VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELGAVI-590018, KARNATAKA

LAB MANUAL

Mobile Application Development Laboratory

(As per VTU revised Syllabus-18CSMP68)

Department of Inforamation Science and Engineering

VIDYA VIKAS INSTITUTE OF ENGINEERING & TECHNOLOGY

127-128, Mysore - Bannur Road Alanahally, Alanahally Post, Mysuru, Karnataka 570028

Course Objectives:

The student should be made to:

- Know the components and structure of mobile application development frameworks for android and windows OS based mobiles.
- Understand how to work with various mobile application development frameworks.
- Learn the basic and important design concepts and issues of development of mobile applications.
- Understand the capabilities and limitations of mobile device

Course Outcomes

Upon completion of the course students should be able to:

- Install and configure Android application development tools.
- Design and develop user Interfaces for the Android platform.
- Save state information across important operating system events.
- Apply Java programming concepts to Android application development.

Laboratory Requirement

Software Requirement:

- Operating System : Windows- 10/8/7(64 Bit)
- IDE: JDK and Android Studio -4.1
- Emulator: AVD Emulator 2.

H/W Requirement:

- 4 GB RAM minimum (8 GB RAM recommended)
- 2 GB of available disk space minimum, 4 GB recommended (500 MB for IDE plus 1.5 GB for Android SDK and emulator system image)

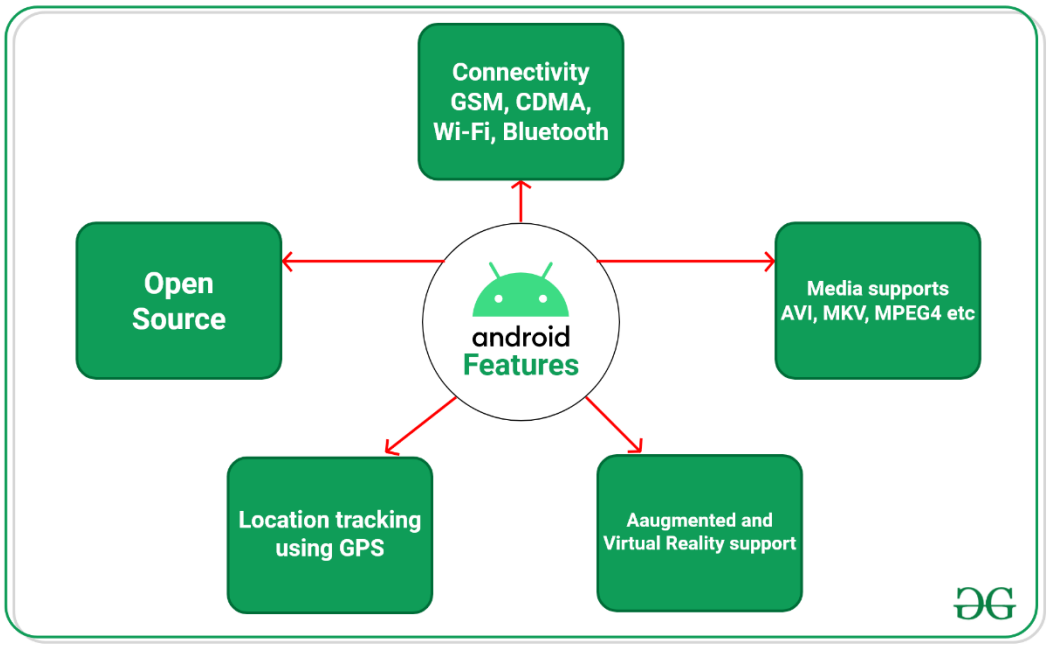
Contents

1. INTRODUCTION TO ANDROID DEVELOPMENT

The Android operating system is the largest installed base among various mobile platforms across the globe. Hundreds of millions of mobile devices are powered by Android in more than 190 countries of the world. It conquered around 75% of the global market share by the end of 2020, and this trend is growing bigger every other day. The company named Open Handset Alliance developed Android for the first time that is based on the modified version of the Linux kernel and other open-source software. Google sponsored the project at initial stages and in the year 2005, it acquired the whole company. In September 2008, the first Android-powered device launched in the market. Android dominates the mobile OS industry because of the long list of features it provides. It's user-friendly, has huge community support, provides a greater extent of customization, and a large number of companies build Android-compatible smartphones. As a result, the market observes a sharp increase in the demand for developing Android mobile applications, and with that companies need smart developers with the right skill set. At first, the purpose of Android was thought of as a mobile operating system. However, with the advancement of code libraries and its popularity among developers of the divergent domain, Android becomes an absolute set of software for all devices like tablets, wearable's, set-top boxes, smart TVs, notebooks, etc.

1.1 Features of Android

Android is a powerful open-source operating system that open-source provides immense features and some of these are listed below.



- Android Open Source Project so we can customize the OS based on our requirements.
- Android supports different types of connectivity for GSM, CDMA, Wi-Fi, Bluetooth, etc. for telephonic conversation or data transfer.
- Using wifi technology we can pair with other devices while playing games or using other applications.
- It contains multiple APIs to support location-tracking services such as GPS.
- We can manage all data storage related activities by using the file manager.
- It contains a wide range of media supports like AVI, MKV, FLV, MPEG4, etc. to play or record a variety of audio/video.
- It also supports different image formats like JPEG, PNG, GIF, BMP, MP3, etc.
- It supports multimedia hardware control to perform playback or recording using a camera and microphone.
- Android has an integrated open-source WebKit layout based web browser to support User Interface like HTML5, CSS3.
- Android supports multi-tasking means we can run multiple applications at a time and can switch in between them.
- It provides support for virtual reality or 2D/3D Graphics

1.2 Android Versions

Google launched the first version of the Android platform on Nov 5, 2007. Since then, Google released a lot of android versions such as Apple Pie, Banana Bread, Cupcake, Donut, Éclair, Froyo, Gingerbread, Jellybeans, Kitkat, Lollipop, marshmallow, Nougat, Oreo, etc. with extra functionalities and new features.

The following table shows the version details of android which is released by Google from 2007 to date.

Code Name	Version	API level	Release date
Apple Pie	Android 1.0	1	September 23, 2008
Banana Bread	Android 1.1	2	February 9, 2009
Cupcake	Android 1.5	3	April 30, 2009
Donut	Android 1.6	4	September 15, 2009
Éclair	Android 2.0 – 2.1	5-7	October 26, 2009
Froyo	Android 2.2 – 2.2.3	8	May 20, 2010
Gingerbread	Android 2.3 – 2.3.4	9-10	December 6, 2010
Honeycomb	Android 3.0.x – 3.2.x	11 – 13	February 22, 2011
Ice Cream Sandwich	Android 4.0 – 4.0.4	14 – 15	October 18, 2011

Code Name	Version	API level	Release date
Jelly Bean	Android 4.1 – 4.1.2	16 – 18	July 9, 2012
Kitkat	Android 4.4 – 4.4.4	19	July 9, 2012
Lollipop	Android 5.0 – 5.1	21 – 22	October 17, 2014
Marshmallow	Android 6.0 – 6.0.1	23	October 5, 2015
Nougat	Android 7.0 – 7.1	24 – 25	August 22, 2016
Oreo	Android 8.0	26	August 21, 2017
Pie	Android 9.0	27	August 6, 2018
Android Q	Android 10.0	29	September 3, 2019
Android 11	Android 11.0	30	September 8, 2020

1.3 Programming Languages used in Developing Android Applications

- **Java**
- **Kotlin**

Developing the Android Application using Kotlin is preferred by Google, as Kotlin is made an official language for Android Development, which is developed and maintained by JetBrains. Previously before the Java is considered the official language for Android Development. Kotlin is made official for Android Development in Google I/O 2017.

1.4 Android Architecture

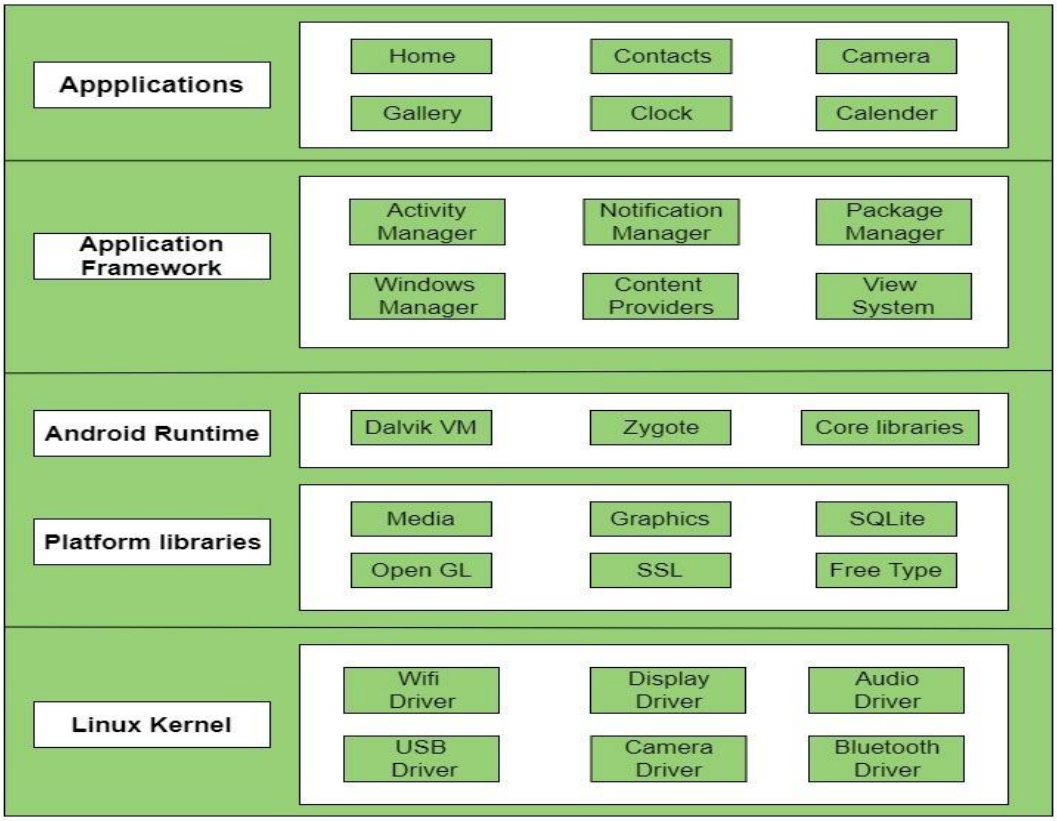
Android architecture contains different number of components to support any android device needs. Android software contains an open-source Linux Kernel having collection of number of C/C++ libraries which are exposed through an application framework services.

Among all the components Linux Kernel provides main functionality of operating system functions to smartphones and Dalvik Virtual Machine (DVM) provide platform for running an android application.

The main components of android architecture are following:-

- Applications
- Application Framework
- Android Runtime
- Platform Libraries
- Linux Kernel

Pictorial representation of android architecture with several main components and their sub components –



1.4.1 Applications –

Applications is the top layer of android architecture. The pre-installed applications like home, contacts, camera, gallery etc and third party applications downloaded from the play store like chat applications, games etc. will be installed on this layer only. It runs within the Android run time with the help of the classes and services provided by the application framework.

1.4.2 Application framework –

Application Framework provides several important classes which are used to create an Android application. It provides a generic abstraction for hardware access and also helps in managing the user interface with application resources. Generally, it provides the services with the help of which we can create a particular class and make that class helpful for the Applications creation. It includes different types of services activity manager, notification manager, view system, package manager etc. which are helpful for the development of our application according to the prerequisite.

1.4.3Application runtime –

Android Runtime environment is one of the most important part of Android. It contains components like core libraries and the Dalvik virtual machine(DVM). Mainly, it provides the base for the application framework and powers our application with the help of the core libraries. Like Java Virtual Machine (JVM), **Dalvik Virtual Machine (DVM)** is a register-based virtual machine and specially designed and optimized for android to ensure that a device can run multiple instances efficiently. It depends on the layer Linux kernel for threading and low-level memory management. The core libraries enable us to implement android applications using the standard JAVA or Kotlin programming languages.

1.4.4 Platform libraries

The Platform Libraries includes various C/C++ core libraries and Java based libraries such as Media, Graphics, Surface Manager, OpenGL etc. to provide a support for android development.

- **Media** library provides support to play and record an audio and video formats.
- **Surface manager** responsible for managing access to the display subsystem.
- **SGL** and **OpenGL** both cross-language, cross-platform application program interface (API) are used for 2D and 3D computer graphics.

- **SQLite** provides database support and **FreeType** provides font support.
- **Web-Kit** This open source web browser engine provides all the functionality to display web content and to simplify page loading.
- **SSL (Secure Sockets Layer)** is security technology to establish an encrypted link between a web server and a web browser.

1.4.5 Linux Kernel

Linux Kernel is heart of the android architecture. It manages all the available drivers such as display drivers, camera drivers, Bluetooth drivers, audio drivers, memory drivers, etc. which are required during the runtime.

The Linux Kernel will provide an abstraction layer between the device hardware and the other components of android architecture. It is responsible for management of memory, power, devices etc.

The features of Linux kernel are:

- **Security:** The Linux kernel handles the security between the application and the system.
- **Memory Management:** It efficiently handles the memory management thereby providing the freedom to develop our apps.
- **Process Management:** It manages the process well, allocates resources to processes whenever they need them.
- **Network Stack:** It effectively handles the network communication.
- **Driver Model:** It ensures that the application works properly on the device and hardware manufacturers responsible for building their drivers into the Linux build.

1.5 Android UI Layouts

Android **Layout** is used to define the user interface which holds the UI controls or widgets that will appear on the screen of an android application or activity.
Generally, every application is combination of View and ViewGroup. As we know, an android application contains a large number of activities and we can say each activity is one page of the application. So, each activities contains multiple user interface components and those components are the instances of the View and ViewGroup.

A **View** is defined as the user interface which is used to create an interactive UI compon ents such as TextView, EditText, Radio Button, etc. and it responsible for event handlin g and drawing.

A **ViewGroup** act as a base class for layouts and layouts parameters which hold other V iews or ViewGroups and to define the layout properties.

The android framework will allow us to use UI elements or widgets in two ways:

- Use UI elements in XML file
- Create elements in Kotlin file dynamically

1.5.1 Types of Android Layout

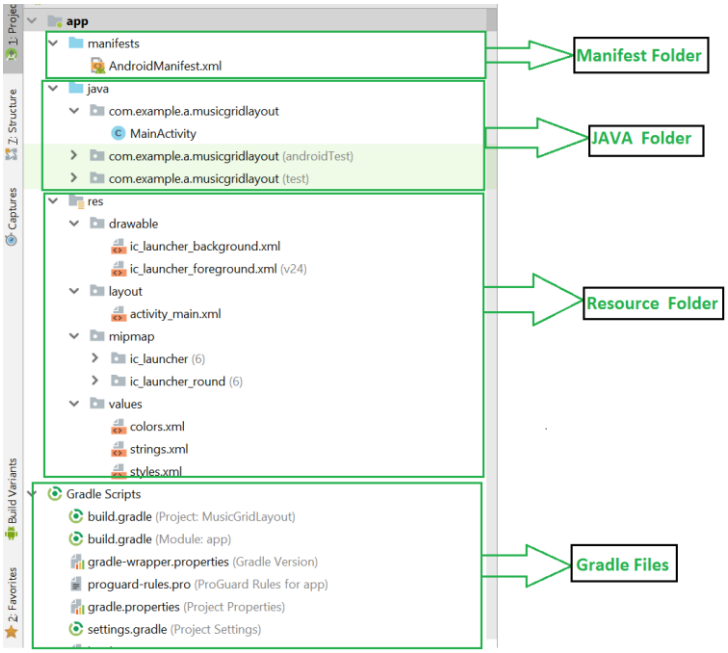
Android Linear Layout: LinearLayout is a ViewGroup subclass, used to provide child View elements one by one either in a particular direction either horizontally or vertically based on the orientation property.

- **Android Relative Layout:** RelativeLayout is a ViewGroup subclass, used to specify the position of child View elements relative to each other like (A to the right of B) or relative to the parent (fix to the top of parent).
- **Android Constraint Layout:** ConstraintLayout is a ViewGroup subclass, used to specify the position of a layout constraints for every child View relative to other views present. A ConstraintLayout is similar to a RelativeLayout, but having more power.
- **Android Frame Layout:** FrameLayout is a ViewGroup subclass, used to specify the position of View elements it contains on the top of each other to display only single View inside the FrameLayout.
- **Android Table Layout:** TableLayout is a ViewGroup subclass, used to display the child View elements in rows and columns.
- **Android Web View:** WebView is a browser which is used to display the web pages in our activity layout.
- **Android List View:** ListView is a ViewGroup, used to display scrollable list of items in single column.
- **Android Grid View:** GridView is a ViewGroup which is used to display scrollable list of items in grid View of rows and columns.

XML attributes	Description
android:id	Used to specify the id of the view.

XML attributes	Description
android:layout_width	Used to declare the width of View and ViewGroup elements in layout.
android:layout_height	Used to declare the height of View and ViewGroup elements in layout.
android:layout_marginLeft	Used to declare the extra space used in the left side of View and ViewGroup elements.
android:layout_marginRight	Used to declare the extra space used in the right side of View and ViewGroup elements.
android:layout_marginTop	Used to declare the extra space used in the top side of View and ViewGroup elements.
android:layout_marginBottom	Used to declare the extra space used in the bottom side of View and ViewGroup elements.
android:layout_gravity	Used to define how child Views are positioned in the layout.

1.6 Structural Layout Of Android Application



The above figure represents the various structure of an app.

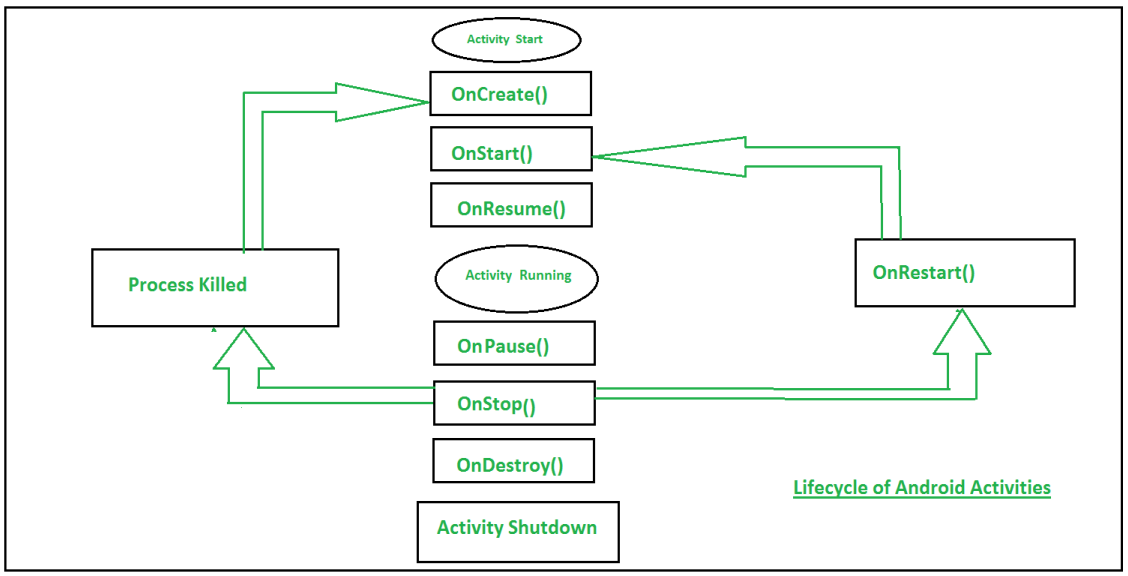
- **Android Mainfest** is a XML file which is the root of the project source set. It describes the essential information about the app and the Android build tools, the Android Operating System and the Google Play. It contains the permission that an app might need in order to perform the specific task. It also contains the Hardware and the Software features of the app, which determines the compatibility of an app on Play Store. It also includes the special activities like services, broadcast receiver, content providers, package name,etc.
- The **JAVA folder** consist of the java files that are required to perform the background task of the app. It consist of the functionality of the buttons, calculation, storing, variables, toast(small popup message) , programming function, etc. The number of these files depends upon the type of activities created.
- **Res** or Resource folder consist of the various resources that are used in the app. This consist of sub-folders like drawable, layout, mipmap, raw and values. The drawable consist of the images. The layout consist of the XML files that defines the user interface layout. These are stored in res.layout and are accessed as R.layout class. The raw consist of the Resources files like audio file or music file,etc. These are accessed through R.raw.filename. values are used to store the hardcoded strings(considered safe to store string values) values, integers and colors. It consist of various other directories like:-

- R.array :arrays.xml for resource arrays
- R.integer : integers.xml for resource integers
- R.bool : bools.xml for resource boolean
- R.color :colors.xml for color values
- R.string : strings.xml for string values
- R.dimen : dimens.xml for dimension values
- R.style : styles.xml for styles

- **Gradle*:*** Gradle is an advance toolkit, which is used to manage the build process, that allows to define the flexible custom build configurations. Each build configuration can define its own set of code and resources, while reusing the parts common to all versions of your app. The Android plugin for Gradle works with the build toolkit to provide processes and configurable settings that are specific to building and testing Android applications. Gradle and the Android plugin run independent of Android Studio. This means that you can build your Android apps from within Android Studio. The flexibility of the Android build system enables you to perform custom build configurations without modifying your app's core source files.

1.7 Lifecycle of Android App

The Lifecycle of the Android App can be shown through this diagram:



1.7.1 States of Android Lifecycle

1. **OnCreate :** This is called when activity is first created.
2. **OnStart :** This is called when activity become visible to the user.
3. **OnResume :** This is called when activity starts to interact with the user.

4. **OnPause :** This is called when activity is not visible to the user.
5. **OnStop :** This is called when activity is no longer visible.
6. **OnRestart :** This is called when activity is stopped, and restarted again.
7. **OnDestroy :** This is called when activity is to be closed or destro

1.8 Advantages of Android Development

- The Android is an open-source Operating system and hence possesses a vast community for support.
- The design of the Android Application has guidelines from Google, which becomes easier for developers to produce more intuitive user applications.
- Fragmentation gives more power to Android Applications. This means the application can run two activities on a single screen.
- Releasing the Android application in the Google play store is easier when it is compared to other platforms.

2. LAB PROGRAMS

Program 1: Create an application to design a Visiting Card. The Visiting card should have a company logo at the top right corner. The company name should be displayed in Capital letters, aligned to the center. Information like the name of the employee, job title, phone number, address, email, fax and the website address is to be displayed. Insert a horizontal line between the job title and the phone number.

XML Code

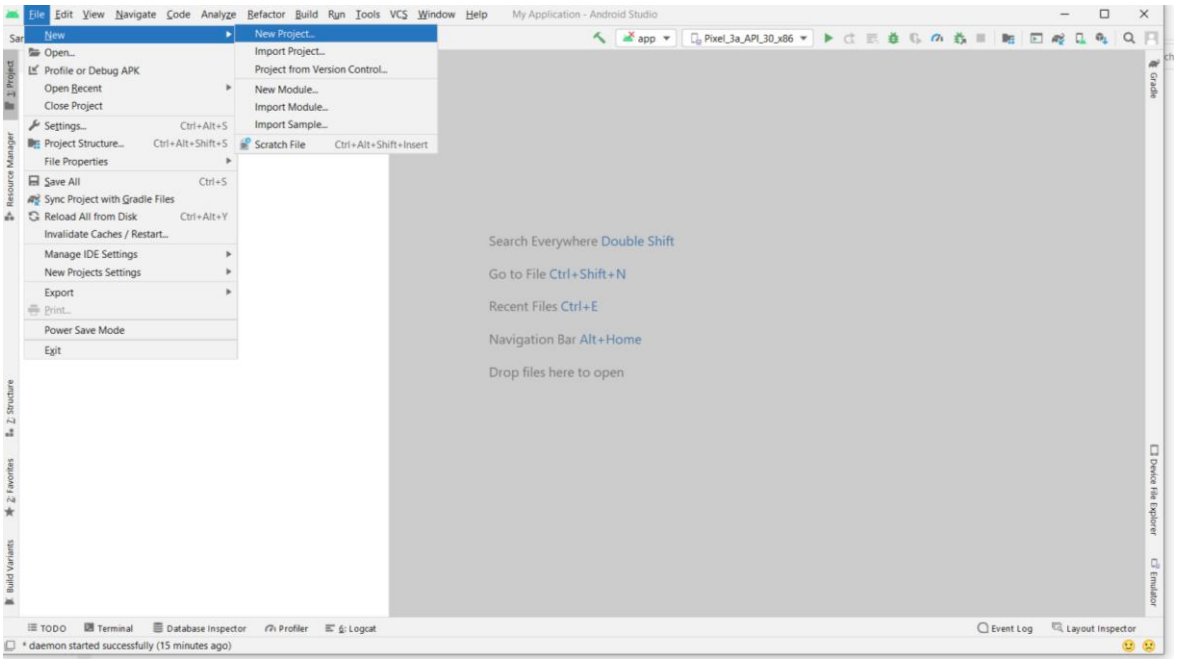
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"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

    <TextView
        android:id="@+id/textView"
        android:layout_width="177dp"
        android:layout_height="65dp"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="138dp"
        android:layout_marginBottom="664dp"
        android:text="mycem"
        android:textAlignment="center"
        android:textSize="60sp"
        app:autoSizeMaxTextSize="28dp"
        tools:layout_editor_absoluteX="84dp"
        tools:layout_editor_absoluteY="82dp" />

    <View
        android:id="@+id/view"
        android:layout_width="wrap_content"
        android:layout_height="4dp"
        android:layout_alignParentBottom="true"
        android:layout_marginBottom="590dp"
        android:background="#4444" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="151dp"
        android:layout_marginBottom="521dp"
        android:text="Person Name"
        android:textAlignment="center"
        android:textSize="18sp"
        android:textStyle="bold" />

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="130dp"
        android:layout_marginBottom="465dp"
        android:text="Designation"
        android:textSize="18sp" />

    <TextView
        android:id="@+id/textView4"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="63dp"
        android:layout_marginBottom="419dp"
        android:text="Company Name "
        android:textAlignment="center"
        android:textSize="18sp" />

    <TextView
        android:id="@+id/textView5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="166dp"
        android:layout_marginBottom="373dp"
        android:text="Phone number"
        android:textSize="14sp" />

    <TextView
```

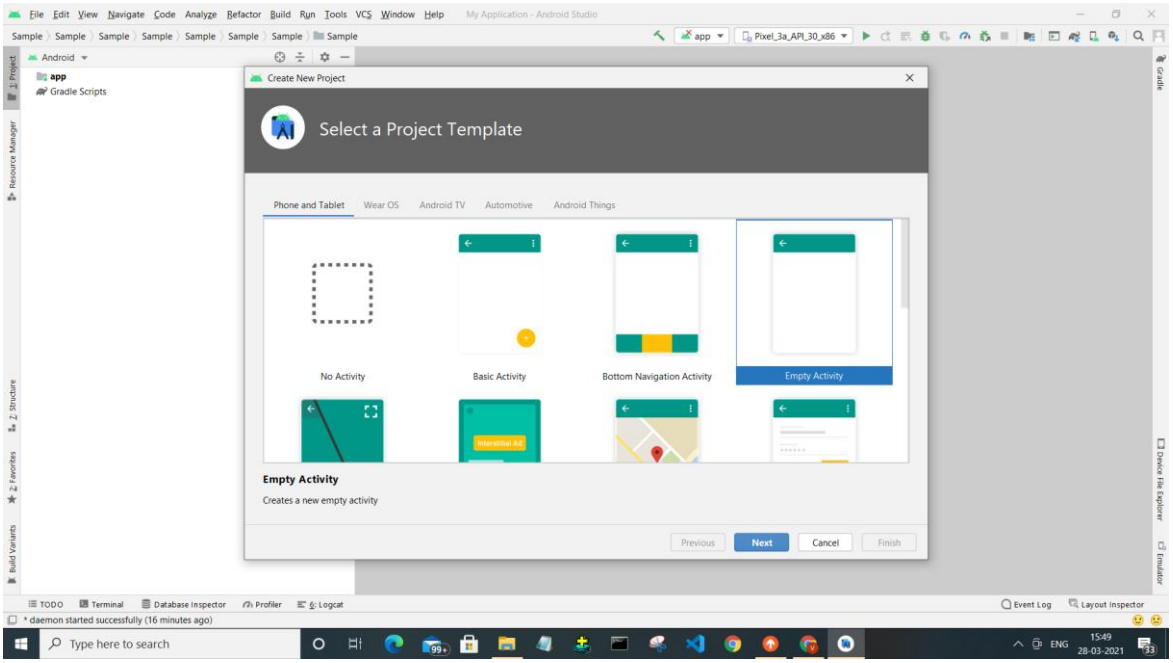
```
        android:id="@+id/textView7"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="123dp"
        android:layout_marginBottom="338dp"
        android:text="email-id"
        android:textAlignment="center"
        android:textSize="14sp" />

    <ImageView
        android:id="@+id/imageView2"
        android:layout_width="96dp"
        android:layout_height="70dp"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="169dp"
        android:layout_marginBottom="589dp"
        app:srcCompat="@drawable/ceclogo" />

</RelativeLayout>
```

EXECUTION STEPS

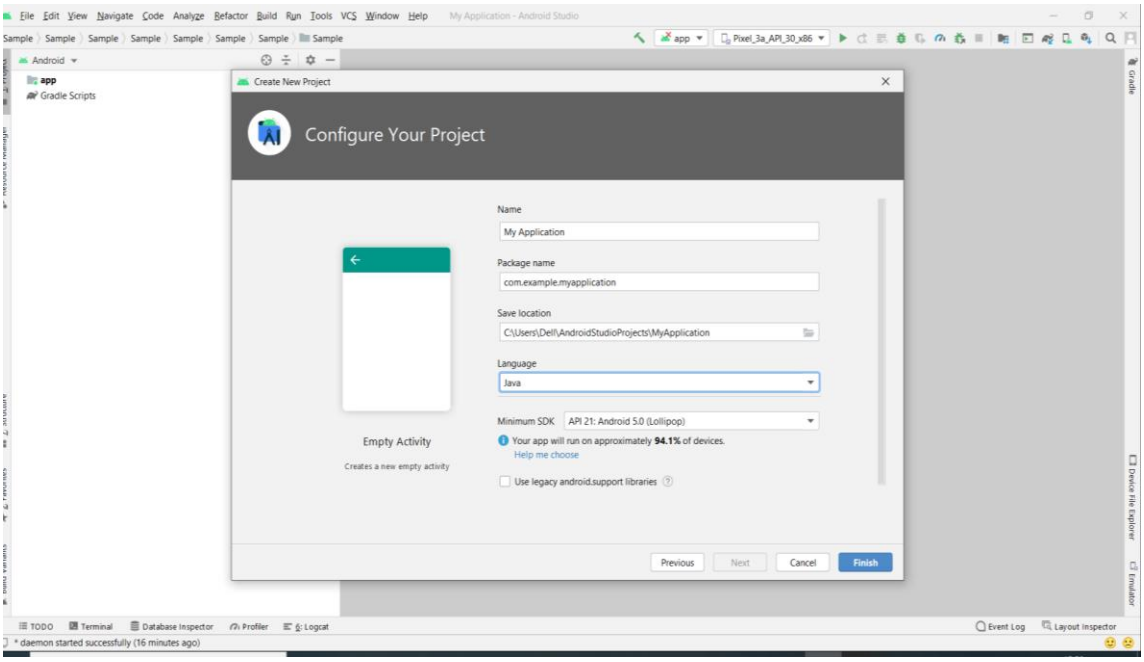
Step 1:Create an empty project and name it, Click on file→new→new project.



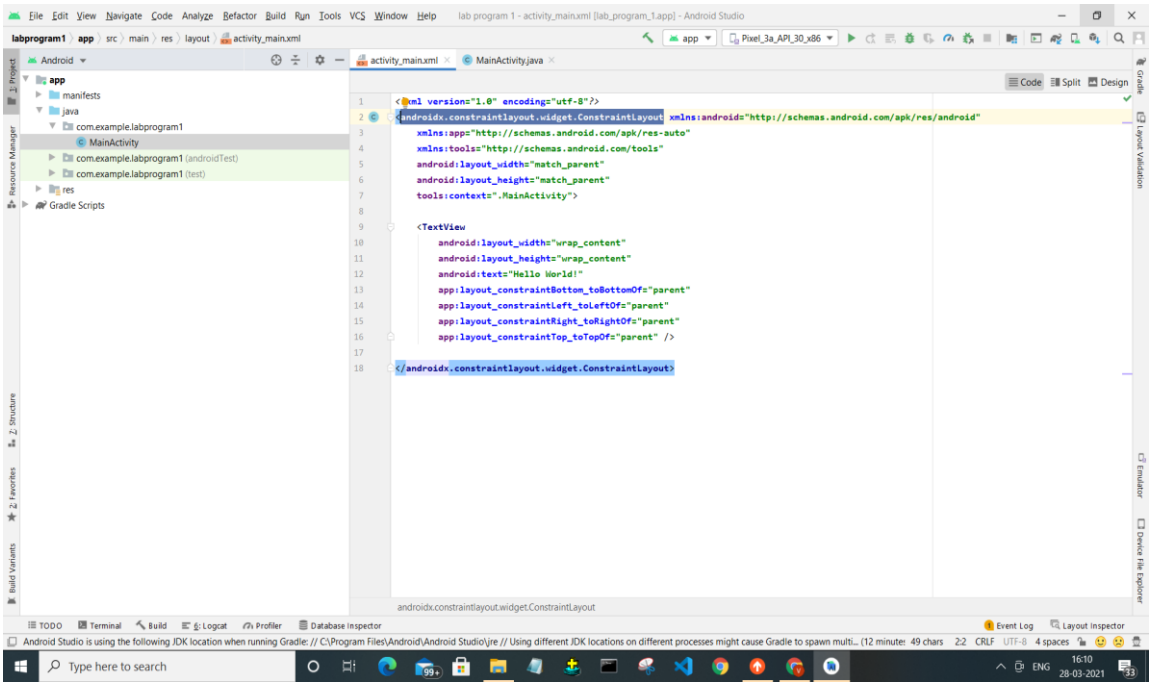
Step-2: Select empty project click on Next.



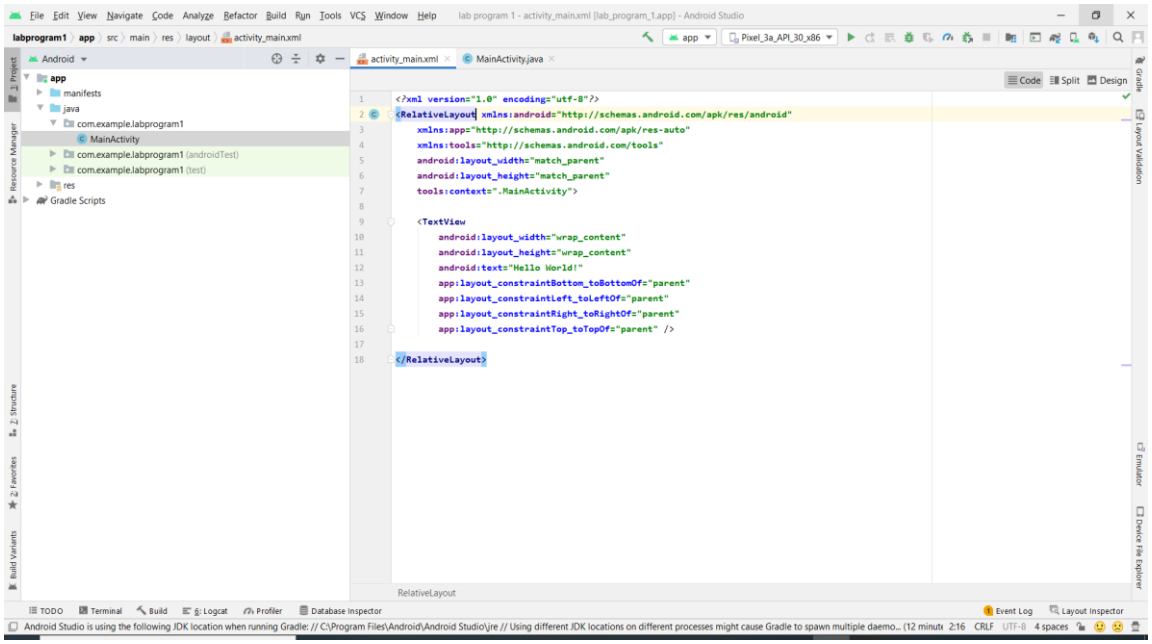
Step 3: Name your application, select language as java and select higher version of Android APIs.



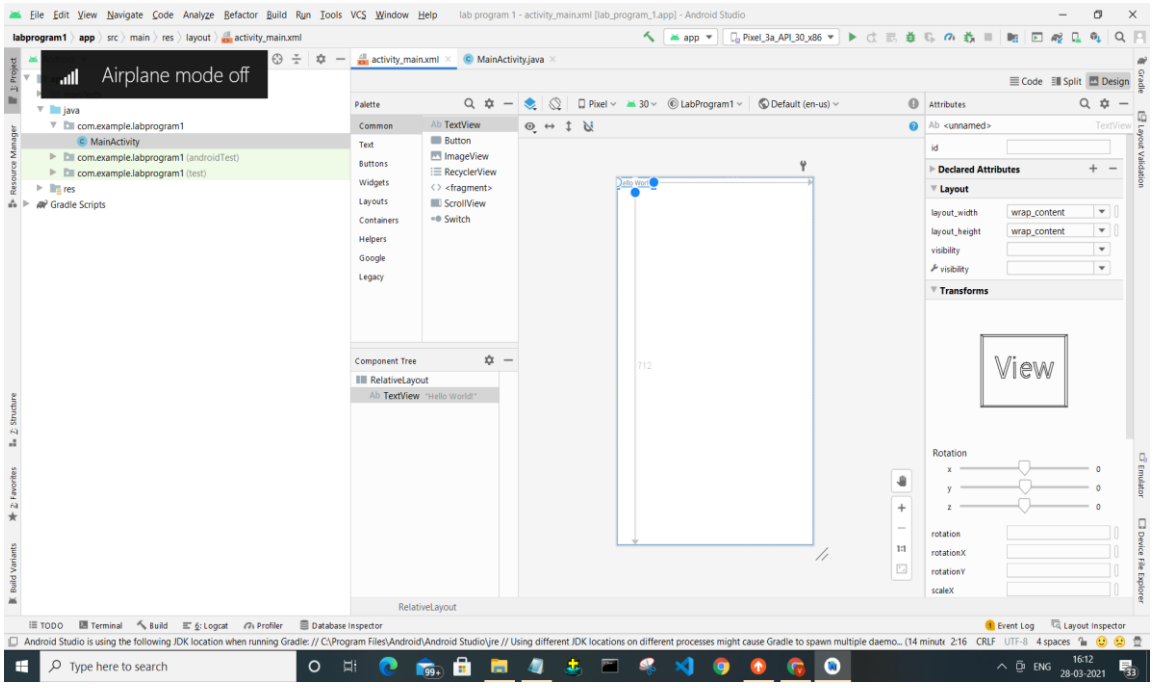
Step 4: Go to XML part in the right side select code in that delete Constraint layout and make it as Relative layout.



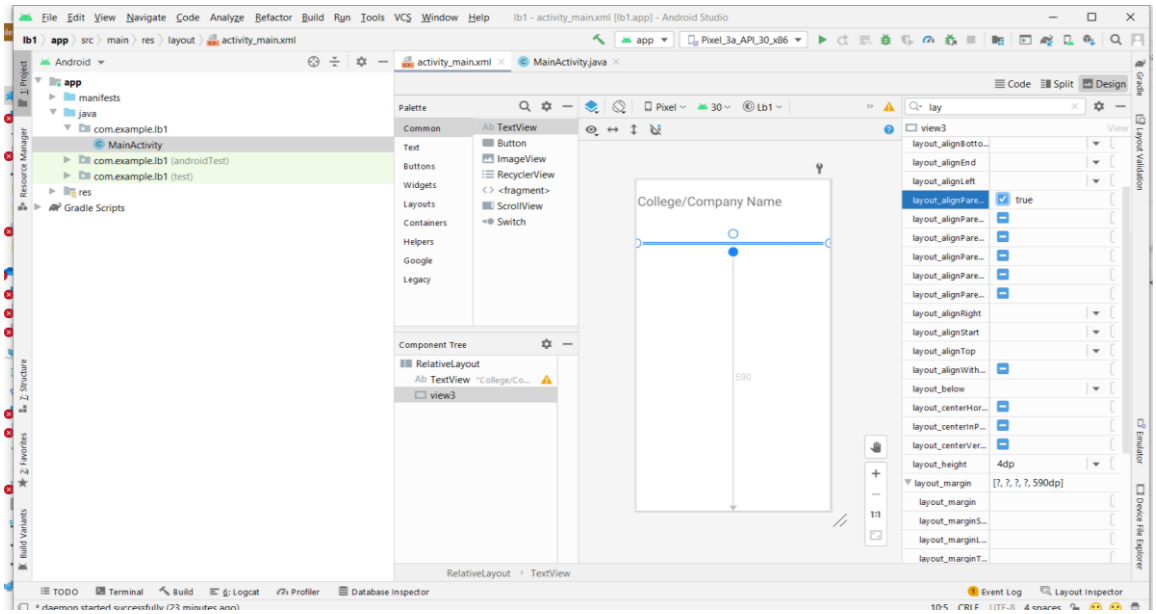
Step 5: Don't type everything just type R in the keyboard and you will be presented with the multiple options in that select relative layout



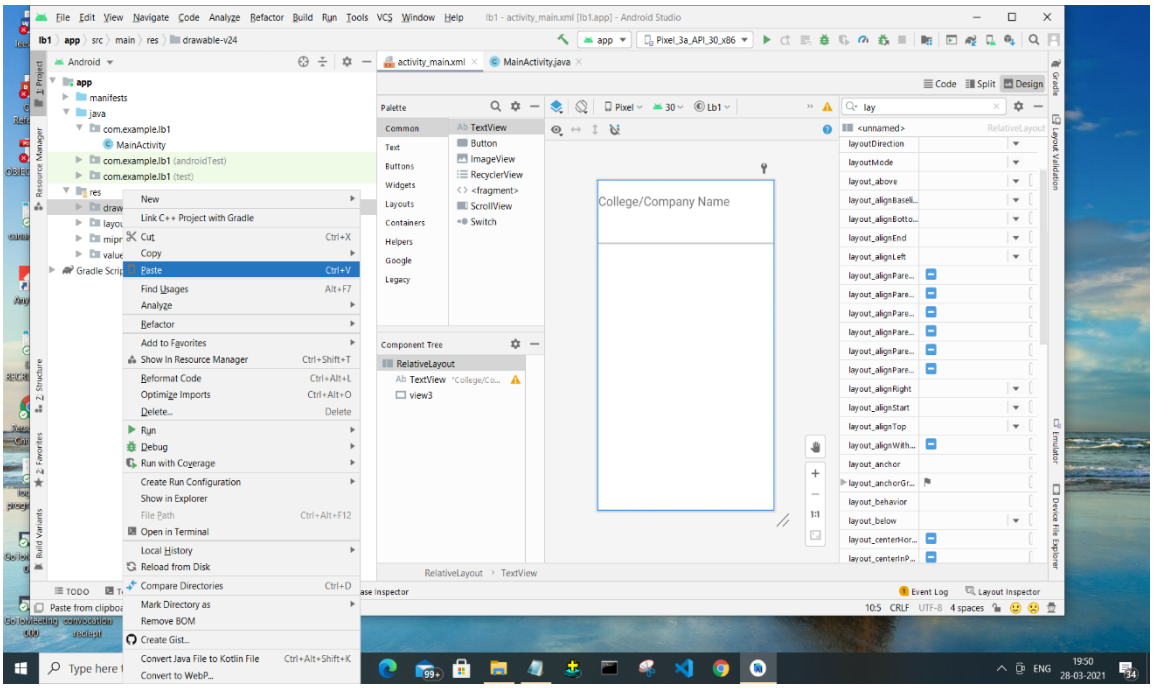
Step 6: In design part drag and drop text view on the terminal and you can change name of the text view in the code part. Like that you can change size, colour and alignment of the text in code part or by searching in the attribute which is present on right side.



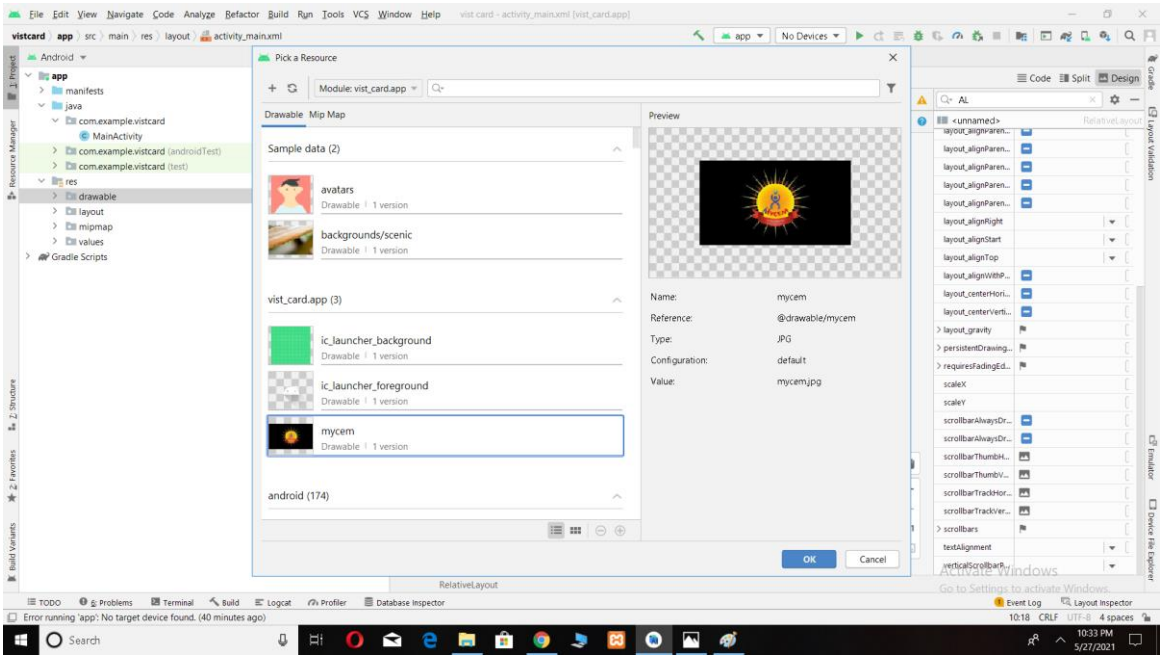
Step 7: Drag and drop another view and set layout height to 4dp, and set layout-parent bottom to true, set layout_marginBottom="590dp"and set background to #4444



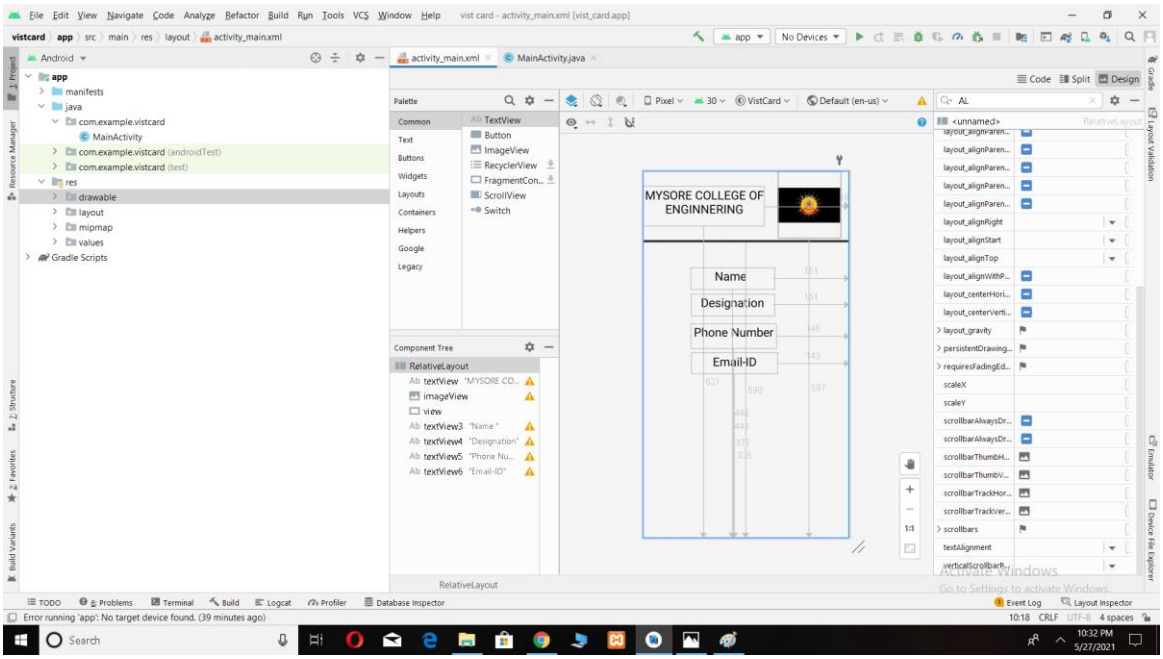
Step 8: To insert any media files like image, video, audio just copy and paste the file into drawable folder and then refactor. Image file extension should be jpeg or png.



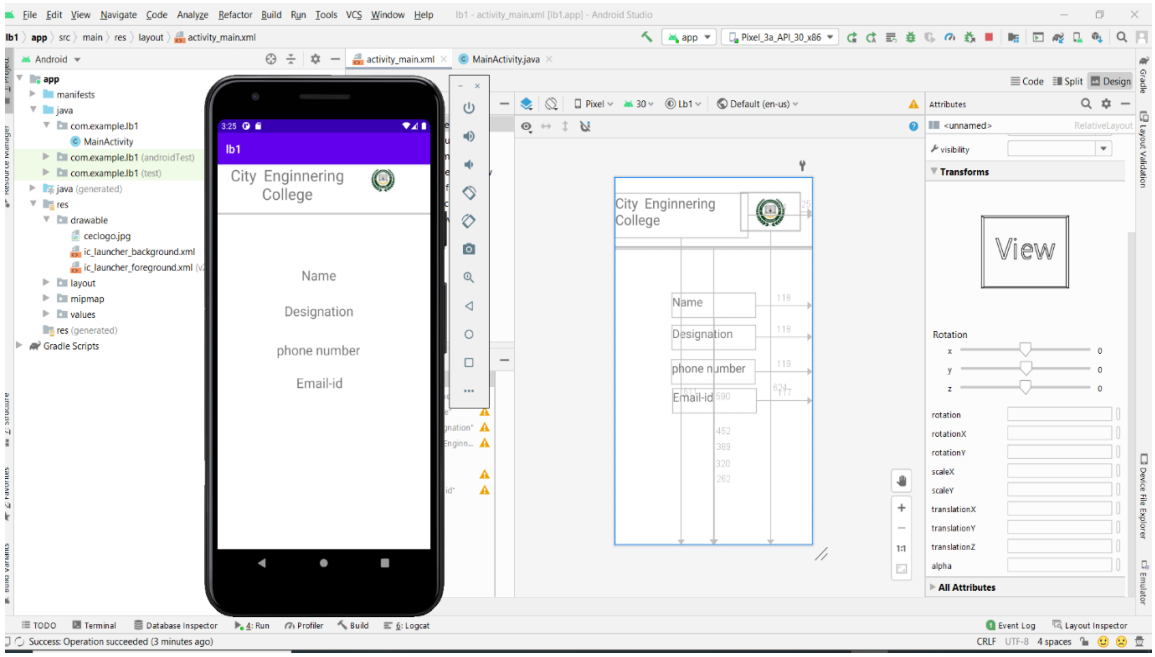
Step 9: Drag and drop image view on to the layout select the file that you have pasted into the drawable. Select your file and click on Ok.



Step 10 : Drag and drop the text views and edit their names size and alignment in the code part



Step 11: Run your application and you will get the out on android emulator as shown in the picture.



Program 2: Develop an Android application using controls like Button, TextView, EditText for designing a calculator having basic functionality like Addition, Subtraction, Multiplication, and Division.

XML-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=".MainActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="46dp"
        android:layout_marginBottom="666dp"
        android:text="Simple Calculator"
        android:textSize="36sp"
        android:textStyle="bold"
        tools:layout_editor_absoluteX="131dp"
        tools:layout_editor_absoluteY="39dp" />

    <EditText
        android:id="@+id/editText2"
        android:layout_width="146dp"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
```

```
        android:layout_marginEnd="227dp"
        android:layout_marginBottom="479dp"
        android:ems="10"
        android:hint="enter number2"
        android:inputType="textPersonName"
        android:text=""
        android:textSize="24sp" />

    <EditText
        android:id="@+id/editText1"
        android:layout_width="146dp"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="225dp"
        android:layout_marginBottom="553dp"
        android:ems="10"
        android:hint="enter number 1"
        android:inputType="textPersonName"
        android:text=""
        android:textSize="24sp" />

    <TextView
        android:id="@+id/tv1"
        android:layout_width="102dp"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="61dp"
        android:layout_marginBottom="520dp"
        android:text="0"
        android:textAlignment="center"
        android:textSize="30sp" />
```

```
<Button
    android:id="@+id/add"
    android:onClick="add"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="289dp"
    android:layout_marginBottom="370dp"
    android:text="ADD" />

<Button
    android:id="@+id/sub"
    android:onClick="sub"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="39dp"
    android:layout_marginBottom="369dp"
    android:text="SUB" />

<Button
    android:id="@+id/div"
    android:onClick="mul"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="283dp"
    android:layout_marginBottom="249dp"
    android:text="MUL" />
```

```
    <Button
        android:id="@+id/mul"
        android:onClick="div"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="33dp"
        android:layout_marginBottom="244dp"
        android:text="DIV" />

</RelativeLayout>
```

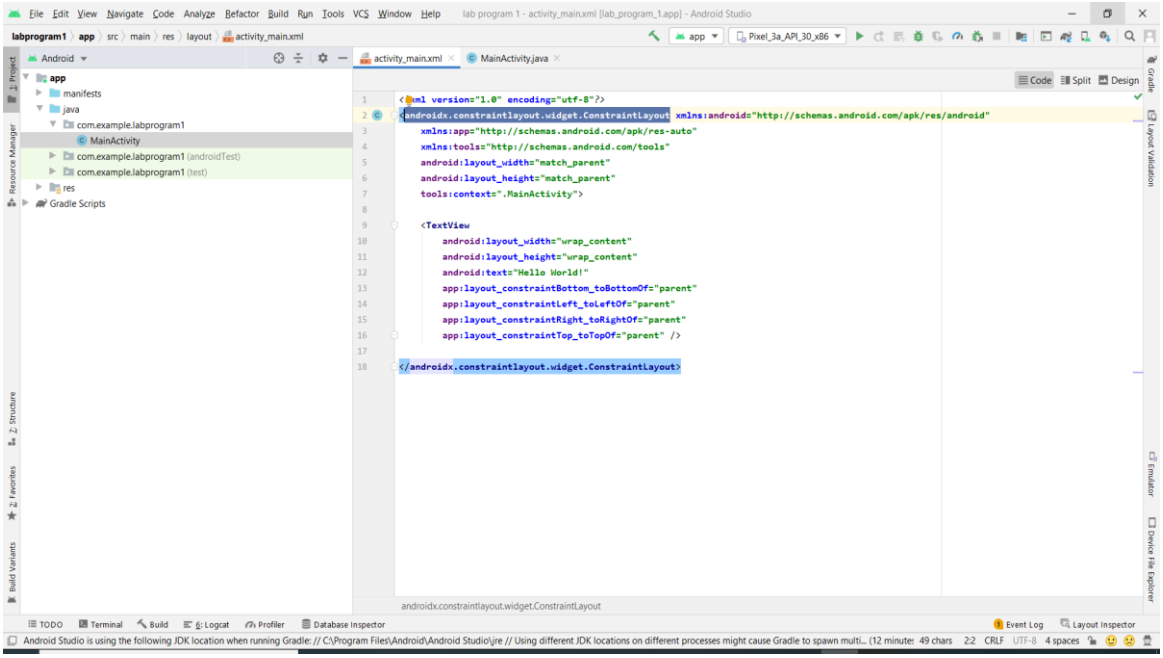
JAVA-Code

```
package com.example.simplecalculator;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
EditText e1,e2;
TextView tv;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        e1=(EditText)findViewById(R.id.editText1);
        e2=(EditText)findViewById(R.id.editText2);
```

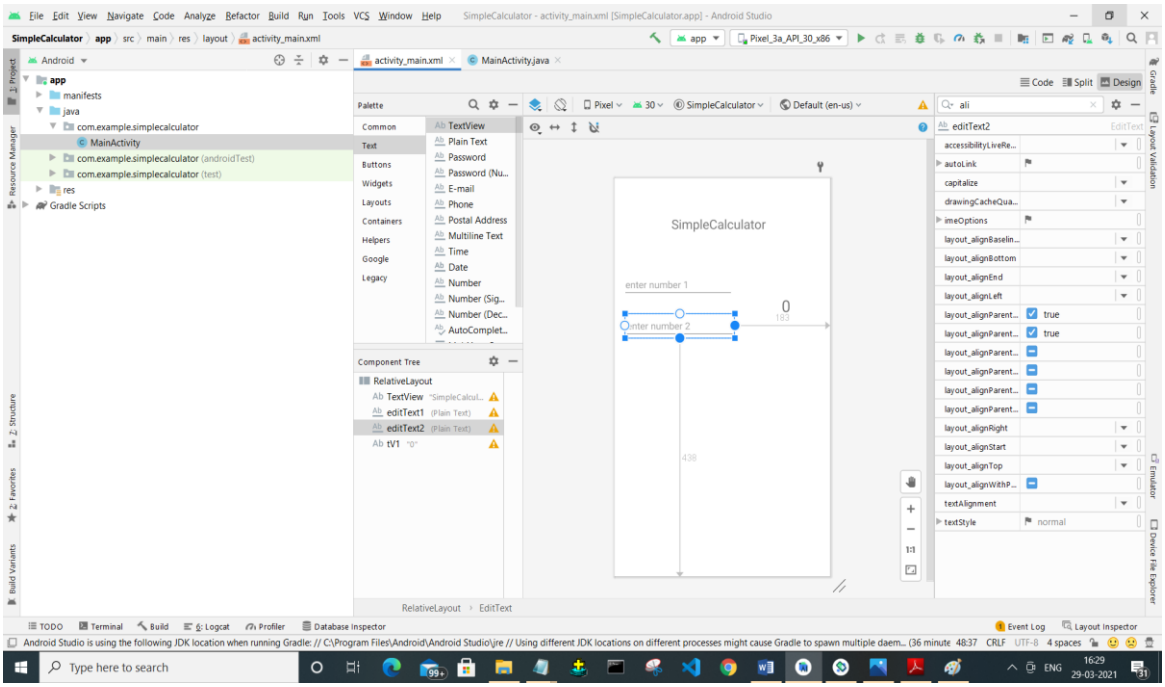
```
        tv=(TextView)findViewById(R.id.tv1);


    }
    public void add(View V)
    {
        int a1=Integer.parseInt(e1.getText().toString());
        int a2=Integer.parseInt(e2.getText().toString());
        int res=a1+a2;
        tv.setText(""+res);
    }
    public void sub(View V)
    {
        int a1=Integer.parseInt(e1.getText().toString());
        int a2=Integer.parseInt(e2.getText().toString());
        int res=a1-a2;
        tv.setText(""+res);
    }
    public void mul(View V)
    {
        int a1=Integer.parseInt(e1.getText().toString());
        int a2=Integer.parseInt(e2.getText().toString());
        int res=a1*a2;
        tv.setText(""+res);
    }
    public void div(View V)
    {
        int a1=Integer.parseInt(e1.getText().toString());
        int a2=Integer.parseInt(e2.getText().toString());
        double res=(double)(a1/a2);
        tv.setText(""+res);
    }
}
```

EXECUTION STEPS:-

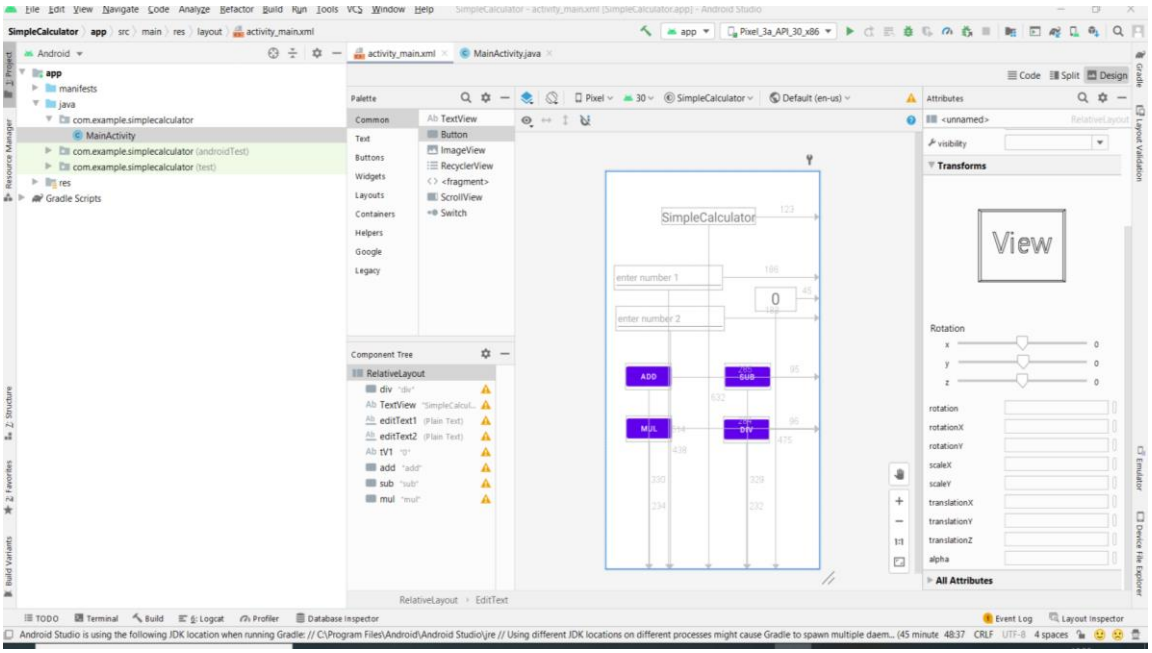
Step 1: Go to XML part in the right side select code in that delete Constraint layout and make it as Relative layout.



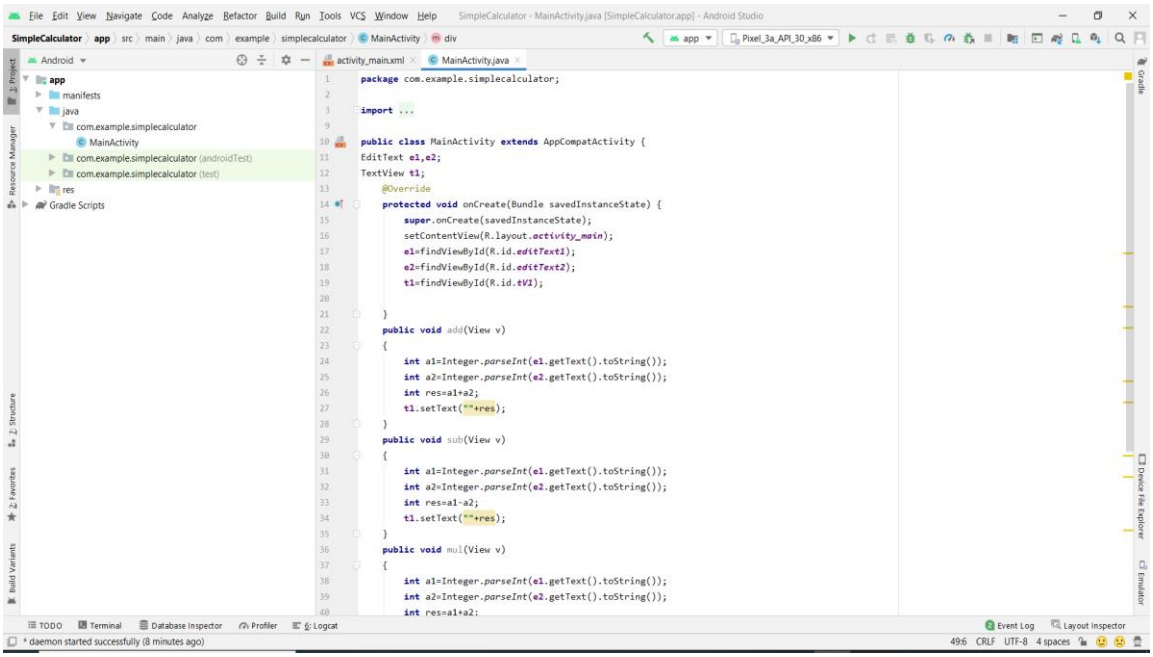
Step 2: In design part drag and drop text view on the terminal and you can change name of the text view in the code part.



Step 3: Drag and drop 4 buttons change their names as add, sub, mul and div respectively and change their ids.

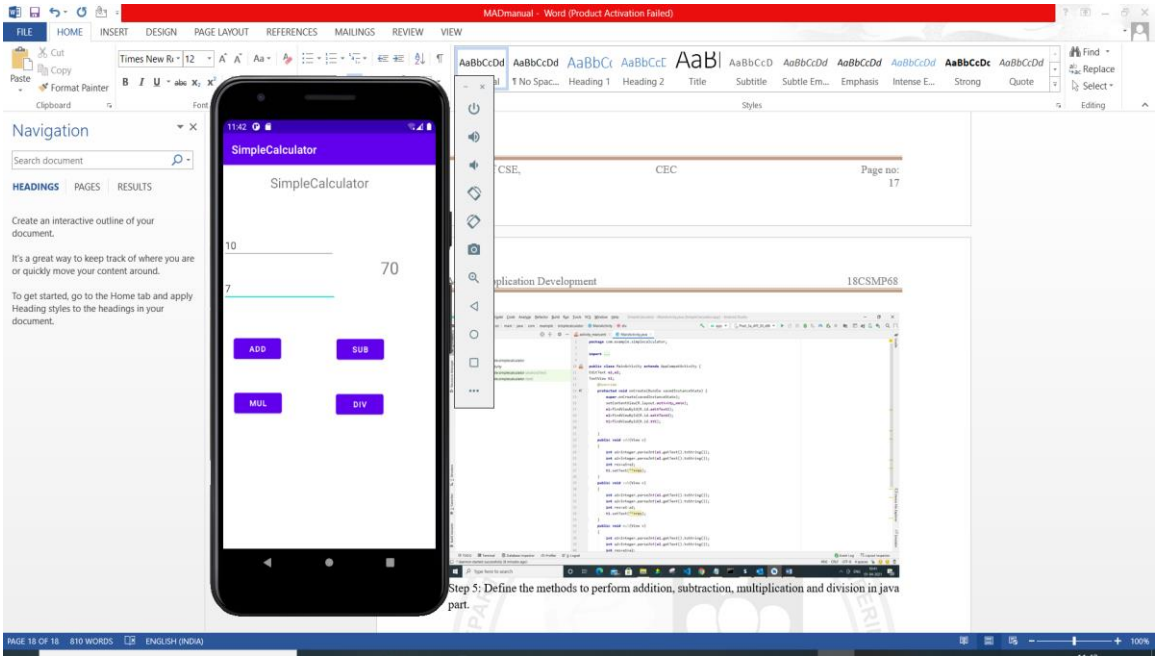


Step 4: Define the methods to perform addition, subtraction, multiplication and division in java part.



```
package com.example.simplecalculator;

import ...

public class MainActivity extends AppCompatActivity {
EditText e1,e2;
TextView t1;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        e1=findViewById(R.id.editText1);
        e2=findViewById(R.id.editText2);
        t1=findViewById(R.id.tV1);

    }
    public void add(View v)
    {
        int a1=Integer.parseInt(e1.getText().toString());
        int a2=Integer.parseInt(e2.getText().toString());
        int res=a1+a2;
        t1.setText(""+res);
    }
    public void sub(View v)
    {
        int a1=Integer.parseInt(e1.getText().toString());
        int a2=Integer.parseInt(e2.getText().toString());
        int res=a1-a2;
        t1.setText(""+res);
    }
    public void mul(View v)
    {
        int a1=Integer.parseInt(e1.getText().toString());
        int a2=Integer.parseInt(e2.getText().toString());
        int res=a1*a2;
```

Step 5: Run your application and get the result on android emulator.



Step 5: Define the methods to perform addition, subtraction, multiplication and division in java part.

Program 3: Create a SIGN Up activity with Username and Password. Validation of password should happen based on the following rules:

- **Password should contain uppercase and lowercase letters.**
- **Password should contain letters and numbers.**
- **Password should contain special characters.**

Minimum length of the password (the default value is 8). On successful SIGN UP proceed to the next Login activity. Here the user should SIGN IN using the Username and Password created during signup activity. If the Username and Password are matched then navigate to the next activity which displays a message saying "Successful Login" or else display a toast message saying "Login Failed". The user is given only two attempts and after that display a toast message saying "Failed Login Attempts" and disable the SIGN IN button. Use Bundle to transfer information from one activity to another.

XML Code for SignUp Activity.

```
<?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=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="157dp"
        android:layout_marginBottom="643dp"
        android:text="sign up"
        android:textSize="36sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
```

```
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/emailEditText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="90dp"
        android:layout_marginBottom="428dp"
        android:ems="10"
        android:hint="email"
        android:inputType="textEmailAddress" />

    <EditText
        android:id="@+id/passwordEditText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="90dp"
        android:layout_marginBottom="315dp"
        android:ems="10"
        android:hint="password"
        android:inputType="textPassword" />

    <Button
        android:id="@+id/signUpBtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
```

```
        android:layout_marginEnd="148dp"
        android:layout_marginBottom="195dp"
        android:text="signup" />


</RelativeLayout>
```

JAVA Code for SignUp activity

```
package com.example.myapplication11;

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;
import android.widget.Toast;

import java.util.regex.Pattern;

public class MainActivity extends AppCompatActivity {
    EditText emailEditText, passwordEditText;
    Button signUpBtn;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        emailEditText = findViewById(R.id.emailEditText);
        passwordEditText =
findViewById(R.id.passwordEditText);
        signUpBtn = findViewById(R.id.signUpBtn);
        signUpBtn.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v)
            {
                String email =
emailEditText.getText().toString();
                String password =
passwordEditText.getText().toString();
                if (!isValidPassword(password))
                {
                    Toast.makeText(MainActivity.this,
"Password doesn't match rules", Toast.LENGTH_SHORT).show();
                    return;

                }
                Intent intent = new
Intent(MainActivity.this, LoginActivity.class);
```

```
                intent.putExtra("email", email);
                intent.putExtra("password", password);
                startActivity(intent);
            }
        });
    }

             Pattern lowerCase = Pattern.compile("^.*[a-
z].*$");
            Pattern upperCase = Pattern.compile("^.*[A-
Z].*$");
            Pattern number = Pattern.compile("^.*[0-9].*$");
            Pattern specialCharacter =
Pattern.compile("^.*[^a-zA-Z0-9].*$");
            private Boolean isValidPassword(String password)
            {
// Checks if password length is less than 8
                if (password.length() < 8)
                {
                    return false;
                }

// Returns false if password doesn't contain a lower case
character
            if (!lowerCase.matcher(password).matches())
               {
                    return false;
                }
            // Returns false if password doesn't contain an
upper case character
            if (!upperCase.matcher(password).matches())
                {
                    return false;
                }
            // Returns false if password doesn't contain a
number
            if (!number.matcher(password).matches())
                {
                return false;
                }
// Returns false if password doesn't contain a special
character
                if
(!specialCharacter.matcher(password).matches())
                {
                    return false;
                }
                return true;
            }
    }
```

XML code for Login Activity

```
<?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=".LoginActivity">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="151dp"
        android:layout_marginBottom="638dp"
        android:text="Login Here"
        android:textSize="36sp" />

    <EditText
        android:id="@+id/emailEditText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="121dp"
        android:layout_marginBottom="468dp"
        android:ems="10"
        android:hint="email"
        android:inputType="textEmailAddress" />
```

```
    <EditText
        android:id="@+id/passwordEditText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="116dp"
        android:layout_marginBottom="371dp"
        android:ems="10"
        android:hint="password"
        android:inputType="textPassword" />

    <Button
        android:id="@+id/loginBtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="180dp"
        android:layout_marginBottom="241dp"
        android:text="Login" />
</RelativeLayout>
```

JAVA Code for LoginActivity

```
package com.example.myapplication11;

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;
import android.widget.Toast;

public class LoginActivity extends AppCompatActivity
```

```
{
    EditText emailEditText, passwordEditText;
    Button loginBtn;
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        emailEditText = findViewById(R.id.emailEditText);
        passwordEditText =
findViewById(R.id.passwordEditText);
        loginBtn = findViewById(R.id.loginBtn);
        String registeredEmail =
getIntent().getStringExtra("email");
        String registeredPassword =
getIntent().getStringExtra("password");
        loginBtn.setOnClickListener(new
View.OnClickListener()
        {
            @Override
            public void onClick(View v)
            {
                String email =
emailEditText.getText().toString();
                String password =
passwordEditText.getText().toString();
                if (registeredEmail.equals(email) &&
registeredPassword.equals(password))
                {
                    Intent intent = new
Intent(LoginActivity.this, LoginSuccessActivity.class);
                    startActivity(intent);
                } else
                    {
                    Toast.makeText(LoginActivity.this,
"Invalid Credentials",         Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}
```

XML Code for LoginSuccessActvity

```
<?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"
```

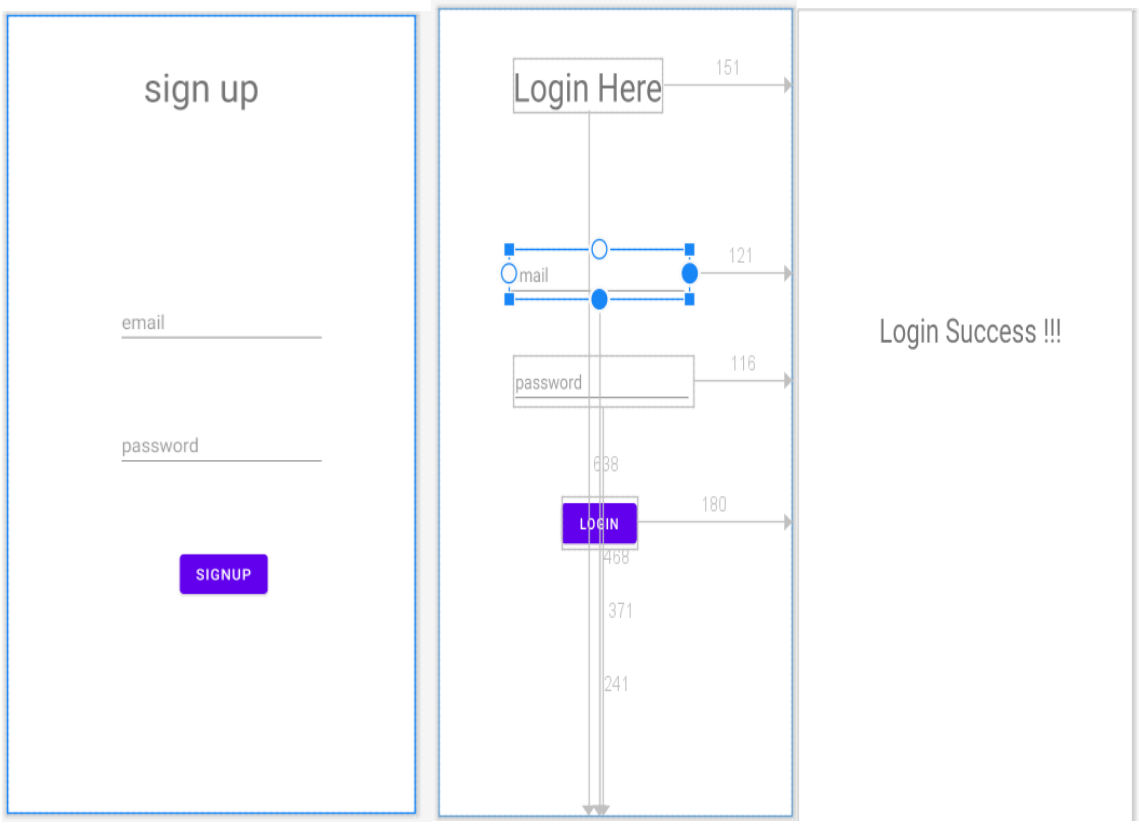
```
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".LoginSuccessActivity">

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="87dp"
        android:layout_marginBottom="426dp"
        android:text="Login Success !!!"
        android:textSize="30sp"
        tools:layout_editor_absoluteX="69dp"
        tools:layout_editor_absoluteY="218dp" />
</RelativeLayout>
```

JAVA Code for LoginSuccessActivity

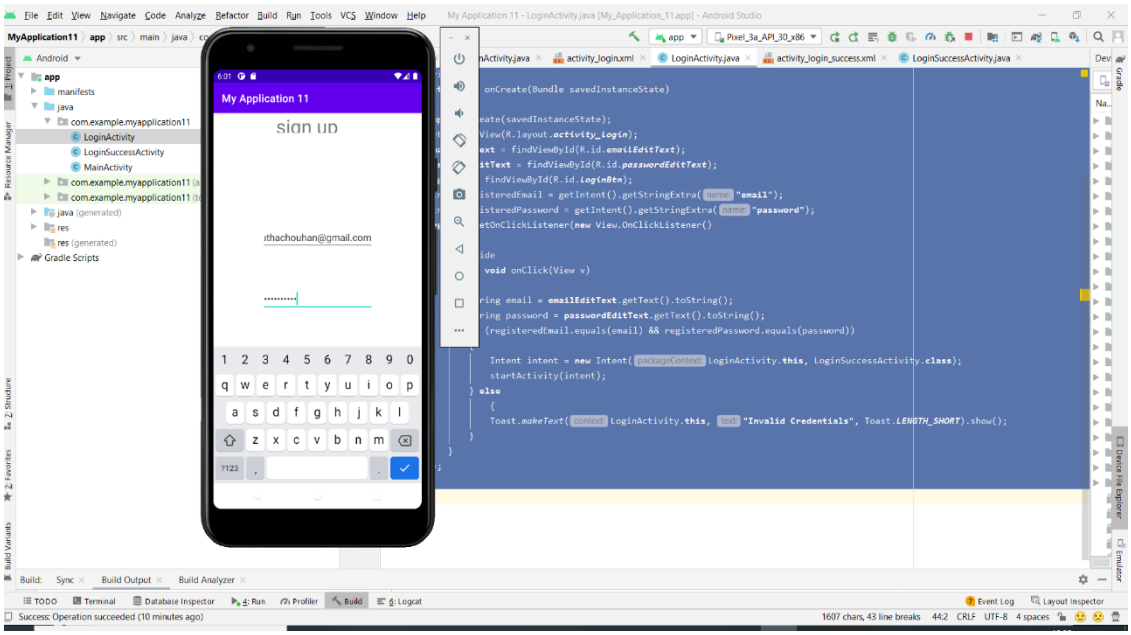
```
package com.example.myapplication11;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

public class LoginSuccessActivity extends AppCompatActivity
{

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

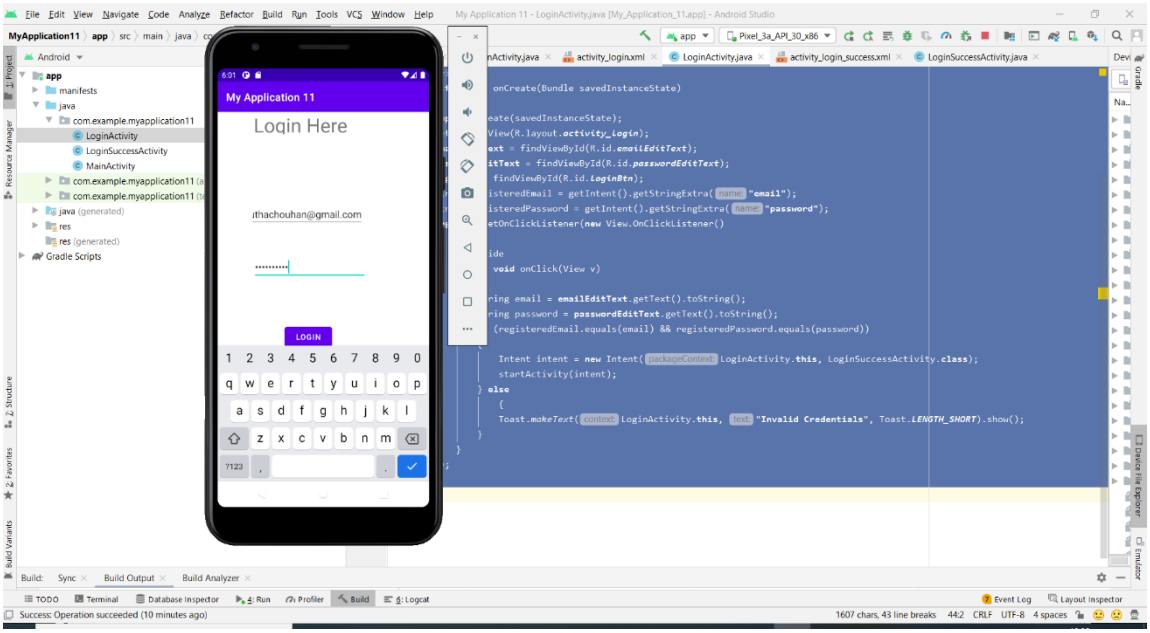
EXECUTION STEPS

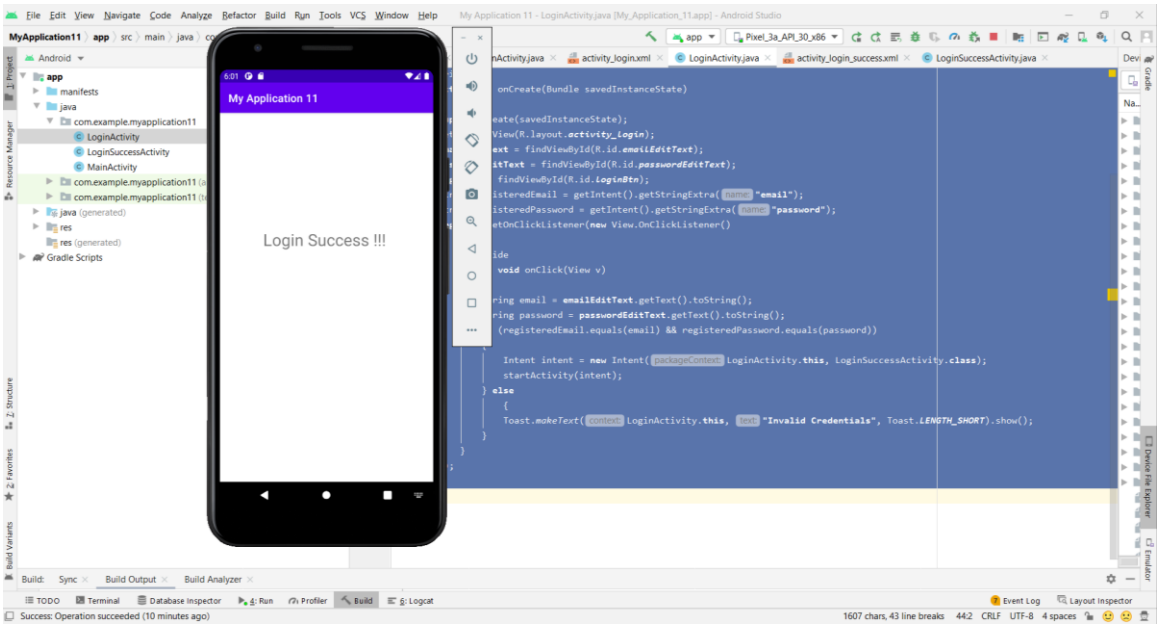
Step 1: Create SignUp Activity followed by Login and LoginSuccess Activty add two edit text view one for email and other for password it should be taken from user during signUp and Login as Shown below



Step 2: Write Java code to perform sign up and login function then execute your application to get output on emulator. Enter email and password to perform sign up activity


My Application 11
sign up
ithachouhan@gmail.com


My Application 11
Login Here
ithachouhan@gmail.com
LOGIN


My Application 11
Login Success !!!

Program 4. Develop an application to set an image as wallpaper. On click of a button, the wallpaper image should start to change randomly every 30 seconds.

XML 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=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="143dp"
        android:layout_marginBottom="616dp"
        android:text="Wall paper App"
        android:textSize="24sp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/button"
        android:layout_width="371dp"
        android:layout_height="wrap_content"
```

```
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="21dp"
        android:layout_marginBottom="393dp"
        android:text="Click to change the wallpaper" />

</RelativeLayout>
```

JAVA Code:

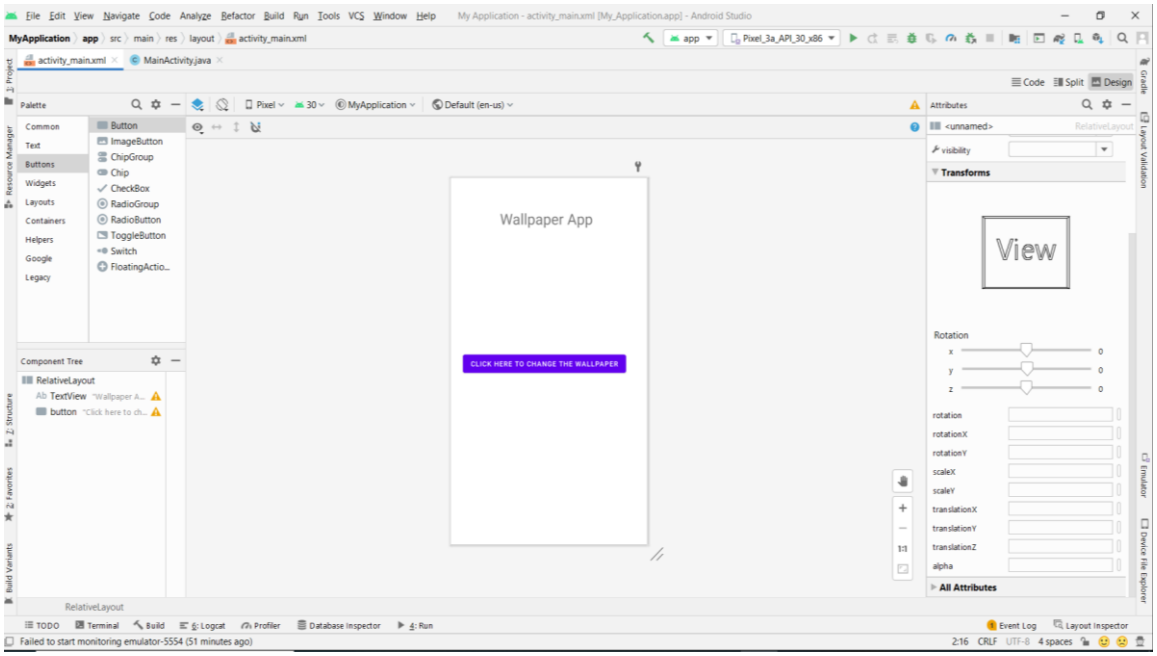
```
package com.example.myapplication;

import androidx.appcompat.app.AppCompatActivity;

import android.app.WallpaperManager;
import android.graphics.Bitmap;
import android.graphics.drawable.BitmapDrawable;
import android.graphics.drawable.Drawable;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

import java.io.IOException;
import java.util.Timer;
import java.util.TimerTask;

public class MainActivity extends AppCompatActivity {
Button chngwallpaper;
Timer mytimer;
Drawable drawable;
WallpaperManager wpm;
int prev=1;
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mytimer=new Timer();
        wpm=WallpaperManager.getInstance(this);
        chngwallpaper=findViewById(R.id.button);
        chngwallpaper.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                setWallpaper();
            }
        });
    }
    private void setWallpaper()
    {
        mytimer.schedule(new TimerTask() {
            @Override
            public void run() {
                if(prev==1)
                {

drawable=getResources().getDrawable(R.drawable.w1);
                     prev=2;
                }
                else if(prev==2)
                {

drawable=getResources().getDrawable(R.drawable.w2);
                    prev=3;
                }
                else if(prev==3)
```

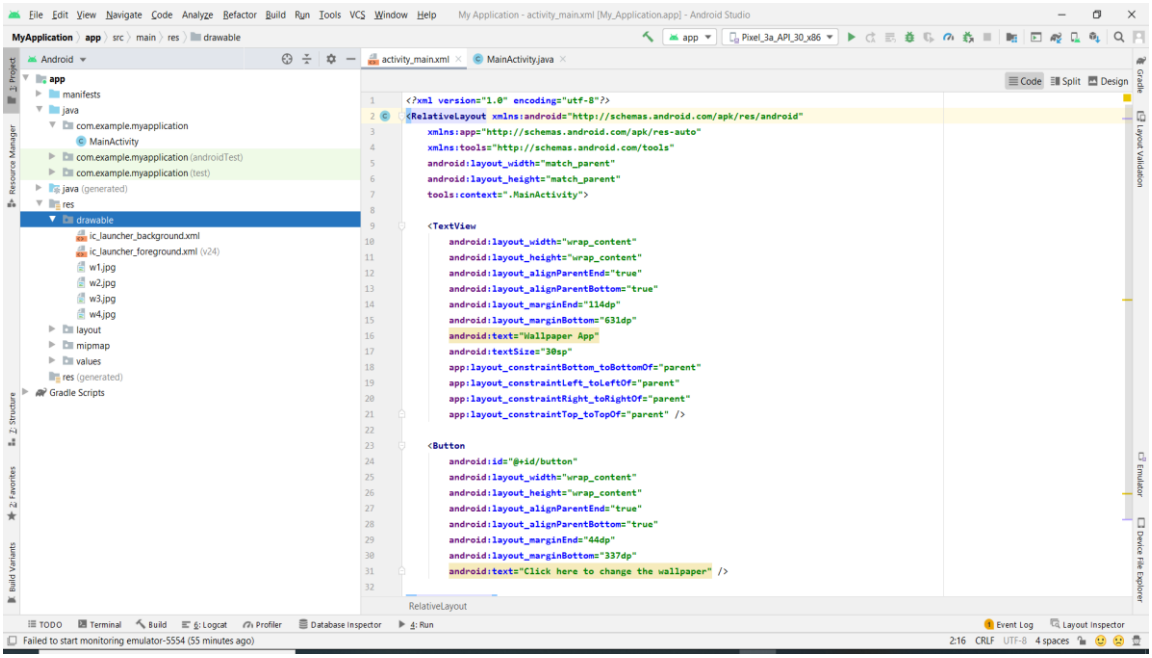
```
                {

drawable=getResources().getDrawable(R.drawable.w3);
                    prev=4;
                }
                else
                {

drawable=getResources().getDrawable(R.drawable.w4);
                  prev=1;
                }
                Bitmap
wallpaper=((BitmapDrawable)drawable).getBitmap();
                try {
                    wpm.setBitmap(wallpaper);
                }
                catch (IOException e)
                {
                    e.printStackTrace();
                }
            }
        },0,30000);
    }
}
```

EXECUTION STEPS:

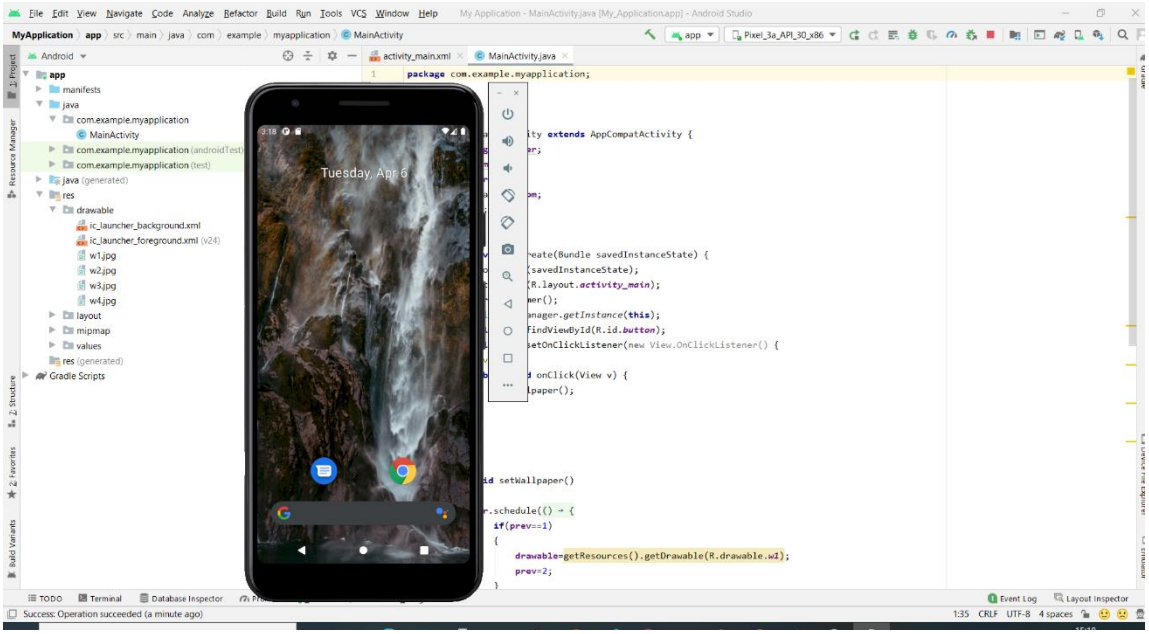
Step 1: Create a layout in xml part drag and drop 1 text view and 1 button and change the button name as “click here to change the wallpaper”.



Step 2: Download 4 wallpapers of jpeg or png format and name them as wallpaper1, wallpaper 2 and so on paste those images into drawable folder.



Step 3: Write the code in java using getDrawable function invoke the wallpaper that you have stored in drawable folder. Execute your application to see the output at emulator.


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
MyApplication app src main java com example myapplication MainActivity
Android
app
manifests
java
com.example.myapplication
MainActivity
com.example.myapplication (androidTest)
com.example.myapplication (test)
java (generated)
res
drawable
ic_launcher_background.xml
ic_launcher_foreground.xml (v24)
w1.jpg
w2.jpg
w3.jpg
w4.jpg
layout
mipmap
values
res (generated)
Gradle Scripts
activity_main.xml
MainActivity.java
package com.example.myapplication;
Tuesday, Apr 6
TODO
Terminal
Database Inspector
Event Log
Layout Inspector
Success: Operation succeeded (a minute ago)
1:35 CRLF UTF-8 4 spaces

Program 5. Write a program to create an activity with two buttons START and STOP. On pressing of the START button, the activity must start the counter by displaying the numbers from one and the counter must keep on counting until the STOP button is pressed. Display the counter value in a TextViewcontrol.

XML 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=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="59dp"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="51dp"
        android:layout_marginBottom="642dp"
        android:text="Counter Application"
        android:textAlignment="center"
        android:textColor="#2196F3"
        android:textSize="36sp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
```

```
<TextView
    android:id="@+id/tv1"
    android:layout_width="175dp"
    android:layout_height="43dp"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="117dp"
    android:layout_marginBottom="502dp"
    android:text="Counter Value"
    android:textAlignment="center" />

<Button
    android:id="@+id/bt1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="282dp"
    android:layout_marginBottom="401dp"
    android:shadowColor="#E91E63"
    android:text="START"
    android:textSize="18sp" />

<Button
    android:id="@+id/bt2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="41dp"
    android:layout_marginBottom="400dp"
```

```
        android:shadowColor="@color/purple_200"
        android:text="STOP"
        android:textColor="@color/white"
        android:textSize="18sp" />

</RelativeLayout>
```

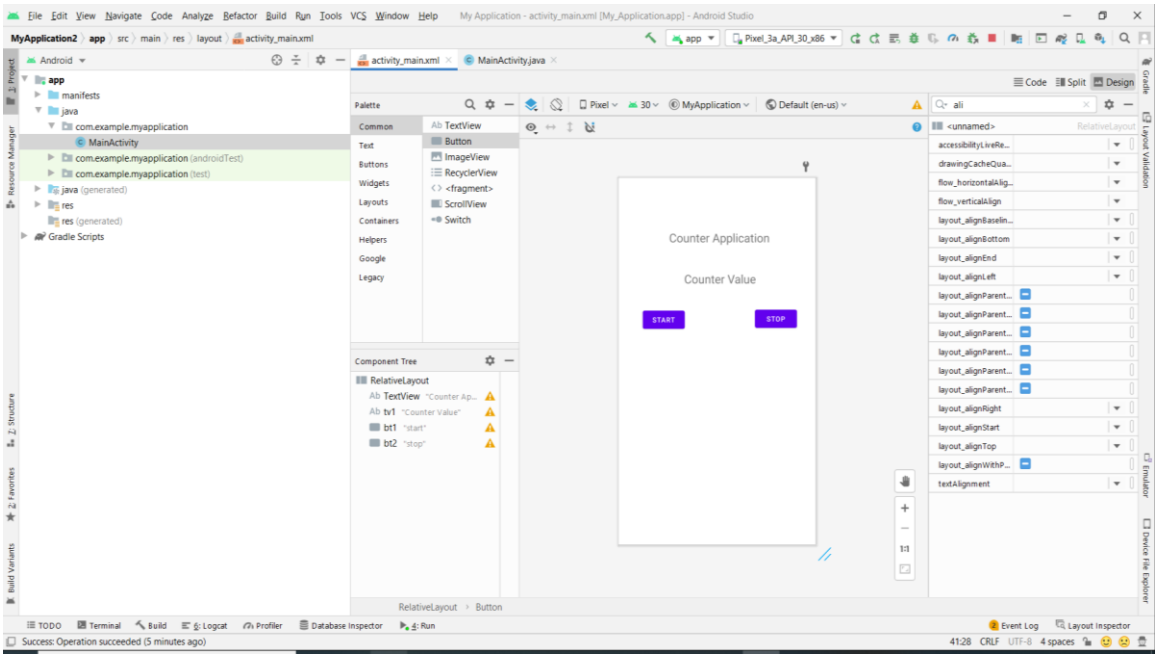
JAVA Code:

```
package com.example.lbp3;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.os.Handler;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
    Button b1,b2;
    TextView tc1;
    int i=1;
    Handler h1=new Handler();


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

        b1=findViewById(R.id.bt1);
        b2=findViewById(R.id.bt2);
```

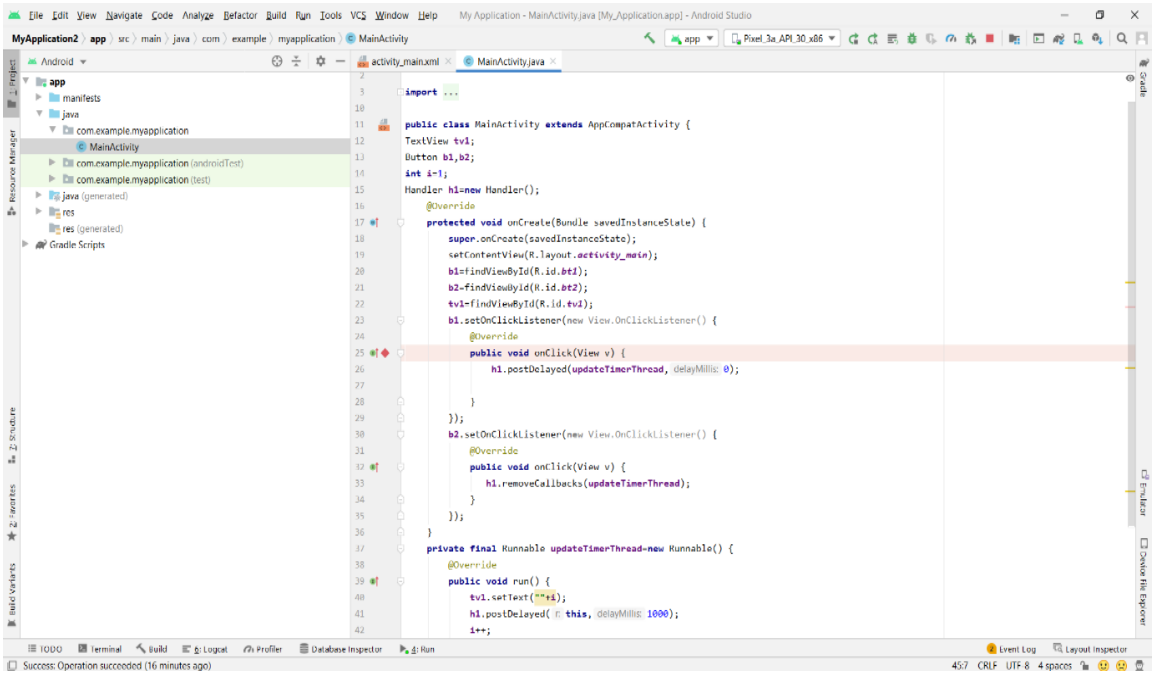
```
        tc1=findViewById(R.id.tv1);
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v)
            {
                h1.postDelayed(updateTimerThread,0);
            }
        });
        b2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                h1.removeCallbacks(updateTimerThread);
            }
        });

    }
    private final Runnable updateTimerThread=new Runnable()
{
        @Override
        public void run() {
            tc1.setText(""+i);
            h1.postDelayed(this,1000);
            i++;

        }
    };
}
```

EXECUTION STEPS:

Step1: create a layout in XML part and drag and drop 1 text view and two buttons, name the button as start and stop button.

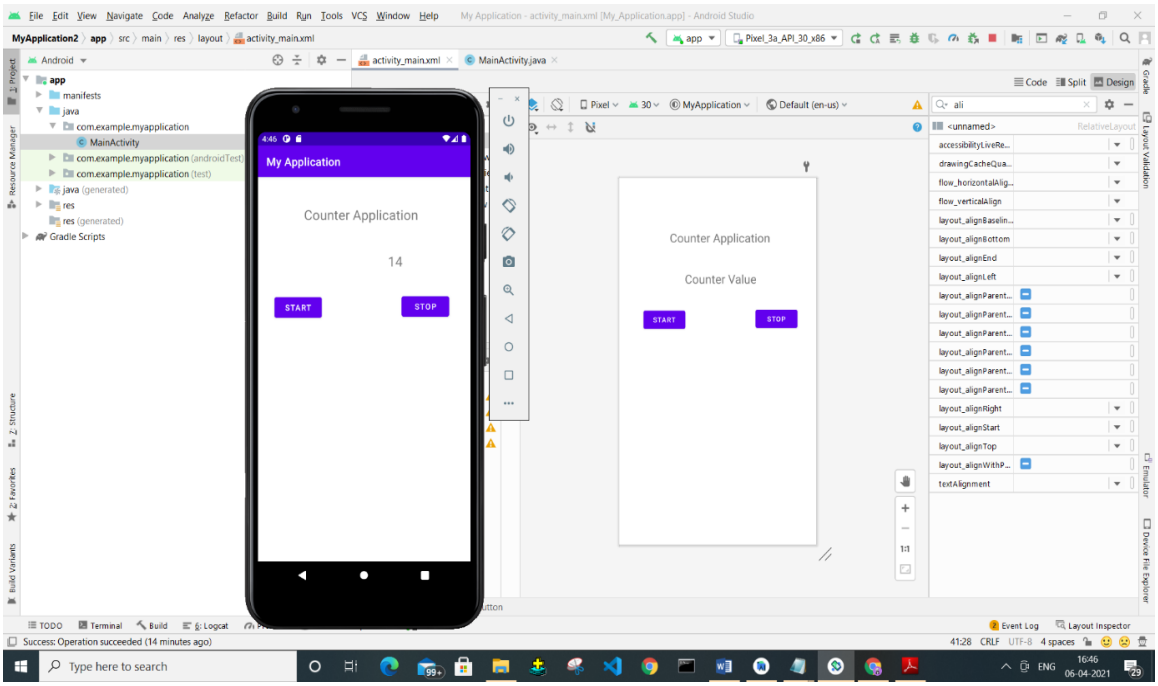


Step 2: Write the code in java part to start the counter and to display the numbers from 1 on clicking start button and on clicking stop button counter application should stop.



```
import ...

public class MainActivity extends AppCompatActivity {
TextView tv1;
Button b1,b2;
int i=1;
Handler h1=new Handler();
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        b1=findViewById(R.id.bt1);
        b2=findViewById(R.id.bt2);
        tv1=findViewById(R.id.tv1);
        b1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                h1.postDelayed(updateTimerThread, delayMillis: 0);

            }
        });
        b2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                h1.removeCallbacks(updateTimerThread);
            }
        });
    }
    private final Runnable updateTimerThread=new Runnable() {
        @Override
        public void run() {
            tv1.setText(""+i);
            h1.postDelayed( r: this, delayMillis: 1000);
            i++;
```

Step3: Execute your application to run your application on the emulator.



Program 6. Create two files of XML and JSON type with values for City_Name, Latitude, Longitude, Temperature, and Humidity. Develop an application to create an activity with two buttons to parse the XML and JSON files which when clicked should display the data in their respective layouts side by side.

XML 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=".MainActivity">

    <TextView
        android:layout_width="380dp"
        android:layout_height="72dp"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="16dp"
        android:layout_marginBottom="621dp"
        android:text="Parser Application"
        android:textAlignment="center"
        android:textSize="30sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintHorizontal_bias="0.128"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.07" />
```

```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="281dp"
    android:layout_marginBottom="464dp"
    android:onClick="parseXml"
    android:text="XML" />

<Button
    android:id="@+id/button2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="69dp"
    android:layout_marginBottom="465dp"
    android:onClick="parseJson"
    android:text="JSON" />

<TextView
    android:id="@+id/display"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="37dp"
    android:layout_marginBottom="264dp"
    android:text=""
```

```
        android:textSize="18sp" />

</RelativeLayout>
```

JAVA Code

```
package com.example.myapplication;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.TextView;
import android.widget.Toast;

import org.json.JSONArray;
import org.json.JSONObject;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

import java.io.InputStream;
import java.nio.charset.StandardCharsets;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;

public class MainActivity extends AppCompatActivity {
TextView display;

    @Override
```

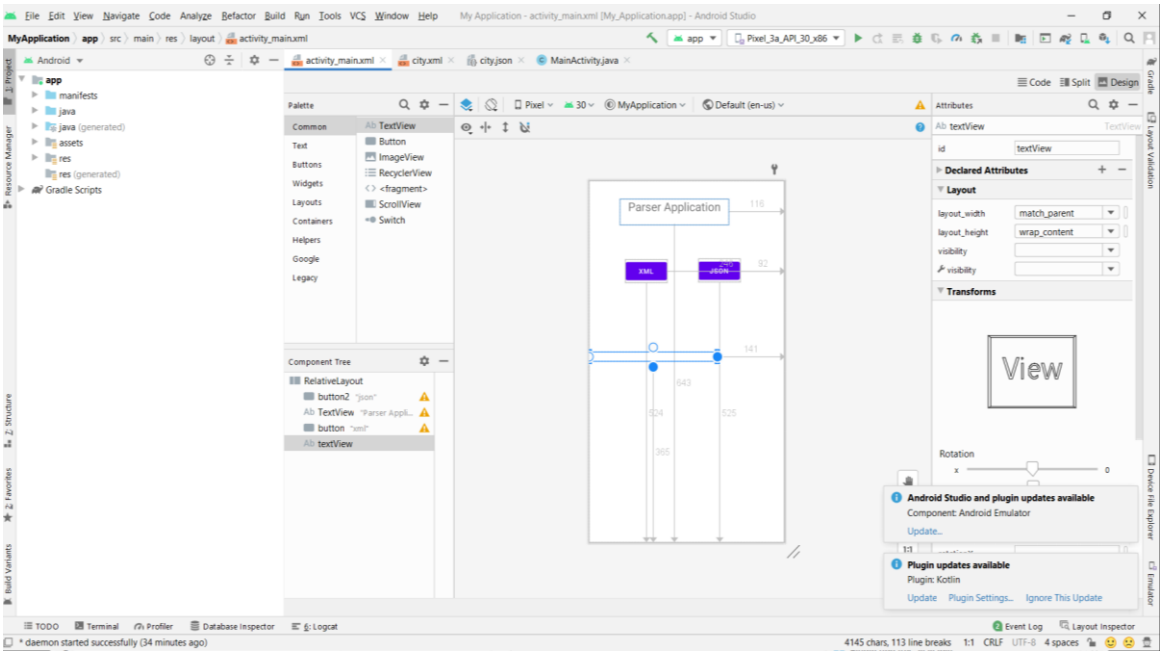
```
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        display=findViewById(R.id.display);
    }
    public void parseXml(View V)
    {
        try {
            InputStream is=getAssets().open("city.xml");
            DocumentBuilderFactory
documentBuilderFactory=DocumentBuilderFactory.newInstance();
            DocumentBuilder        documentBuilder        =
documentBuilderFactory.newDocumentBuilder();
            Document document=documentBuilder.parse(is);
            StringBuilder stringBuilder=new StringBuilder();
            stringBuilder.append("XML DATA");
            stringBuilder.append("\n-------");
            NodeList
nodeList=document.getElementsByTagName("place");
            for (int i=0;i<nodeList.getLength();i++)
            {
                Node node=nodeList.item(i);
                if(node.getNodeType()==Node.ELEMENT_NODE)
                {
                    Element element=(Element)node;
                    stringBuilder.append("\n
Lame").append(getValue("name",element));
                    stringBuilder.append("\n
Lattitude").append(getValue("lat",element));
                    stringBuilder.append("\n
Longitude").append(getValue("long",element));
                    stringBuilder.append("\n
Temperature").append(getValue("temperature",element));
```

```
                    stringBuilder.append("\n
Humidity").append(getValue("humidity",element));
                    stringBuilder.append("\n------");


                }
            }
            display.setText(stringBuilder.toString());
        }
        catch (Exception e)
        {
               e.printStackTrace();
            Toast.makeText(MainActivity.this,"Error       in
reading xml file",Toast.LENGTH_SHORT).show();
        }
    }
    public void parseJson(View V)
    {
        String json;
        StringBuilder stringBuilder=new StringBuilder();
        try
        {
            InputStream is=getAssets().open("city.json");
            int size=is.available();
            byte[] buffer=new byte[size];
            is.read(buffer);
            json=new String(buffer, StandardCharsets.UTF_8);
            JSONArray jsonArray=new JSONArray(json);
            stringBuilder.append("JSON DATA");
            stringBuilder.append("\n-----");
            for (int i=0;i<jsonArray.length();i++)
            {
                JSONObject
jsonObject=jsonArray.getJSONObject(i);
```

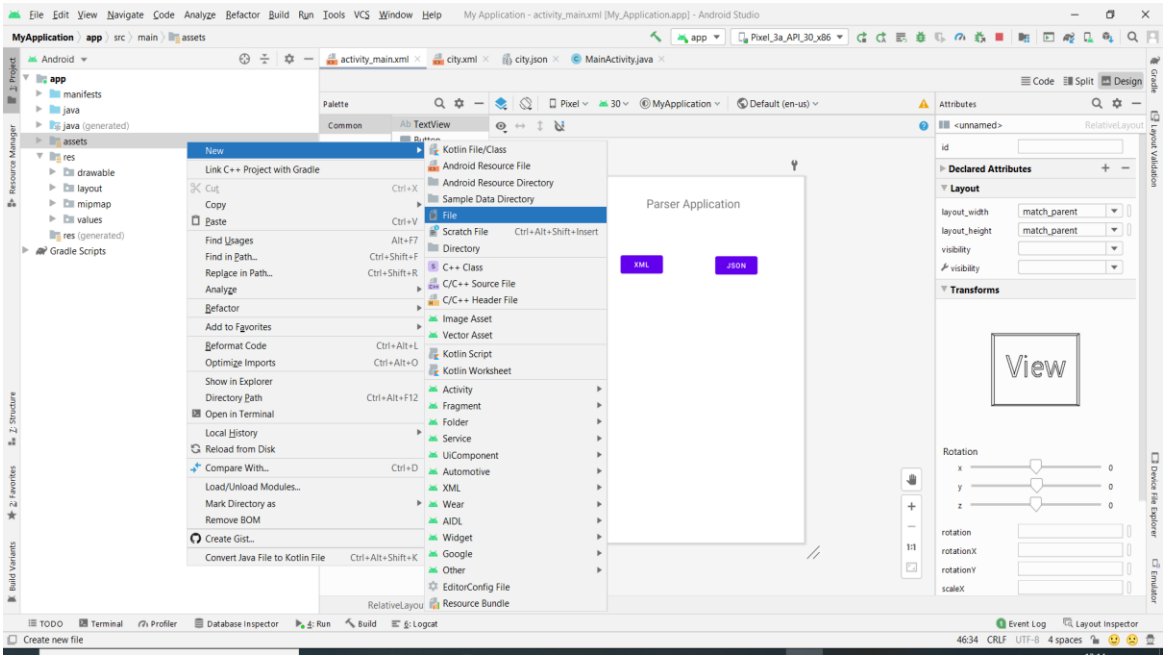
```
                stringBuilder.append("\n
Name").append(jsonObject.getString("name"));

                stringBuilder.append("\n
Latitude").append(jsonObject.getString("lat"));

                stringBuilder.append("\n
Longitude").append(jsonObject.getString("long"));

                stringBuilder.append("\n
Temperature").append(jsonObject.getString("temperature"));

                stringBuilder.append("\n
Humidity").append(jsonObject.getString("humidity"));

                stringBuilder.append("\n-----");

            }

            display.setText(stringBuilder.toString());

            is.close();

        }

        catch (Exception e)

        {

            e.printStackTrace();

            Toast.makeText(MainActivity.this,"Error       in
reading json file",Toast.LENGTH_SHORT).show();

        }

    }

    private String getValue(String tag,Element element)

    {

     return
element.getElementsByTagName(tag).item(0).getChildNodes().it
em(0).getNodeValue();

    }

}
```

EXECUTION STEPS:

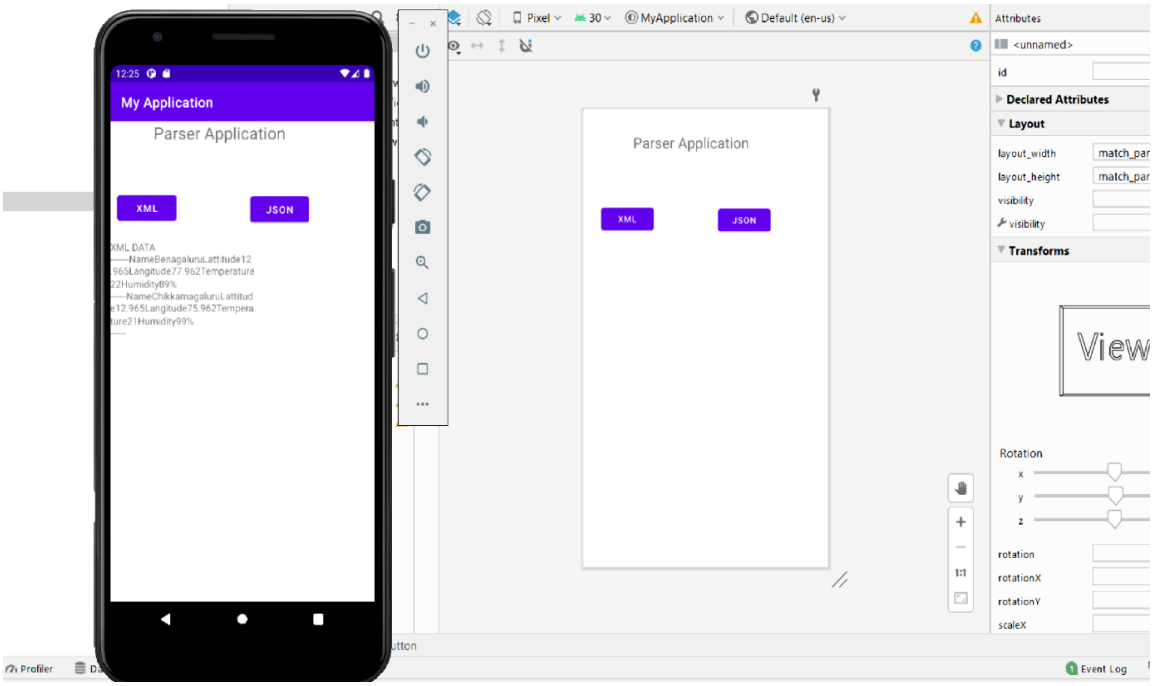
Step1: Create a relative layout in that drag and drop 1 text and name it as Parser Application. Drag and drop two button name them as XML and JSON respectively. Drag and drop one text view so that on clicking XML button it should display content present inside the xml file and on clicking JSON button it should display content that is present inside the json file.



Step 2:Click on app→new→folder→select Asset folder, inside asset folder create two file with extension .xml and .json paste the city information inside those file.



Step 3: write the code in java to parse the information that is present inside xml and json file on to emulator.



Program 7. Develop a simple application with one EditText so that the user can write some text in it. Create a button called "Convert Text to Speech" that converts the user input text into voice.

XML 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"
    android:textAlignment="center"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="291dp"
        android:layout_height="58dp"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="67dp"
        android:layout_marginBottom="617dp"
        android:text="Text to Speech"
        android:textAlignment="center"
        android:textSize="36sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/editText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="113dp"
        android:layout_marginBottom="484dp"
        android:ems="10"
        android:inputType="textPersonName"
        android:hint="enter text here"
        android:text="" />

    <Button
        android:id="@+id/b1"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
       android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:onClick="convert"
        android:layout_marginEnd="185dp"
        android:layout_marginBottom="359dp"
        android:text="convert" />

</RelativeLayout>
```

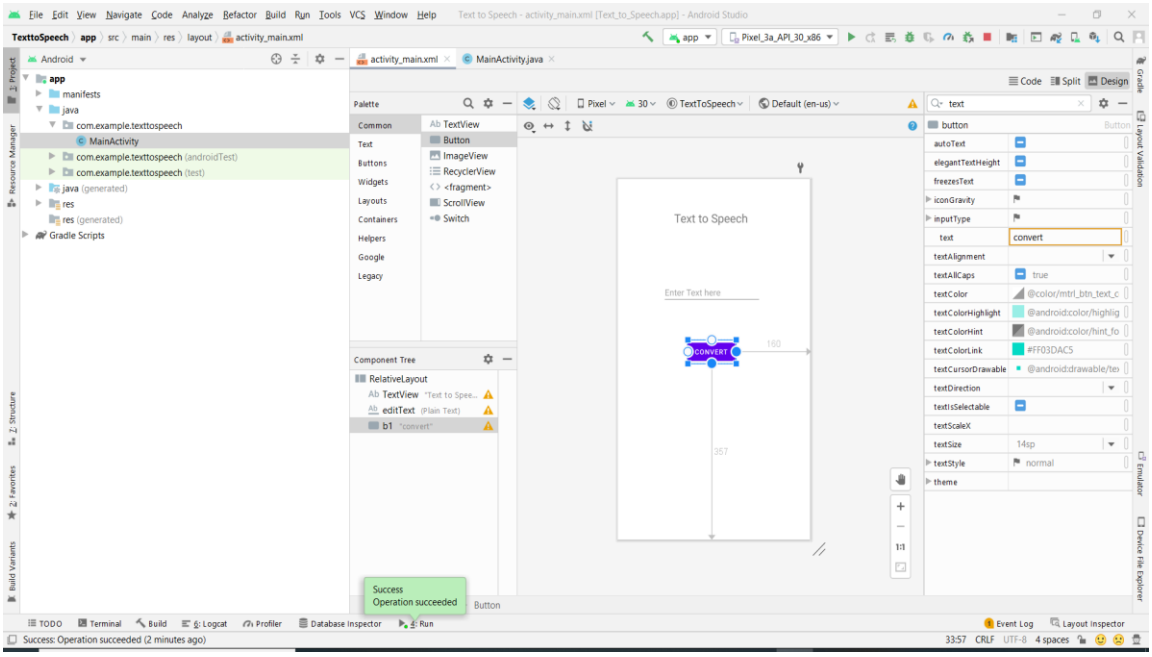
JAVA Code

```
package com.example.lb7;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.speech.tts.TextToSpeech;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

import java.util.Locale;

public class MainActivity extends AppCompatActivity {
EditText e1;
TextToSpeech t1;
    @Override
    protected void onCreate(Bundle savedInstanceState)
 {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        e1=findViewById(R.id.editText);
        t1=new   TextToSpeech(getApplicationContext(),   new
TextToSpeech.OnInitListener() {
            @Override
            public void onInit(int status)
            {
             if(status!=TextToSpeech.ERROR)
             {
                 t1.setLanguage(Locale.UK);
             }
            }
        });
    }
```

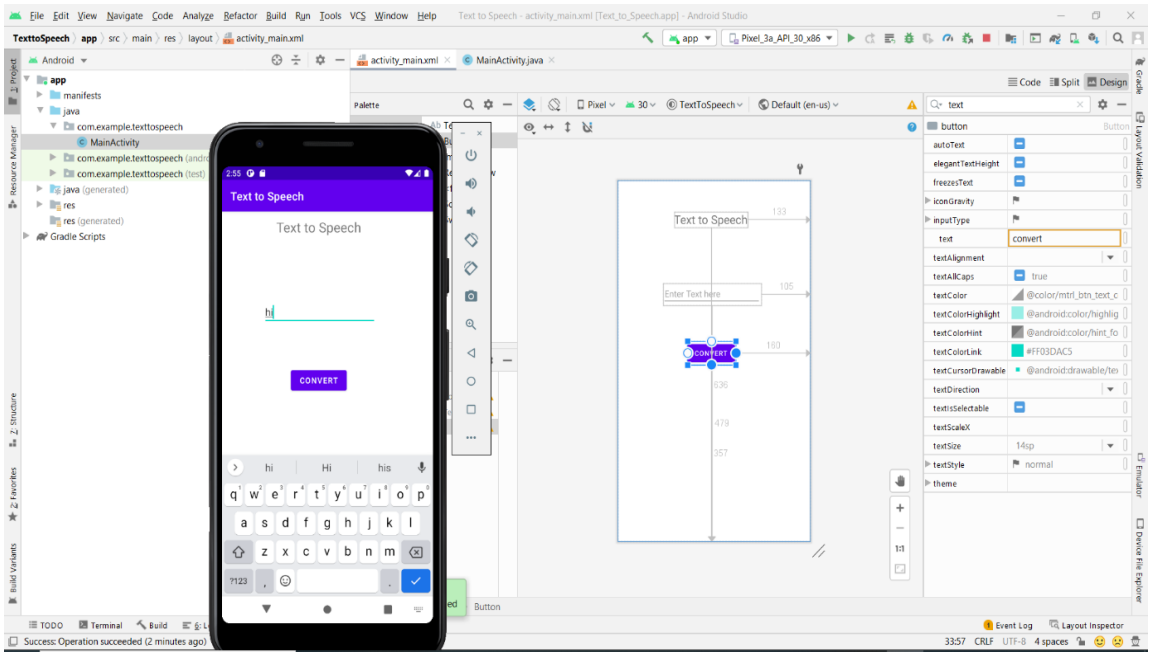
```
    public void convert(View V)
    {
        String tospeak=e1.getText().toString();
        t1.speak(tospeak,TextToSpeech.QUEUE_FLUSH,null);
    }
}
```

EXECUTION STEPS:

Step1: Create a Relative layout drag and drop one text view for title drag and drop one EditText to type text you convert to speech. Add one button and name it as Convert on clicking convert button whatever the text present inside the EditText field



Step 2: Write Java code to convert text to speech and get your output on emulator.


Text to Speech
Text to Speech
hi
CONVERT
Text to Speech
Enter Text here
CONVERT

Program 8. Create an activity like a phone dialer withCALLand SAVE buttons. On pressing the CALL button, it must call the phone number and on pressing the SAVE button it must save the number to the phone contacts.

XML-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=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="161dp"
        android:layout_marginBottom="651dp"
        android:text="Call App"
        android:textSize="24sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/editText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="191dp"
        android:layout_marginBottom="541dp"
        android:ems="10"
        android:hint="Enter phone number"
        android:inputType="" />

    <Button
        android:id="@+id/clearBtn"
        android:layout_width="82dp"
```

```
        android:layout_height="37dp"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="68dp"
        android:layout_marginBottom="546dp"
        android:text="Clear" />

    <Button
        android:id="@+id/btn1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="301dp"
        android:layout_marginBottom="471dp"
        android:onClick="inputNumber"
        android:background="#D0173F49"

        android:text="1" />

    <Button
        android:id="@+id/btn2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="193dp"
        android:layout_marginBottom="472dp"
        android:onClick="inputNumber"
        android:background="#D0173F49"

        android:text="2" />

    <Button
        android:id="@+id/btn3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:onClick="inputNumber"
        android:layout_marginEnd="80dp"
        android:layout_marginBottom="471dp"
        android:background="#D0173F49"

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

```
<Button
    android:id="@+id/btn4"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="302dp"
    android:layout_marginBottom="403dp"
    android:background="#D0173F49"

    android:onClick="inputNumber"
    android:text="4" />

<Button
    android:id="@+id/btn5"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="193dp"
    android:layout_marginBottom="402dp"
    android:background="@color/purple_200"

    android:onClick="inputNumber"
    android:text="5" />

<Button
    android:id="@+id/btn6"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="77dp"
    android:layout_marginBottom="403dp"
    android:background="#D0173F49"
    android:onClick="inputNumber"

    android:text="6" />

<Button
    android:id="@+id/btn7"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
```

```
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="302dp"
        android:layout_marginBottom="336dp"
        android:background="#D0173F49"

        android:onClick="inputNumber"
        android:text="7" />

    <Button
        android:id="@+id/btn8"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="193dp"
        android:layout_marginBottom="338dp"
        android:onClick="inputNumber"
        android:text="8" />

    <Button
        android:id="@+id/btn9"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="69dp"
        android:layout_marginBottom="336dp"
        android:background="#D0173F49"
        android:onClick="inputNumber"

        android:text="9" />

    <Button
        android:id="@+id/btn10"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="302dp"
        android:layout_marginBottom="283dp"
        android:background="#D0173F49"

        android:onClick="inputNumber"
        android:text="*" />
```

```
<Button
    android:id="@+id/btn11"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="190dp"
    android:layout_marginBottom="283dp"
    android:background="#D0173F49"

    android:onClick="inputNumber"
    android:text="0" />

<Button
    android:id="@+id/btn12"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="78dp"
    android:layout_marginBottom="282dp"
    android:background="#D0173F49"

    android:onClick="inputNumber"
    android:text="#" />

<Button
    android:id="@+id/callBtn"
    android:layout_width="127dp"
    android:layout_height="63dp"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="232dp"
    android:layout_marginBottom="194dp"
    android:text="Call" />

<Button
    android:id="@+id/saveBtn"
    android:layout_width="107dp"
    android:layout_height="60dp"
    android:layout_alignParentEnd="true"
    android:layout_alignParentBottom="true"
    android:layout_marginEnd="59dp"
    android:layout_marginBottom="193dp"
    android:text="Save" />
```

```
</RelativeLayout>
```

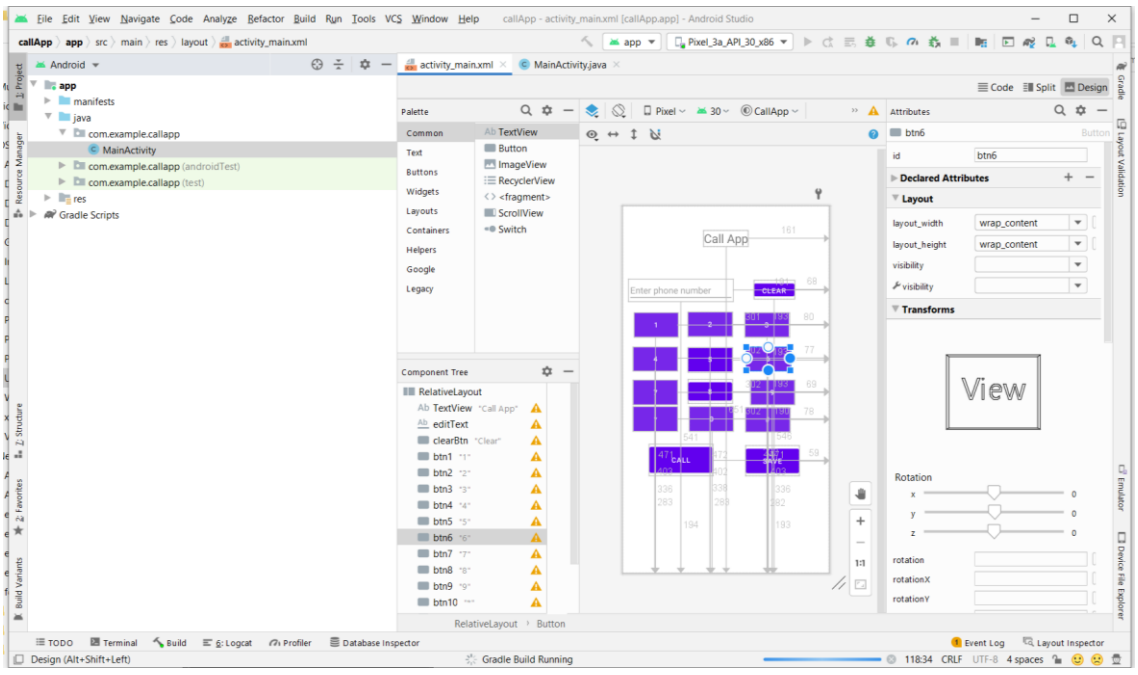
JAVA Code

```
package com.example.myapplication;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.provider.ContactsContract;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

public class MainActivity extends AppCompatActivity {
    EditText editText;
    Button clearBtn,callBtn,saveBtn;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        editText=findViewById(R.id.editText);
        clearBtn=findViewById(R.id.clearBtn);
        callBtn=findViewById(R.id.callBtn);
        saveBtn=findViewById(R.id.saveBtn);
        clearBtn.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                editText.setText("");
            }
        });
        callBtn.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String phnum=editText.getText().toString();
                Intent                              intent=new
Intent(Intent.ACTION_DIAL);
                intent.setData(Uri.parse("tel:"+phnum));
                startActivity(intent);
            }
        });
```

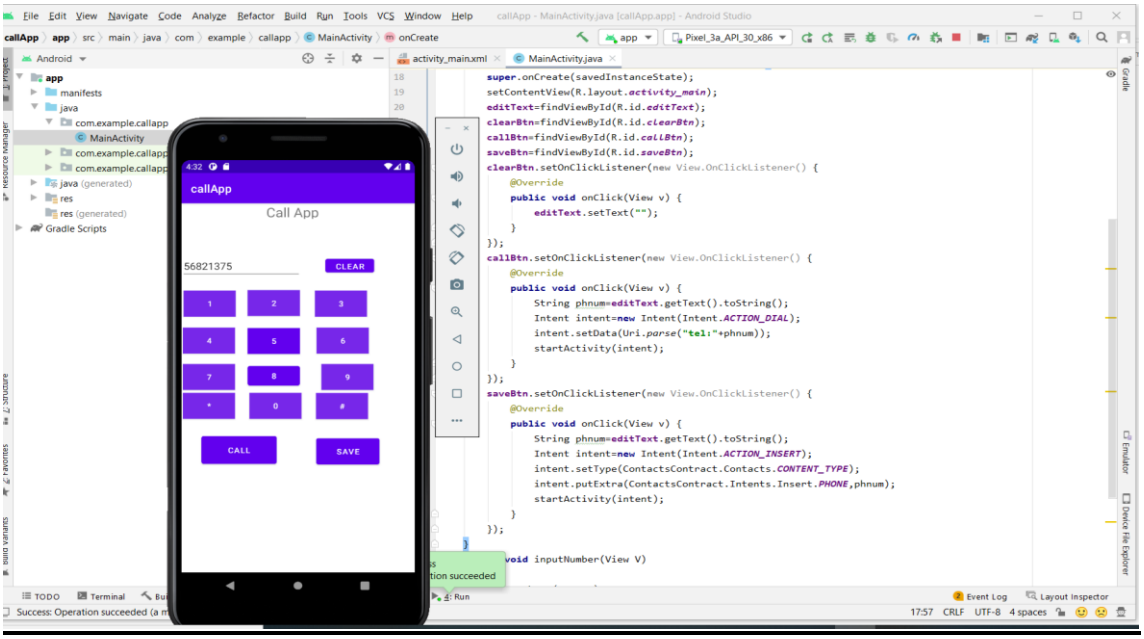
```
        saveBtn.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String phnum=editText.getText().toString();
                Intent                                  intent=new
Intent(Intent.ACTION_INSERT);

intent.setType(ContactsContract.Contacts.CONTENT_TYPE);

intent.putExtra(ContactsContract.Intents.Insert.PHONE,phnum)
;
                startActivity(intent);
            }
        });
    }
    public void inputNumber(View V)
    {
        Button btn=(Button)V;
        String digit=btn.getText().toString();
        String phnum=editText.getText().toString();
        editText.setText(phnum + digit);
    }
}
```

EXECUTION STEPS:

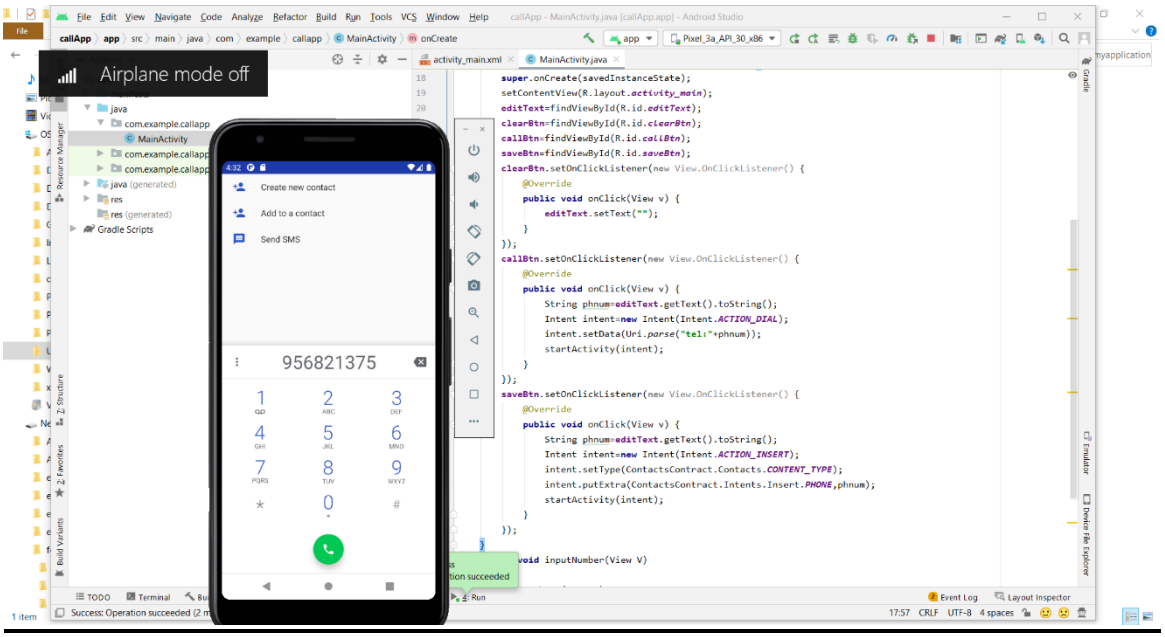
Step1: Select edit text view and few button to create a call application as shown below



Step 2: Write the java code to perform phone call operation then execute your application to see the output on emulator



Step 3: On pressing call button the page will be redirected to call that number entered by the user and by pressing save button we can the save the number



Program 9. (PART-B): 5. Create an application to demonstrate a basic media player that allows the user to Forward Backward, Play and Pause an audio. Also, make use of the indicator in the seek bar to move the audio forward or backward as required.

XML 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=".MainActivity">
    <SeekBar
        android:id="@+id/seekBar"
        android:layout_width="255dp"
        android:layout_height="28dp"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="66dp"
        android:layout_marginBottom="311dp" />
    <ImageButton
        android:id="@+id/rewind"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="256dp"
        android:layout_marginBottom="219dp"
        app:srcCompat="@android:drawable/ic_media_rew" />
    <ImageButton
        android:id="@+id/playButton"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="168dp"
        android:layout_marginBottom="223dp"
        app:srcCompat="@android:drawable/ic_lock_power_off"
/>
    <ImageButton
        android:id="@+id/forward"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="79dp"
        android:layout_marginBottom="220dp"
        app:srcCompat="@android:drawable/ic_media_ff" />


</RelativeLayout>
```

JAVA Code

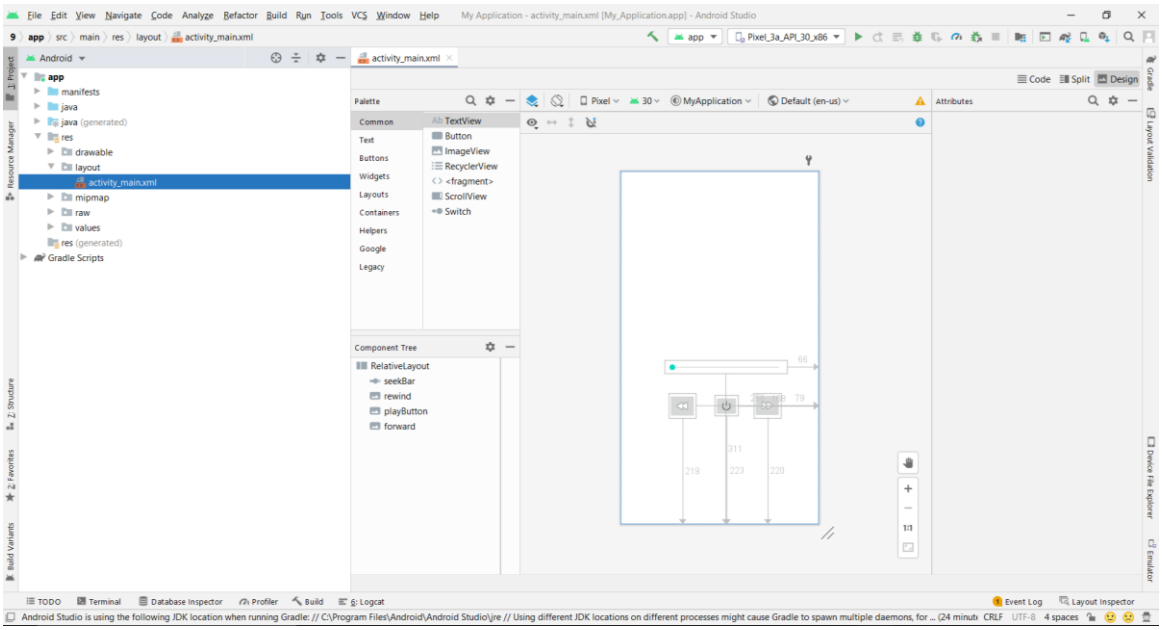
```
ackage com.example.myapplication;

import androidx.appcompat.app.AppCompatActivity;

import android.media.MediaPlayer;
import android.os.Bundle;
import android.os.Handler;
import android.view.View;
import android.widget.ImageButton;
import android.widget.SeekBar;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    private ImageButton playButton,forward,rewind;
    private SeekBar seekbar;
    private MediaPlayer mediaPlayer;
    private Handler handler = new Handler();
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        playButton = findViewById(R.id.playButton);
        forward = findViewById(R.id.forward);
        rewind = findViewById(R.id.rewind);
        seekbar = findViewById(R.id.seekBar);
        prepareMediaPlayer();
        seekbar.setMax(100);
        playButton.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if(mediaPlayer.isPlaying()){
                    handler.removeCallbacks(updater);
                    mediaPlayer.pause();
                }else {
                    mediaPlayer.start();
                    updateSeekBar();
                }
            }
        });
        forward.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {

if(mediaPlayer.getDuration()>mediaPlayer.getCurrentPosition(
) + 10000){

mediaPlayer.seekTo(mediaPlayer.getCurrentPosition()        +
10000);
                    updateSeekBar();
                }
            }
        });
        rewind.setOnClickListener(new View.OnClickListener()
{
            @Override
            public void onClick(View v) {
                if(mediaPlayer.getCurrentPosition()>10000){

mediaPlayer.seekTo(mediaPlayer.getCurrentPosition()        -
10000);
                    updateSeekBar();
                }
            }
```

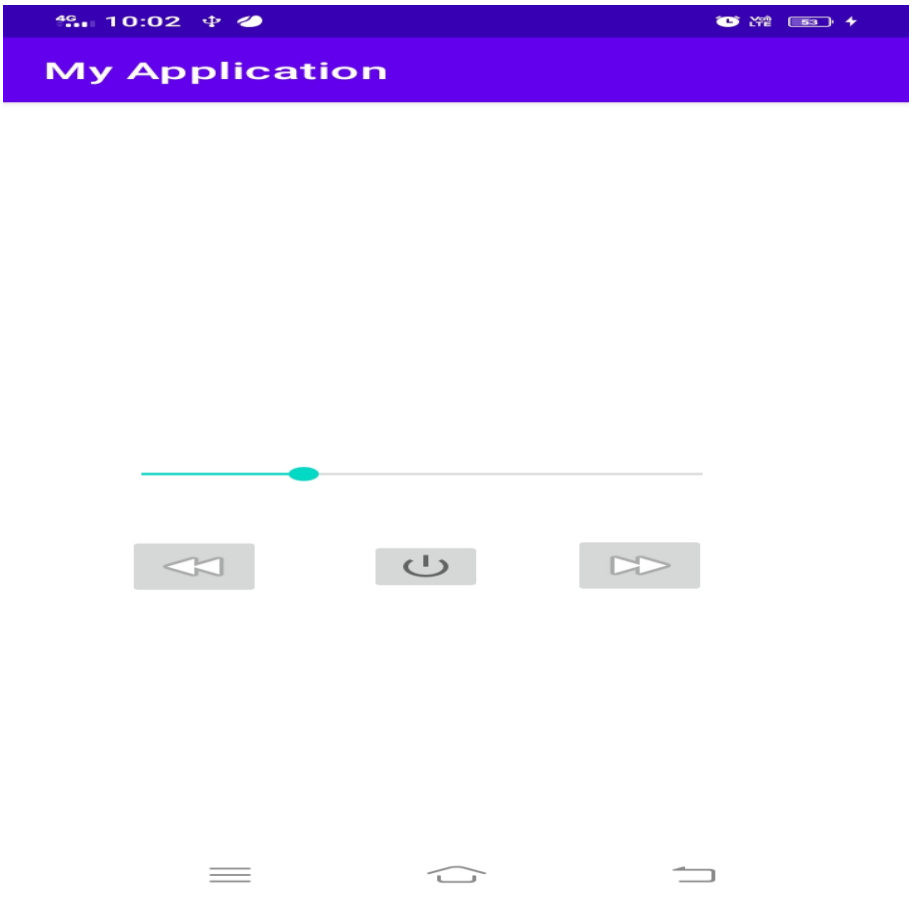
```
        });
        seekbar.setOnTouchListener((v, event) -> {
            SeekBar s = (SeekBar) v;
            int                  position                  =
(mediaPlayer.getDuration()/100)*s.getProgress();
            mediaPlayer.seekTo(position);
            return false;
        });
        mediaPlayer.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
            @Override
            public void onCompletion(MediaPlayer mp) {
                seekbar.setProgress(0);
                mediaPlayer.reset();
                prepareMediaPlayer();
            }
        });
    }
    private void prepareMediaPlayer (){
        try {
            mediaPlayer =MediaPlayer.create(this,R.raw.m);
        }catch (Exception e){
            Toast.makeText(this,             e.getMessage(),
Toast.LENGTH_LONG).show();
        }
    }
    private Runnable updater = new Runnable() {
        @Override
        public void run() {
            updateSeekBar();
        }
    };
    private void updateSeekBar(){
        if(mediaPlayer.isPlaying()){

seekbar.setProgress((int)((float)mediaPlayer.getCurrentPosit
ion()/mediaPlayer.getDuration()*100));
            handler.postDelayed(updater,1000);
        }
    }
}
```

EXECUTION STEPS

Step 1: Create a relative layout drag and drop one seek bar to represent playing music then add 3 image button for play forward, play backward, for pause/resume. As shown below



Step 2: Create a raw folder copy and paste the mp3 song then write java code to perform the operation then connect your phone to the system by activating developer option in your phone the execute your application to play the music



Program 10: (PART-B): 6. Develop an application to demonstrate the use of Asynchronous tasks in android. The asynchronous task should implement the functionality of a simple moving banner. On pressing the Start Task button, the banner message should scroll from right to left. On pressing the Stop Task button, the banner message should stop. Let the banner message be "Demonstration of Asynchronous Task".

XML 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=".MainActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="123dp"
        android:layout_marginBottom="630dp"
        android:text="Async Task"
        android:textSize="36sp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
    <Button
        android:id="@+id/buttonstart"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
```

```
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="167dp"
        android:layout_marginBottom="441dp"
        android:text="Start" /> <Button
android:id="@+id/buttonstop"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="169dp"
        android:layout_marginBottom="328dp"
        android:text="Stop" />
    <TextView
        android:id="@+id/marqueeText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginStart="50dp"
        android:layout_marginTop="250dp"
        android:layout_marginEnd="117dp"
        android:layout_marginBottom="207dp"
        android:ellipsize="marquee"
        android:marqueeRepeatLimit="marquee_forever"
        android:scrollHorizontally="true"
        android:singleLine="true"
        android:text="Demonstration of Asynchronous Task
!!!!"
        android:textSize="20sp"
        android:textStyle="bold"
        android:visibility="invisible" />
</RelativeLayout>
```

JAVA Code

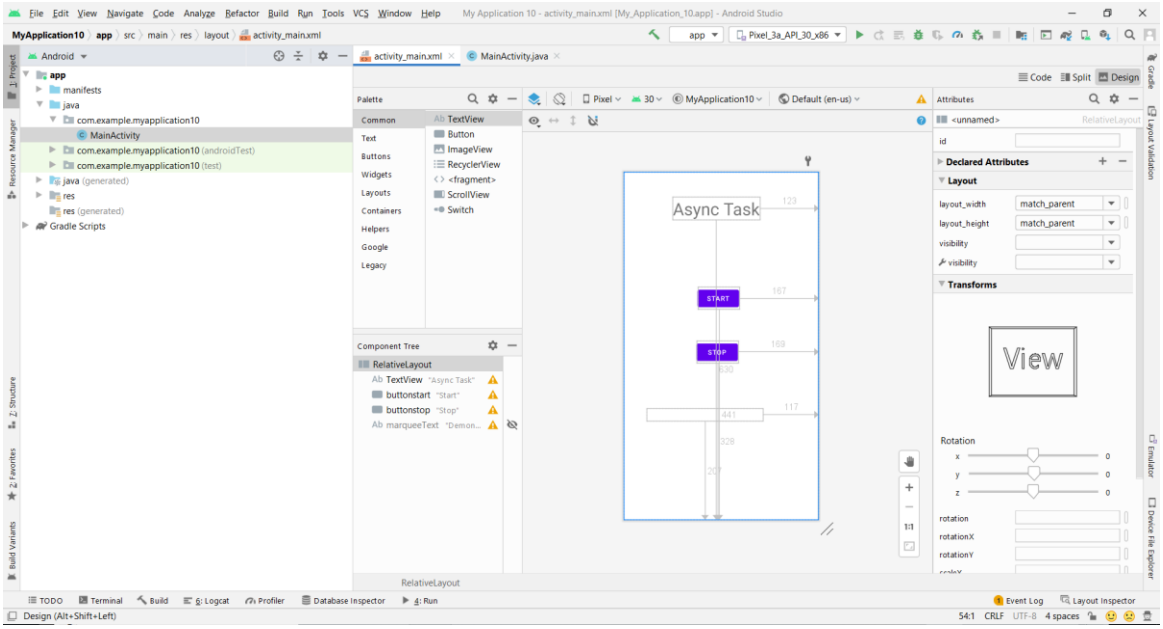
```
package com.example.myapplication10;


import androidx.appcompat.app.AppCompatActivity;


import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;


public class MainActivity extends AppCompatActivity {
   TextView marqtxt;
   Button btnstart, btnstop;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        marqtxt = (TextView) findViewById(R.id.marqueeText);
        btnstart = (Button) findViewById(R.id.buttonstart);
        btnstop = (Button) findViewById(R.id.buttonstop);
        btnstart.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                ExampleAsyncTask         exampleAsncTask=new
ExampleAsyncTask();
                exampleAsncTask.execute();
            }
        });
```

```
        btnstop.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                marqtxt.setSelected(false);
                marqtxt.setVisibility(View.INVISIBLE);
            }
        });
    }
    private class ExampleAsyncTask extends AsyncTask<String,
String,String> {
        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            Toast.makeText(getBaseContext(),"Async      Task
Started!!!!!!!!",Toast.LENGTH_SHORT).show();
        } @Override
        protected String doInBackground(String... strings) {
            try
            {
                Thread.sleep(250);
            }
            catch (InterruptedException e)
            {
                e.printStackTrace();
            } return null;
        } @Override
        protected void onPostExecute(String s) {
            super.onPostExecute(s);
            marqtxt.setVisibility(View.VISIBLE);
            marqtxt.setSelected(true);
        }
    }
```

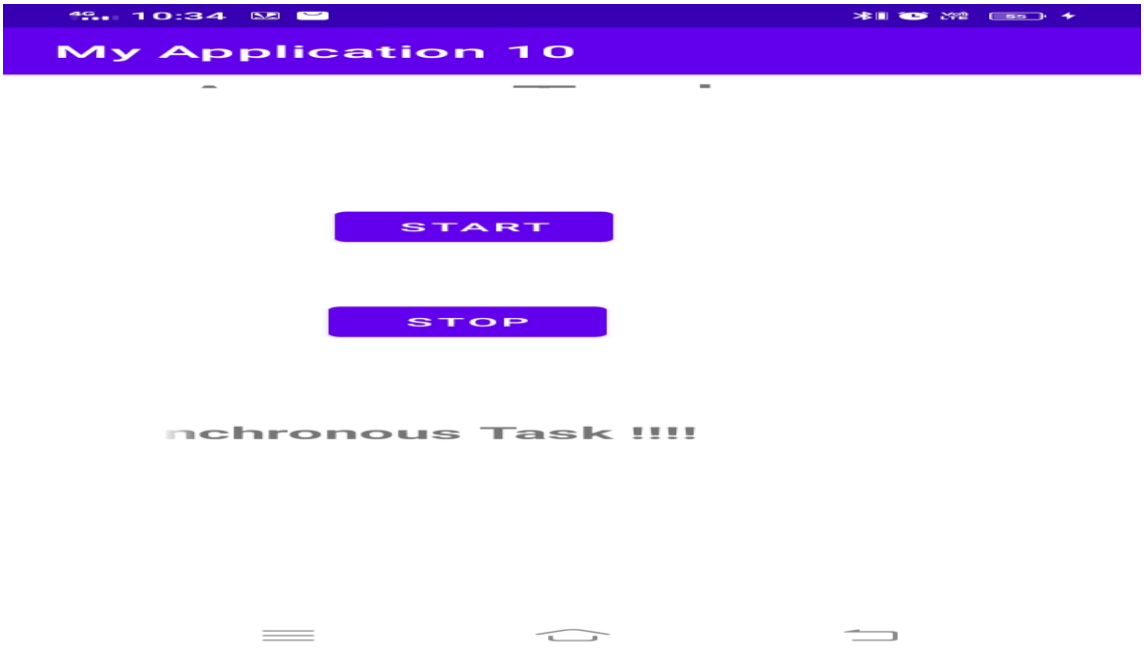
}

EXECUTION STEPS

Steps 1: Create a relative layout add two button to start and stop the movement of text and one marque text view for asynchronous movement of text present in the field as shown below.



Step 2: Write the java code to perform the operation the execute your application to see the output emulator



Program 11: (PART-B): 7. Develop an application that makes use of the clipboard framework for copying and pasting of the text. The activity consists of two EditText controls and two Buttons to trigger the copy and paste functionality.

XML 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=".MainActivity">
    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="152dp"
        android:layout_marginBottom="564dp"
        android:text="ClipBoard"
        android:textSize="36sp" />
    <EditText
        android:id="@+id/editText1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="127dp"
        android:layout_marginBottom="496dp"
        android:ems="10"
```

```
        android:hint="Enter the text here"
        android:inputType="textPersonName"
        android:text="" />
    <EditText
        android:id="@+id/editText2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="122dp"
        android:layout_marginBottom="411dp"
        android:ems="10"
        android:hint="Copied Text"
        android:inputType="textPersonName"
        android:text="" />
    <Button
        android:id="@+id/copy"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="279dp"
        android:onClick="copy"
        android:layout_marginBottom="312dp"
        android:text="Copy" />
    <Button
        android:id="@+id/paste"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_alignParentBottom="true"
```

```
        android:layout_marginEnd="92dp"
        android:onClick="paste"
        android:layout_marginBottom="313dp"
        android:text="Paste" />
</RelativeLayout>
```

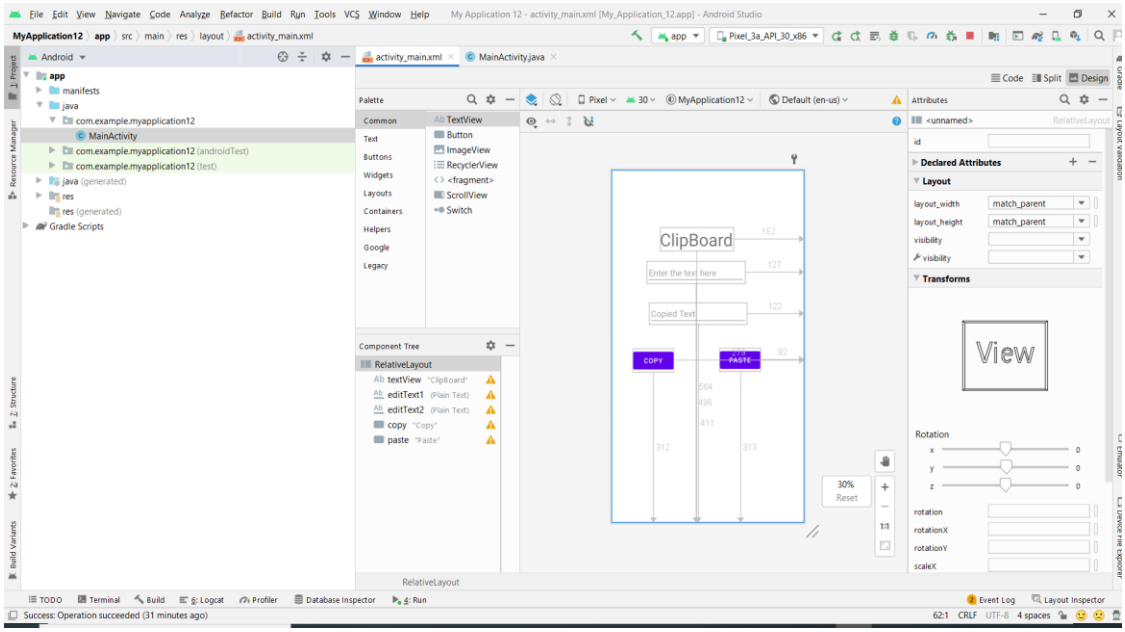
JAVA Code

```
package com.example.myapplication12;

import androidx.appcompat.app.AppCompatActivity;

import android.content.ClipData;
import android.content.ClipboardManager;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;

public class MainActivity extends AppCompatActivity {
    ClipboardManager cbm;
    ClipData cd;
    EditText e1,e2;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        e1 = (EditText)findViewById(R.id.editText1);
        e2 = (EditText)findViewById(R.id.editText2);
        cbm
=(ClipboardManager)getSystemService(CLIPBOARD_SERVICE);
    }
    public void copy(View V)
    {
        String text = e1.getText().toString();
```

```
        cd = ClipData.newPlainText("text",text);
        cbm.setPrimaryClip(cd);
    }
    public void paste(View V)
    {
        ClipData cd2 = cbm.getPrimaryClip();
        ClipData.Item item = cd2.getItemAt(0);
        String copied = item.getText().toString();
        e2.setText(copied);
    }
}
```

EXECUTION STEPS

Step 1: Create a relative layout and the add two edit text views one to get the input from the user and other to display the copied text respectively and add two button one to copy the text an d other to paste the text respectively.



3. VIVA QUESTIONS

1. **What is Android?**

   It is an open-sourced operating system that is used primarily on mobile devices, such as cell phones and tablets. It is a Linux kernel-based system that's been equipped with rich components that allows developers to create and run apps that can perform both basic and advanced functions.

2. **What Is the Google Android SDK?**

   The Google Android SDK is a toolset that developers need in order to write apps on Android enabled devices. It contains a graphical interface that emulates an Android driven handheld environment, allowing them to test and debug their codes.

3. **What is the Android Architecture?**

   Android Architecture is made up of 4 key components:

   - Linux Kernel
   - Libraries
   - Android Framework
   - Android Applications

4. **Describe the Android Framework.**

   The Android Framework is an important aspect of the Android Architecture. Here you can find all the classes and methods that developers would need in order to write applications on the Android environment.

5. **What is AAPT?**

   AAPT is short for Android Asset Packaging Tool. This tool provides developers with the ability to deal with zip-compatible archives, which includes creating, extracting as well as viewing its contents.

6. **What is the importance of having an emulator within the Android environment?**

   The emulator lets developers "play" around an interface that acts as if it were an actual mobile device. They can write and test codes, and even debug. Emulators are a safe place for testing codes especially if it is in the early design phase.

7. **What is the use of an activity Creator?**

   An activity Creator is the first step towards the creation of a new Android project. It is made up of a shell script that will be used to create new file system structure necessary for writing codes within the Android IDE.

8. **Describe Activities.**

Activities are what you refer to as the window to a user interface. Justas you create windows in order to display output or to ask for an input in the form of dialog boxes, activities play the same role, though it may not always be in the form of a user interface.

9. What are Intents?

Intents displays notification messages to the user from within the Android enabled device. It can be used to alert the user of a particular state that occurred. Users can be made to respond to intents.

10. Differentiate Activities from Services.

Activities can be closed, or terminated anytime the user wishes. On the other hand, services are designed to run behind the scenes, and can act independently. Most services run continuously, regardless of whether there are certain or no activities being executed.

11. What items are important in every Android project?

These are the essential items that are present each time an Android project is created:

- Android Manifest.xml
- build.xml
- bin/
- src/
- res/
- assets/

12. What is the importance of XML-based layouts?

The use of XML-based layouts provides a consistent and somewhat standard means of setting GUI definition format. In common practice, layout details are placed in XML files while other items are placed insource files.

13. What are containers?

Containers, as the name itself implies, holds objects and widgets together, depending on which specific items are needed and in what particular arrangement that is wanted. Containers may hold labels, fields, buttons, or even child containers, as examples.

14. What is Orientation?

Orientation, which can be set using setOrientation(), dictates if the Linear Layout is represented as a row or as a column. Values are set as either HORIZONTAL or VERTICAL.

15. What is the importance of Android in the mobile market?

Developers can write and register apps that will specifically run under the Android environment. This means that every mobile device that is Android enabled will be able

to support and run these apps. With the growing popularity of Android mobile devices, developers can take advantage of this trend by creating and uploading their apps on the Android Market for distribution to anyone who wants to download it.

16. What do you think are some disadvantages of Android?

Given that Android is an open-source platform, and the fact that different Android operating systems have been released on different mobile devices, there's no clear cut policy to how applications can adapt with various OS versions and upgrades. –> One app that runs on this particular version of Android OS may or may not run on another version. –> Another disadvantage is that since mobile devices such as phone sand tabs come in different sizes and forms, it poses a challenge for developers to create apps that can adjust correctly to the right screen size and other varying features and specs.

17. What is adb?

Adb is short for "Android Debug Bridge". It allows developers the power to execute remote shell commands. Its basic function is to allow and control communication towards and from the emulator port.

18. What are the four essential states of an activity?

- Active – if the activity is at the foreground
- Paused – if the activity is at the background and still visible
- Stopped – if the activity is not visible and therefore is hidden or obscured by another activity
- Destroyed – when the activity process is killed.

19. What is ANR?

ANR is short for Application Not Responding. This is actually a dialog that appears to the user whenever an application have been unresponsive for a long period of time.

20. How are escape characters used as attribute?

Escape characters are preceded by double backslashes. For example, a newline character is created using '\\n'

21. What is the importance of settings permissions in app development?

Permissions allow certain restrictions to be imposed primarily to protect data and code. Without these, codes could be compromised, resulting to defects in functionality.

22. What is the function of an intent filter?

Because every component needs to indicate which intents they can respond to, intent filters are used to filter out intents that these components are willing to receive. One or

more intent filters are possible, depending on the services and activities that is going to make use of it.

23. Enumerate the three key loops when monitoring an activity?

Entire lifetime – activity happens between on Create and on Destroy

Visible lifetime – activity happens between on Start and on Stop

Foreground lifetime – activity happens between on Resume and on Pause

24. When is the onStop() method invoked?

A call to on Stop method happens when an activity is no longer visible to the user, either because another activity has taken over or if in front

of that activity.

25. Is there a case wherein other qualifiers in multiple resources take precedence over locale?

Yes, there are actually instances wherein some qualifiers can take precedence over locale. There are two known exceptions, which are the MCC (mobile country code. and MNC (mobile network code. qualifiers.

26. What are the different states wherein a process is based?

There are 4 possible states:

- foreground activity
- visible activity
- background activity
- empty process

27. How can the ANR be prevented?

One technique that prevents the Android system from concluding a code that has been responsive for a long period of time is to create a child thread. Within the child thread, most of the actual workings of the codes can be placed, so that the main thread runs with minimal periods of unresponsive times.

28. What role does Dalvik play in Android development?

Dalvik serves as a virtual machine, and it is where every Android application runs. Through Dalvik, a device is able to execute multiple virtual machines efficiently through better memory management.

29. What is the Android Manifest.xml?

This file is essential in every application. It is declared in the root directory and contains information about the application that the Android system must know before the codes can be executed.

30. What is the proper way of setting up an Android-powered device for app development?

The following are steps to be followed prior to actual application development in an Android-powered device:

Declare your application as "debuggable" in your Android Manifest.

Turn on "USB Debugging" on your device.

Set up your system to detect your device.

31. Enumerate the steps in creating a bounded service through AIDL.

1. create the .aidl file, which defines the programming interface

2. implement the interface, which involves extending the inner abstract Stub class as well as implanting its methods.

3. expose the interface, which involves implementing the service to the clients.

32. What is the importance of Default Resources?

When default resources, which contain default strings and files, are not present, an error will occur and the app will not run. Resources are placed in specially named subdirectories under the project res/directory.

33. When dealing with multiple resources, which one takes precedence?

Assuming that all of these multiple resources are able to match the configuration of a device, the 'locale' qualifier almost always takes the highest precedence over the others.

34. When does ANR occur?

The ANR dialog is displayed to the user based on two possible conditions. One is when there is no response to an input event within5 seconds, and the other is when a broadcast receiver is not done executing within 10 seconds.

35. What is AIDL?

AIDL, or Android Interface Definition Language, handles the interface requirements between a client and a service so both can communicate at the same level through inter process communication or IPC. This process involves breaking down objects into primitives that Android can understand. This part is required simply because a process cannot access the memory of the other process.

36. What data types are supported by AIDL?

AIDL has support for the following data types:

string

charSequence

List

Map

all native Java data types like int,long, char and Boolean

37. What is a Fragment?

A fragment is a part or portion of an activity. It is modular in a sense that you can move around or combine with other fragments in a single activity. Fragments are also reusable.

38. What is a visible activity?

A visible activity is one that sits behind a foreground dialog. It is actually visible to the user, but not necessarily being in the foreground itself.

39. When is the best time to kill a foreground activity?

The foreground activity, being the most important among the other states, is only killed or terminated as a last resort, especially if it is already consuming too much memory. When a memory paging state has been reach by a foreground activity, then it is killed so that the user interface can retain its responsiveness to the user.

40. Is it possible to use or add a fragment without using a user interface?

Yes, it is possible to do that, such as when you want to create a background behaviour for a particular activity. You can do this by using addFragment() method to add a fragment from the activity.

41. How do you remove icons and widgets from the main screen of the Android device?

To remove an icon or shortcut, press and hold that icon. You then drag it downwards to the lower part of the screen where a remove button appears.

42. What are the core components under the Android application architecture?

There are 5 key components under the Android application architecture:

- services
- intent
- resource externalization
- notifications
- content providers

43. What composes a typical Android application project?

A project under Android development, upon compilation, becomes an .apk file. This .apk file format is actually made up of theAndroidManifest.xml file, application code, resource files, and other related files.

44. What is a Sticky Intent?

A Sticky Intent is a broadcast from sendStickyBroadcast() method such that the intent floats around even after the broadcast, allowing others to collect data from it.

45. Do all mobile phones support the latest Android operating system?

Some Android-powered phone allows you to upgrade to the higher Android operating system version. However, not all upgrades would allow you to get the latest version. It depends largely on the capability and specs of the phone, whether it can support the newer features available under the latest Android version.

46. What is portable wi-fi hotspot?

Portable Wi-Fi Hotspot allows you to share your mobile internet connection to other wireless device. For example, using your Android-powered phone as a Wi-Fi Hotspot, you can use your laptop to connect to the Internet using that access point.

47. What is an action?

In Android development, an action is what the intent sender wants to do or expected to get as a response. Most application functionality is based on the intended action.

48. What is the difference between a regular bitmap and a nine-patch image?

In general, a Nine-patch image allows resizing that can be used as background or other image size requirements for the target device. The Nine-patch refers to the way you can resize the image: 4 corners that are unscaled, 4 edges that are scaled in 1 axis, and the middle one that can be scaled into both axes.

49. What language is supported by Android for application development?

The main language supported is Java programming language. Java is the most popular language for app development, which makes it ideal even for new Android developers to quickly learn to create and deploy applications in the Android environment.