android:layout_height="33dp"
86     android:text="Other"
87     app:layout_constraintEnd_toEndOf="parent"
88     app:layout_constraintHorizontal_bias="0.949"
89     app:layout_constraintStart_toStartOf="parent"
90     app:layout_constraintTop_toTopOf="parent"
91     android:onClick="radioButtonhandler"
92     tools:ignore="OnClick" />
93
94 </RadioGroup>
95
96 <EditText
97     android:id="@+id/current_weight"
98     android:layout_width="204dp"
99     android:layout_height="44dp"
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
```

The screenshot shows the Android Studio interface with the project 'LAYOUT\_FORM' open. The left sidebar displays the project structure under 'app'. The main editor window shows the XML code for 'activity\_main.xml'. The code defines two EditText fields ('height' and 'age') with specific dimensions, margins, and input types. The XML includes attributes like android:id, android:layout\_width, android:layout\_height, and android:layout\_marginTop. The code is color-coded for syntax highlighting.

```
99     android:layout_marginTop="224dp"
100    android:ems="10"
101    android:hint="Current Weight"
102    android:inputType="number"
103    app:layout_constraintEnd_toEndOf="parent"
104    app:layout_constraintHorizontal_bias="0.917"
105    app:layout_constraintStart_toStartOf="parent"
106    app:layout_constraintTop_toTopOf="parent" />
107
108 <EditText
109     android:id="@+id/height"
110     android:layout_width="193dp"
111     android:layout_height="45dp"
112     android:layout_marginTop="292dp"
113     android:ems="10"
114     android:hint="Height"
115     android:inputType="number"
116     app:layout_constraintEnd_toEndOf="parent"
117     app:layout_constraintHorizontal_bias="0.876"
118     app:layout_constraintStart_toStartOf="parent"
119     app:layout_constraintTop_toTopOf="parent" />
120
121 <EditText
122     android:id="@+id/age"
123     android:layout_width="190dp"
124     android:layout_height="45dp"
125     android:layout_marginTop="372dp"
126     android:ems="10"
127     android:hint="Age"
128     android:inputType="number"
129     app:layout_constraintEnd_toEndOf="parent"
130     app:layout_constraintHorizontal_bias="0.864"
```

The screenshot shows the Android Studio interface with the project 'LAYOUT\_FORM' open. The main window displays the XML code for 'activity\_main.xml'. The code defines three UI components: an EditText for a phone number, an EditText for an address, and a CheckBox for accepting terms. The XML uses standard Android and AppCompat attributes for styling and layout constraints.

```
<EditText  
    android:id="@+id/Phone"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_marginTop="440dp"  
    android:ems="10"  
    android:hint="Phone"  
    android:inputType="phone"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.945"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent" />  
  
<EditText  
    android:id="@+id/address"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_marginTop="516dp"  
    android:ems="10"  
    android:hint="Address"  
    android:inputType="text"  
    android:maxLines="2"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.92"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent" />  
  
<CheckBox  
    android:id="@+id/conditions"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_marginTop="580dp"  
    android:layout_marginBottom="50dp" />
```

This screenshot shows the same XML code for 'activity\_main.xml' as the first one, but with several lines highlighted in various colors (pink, blue, green). These highlights likely indicate specific annotations or regions of interest for review or analysis. The code itself remains identical to the first screenshot.

```
166: android:backgroundTint="#0500F9"  
167: android:buttonTint="#D500F9"  
168: android:text="I have read,understood and accepted Membership Rules."  
169: app:layout_constraintEnd_toEndOf="parent"  
170: app:layout_constraintHorizontal_bias="0.485"  
171: app:layout_constraintStart_toStartOf="parent"  
172: app:layout_constraintTop_toTopOf="parent" />  
  
<Button  
    android:id="@+id/button"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_marginTop="684dp"  
    android:backgroundTint="#0500F9"  
    android:onClick="submitbuttonHandler"  
    android:text="SUBMIT"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.425"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    tools:ignore="OnClick" />  
  
<TextView  
    android:id="@+id/textView2"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:background="#color:teal_200"  
    android:text="Full Name"  
    android:textSize="18sp"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.147"
```

```
<TextView  
    android:id="@+id/textView3"  
    android:layout_width="93dp"  
    android:layout_height="33dp"  
    android:background="@color/teal_200"  
    android:text="Phone No."  
    android:textSize="18sp"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.091"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.632" />  
  
<TextView  
    android:id="@+id/textView4"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:background="@color/teal_200"  
    android:text="Address"  
    android:textSize="18sp"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.142"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.742" />  
  
<TextView  
    android:id="@+id/textView5"  
    android:layout_width="90dp"  
    android:layout_height="28dp"
```

```
        android:background="@color/teal_200"  
        android:text="Age"  
        android:textSize="18sp"  
        app:layout_constraintBottom_toBottomOf="parent"  
        app:layout_constraintEnd_toEndOf="parent"  
        app:layout_constraintHorizontal_bias="0.099"  
        app:layout_constraintStart_toStartOf="parent"  
        app:layout_constraintTop_toTopOf="parent"  
        app:layout_constraintVertical_bias="0.553" />  
  
<TextView  
    android:id="@+id/textView6"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:background="@color/teal_200"  
    android:text="Weight"  
    android:textSize="18sp"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.168"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.335" />  
  
<TextView  
    android:id="@+id/textView7"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:background="@color/teal_200"  
    android:text="Height"  
    android:textSize="18sp"  
    app:layout_constraintBottom_toBottomOf="parent"
```

# OUTPUT:-

The screenshot shows a mobile application interface for a "Membership Form". The top status bar indicates the time is 8:05 PM, the data speed is 43.6KB/s, and battery level is 50%. The title "LAYOUT\_FORM" is displayed above the form fields.

**Membership Form**

Full Name Jaspreet (1706861)

Gender  M  F  Other

Weight 50

Height 5

Age 21

Phone No. 7986763507

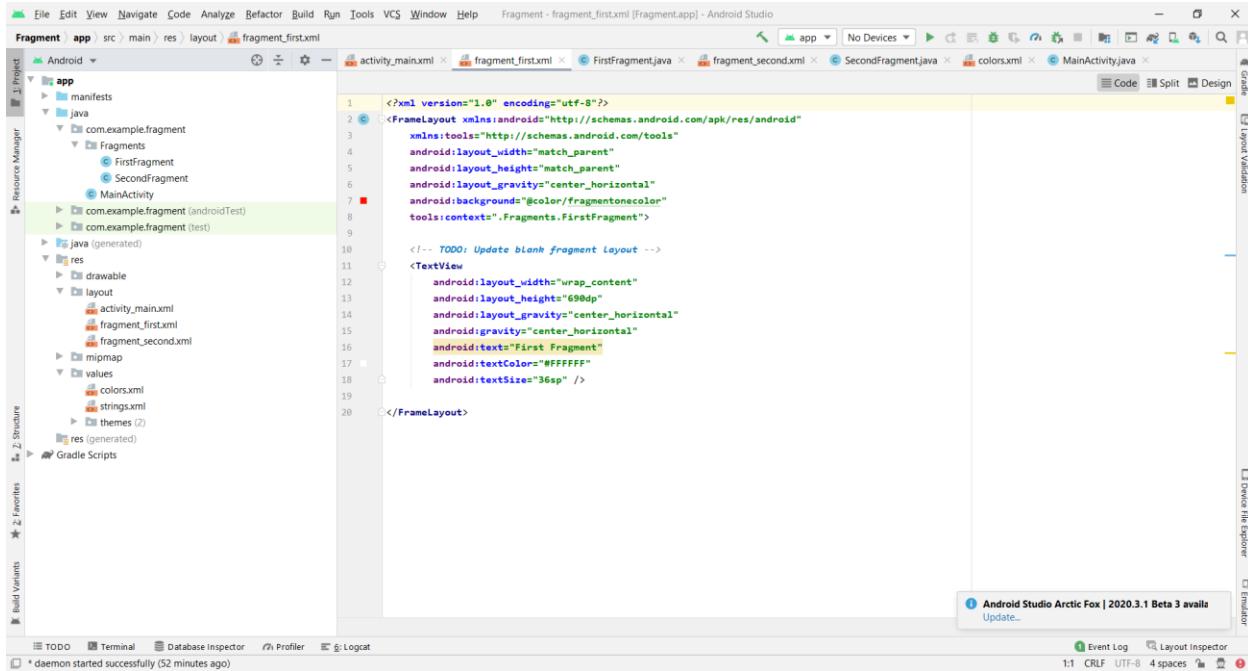
Address Samrala

I have read, understood and accepted Membership Rules

**SUBMIT**

# Practical 6:- Create a Fragment

## Xml File:-

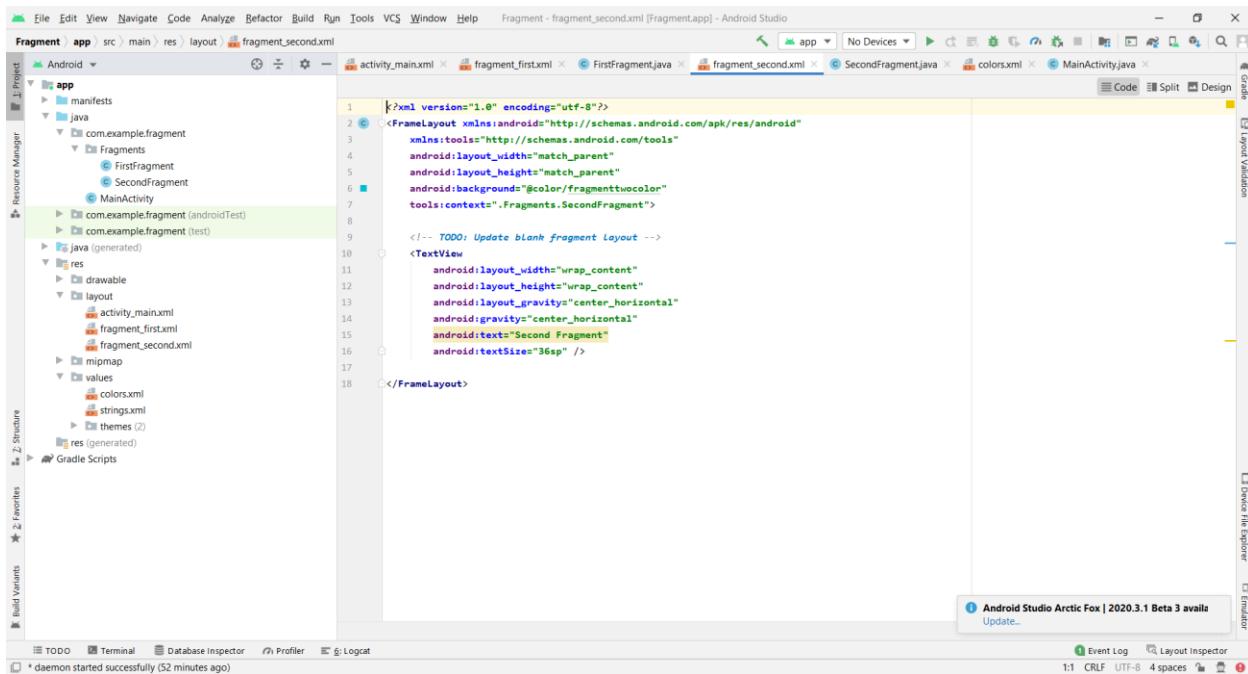


The screenshot shows the Android Studio interface with the project 'Fragment' open. The 'fragment\_first.xml' file is selected in the navigation bar. The code editor displays the XML layout for the first fragment:

```
<?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"
    android:layout_gravity="center_horizontal"
    android:background="@color/fragmentonecolor"
    tools:context=".Fragments.FirstFragment">

    <!-- TODO: Update blank fragment layout -->
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="690dp"
        android:layout_gravity="center_horizontal"
        android:gravity="center_horizontal"
        android:text="First Fragment"
        android:textColor="#FFFFFF"
        android:textSize="36sp" />
</FrameLayout>
```

The code editor has syntax highlighting for XML tags and Java code. The status bar at the bottom indicates 'daemon started successfully (52 minutes ago)'.



The screenshot shows the Android Studio interface with the project 'Fragment' open. The 'fragment\_second.xml' file is selected in the navigation bar. The code editor displays the XML layout for the second fragment:

```
<?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"
    android:background="@color/fragmenttwo_color"
    tools:context=".Fragments.SecondFragment">

    <!-- TODO: Update blank fragment layout -->
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:gravity="center_horizontal"
        android:text="Second Fragment"
        android:textSize="36sp" />
</FrameLayout>
```

The code editor has syntax highlighting for XML tags and Java code. The status bar at the bottom indicates 'daemon started successfully (52 minutes ago)'.

The screenshot shows the Android Studio interface with the XML layout file `activity_main.xml` open. The code defines a vertical linear layout containing two buttons, each with a different background color and text. The layout uses wrap\_content for most dimensions and match\_parent for others.

```
<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="wrap_content"
    android:background="@color/teal_200"
    android:orientation="vertical"
    tools:context=".MainActivity"

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="@color/purple_700"
        android:orientation="horizontal"

        <Button
            android:id="@+id/btnFirst"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="5dp"
            android:layout_weight="1"
            android:text="FIRST"
            android:textSize="24sp"
            app:backgroundTint="#FFEB3B" />

        <Button
            android:id="@+id/btnSecond"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="5dp"
            android:layout_weight="1"
            android:text="SECOND"
            android:textSize="24sp"
            app:backgroundTint="@color/fragmentttwoColor" />
    </LinearLayout>
</LinearLayout>
```

## Java File:-

The screenshot shows the Android Studio interface with the Java code for `FirstFragment`. The code defines a fragment class that inflates a layout from `fragment_first.xml`.

```
package com.example.fragment.Fragments;

import ...

public class FirstFragment extends Fragment {

    public FirstFragment() {
        // Required empty public constructor
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                           Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        View view = inflater.inflate(R.layout.fragment_first, container, false);

        return view;
    }
}
```

The screenshot shows the Android Studio interface with the SecondFragment.java file open in the main editor. The code defines a fragment named SecondFragment that inflates a layout from R.layout.fragment\_second. The Java code is as follows:

```
package com.example.fragment.Fragments;
import ...;

public class SecondFragment extends Fragment {

    public SecondFragment() {
        // Required empty public constructor
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                           Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        View view= inflater.inflate(R.layout.fragment_second, container, false);

        return view;
    }
}
```

The Project tool window on the left shows the project structure with files like activity\_main.xml, fragment\_first.xml, FirstFragment.java, fragment\_second.xml, SecondFragment.java, colors.xml, strings.xml, and MainActivity.java. The Java tab in the bottom navigation bar is selected.

The screenshot shows the Android Studio interface with the MainActivity.java file open in the main editor. The code sets up two buttons (btnFirst, btnSecond) and a linear layout (linearLayout). It then handles button clicks to replace the first fragment with either the first or second fragment. The Java code is as follows:

```
package com.example.fragment;
import ...;

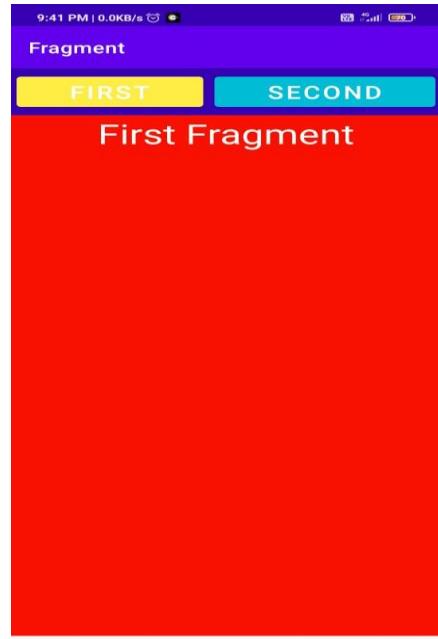
public class MainActivity extends AppCompatActivity {
    Button btnFirst,btnSecond;
    LinearLayout linearLayout;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

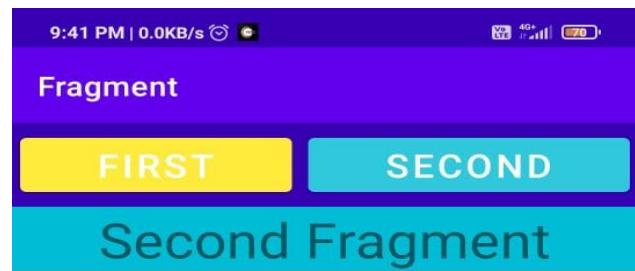
        btnFirst=findViewByIdById(R.id.btnFirst);
        btnSecond=findViewByIdById(R.id.btnSecond);
        linearLayout=findViewByIdById(R.id.linearlayout);

        btnFirst.setOnClickListener(new View.OnClickListener(){
            @Override
            public void onClick(View v){
                FirstFragment firstFragment=new FirstFragment();
                FragmentTransaction transaction=getSupportFragmentManager().beginTransaction();
                transaction.replace(R.id.linearlayout,firstFragment);
                transaction.commit();
            }
        });
        btnSecond.setOnClickListener(new View.OnClickListener(){
            @Override
            public void onClick(View v){
                SecondFragment firstFragment=new SecondFragment();
                FragmentTransaction transaction=getSupportFragmentManager().beginTransaction();
                transaction.replace(R.id.linearlayout,firstFragment);
                transaction.commit();
            }
        });
    }
}
```

The Project tool window on the left shows the project structure with files like activity\_main.xml, fragment\_first.xml, FirstFragment.java, fragment\_second.xml, SecondFragment.java, colors.xml, strings.xml, and MainActivity.java. The Java tab in the bottom navigation bar is selected.

# OUTPUT:-

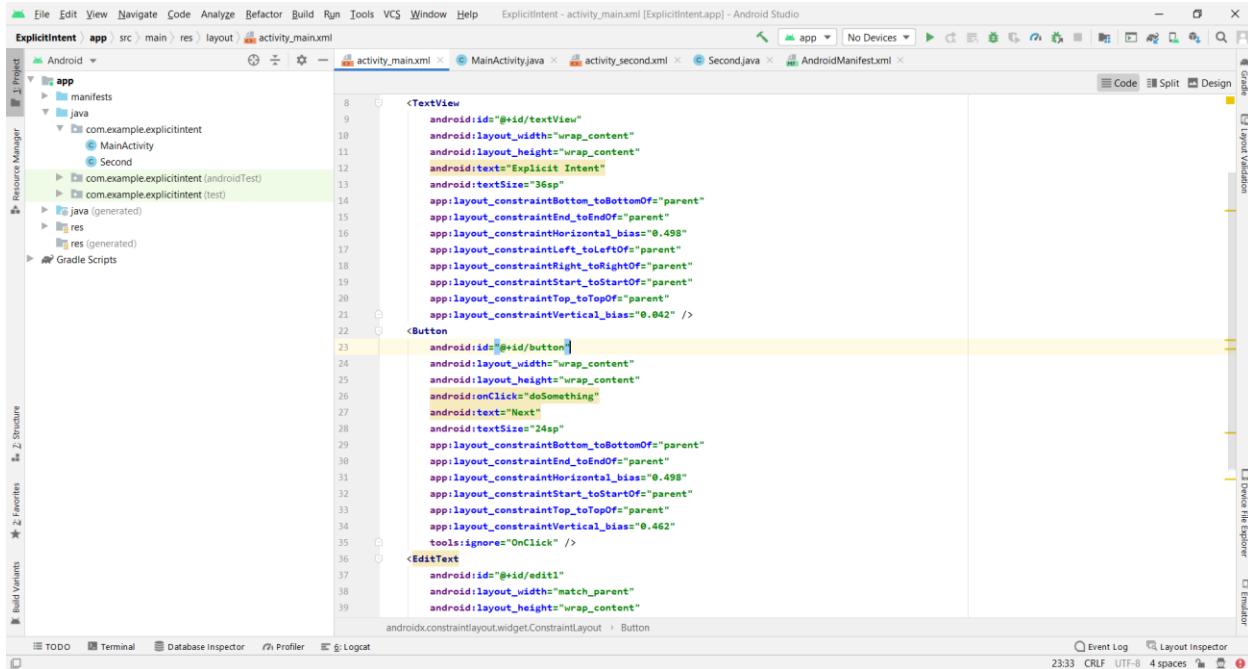




---

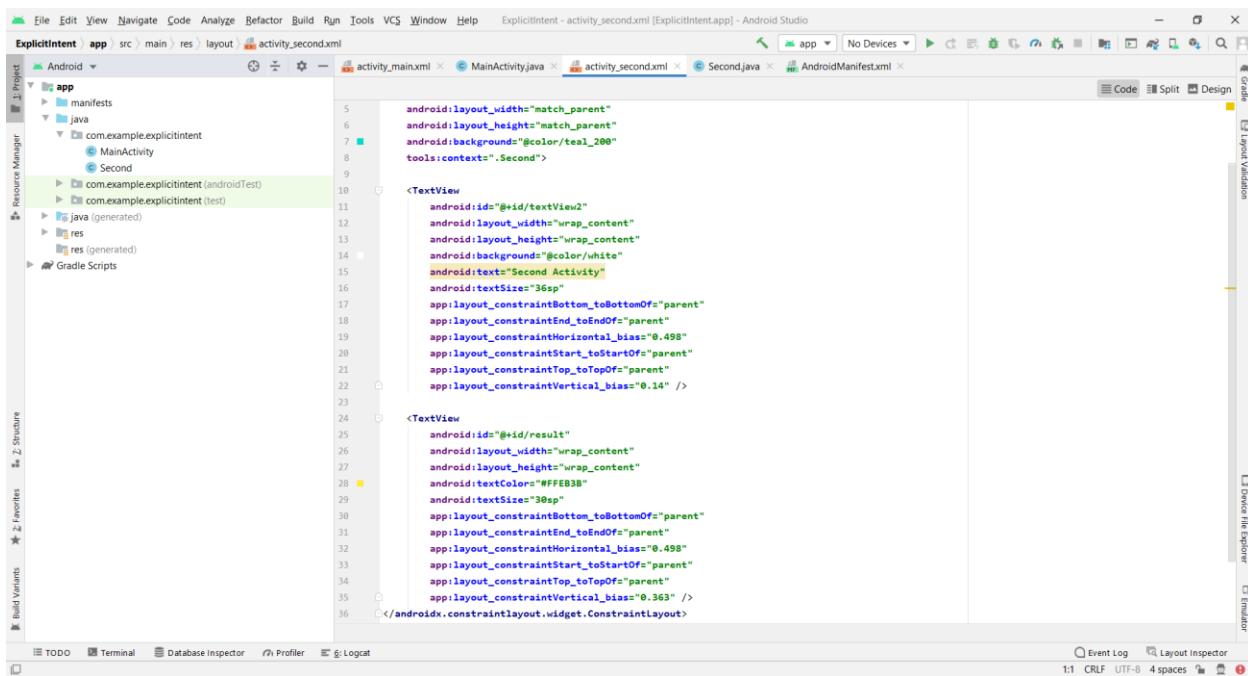
# Practical 7:- Create an activity that links two activities.

## Xml File:-



The screenshot shows the Android Studio interface with the project 'ExplicitIntent' open. The main window displays the XML code for 'activity\_main.xml'. The code defines a ConstraintLayout containing a TextView, a Button, and an EditText. The Button has an onClick attribute set to 'doSomething'. The XML uses various constraints like 'app:layout\_constraintBottom\_toBottomOf="parent"' and 'app:layout\_constraintHorizontal\_bias="0.498"'. The code is as follows:

```
<TextView  
    android:id="@+id/textView"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Explicit Intent"  
    android:textSize="36sp"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.498"  
    app:layout_constraintLeft_toLeftOf="parent"  
    app:layout_constraintRight_toRightOf="parent"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.042" />  
  
<Button  
    android:id="@+id/button"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:onClick="doSomething"  
    android:text="Next"  
    android:textSize="24sp"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.498"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.462"  
    tools:ignore="OnClick" />  
  
<EditText  
    android:id="@+id/editText"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    androidx.constraintlayout.widget.ConstraintLayout > Button
```



The screenshot shows the Android Studio interface with the project 'ExplicitIntent' open. The main window displays the XML code for 'activity\_second.xml'. This XML defines a ConstraintLayout with two TextViews. The first TextView's text is 'Second Activity' and its color is '#FFEB3B'. The second TextView's text is 'Result' and its color is '#00897B'. Both TextViews have a size of 30sp and are constrained to the parent with various bias values. The code is as follows:

```
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:background="#color/teal_200"  
    tools:context=".Second">  
  
<TextView  
    android:id="@+id/textView2"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:background="#color/white"  
    android:text="Second Activity"  
    android:textSize="36sp"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.498"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.14" />  
  
<TextView  
    android:id="@+id/result"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:textColor="#00897B"  
    android:textSize="30sp"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.498"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.363" />  
  </androidx.constraintlayout.widget.ConstraintLayout>
```

## Java File:-

The screenshot shows the Android Studio interface with the project 'ExplicitIntent' open. The 'Second.java' file is selected in the top navigation bar. The code in the editor is:

```
1 package com.example.explicitintent;
2
3 import ...
4
5 public class Second extends AppCompatActivity {
6     TextView txt1;
7     @Override
8     protected void onCreate(Bundle savedInstanceState) {
9         super.onCreate(savedInstanceState);
10        setContentView(R.layout.activity_second);
11        txt1=(TextView)findViewById(R.id.result);
12
13        Bundle b1= getIntent().getExtras();
14        String s1=b1.getString( key: "user");
15        txt1.setText(s1);
16    }
17}
```

The 'Resource Manager' tool window on the left shows the project structure with 'Second.java' highlighted. The bottom status bar indicates '1:1 CRLF UTF-8 4 spaces'.

The screenshot shows the Android Studio interface with the project 'ExplicitIntent' open. The 'MainActivity.java' file is selected in the top navigation bar. The code in the editor is:

```
1 package com.example.explicitintent;
2
3 import ...
4
5 public class MainActivity extends AppCompatActivity {
6     EditText e1;
7     @Override
8     protected void onCreate(Bundle savedInstanceState) {
9         super.onCreate(savedInstanceState);
10        setContentView(R.layout.activity_main);
11        e1 = (EditText)findViewById(R.id.edit1);
12
13
14
15
16
17
18
19
20
21
22
23
24
25 }
```

The 'Resource Manager' tool window on the left shows the project structure with 'MainActivity.java' highlighted. The bottom status bar indicates '1:1 CRLF UTF-8 4 spaces'.

# OUTPUT:-

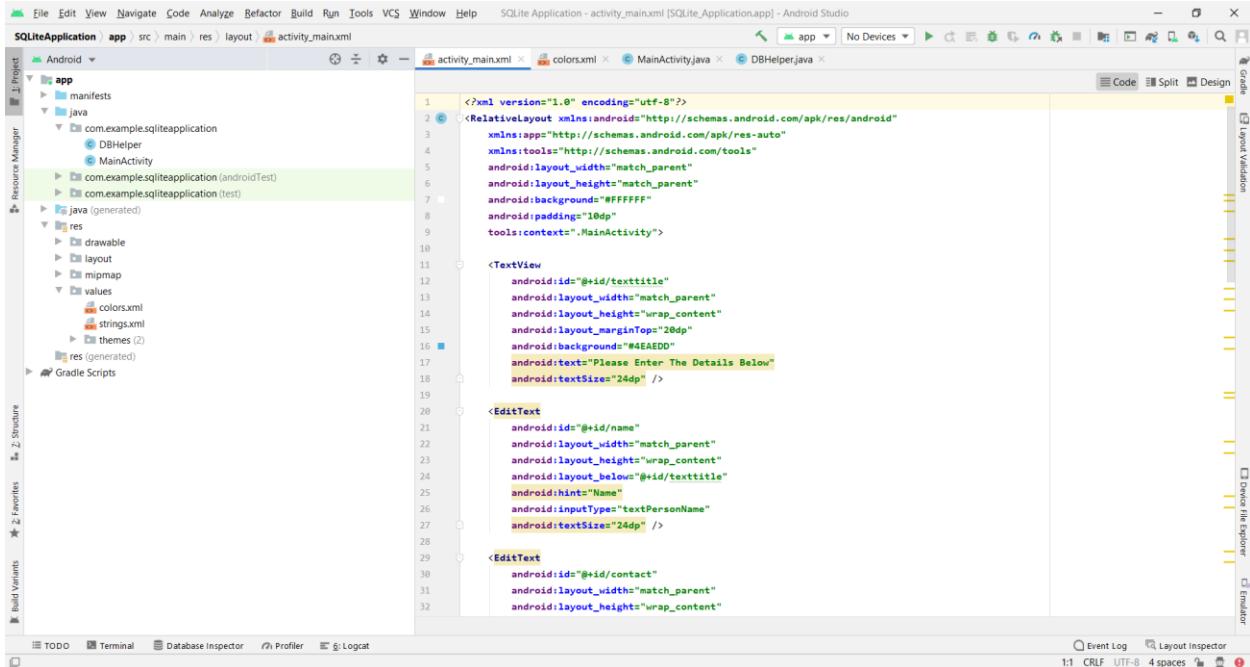


NEXT



# Practical 8 :- SQL APPLICATION

## Xml File



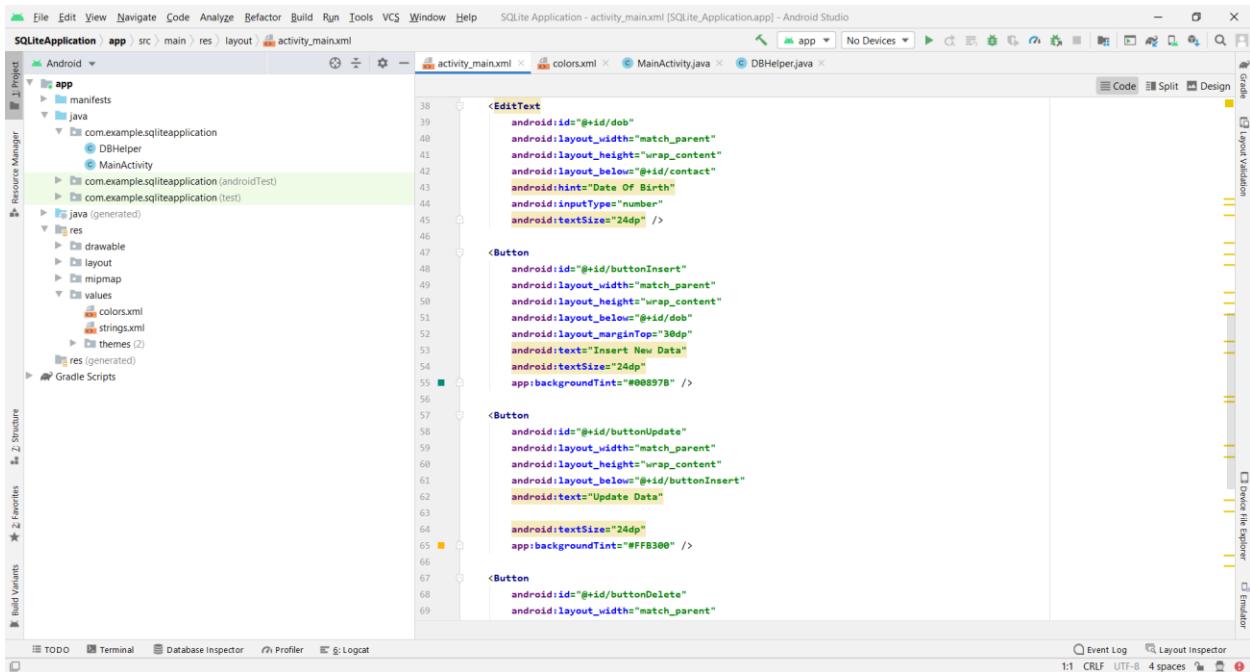
The screenshot shows the Android Studio interface with the project 'SQLiteApplication' open. The 'activity\_main.xml' file is selected in the navigation bar. The left sidebar displays the project structure, including the 'app' module with its Java and resources. The main editor pane shows the XML code for the activity layout:

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

    <TextView
        android:id="@+id/texttitle"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:background="#4EAEDD"
        android:text="Please Enter The Details Below"
        android:textSize="24dp" />

    <EditText
        android:id="@+id/name"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/texttitle"
        android:hint="Name"
        android:inputType="textPersonName"
        android:textSize="24dp" />

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



The screenshot shows the same Android Studio environment with the 'activity\_main.xml' file open. The XML code has been modified to include three buttons at the bottom:

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

    <EditText
        android:id="@+id/dob"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/contact"
        android:hint="Date Of Birth"
        android:inputType="number"
        android:textSize="24dp" />

    <Button
        android:id="@+id/buttonInsert"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/dob"
        android:layout_marginTop="3dp"
        android:text="Insert New Data"
        android:textSize="24dp"
        app:backgroundTint="#00897B" />

    <Button
        android:id="@+id/buttonUpdate"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/buttonInsert"
        android:text="Update Data"
        android:textSize="24dp"
        app:backgroundTint="#FFB300" />

    <Button
        android:id="@+id/buttonDelete"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/buttonUpdate" />
```

```

<Button
    android:id="@+id/buttonUpdate"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/buttonDelete"
    android:text="Update Data"
    android:textSize="24dp"
    app:backgroundTint="#FFB300" />

<Button
    android:id="@+id/buttonDelete"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@+id/buttonUpdate"
    android:text="Delete Data"
    android:textSize="24dp"
    app:backgroundTint="#D81B60" />

```

## Java File:-

```

public class MainActivity extends AppCompatActivity {
    EditText name,contact,dob;
    Button insert,update,delete,view;
    DBHelper DB;

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

        name=findViewById(R.id.name);
        contact=findViewById(R.id.contact);
        dob=findViewById(R.id.dob);

        insert=findViewById(R.id.buttonInsert);
        update=findViewById(R.id.buttonUpdate);
        delete=findViewById(R.id.buttonDelete);
        view=findViewById(R.id.buttonView);
        DB=new DBHelper(context: this);

        insert.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String nameTXT=name.getText().toString();
                String contactTXT=contact.getText().toString();
                String dobTXT=dob.getText().toString();

                Boolean checkinsertdata=DB.insertuserdata(nameTXT,contactTXT,dobTXT);
                if(checkinsertdata==true){
                    Toast.makeText( context: MainActivity.this, text: "New Entry Inseted",Toast.LENGTH_SHORT).show();
                }else{
                    Toast.makeText( context: MainActivity.this, text: "New Entry Not Inseted",Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}

```

SQLite Application - MainActivity.java [SQLite\_Application.app] - Android Studio

```
activity_main.xml colors.xml MainActivity.java DBHelper.java
```

```
update.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String nameTXT=name.getText().toString();
        String contactTXT=contact.getText().toString();
        String dobTXT=dob.getText().toString();

        Boolean checkupdatedata=DB.updateuserdata(nameTXT,contactTXT,dobTXT);
        if(checkupdatedata==true){
            Toast.makeText(context: MainActivity.this, text: "Entry Updated",Toast.LENGTH_SHORT).show();
        }else{
            Toast.makeText(context: MainActivity.this, text: "Entry Not Updated",Toast.LENGTH_SHORT).show();
        }
    }
});
```

```
delete.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String nameTXT=name.getText().toString();

        Boolean checkdeletedata=DB.deletedata(nameTXT);
        if(checkdeletedata==true){
            Toast.makeText(context: MainActivity.this, text: "Entry Deleted",Toast.LENGTH_SHORT).show();
        }else{
            Toast.makeText(context: MainActivity.this, text: "Entry Not Deleted",Toast.LENGTH_SHORT).show();
        }
    }
});
```

```
view.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Cursor res=DB.getdata();
        if(res.getCount()==0){
```

Event Log Layout Inspector

1:1 CRLF UTF-8 4 spaces

SQLite Application - MainActivity.java [SQLite\_Application.app] - Android Studio

```
activity_main.xml colors.xml MainActivity.java DBHelper.java
```

```
        Toast.makeText(context: MainActivity.this, text: "Entry Not Deleted",Toast.LENGTH_SHORT).show();
    }
});
```

```
view.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Cursor res=DB.getdata();
        if(res.getCount()==0){
            Toast.makeText(context: MainActivity.this, text: "No Entry Exists",Toast.LENGTH_SHORT).show();
            return;
        }
        StringBuffer buffer=new StringBuffer();
        while (res.moveToNext()){
            buffer.append("Name :"+res.getString( columnIndex: 0)+"\n");
            buffer.append("Contact :"+res.getString( columnIndex: 1)+"\n");
            buffer.append("Date of Birth :"+res.getString( columnIndex: 2)+"\n\n");
        }
        AlertDialog.Builder builder=new AlertDialog.Builder( context: MainActivity.this);
        builder.setCancelable(true);
        builder.setTitle("User Entries");
        builder.setMessage(buffer.toString());
        builder.show();
    }
});
```

Event Log Layout Inspector

1:1 CRLF UTF-8 4 spaces

```
public class DBHelper extends SQLiteOpenHelper {
    public DBHelper(Context context) { super(context, name: "userdata.db", factory: null, version: 1); }

    @Override
    public void onCreate(SQLiteDatabase DB) {
        DB.execSQL("create Table Userdetails(name TEXT primary key, contact TEXT, dob TEXT)");
    }

    @Override
    public void onUpgrade(SQLiteDatabase DB, int i, int ii) {
        DB.execSQL("drop Table if exists Userdetails");
    }

    public Boolean insertuserdata(String name, String contact, String dob) {
        SQLiteDatabase DB=this.getWritableDatabase();
        ContentValues contentValues=new ContentValues();
        contentValues.put("name",name);
        contentValues.put("contact",contact);
        contentValues.put("dob",dob);
        long result=DB.insert( table: "Userdetails", nullColumnHack: null, contentValues);
        if(result== -1){
            return false;
        }
        else{
            return true;
        }
    }

    public Boolean updateuserdata(String name, String contact, String dob) {
        SQLiteDatabase DB=this.getWritableDatabase();
        ContentValues contentValues=new ContentValues();

        contentValues.out("contact".contact);
    }
}
```

```
}

    public Boolean deletedata(String name) {
        SQLiteDatabase DB=this.getWritableDatabase();

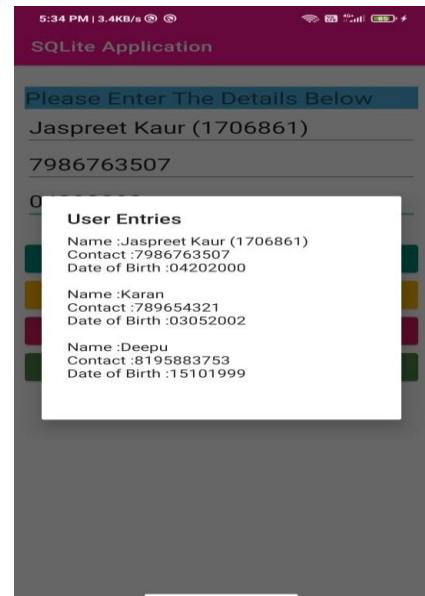
        Cursor cursor=DB.rawQuery( sql: "Select * from Userdetails where name = ? ", new String[] { name });
        if(cursor.getCount() > 0) {

            long result = DB.delete( table: "Userdetails", whereClause: "name=?", new String[] { name });
            if (result == -1) {
                return false;
            } else {
                return true;
            }
        } else {
            return false;
        }
    }

    public Cursor getdata() {
        SQLiteDatabase DB=this.getWritableDatabase();

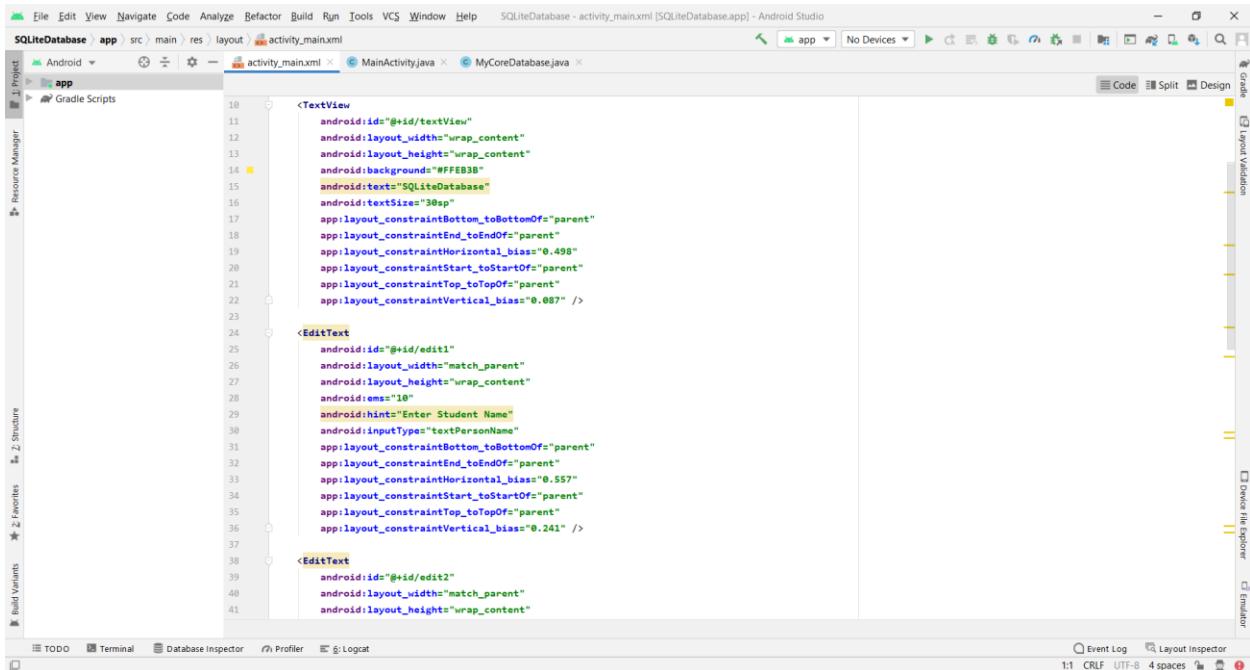
        Cursor cursor=DB.rawQuery( sql: "Select * from Userdetails", selectionArgs: null );
        return cursor;
    }
}
```

# OUTPUT:-



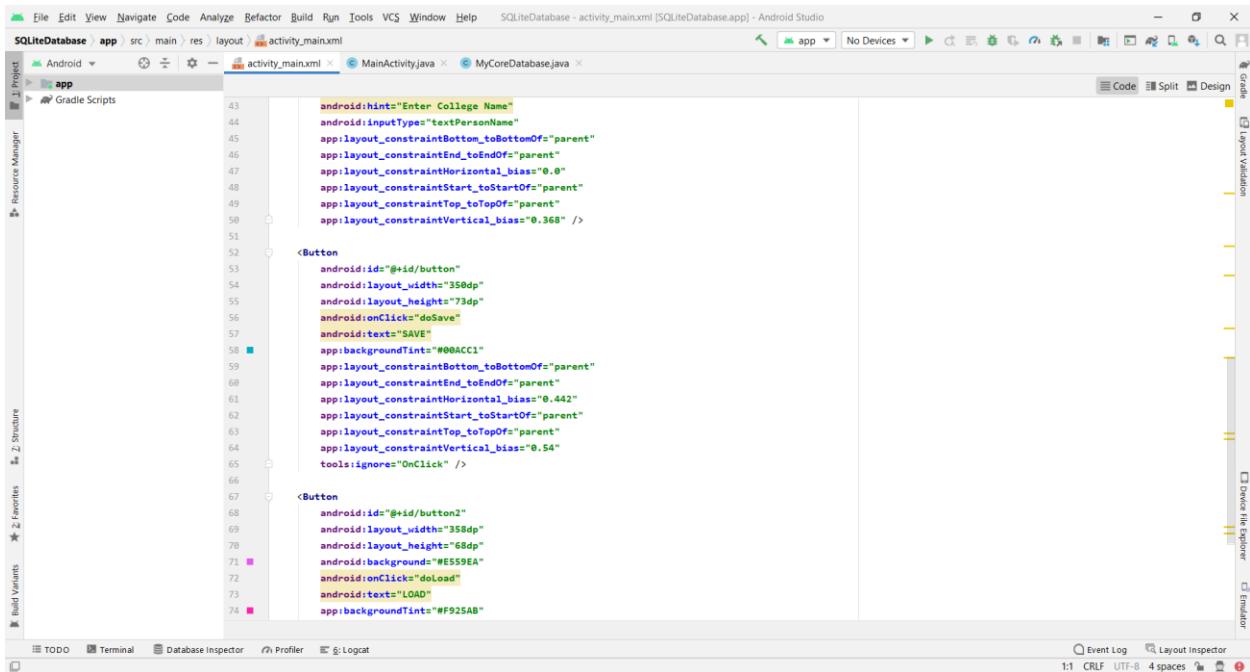
# Practical 9 :- Store Data Using SQLite Database

## Xml File:-



The screenshot shows the Android Studio interface with the XML code for `activity_main.xml`. The code defines a layout with three text views and two edit texts. The first text view has a placeholder "SQLiteDatabase". The second edit text has a placeholder "Enter Student Name". The third edit text has a placeholder "Enter College Name".

```
<TextView  
    android:id="@+id/textView"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:background="#FFEB3B"  
    android:text="SQLiteDatabase"  
    android:textSize="30sp"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.498"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.087" />  
  
<EditText  
    android:id="@+id/edit1"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:ems="10"  
    android:hint="Enter Student Name"  
    android:inputType="textPersonName"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.557"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.241" />  
  
<EditText  
    android:id="@+id/edit2"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"
```



The screenshot shows the Android Studio interface with the XML code for `activity_main.xml`. It includes two buttons: one for saving data and another for loading data. The save button has a placeholder "SAVE" and a background color "#00ACC1". The load button has a placeholder "LOAD" and a background color "#F925AB".

```
        android:hint="Enter College Name"  
        android:inputType="textPersonName"  
        app:layout_constraintBottom_toBottomOf="parent"  
        app:layout_constraintEnd_toEndOf="parent"  
        app:layout_constraintHorizontal_bias="0.0"  
        app:layout_constraintStart_toStartOf="parent"  
        app:layout_constraintTop_toTopOf="parent"  
        app:layout_constraintVertical_bias="0.368" />  
  
<Button  
    android:id="@+id/button"  
    android:layout_width="350dp"  
    android:layout_height="73dp"  
    android:onClick="doSave"  
    android:text="SAVE"  
    app:backgroundTint="#00ACC1"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.442"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.54"  
    tools:ignore="OnClick" />  
  
<Button  
    android:id="@+id/button2"  
    android:layout_width="350dp"  
    android:layout_height="68dp"  
    android:background="#E559E4"  
    android:onClick="doLoad"  
    android:text="LOAD"  
    app:backgroundTint="#F925AB"
```

## Java File:-

The screenshot shows the Android Studio interface with the code editor open to the MyCoreDatabase.java file. The code implements a SQLiteOpenHelper class named MyCoreDatabase. It defines static final variables for the database name ('DB\_NAME'), table name ('DB\_TABLE'), and version ('DB\_VER'). The constructor takes a Context parameter and initializes the SQLiteDatabase object. The onCreate method creates the table if it doesn't exist. The onUpgrade method drops the table if it exists and then recreates it. The insertData method inserts data into the table. The getAll method retrieves all data from the table and displays it in a toast message.

```
public class MyCoreDatabase extends SQLiteOpenHelper {
    static final private String DB_NAME="Education";
    static final private String DB_TABLE="students";
    static final private int DB_VER=1;
    Context ctx;
    SQLiteDatabase myDb;
    public MyCoreDatabase(Context ct){
        super(ct,DB_NAME, factory: null,DB_VER);
        ctx =ct;
    }
    @Override
    public void onCreate(SQLiteDatabase sqLiteDatabase){
        sqLiteDatabase.execSQL("create table "+DB_TABLE+" (_id integer primary key autoincrement,stu_name text,college_name text);");
        Log.i( tag: "Database", msg: "Table Created");
    }
    @Override
    public void onUpgrade(SQLiteDatabase sqLiteDatabase,int i,int ii){
        sqLiteDatabase.execSQL("drop table if exists "+DB_TABLE );
        onCreate(sqLiteDatabase);
    }
    public void insertData(String s1,String s2){
        myDb=getWritableDatabase();
        myDb.execSQL("insert into "+DB_TABLE+" (stu_name,college_name) values('"+s1+"','"+s2+"')");
        Toast.makeText(ctx, text: "Data Saved Successfully",Toast.LENGTH_SHORT).show();
    }
    public void getAll(){
        myDb=getReadableDatabase();
        Cursor cr=myDb.rawQuery( sql: "Select * from "+DB_TABLE , selectionArgs: null);
        StringBuilder str=new StringBuilder();
        while (cr.moveToFirst()){
            String s1=cr.getString( columnIndex: 1);
            String s2=cr.getString( columnIndex: 2);
            str.append(s1+
                    "+s2+"
                    "\n");
        }
        Toast.makeText(ctx,str.toString(),Toast.LENGTH_LONG).show();
    }
}
```

This screenshot shows the same Java code as above, but with modifications. The getAll() method has been updated to use a StringBuilder to append the retrieved data instead of concatenating strings directly. This results in a more efficient and readable code. The rest of the logic remains the same, including the database creation and data insertion methods.

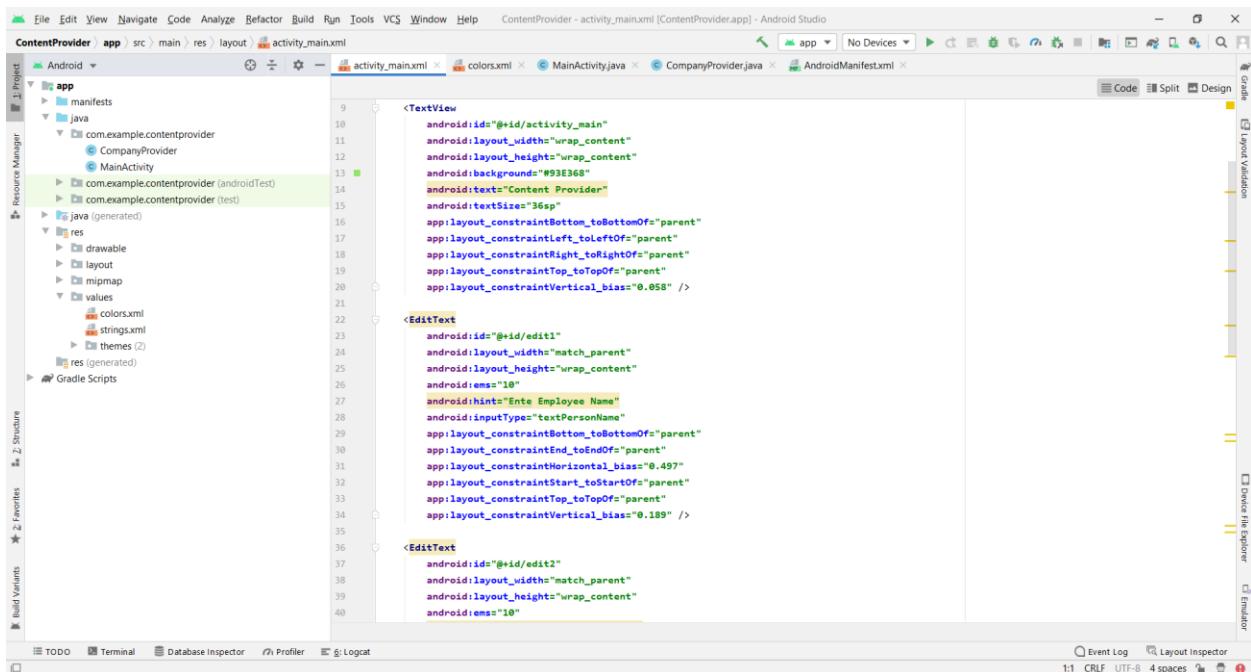
```
Log.i( tag: "Database", msg: "Table Created");
    @Override
    public void onUpgrade(SQLiteDatabase sqLiteDatabase,int i,int ii){
        sqLiteDatabase.execSQL("drop table if exists "+DB_TABLE );
        onCreate(sqLiteDatabase);
    }
    public void insertData(String s1,String s2){
        myDb=getWritableDatabase();
        myDb.execSQL("insert into "+DB_TABLE+" (stu_name,college_name) values('"+s1+"','"+s2+"')");
        Toast.makeText(ctx, text: "Data Saved Successfully",Toast.LENGTH_SHORT).show();
    }
    public void getAll(){
        myDb=getReadableDatabase();
        Cursor cr=myDb.rawQuery( sql: "Select * from "+DB_TABLE , selectionArgs: null);
        StringBuilder str=new StringBuilder();
        while (cr.moveToFirst()){
            String s1=cr.getString( columnIndex: 1);
            String s2=cr.getString( columnIndex: 2);
            str.append(s1+
                    "+s2+"
                    "\n");
        }
        Toast.makeText(ctx,str.toString(),Toast.LENGTH_LONG).show();
    }
}
```

# OUTPUT:-



# PRACTICAL 10:- CONTENT PROVIDER

## Xml File:-



The screenshot shows the Android Studio interface with the project 'ContentProvider' open. The 'activity\_main.xml' file is selected in the top navigation bar. The left sidebar displays the project structure, including 'app' (manifests, java, com.example.contentprovider, res), 'Resource Manager' (values, strings.xml), and 'Gradle Scripts'. The main editor area contains the XML code for the layout:

```
<TextView  
    android:id="@+id/activity_main"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:background="#93E6EB"  
    android:text="Content Provider"  
    android:textSize="36sp"  
    android:layout_constraintBottom_toBottomOf="parent"  
    android:layout_constraintLeft_toLeftOf="parent"  
    android:layout_constraintRight_toRightOf="parent"  
    android:layout_constraintTop_toTopOf="parent"  
    android:layout_constraintVertical_bias="0.058" />  
  
<EditText  
    android:id="@+id/edit1"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:ems="10"  
    android:hint="Enter Employee Name"  
    android:inputType="textPersonName"  
    android:layout_constraintBottom_toBottomOf="parent"  
    android:layout_constraintEnd_toEndOf="parent"  
    android:layout_constraintHorizontal_bias="0.497"  
    android:layout_constraintStart_toStartOf="parent"  
    android:layout_constraintTop_toTopOf="parent"  
    android:layout_constraintVertical_bias="0.189" />  
  
<EditText  
    android:id="@+id/edit2"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:ems="10"
```

```

<Button
    android:id="@+id/button"
    android:layout_width="336dp"
    android:layout_height="60dp"
    android:onClick="doSaveContent"
    android:text="SAVE"
    app:backgroundTint="#DE57F6"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.29" />

<Button
    android:id="@+id/button2"
    android:layout_width="340dp"
    android:layout_height="64dp"
    android:onClick="doLoadContent"
    android:text="LOAD"
    app:backgroundTint="#6EBBFA"
    tools:ignore=".onClick" />

```

## Java File:-

```

public class MainActivity extends AppCompatActivity {
    EditText e1,e2;
    ContentValues values=new ContentValues();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        e1=(EditText)findViewById(R.id.edit1);
        e2=(EditText)findViewById(R.id.edit2);
    }

    public void doSaveContent(View view) {
        values.put("emp_name",e1.getText().toString());
        values.put("profile",e2.getText().toString());

        Uri uri=getContentResolver().insert(CompanyProvider.CONTENT_URI,values);
        Toast.makeText(context: this,uri.toString(),Toast.LENGTH_SHORT).show();
    }

    public void doLoadContent(View view) {
        Cursor cr =getContentResolver().query(CompanyProvider.CONTENT_URI, projection: null, selection: null, selectionArgs: null, sortOrder: "_id");
        StringBuilder stringBuilder=new StringBuilder();
        while (cr.moveToFirst()){
            int id=cr.getInt( columnIndex: 0);
            String s1=cr.getString( columnIndex: 1);
            String s2=cr.getString( columnIndex: 2);
            stringBuilder.append(id+" "+s1+" "+s2+"\n");
        }
        Toast.makeText(context: this,stringBuilder.toString(),Toast.LENGTH_SHORT).show();
    }
}

```

ContentProvider - CompanyProvider.java [ContentProvider.app] - Android Studio

```
public class CompanyProvider extends ContentProvider {
    SQLiteDatabase myDb;
    private static final String DB_NAME="company";
    private static final String DB_TABLE="emp";
    private static final int DB_VER=1;
    public CompanyProvider() {
    }

    public static final String AUTHORITY="com.lalit.my.company.provider";
    public static final Uri CONTENT_URI=Uri.parse("content://"+AUTHORITY+"/emp");

    public static final int EMP_ID=1;
    public static final int EMP_ID=2;

    static {
        myUri.addURI(AUTHORITY, path: "emp",EMP_ID);
        myUri.addURI(AUTHORITY, path: "emp/#",EMP_ID);
    }

    private class MyOwnDatabase extends SQLiteOpenHelper{
        public MyOwnDatabase(Context ct) { super(ct,DB_NAME, factory: null,DB_VER); }

        @Override
        public void onCreate(SQLiteDatabase sqLiteDatabase) {
            sqLiteDatabase.execSQL("create table "+DB_TABLE+" (_id integer primary key autoincrement,emp_name text,profile_text);");
        }

        @Override
        public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int ii) {
    
```

ContentProvider - CompanyProvider.java [ContentProvider.app] - Android Studio

```
@Override
public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int ii) {
    sqLiteDatabase.execSQL("drop table if exists "+DB_TABLE);
}

@Override
public int delete(Uri uri, String selection, String[] selectionArgs) {
    // Implement this to handle requests to delete one or more rows.
    throw new UnsupportedOperationException("Not yet implemented");
}

@Override
public String getType(Uri uri) {
    // TODO: Implement this to handle requests for the MIME type of the data
    // at the given URI.
    throw new UnsupportedOperationException("Not yet implemented");
}

@Override
public Uri insert(Uri uri, ContentValues values) {
    long row=myDb.insert(DB_TABLE, nullColumnHack: null,values);
    if(row>0){
        uri=ContentUri.withAppendedId(CONTENT_URI,row);
        getContext().getContentResolver().notifyChange(uri, observer: null);
    }
    return uri;
}

@Override
public boolean onCreate() {
    MyOwnDatabase myHelper=new MyOwnDatabase(getContext());
    myDb=myHelper.getWritableDatabase();
    if (myDb !=null){
    
```

## **OUTPUT:-**



# SERVICES

## Xml File:-

The screenshot shows the Android Studio interface with the project navigation bar at the top. Below it is the 'activity\_main.xml' file in the 'Layout' tab of the code editor. The XML code defines a ConstraintLayout with two buttons. The first button has an ID of '@+id/button' and an onClick event set to 'StartSomething'. The second button has an ID of '@+id/button2' and an onClick event set to 'StopSomething'. Both buttons have specific styling and constraints defined.

```
<?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">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="#000000"
        android:text="Services"
        android:textSize="30sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.139" />

    <Button
        android:id="@+id/button"
        android:layout_width="417dp"
        android:layout_height="63dp"
        android:onClick="StartSomething"
        android:text="Start Service"
        app:backgroundTint="#0099EE"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.866"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.377"
        tools:ignore="OnClick" />

    <Button
        android:id="@+id/button2"
        android:layout_width="414dp"
        android:layout_height="58dp"
        android:onClick="StopSomething"
        android:text="Stop Service"
        app:backgroundTint="#CC00AA"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.534"
        tools:ignore="OnClick" />

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

This screenshot shows the same XML layout file after modifications. The first button's onClick event is now 'StartService' and its background color is '#0099EE'. The second button's onClick event is now 'StopService' and its background color is '#CC00AA'. The rest of the XML structure remains the same.

```
<?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">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="#000000"
        android:text="Services"
        android:textSize="30sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.139" />

    <Button
        android:id="@+id/button"
        android:layout_width="417dp"
        android:layout_height="63dp"
        android:onClick="StartService"
        android:text="Start Service"
        app:backgroundTint="#0099EE"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.866"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.377"
        tools:ignore="OnClick" />

    <Button
        android:id="@+id/button2"
        android:layout_width="414dp"
        android:layout_height="58dp"
        android:onClick="StopService"
        android:text="Stop Service"
        app:backgroundTint="#CC00AA"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="1.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.534"
        tools:ignore="OnClick" />

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

## Java File:-

MainActivity.java

```
package com.example.services;

import ...

public class MainActivity extends AppCompatActivity {

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

    public void StartSomething(View view) {
        Intent ii=new Intent( mContext,MyOwnService.class);
        startService(ii);
    }

    public void StopSomething(View view) {
        Intent ii=new Intent( mContext,MyOwnService.class);
        stopService(ii);
    }
}
```

MyOwnService.java

```
package com.example.services;

import ...

public class MyOwnService extends Service {
    public MyOwnService() {
    }

    @Override
    public void onCreate(){
        super.onCreate();
        Toast.makeText( context, "Service Created",Toast.LENGTH_SHORT).show();
        Log.i( tag: "MyService", msg: "Service has Created");
    }

    @Override
    public int onStartCommand(Intent intent,int flags,int startId){
        Toast.makeText( context, "Service Started",Toast.LENGTH_SHORT).show();
        Log.i( tag: "MyService", msg: "Service has started");
        return START_STICKY;
    }

    @Override
    public void onDestroy(){
        Toast.makeText( context, "Service Stopped",Toast.LENGTH_SHORT).show();
        Log.i( tag: "MyService", msg: "Service has stopped");
        super.onDestroy();
    }

    @Override
    public IBinder onBind(Intent intent) { return null; }
}
```

# OUTPUT:-



## Services

START SERVICE

STOP SERVICE

Service Started



## Services

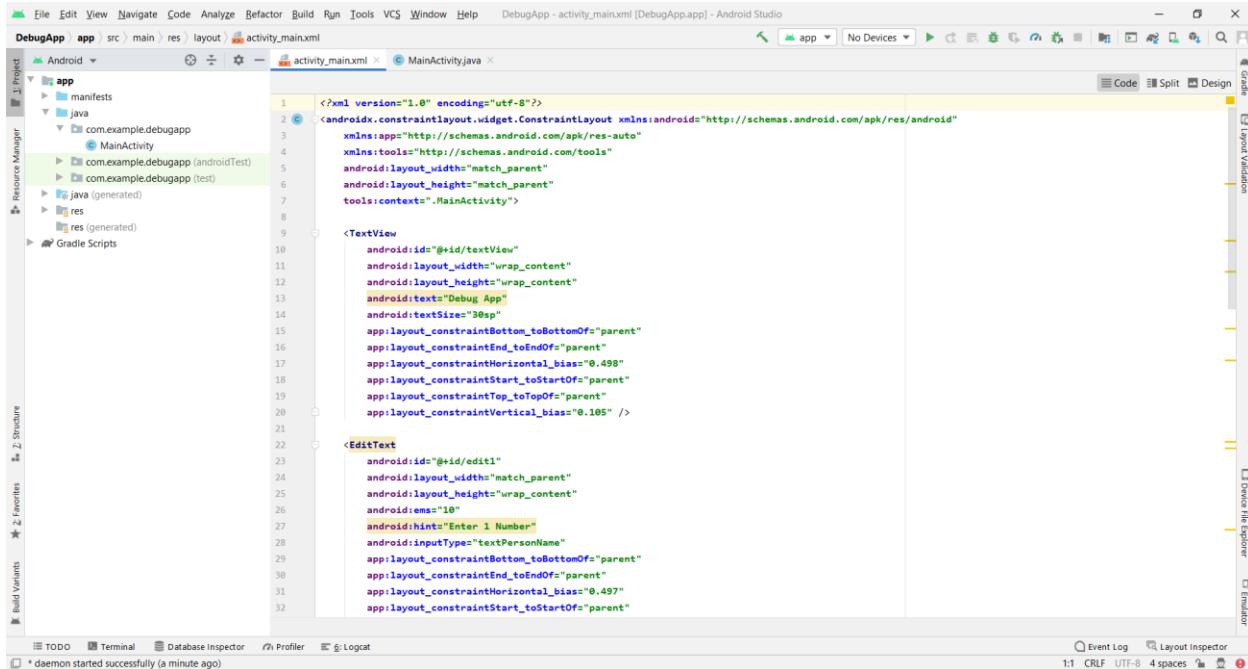
START SERVICE

STOP SERVICE

Service Stopped

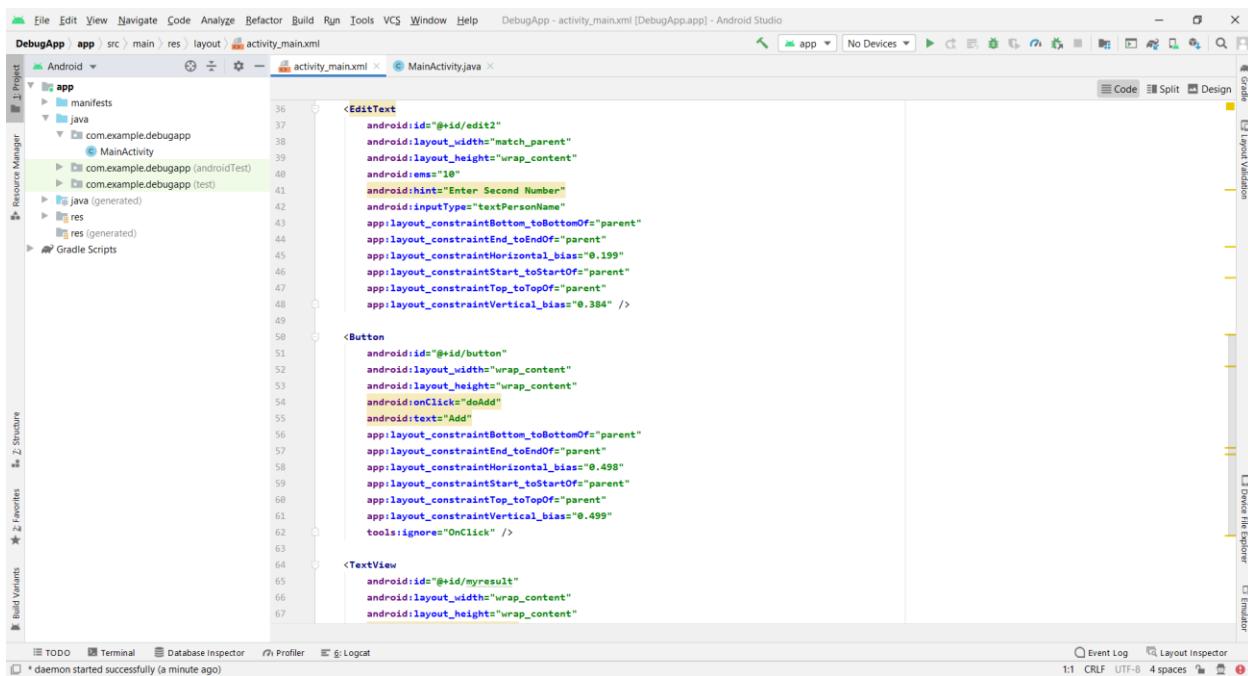
# Practical 11:- ANDROID SECURITY AND DEBUGGING

## Xml File:-



The screenshot shows the Android Studio interface with the XML code for `activity_main.xml`. The code defines a single `EditText` field with the ID `@+id/edit1`, which has a placeholder text of "Enter 1 Number".

```
<EditText  
    android:id="@+id/edit1"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:ems="10"  
    android:hint="Enter 1 Number"  
    android:inputType="textPersonName"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.498"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.105" />
```



The screenshot shows the Android Studio interface with the XML code for `activity_main.xml`. It includes an `EditText` for the first number and a `Button` labeled "Add" with an `onClick` event set to "doAdd".

```
<EditText  
    android:id="@+id/edit2"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:ems="10"  
    android:hint="Enter Second Number"  
    android:inputType="textPersonName"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.199"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.384" />  
  
<Button  
    android:id="@+id/button"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:onClick="doAdd"  
    android:text="Add"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.498"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.499"  
    tools:ignore="OnClick" />  
  
<TextView  
    android:id="@+id/myresult"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"
```

The screenshot shows the Android Studio interface with the XML layout file `activity_main.xml` open. The code defines a `ConstraintLayout` containing a button and a text view. The button's text is "Add" and it has an `onClick` event. The text view's text is "Your Result". Both views are constrained to the bottom and end of their parent.

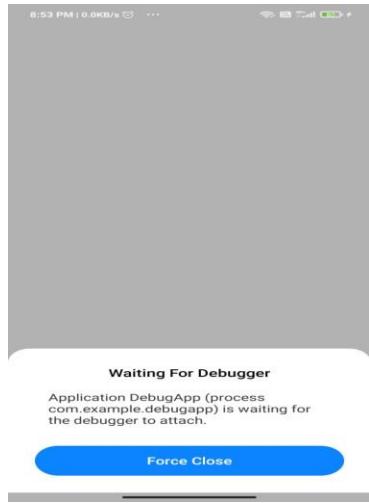
```
<Button  
    android:id="@+id/button"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:onClick="doAdd"  
    android:text="Add"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.498"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.499"  
    tools:ignore="OnClick" />  
  
<TextView  
    android:id="@+id/myresult"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Your Result"  
    android:textSize="36sp"  
    app:layout_constraintBottom_toBottomOf="parent"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintHorizontal_bias="0.07"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="parent"  
    app:layout_constraintVertical_bias="0.604" />
```

## Java File:-

The screenshot shows the Java code for `MainActivity`. It extends `AppCompatActivity` and overrides the `onCreate` method. In `onCreate`, it sets the content view to `activity_main` and finds three views by ID: `edit1`, `edit2`, and `tv1`. It also defines a `doAdd` method that adds the integer values from `edit1` and `edit2`, then logs the results and updates `tv1`'s text.

```
package com.example.debugapp;  
  
import ...  
  
public class MainActivity extends AppCompatActivity {  
    EditText e1,e2;  
    TextView tv1;  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
        e1=(EditText)findViewById(R.id.edit1);  
        e2=(EditText)findViewById(R.id.edit2);  
        tv1=(TextView)findViewById(R.id.myresult);  
    }  
  
    public void doAdd(View view) {  
        Log.i( tag: "Mainactivity", msg: "Executed 1");  
        int a1=Integer.parseInt(e1.getText().toString());  
        Log.i( tag: "Mainactivity", msg: "Executed 2");  
        int a2=Integer.parseInt(e2.getText().toString());  
        Log.i( tag: "Mainactivity", msg: "Executed 3");  
        int result=a1+a2;  
        Log.i( tag: "Mainactivity", msg: "Executed 4");  
        tv1.setText(""+result);  
        Log.i( tag: "Mainactivity", msg: "Executed 5");  
    }  
}
```

# OUTPUT:-



57