

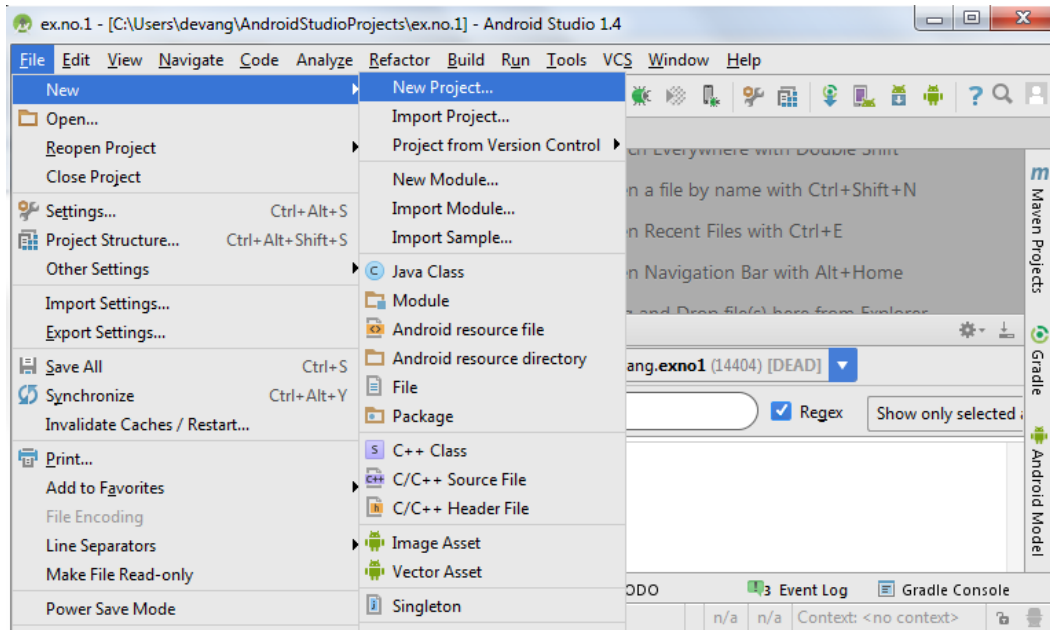
Date:

Aim:

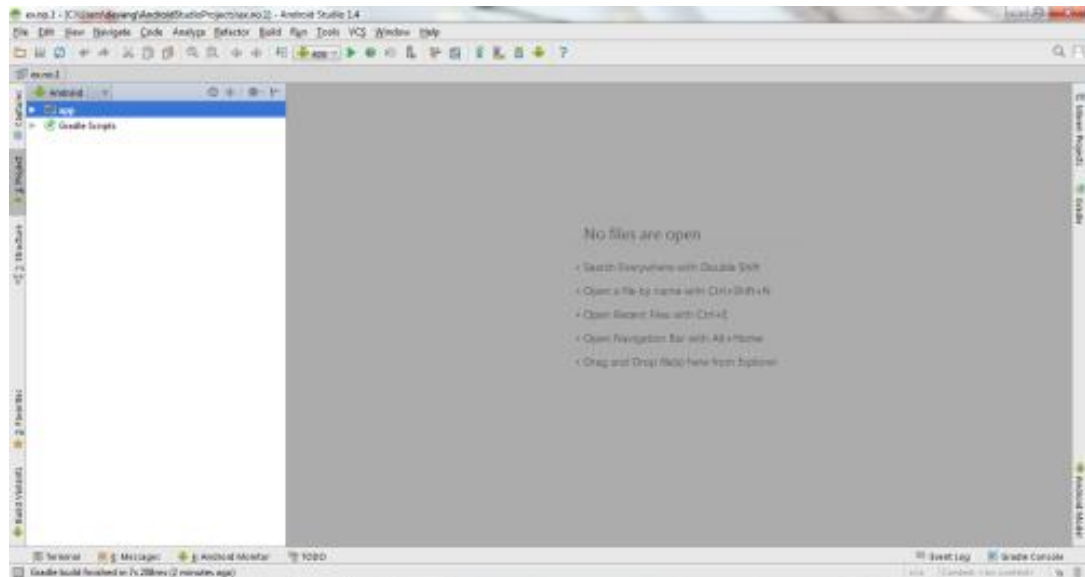
To develop a Simple Android Application that uses GUI components, Font and Colors.

Procedure:**Creating a New project:**

- Open Android Studio and then click on **File -> New -> New project.**



- Then type the Application name as **"exno1"** and click Next.
- Then **select the Minimum SDK** as shown below and click Next.
- Then select the **Empty Activity** and click Next.
- Finally click **Finish**.
- It will take some time to build and load the project.
- After completion it will look as given below.



Designing layout for the Android Application:

- Click on **app** -> **res** -> **layout** -> **activity_main.xml**.
- Now click on Text as shown below.
- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

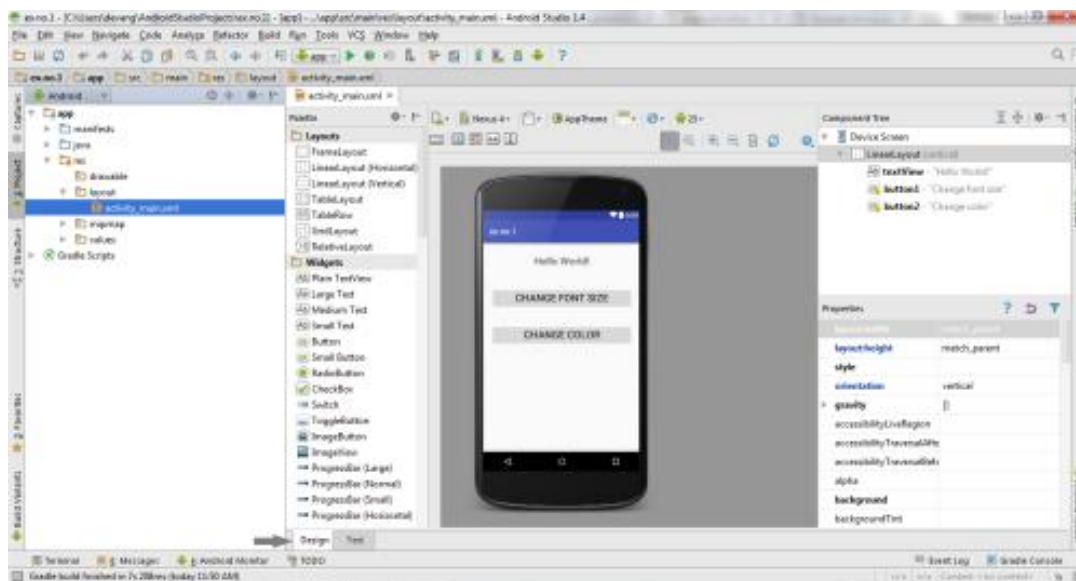
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
<TextView
    android:id="@+id/textView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="30dp"
    android:gravity="center"
    android:text="Hello World!"
    android:textSize="25sp"
    android:textStyle="bold" />
<Button
    android:id="@+id/button1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="20dp"
    android:gravity="center"
    android:text="Change font size"
    android:textSize="25sp" />
```

<Button

```
android:id="@+id/button2"  
android:layout_width="match_parent"  
android:layout_height="wrap_content"  
android:layout_margin="20dp"  
android:gravity="center"  
android:text="Change color"  
android:textSize="25sp" />
```

</LinearLayout>

- Now click on Design and your application will look as given below.



- So now the designing part is completed.

Java Coding for the Android Application:

- Click on **app -> java -> com.example.exno1 -> MainActivity**.
- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

```
package com.example.exno1;  
import android.graphics.Color;  
//import android.support.v7.app.AppCompatActivity;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.TextView;  
import androidx.appcompat.app.AppCompatActivity;  
public class MainActivity extends AppCompatActivity  
{  
    int ch=1;  
    float font=30;
```

```

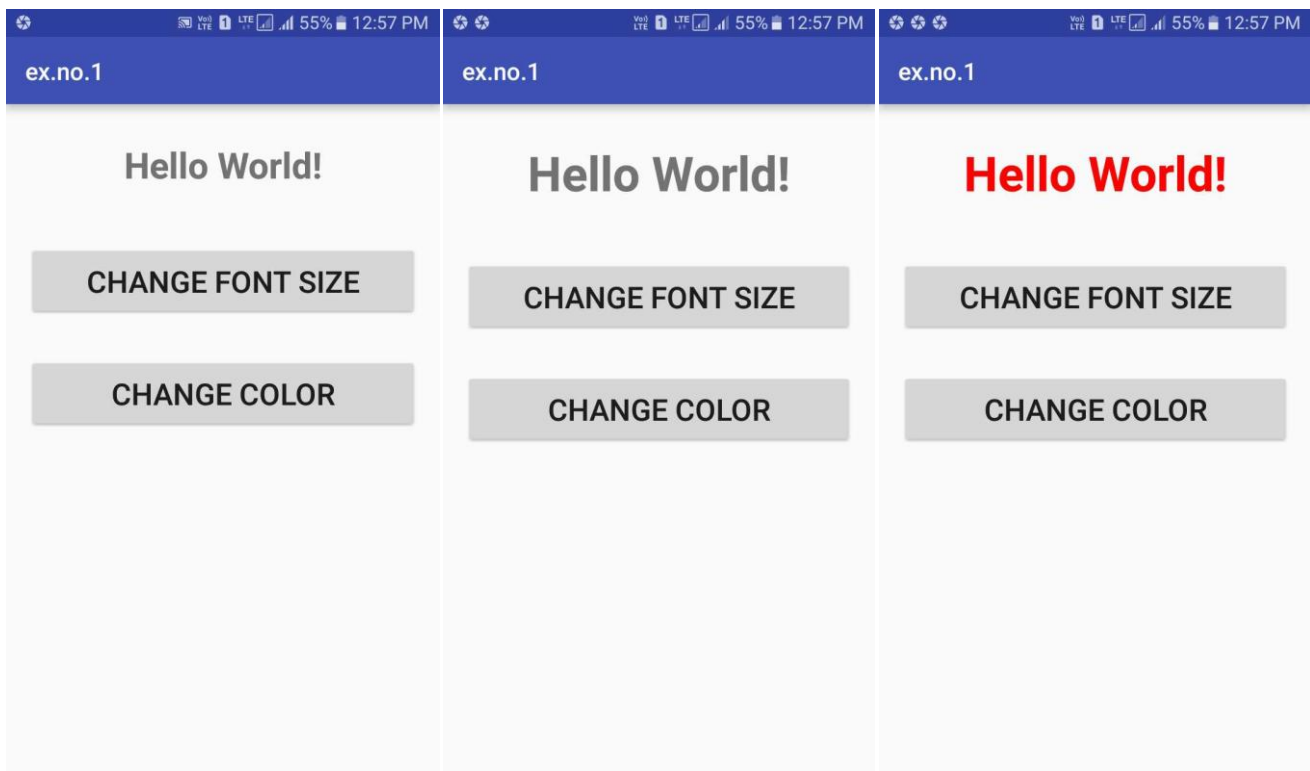
@Override
protected void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    final TextView t= (TextView) findViewById(R.id.textView);
    Button b1= (Button) findViewById(R.id.button1);
    b1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            t.setTextSize(font);
            font = font + 5;
            if (font == 50)
                font = 30;
        }
    });
    Button b2= (Button) findViewById(R.id.button2);
    b2.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            switch (ch) {
                case 1:
                    t.setTextColor(Color.RED);
                    break;
                case 2:
                    t.setTextColor(Color.GREEN);
                    break;
                case 3:
                    t.setTextColor(Color.BLUE);
                    break;
                case 4:
                    t.setTextColor(Color.CYAN);
                    break;
                case 5:
                    t.setTextColor(Color.YELLOW);
                    break;
                case 6:
                    t.setTextColor(Color.MAGENTA);
                    break;
            }
            ch++;
            if (ch == 7)

```

```
        ch = 1;
    }
    });
}
}
```

- So now the Coding part is also completed.
- Now run the application to see the output.

Output:



Result:

Thus a Simple Android Application that uses GUI components, Font and Colors is developed and executed successfully.

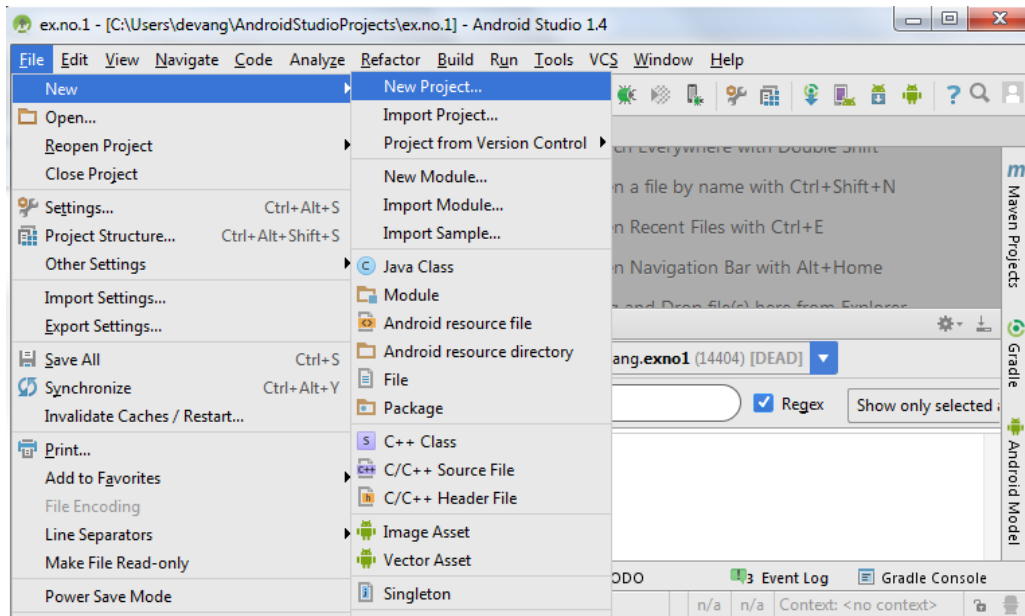
Date:

Aim:

To develop a Simple Android Application that uses Layout Managers and Event Listeners.

Procedure:**Creating a New project:**

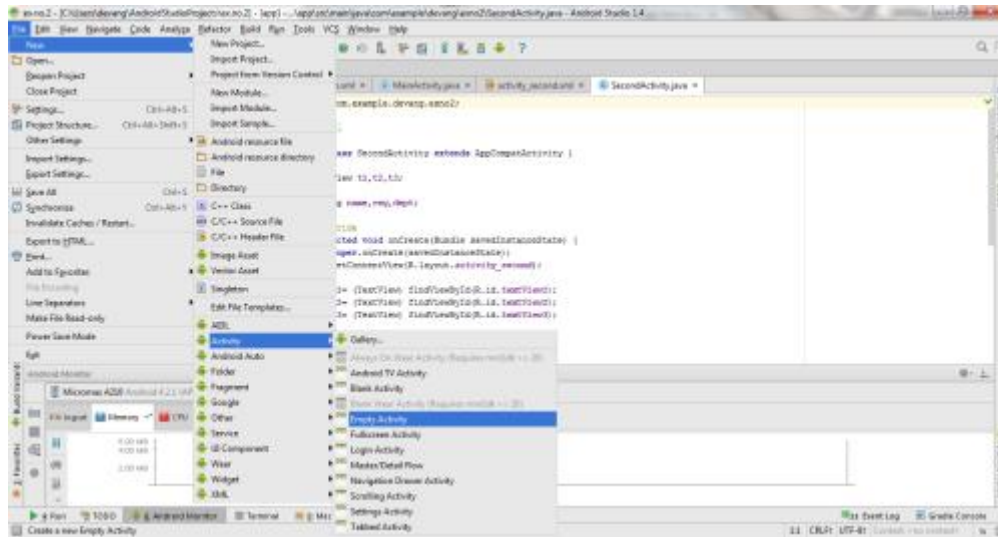
- Open Android Studio and then click on **File -> New -> New project**.



- Then type the Application name as **“exno2”** and click Next.
- Then **select the Minimum SDK** as shown below and click Next.
- Then **select the Empty Activity** and click Next.
- Finally click **Finish**.
- It will take some time to build and load the project.
- After completion it will look as given below.

Creating Second Activity for the Android Application:

- Click on **File -> New -> Activity -> Empty Activity**.



- Type the Activity Name as **SecondActivity** and click Finish button.
- Thus Second Activity For the application is created.

Designing Layout for Main Activity:

- Click on **app -> res -> layout -> activity_main.xml**.
- Now click on Text as shown below.
- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

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

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="100dp">

        <TextView
            android:id="@+id/textView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_margin="30dp"
            android:text="Details Form"
            android:textSize="25sp"
            android:gravity="center"/>

    </LinearLayout>
```

```
<GridLayout
    android:id="@+id/gridLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginTop="100dp"
    android:layout_marginBottom="200dp"
    android:columnCount="2"
    android:rowCount="3">
<TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:layout_row="0"
    android:layout_column="0"
    android:text="Name"
    android:textSize="20sp"
    android:gravity="center"/>

<EditText
    android:id="@+id/editText"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:layout_row="0"
    android:layout_column="1"
    android:ems="10"/>

<TextView
    android:id="@+id/textView2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:layout_row="1"
    android:layout_column="0"
    android:text="Reg.No"
    android:textSize="20sp"
    android:gravity="center"/>
```



```
<EditText
    android:id="@+id/editText2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:layout_row="1"
    android:layout_column="1"
    android:inputType="number"
    android:ems="10"/>
```

```
<TextView
    android:id="@+id/textView3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:layout_row="2"
    android:layout_column="0"
    android:text="Dept"
    android:textSize="20sp"
    android:gravity="center"/>
```

```
<Spinner
    android:id="@+id/spinner"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:layout_row="2"
    android:layout_column="1"
    android:spinnerMode="dropdown"/>
```

```
</GridLayout>
```

```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:layout_centerInParent="true"
    android:layout_marginBottom="150dp"
    android:text="Submit"/>
```

```
</RelativeLayout>
```

Designing Layout for Second Activity:

- Click on app -> res -> layout -> activity_second.xml.
- Now click on Text as shown below.
- Then delete the code which is there and type the code as given below.

Code for Activity_second.xml:

```
<?xmlversion="1.0"encoding="utf-8"?>
<LinearLayoutxmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.devang.exno2.SecondActivity"
    android:orientation="vertical"
    android:gravity="center">

    <TextView
        android:id="@+id/textView1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="20dp"
        android:text="New Text"
        android:textSize="30sp"/>

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="20dp"
        android:text="New Text"
        android:textSize="30sp"/>

    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="20dp"
        android:text="New Text"
        android:textSize="30sp"/>
</LinearLayout>
```

- Now click on Design and your activity will look as given below.
- So now the designing part of Second Activity is also completed.

Java Coding for the Android Application:

- Java Coding for Main Activity:
- Click on **app -> java -> com.example.exno2 -> MainActivity**.
- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

```
package com.example.exno2;

import android.content.Intent;
//import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    //Defining the Views
    EditText e1,e2;
    Button bt;
    Spinner s;

    //Data for populating in Spinner
    String [] dept_array={"CSE","ECE","IT","Mech","Civil"};

    String name,reg,dept;

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

        //Referring the Views
        e1= (EditText) findViewById(R.id.editText);
```

```

e2= (EditText) findViewById(R.id.editText2);

bt= (Button) findViewById(R.id.button);

s= (Spinner) findViewById(R.id.spinner);

//Creating Adapter for Spinner for adapting the data from array to Spinner
ArrayAdapter<String> adapter= new
ArrayAdapter(MainActivity.this,android.R.layout.simple_spinner_item,dept_array);
s.setAdapter(adapter);

//Creating Listener for Button
bt.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

        //Getting the Values from Views(Edittext & Spinner)
        name=e1.getText().toString();
        reg=e2.getText().toString();
        dept=s.getSelectedItem().toString();

        //Intent For Navigating to Second Activity
        Intent i = new Intent(MainActivity.this,SecondActivity.class);

        //For Passing the Values to Second Activity
        i.putExtra("name_key", name);
        i.putExtra("reg_key",reg);
        i.putExtra("dept_key", dept);

        startActivity(i);
    }
});
}
}

```

Java Coding for Second Activity:

- Click on **app -> java -> com.example.exno2 -> SecondActivity**.
- Then delete the code which is there and type the code as given below.

Code for SecondActivity.java:

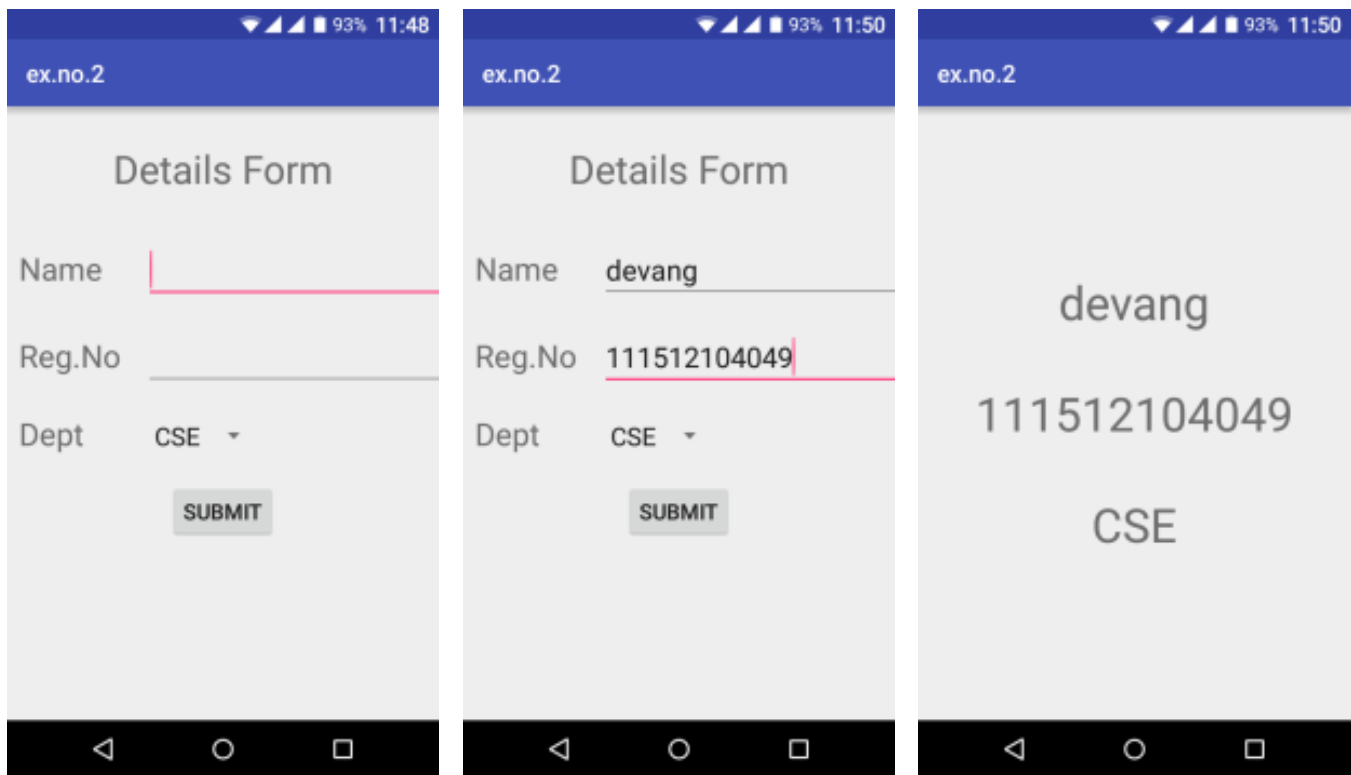
```
package com.example.exno2;

import android.content.Intent;
//import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;

public class SecondActivity extends AppCompatActivity {
    TextView t1,t2,t3;
    String name,reg,dept;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);
        t1= (TextView) findViewById(R.id.textView1);
        t2= (TextView) findViewById(R.id.textView2);
        t3= (TextView) findViewById(R.id.textView3);
        //Getting the Intent
        Intent i = getIntent();
        //Getting the Values from First Activity using the Intent received
        name=i.getStringExtra("name_key");
        reg=i.getStringExtra("reg_key");
        dept=i.getStringExtra("dept_key");
        //Setting the Values to Intent
        t1.setText(name);
        t2.setText(reg);
        t3.setText(dept);
    }
}
```

- So now the Coding part of Second Activity is also completed.
- Now run the application to see the output.

Output:



Result:

Thus a Simple Android Application that uses Layout Managers and Event Listeners is developed and executed successfully.

Date:

Aim:

To develop a Simple Android Application that draws basic Graphical Primitives on the screen.

Procedure:**Creating a New project:**

- Open Android Studio and then click on **File -> New -> New project**.
- Then type the Application name as **"exno3"** and click Next.
- Then **select the Minimum SDK** as shown below and click Next.
- Then **select the Empty Activity** and click Next.
- Finally click **Finish**.
- It will take some time to build and load the project.
- After completion it will look as given below.

Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml**.
- Now click on Text as shown below.
- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

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

<ImageView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/imageView" />
</RelativeLayout>
```

- Now click on Design and your application will look as given below.
- So now the designing part is completed.

Java Coding for the Android Application:

- Click on **app -> java -> com.example.exno3 -> MainActivity**.
- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

```
package com.example.exno3;

import android.app.Activity;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.drawable.BitmapDrawable;
import android.os.Bundle;
import android.widget.ImageView;

public class MainActivity extends Activity
{
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        //Creating a Bitmap
        Bitmap bg = Bitmap.createBitmap(720, 1280, Bitmap.Config.ARGB_8888);

        //Setting the Bitmap as background for the ImageView
        ImageView i = (ImageView) findViewById(R.id.imageView);
        i.setBackgroundDrawable(new BitmapDrawable(bg));

        //Creating the Canvas Object
        Canvas canvas = new Canvas(bg);

        //Creating the Paint Object and set its color & TextSize
        Paint paint = new Paint();
        paint.setColor(Color.BLUE);
        paint.setTextSize(50);

        //To draw a Rectangle
        canvas.drawText("Rectangle", 420, 150, paint);
        canvas.drawRect(400, 200, 650, 700, paint);

        //To draw a Circle
        canvas.drawText("Circle", 120, 150, paint);
        canvas.drawCircle(200, 350, 150, paint);
    }
}
```



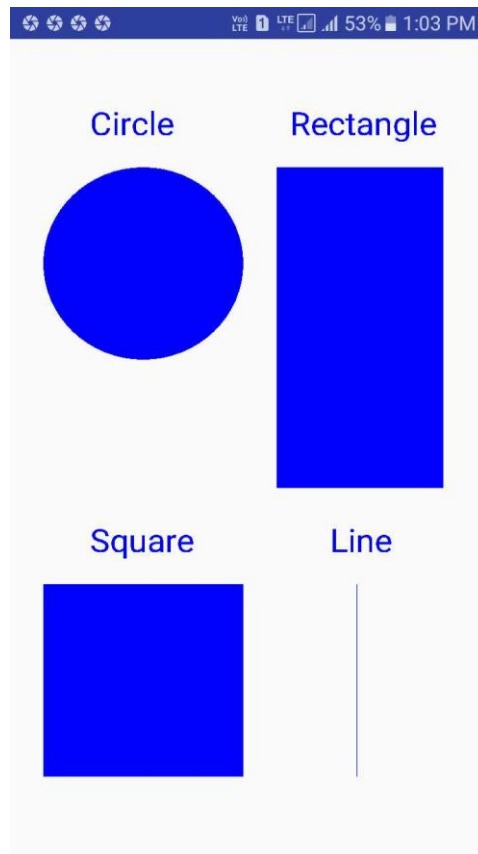
```
//To draw a Square  
canvas.drawText("Square", 120, 800, paint);  
canvas.drawRect(50, 850, 350, 1150, paint);
```

```
//To draw a Line  
canvas.drawText("Line", 480, 800, paint);  
canvas.drawLine(520, 850, 520, 1150, paint);
```

```
}  
}
```

- So now the Coding part is also completed.
- Now run the application to see the output.

Output:



Result:

Thus a Simple Android Application that draws basic Graphical Primitives on the screen is developed and executed successfully.

Date:

Aim:

To develop a Simple Android Application that makes use of Database.

Procedure:**Creating a New project:**

- Open Android Studio and then click on **File -> New -> New project**.
- Then type the Application name as **"exno4"** and click Next.
- Then **select the Minimum SDK** as shown below and click Next.
- Then **select the Empty Activity** and click Next.
- Finally click **Finish**.
- It will take some time to build and load the project.
- After completion it will look as given below.

Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml**.
- Now click on Text as shown below.
- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<AbsoluteLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="50dp"
    android:layout_y="20dp"
    android:text="Student Details"
    android:textSize="30sp" />

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="20dp"
    android:layout_y="110dp"
    android:text="Enter Rollno:"
    android:textSize="20sp" />
```

```
<EditText
    android:id="@+id/Rollno"
    android:layout_width="150dp"
    android:layout_height="wrap_content"
    android:layout_x="175dp"
    android:layout_y="100dp"
    android:inputType="number"
    android:textSize="20sp" />
```

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="20dp"
    android:layout_y="160dp"
    android:text="Enter Name:"
    android:textSize="20sp" />
```

```
<EditText
    android:id="@+id/Name"
    android:layout_width="150dp"
    android:layout_height="wrap_content"
    android:layout_x="175dp"
    android:layout_y="150dp"
    android:inputType="text"
    android:textSize="20sp" />
```

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_x="20dp"
    android:layout_y="210dp"
    android:text="Enter Marks:"
    android:textSize="20sp" />
```

```
<EditText
    android:id="@+id/Marks"
    android:layout_width="150dp"
    android:layout_height="wrap_content"
```

```
android:layout_x="175dp"  
android:layout_y="200dp"  
android:inputType="number"  
android:textSize="20sp" />
```

<Button

```
android:id="@+id/Insert"  
android:layout_width="150dp"  
android:layout_height="wrap_content"  
android:layout_x="25dp"  
android:layout_y="300dp"  
android:text="Insert"  
android:textSize="30dp" />
```

<Button

```
android:id="@+id/Delete"  
android:layout_width="150dp"  
android:layout_height="wrap_content"  
android:layout_x="200dp"  
android:layout_y="300dp"  
android:text="Delete"  
android:textSize="30dp" />
```

<Button

```
android:id="@+id/Update"  
android:layout_width="150dp"  
android:layout_height="wrap_content"  
android:layout_x="25dp"  
android:layout_y="400dp"  
android:text="Update"  
android:textSize="30dp" />
```

<Button

```
android:id="@+id/View"  
android:layout_width="150dp"  
android:layout_height="wrap_content"  
android:layout_x="200dp"  
android:layout_y="400dp"  
android:text="View"
```

```
android:textSize="30dp" />
```

```
<Button
```

```
    android:id="@+id/ViewAll"  
    android:layout_width="200dp"  
    android:layout_height="wrap_content"  
    android:layout_x="100dp"  
    android:layout_y="500dp"  
    android:text="View All"  
    android:textSize="30dp" />
```

```
</AbsoluteLayout>
```

- Now click on Design and your application will look as given below.
- So now the designing part is completed.

Java Coding for the Android Application:

- Click on **app -> java -> com.example.exno4 -> MainActivity**.
- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

```
package com.example.exno4;  
import android.app.Activity;  
import android.app.AlertDialog.Builder;  
import android.content.Context;  
import android.database.Cursor;  
import android.database.sqlite.SQLiteDatabase;  
import android.os.Bundle;  
import android.view.View;  
import android.view.View.OnClickListener;  
import android.widget.Button;  
import android.widget.EditText;  
  
public class MainActivity extends Activity implements OnClickListener  
{  
    EditText Rollno, Name, Marks;  
    Button Insert, Delete, Update, View, ViewAll;  
    SQLiteDatabase db;  
    /** Called when the activity is first created. */  
    @Override  
    public void onCreate(Bundle savedInstanceState)  
    {  
        super.onCreate(savedInstanceState);
```

```

setContentView(R.layout.activity_main);

Rollno=(EditText)findViewById(R.id.Rollno);
Name=(EditText)findViewById(R.id.Name);
Marks=(EditText)findViewById(R.id.Marks);
Insert=(Button)findViewById(R.id.Insert);
Delete=(Button)findViewById(R.id.Delete);
Update=(Button)findViewById(R.id.Update);
View=(Button)findViewById(R.id.View);
ViewAll=(Button)findViewById(R.id.ViewAll);

Insert.setOnClickListener(this);
Delete.setOnClickListener(this);
Update.setOnClickListener(this);
View.setOnClickListener(this);
ViewAll.setOnClickListener(this);

// Creating database and table
db=openOrCreateDatabase("StudentDB", Context.MODE_PRIVATE, null);
db.execSQL("CREATE TABLE IF NOT EXISTS student(rollno VARCHAR,name VARCHAR,marks
VARCHAR);");
}
public void onClick(View view)
{
// Inserting a record to the Student table
if(view==Insert)
{
// Checking for empty fields
if(Rollno.getText().toString().trim().length()==0||
Name.getText().toString().trim().length()==0||
Marks.getText().toString().trim().length()==0)
{
showMessage("Error", "Please enter all values");
return;
}
db.execSQL("INSERT INTO student VALUES('"+Rollno.getText()+"','"+Name.getText()+"
','"+Marks.getText()+"');");
showMessage("Success", "Record added");
clearText();
}
// Deleting a record from the Student table
if(view==Delete)

```

```

{
// Checking for empty roll number
if(Rollno.getText().toString().trim().length()==0)
{
    showMessage("Error", "Please enter Rollno");
    return;
}
Cursor c=db.rawQuery("SELECT * FROM student WHERE rollno='"+Rollno.getText()+"'", null);
if(c.moveToFirst())
{
    db.execSQL("DELETE FROM student WHERE rollno='"+Rollno.getText()+"'");
    showMessage("Success", "Record Deleted");
}
else
{
    showMessage("Error", "Invalid Rollno");
}
clearText();
}
// Updating a record in the Student table
if(view==Update)
{
// Checking for empty roll number
if(Rollno.getText().toString().trim().length()==0)
{
    showMessage("Error", "Please enter Rollno");
    return;
}
Cursor c=db.rawQuery("SELECT * FROM student WHERE rollno='"+Rollno.getText()+"'", null);
if(c.moveToFirst()) {
    db.execSQL("UPDATE student SET name='"+ Name.getText() + "',marks='"+ Marks.getText() +
        "' WHERE rollno='"+Rollno.getText()+"'");
    showMessage("Success", "Record Modified");
}
else {
    showMessage("Error", "Invalid Rollno");
}
clearText();
}
// Display a record from the Student table
if(view==View)
{

```

```

// Checking for empty roll number
if(Rollno.getText().toString().trim().length()==0)
{
    showMessage("Error", "Please enter Rollno");
    return;
}
Cursor c=db.rawQuery("SELECT * FROM student WHERE rollno='"+Rollno.getText()+"'", null);
if(c.moveToFirst())
{
    Name.setText(c.getString(1));
    Marks.setText(c.getString(2));
}
else
{
    showMessage("Error", "Invalid Rollno");
    clearText();
}
}
// Displaying all the records
if(view==ViewAll)
{
    Cursor c=db.rawQuery("SELECT * FROM student", null);
    if(c.getCount()==0)
    {
        showMessage("Error", "No records found");
        return;
    }
    StringBuffer buffer=new StringBuffer();
    while(c.moveToNext())
    {
        buffer.append("Rollno: "+c.getString(0)+"\n");
        buffer.append("Name: "+c.getString(1)+"\n");
        buffer.append("Marks: "+c.getString(2)+"\n\n");
    }
    showMessage("Student Details", buffer.toString());
}
}
public void showMessage(String title,String message)
{
    Builder builder=new Builder(this);
    builder.setCancelable(true);
    builder.setTitle(title);
}

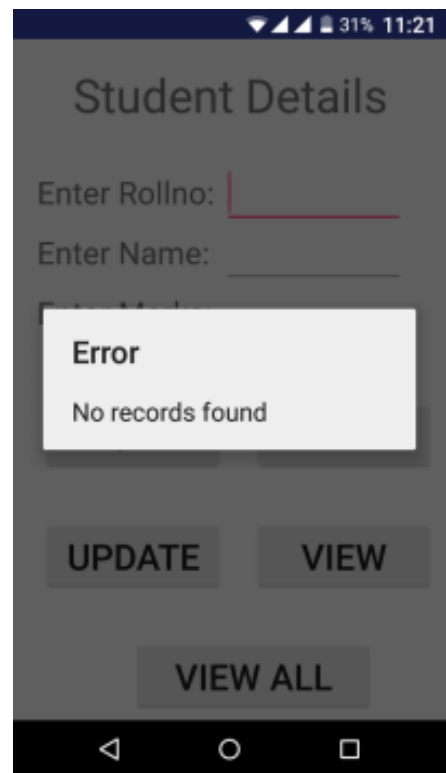
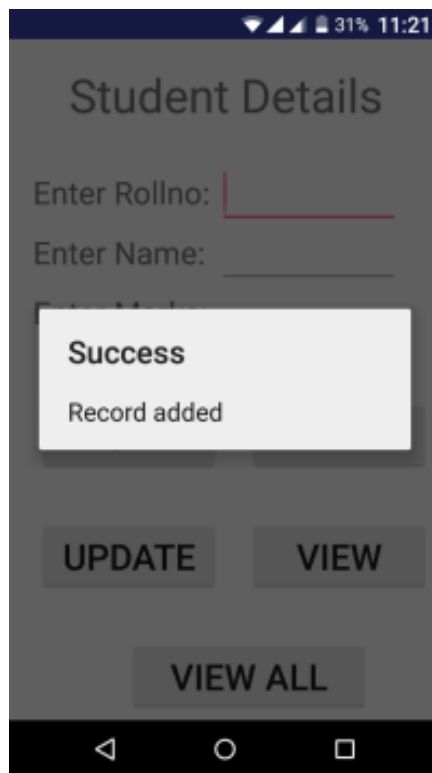
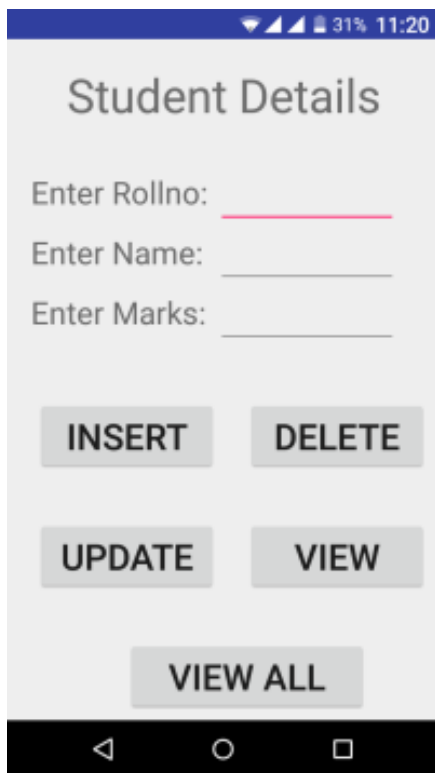
```

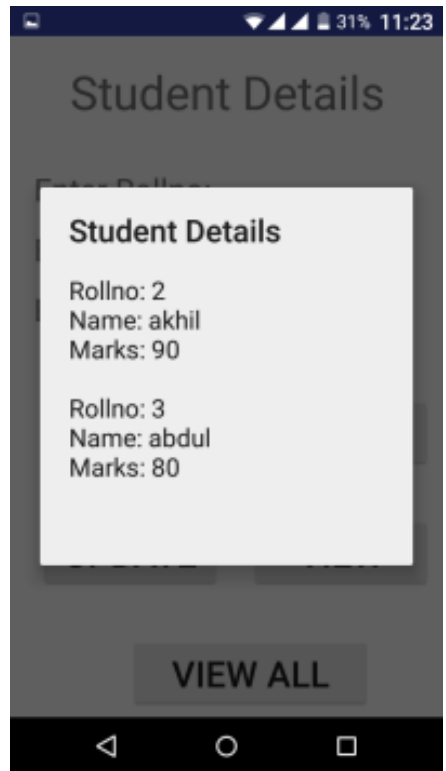
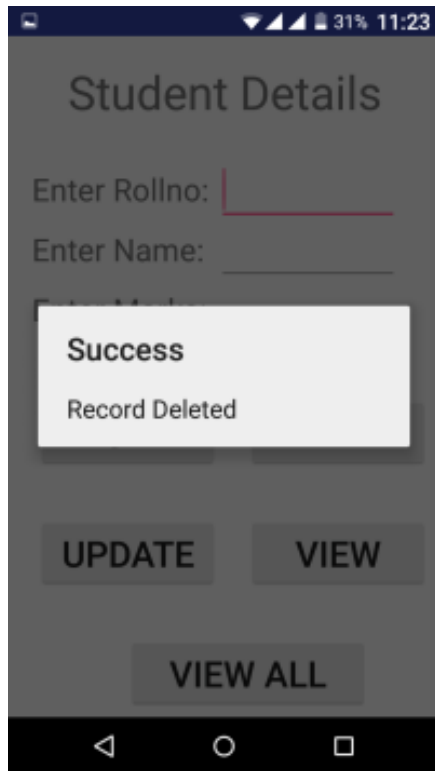
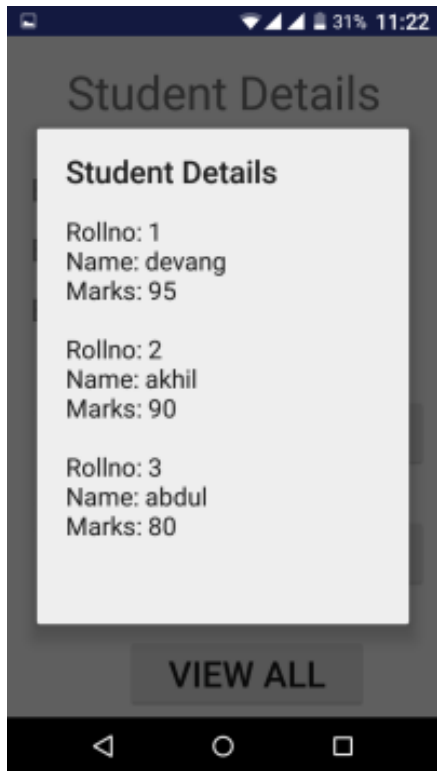
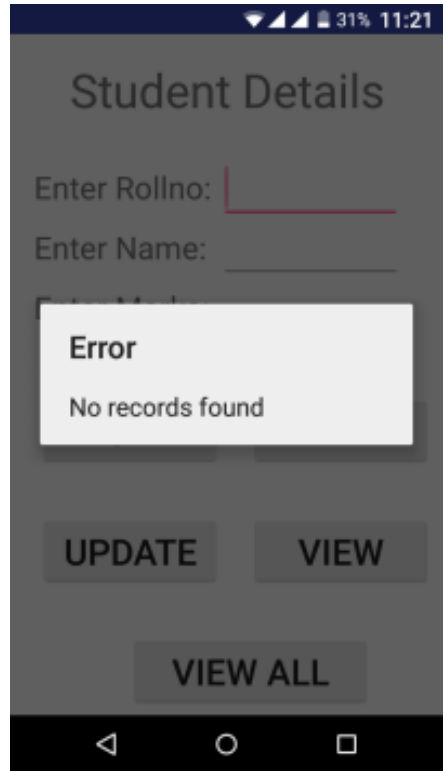
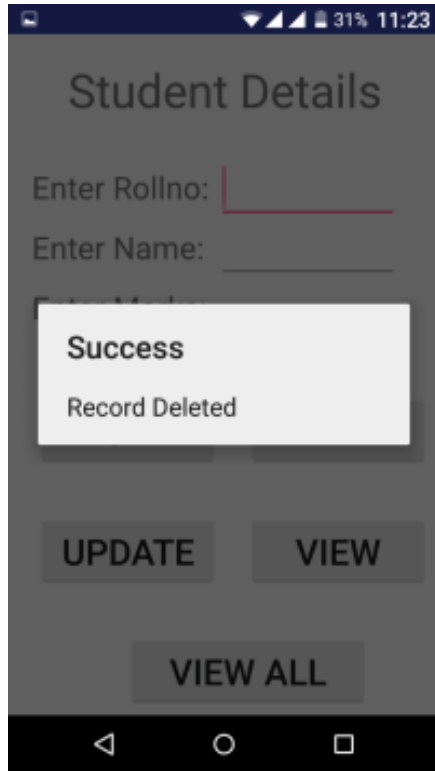
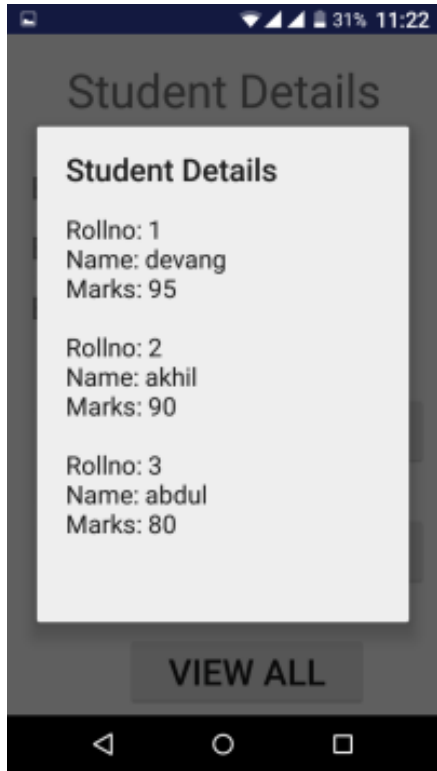


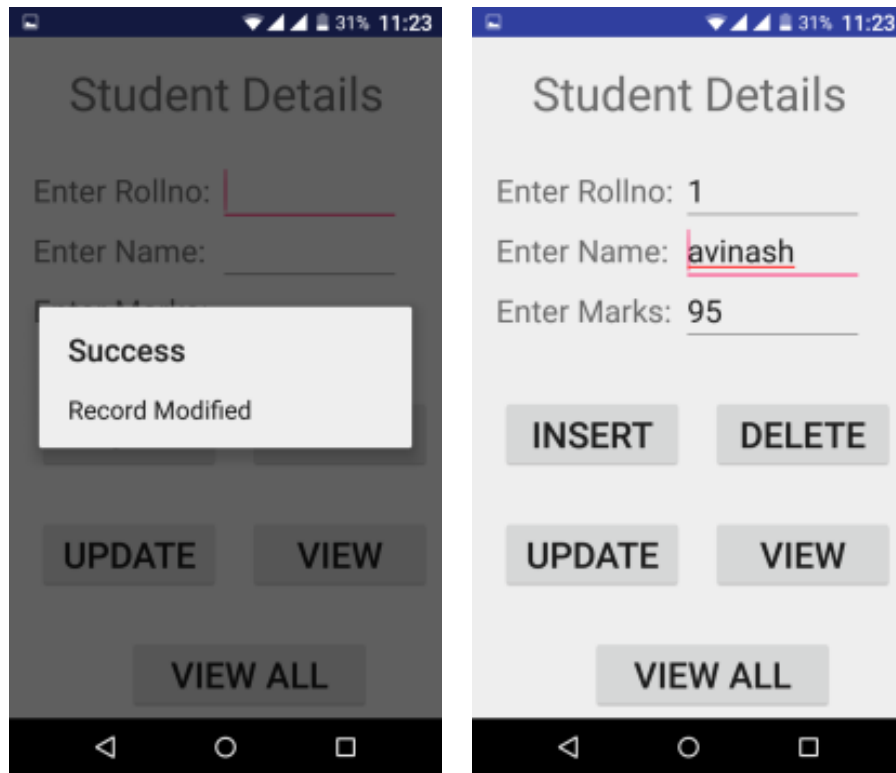
```
builder.setMessage(message);
builder.show();
}
public void clearText()
{
    Rollno.setText("");
    Name.setText("");
    Marks.setText("");
    Rollno.requestFocus();
}
}
```

- So now the Coding part is also completed.
- Now run the application to see the output.

Output:







Result:

Thus a Simple Android Application that makes use of Database is developed and executed successfully.

Date:

Aim:

To develop an Android Application that makes use of Notification Manager.

Procedure:**Creating a New project:**

- Open Android Studio and then click on **File -> New -> New project**.
- Then type the Application name as **"exno5"** and click Next.
- Then **select the Minimum SDK** as shown below and click Next.
- Then **select the Empty Activity** and click Next.
- Finally click **Finish**.
- It will take some time to build and load the project.
- After completion it will look as given below.

Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml**.
- Now click on Text as shown below.
- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

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

    <Button
        android:id="@+id/btnSimpleNotification"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Simple Notification" />

    <Button
        android:id="@+id/btnNotificationIcon"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
android:text="Notification With Icon" />
```

```
<Button  
    android:id="@+id/btnNotificationImage"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Notification With Image" />
```

```
<Button  
    android:id="@+id/btnNotificationWithGroupConvo"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Notification With Group Conversation" />
```

```
<Button  
    android:id="@+id/btnNotificationSemantic"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:text="Notification Semantic Action" />
```

```
</LinearLayout>
```

Java Coding for the Android Application:

- Click on **app -> java -> com.example.exno5 -> MainActivity**.
- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

```
package com.example.exno5;  
  
import android.app.NotificationChannel;  
import android.app.NotificationManager;  
import android.app.PendingIntent;  
import android.content.Context;  
import android.content.Intent;  
import android.net.Uri;  
import androidx.core.app.NotificationCompat;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.core.app.Person;  
import androidx.core.graphics.drawable.IconCompat;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;
```

```

import android.widget.Toast;
import java.util.Date;
public class MainActivity extends AppCompatActivity implements View.OnClickListener {

    NotificationManager notificationManager;
    NotificationCompat.Builder builder;
    NotificationChannel channel;

    CharSequence charSequence = "";

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

        Button btnSimpleNotification = findViewById(R.id.btnSimpleNotification);
        Button btnNotificationIcon = findViewById(R.id.btnNotificationIcon);
        Button btnNotificationImage = findViewById(R.id.btnNotificationImage);
        Button btnNotificationWithGroupConvo = findViewById(R.id.btnNotificationWithGroupConvo);
        Button btnNotificationSemantic = findViewById(R.id.btnNotificationSemantic);

        charSequence = btnNotificationIcon.getText();

        btnSimpleNotification.setOnClickListener(this);
        btnNotificationIcon.setOnClickListener(this);
        btnNotificationImage.setOnClickListener(this);
        btnNotificationWithGroupConvo.setOnClickListener(this);
        btnNotificationSemantic.setOnClickListener(this);

        notificationManager = (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);
        CharSequence name = "My Notification";
        String description = "yadda yadda";
        int importance = NotificationManager.IMPORTANCE_DEFAULT;

        channel = new NotificationChannel("1", name, importance);
        channel.setDescription(description);

        builder = new NotificationCompat.Builder(MainActivity.this, channel.getId())
                .setSmallIcon(R.mipmap.ic_launcher);

        notificationManager.createNotificationChannel(channel);
    }
}

```

```

}

@Override
public void onClick(View v) {

    switch (v.getId()) {
        case R.id.btnSimpleNotification:
            simpleNotification();
            break;
        case R.id.btnNotificationIcon:
            notificationWithIcon();
            break;
        case R.id.btnNotificationImage:
            notificationWithImage();
            break;

        case R.id.btnNotificationWithGroupConvo:
            notificationWithGroupConvo();
            break;

        case R.id.btnNotificationSemantic:
            notificationSemantic();
            break;
    }
}

private void simpleNotification() {
    Person jd = new Person.Builder().setName("JournalDev ").setImportant(true).build();

    new NotificationCompat.MessagingStyle(jd)
        .addMessage("Check me out", new Date().getTime(), jd).setBuilder(builder);

    notificationManager.notify(1, builder.build());
}

private void notificationWithIcon() {
    Person anupam = new Person.Builder()
        .setName("Anupam")
        .setIcon(IconCompat.createWithResource(this, R.drawable.index))
        .setImportant(true).build();
    new NotificationCompat.MessagingStyle(anupam)
        .addMessage("Check out my latest article!", new Date().getTime(), anupam)
        .setBuilder(builder);
}

```

```

notificationManager.notify(2, builder.build());
}
private void notificationWithImage() {
    Person bot = new Person.Builder()
        .setName("Bot") .setImportant(true)
        .setBot(true) .build();

    Uri uri = Uri.parse("android.resource://com.journaldev.androidpnotifications/drawable/"+R.drawable.bg);

    NotificationCompat.MessagingStyle.Message message = new
    NotificationCompat.MessagingStyle.Message("Check out my latest article!", new Date().getTime(), bot);
    message.setData("image/*",uri);

    new NotificationCompat.MessagingStyle(bot)
        .addMessage(message) .setGroupConversation(true).setBuilder(builder);

    notificationManager.notify(3, builder.build());
}
private void notificationWithGroupConvo()
{
    Person jd = new Person.Builder()
        .setName("JournalDev") .build();

    Person anupam = new Person.Builder()
        .setName("Anupam")
        .setIcon(IconCompat.createWithResource(this, R.drawable.samindexple_photo))
        .setImportant(true).build();

    Person bot = new Person.Builder()
        .setName("Bot").setBot(true) .build();

    Uri uri = Uri.parse("android.resource://com.journaldev.androidpnotifications/drawable/"+R.drawable.bg);

    NotificationCompat.MessagingStyle.Message message = new
    NotificationCompat.MessagingStyle.Message("", new Date().getTime(), bot);
    message.setData("image/*",uri);
    new NotificationCompat.MessagingStyle(bot)
        .addMessage("Hi. How are you?", new Date().getTime(), anupam)
        .addMessage(message)
        .addMessage("Does this image look good?", new Date().getTime(), bot)
        .addMessage("Looks good!", new Date().getTime(), jd)
        .setGroupConversation(true)

```



```

        .setConversationTitle("Sample Conversation")
        .setBuilder(builder);

notificationManager.notify(4, builder.build());

}
private void notificationSemantic()
{
    Person jd = new Person.Builder()
        .setName("JournalDev")
        .build();

    Person anupam = new Person.Builder()
        .setName("Anupam")
        .setIcon(IconCompat.createWithResource(this, R.drawable.index))
        .setImportant(true)
        .build();

    Person bot = new Person.Builder()
        .setName("Bot")
        .setBot(true)
        .build();

    Uri uri = Uri.parse("android.resource://com.journaldev.androidnotifications/drawable/"+R.drawable.bg);

    Intent intent = new Intent(this, MainActivity.class);
    intent.putExtra("hi","Notifications were read");
    PendingIntent pendingIntent = PendingIntent.getActivity(this, 0, intent, 0);

    NotificationCompat.MessagingStyle.Message message = new
    NotificationCompat.MessagingStyle.Message("", new Date().getTime(), bot);
    message.setData("image/*",uri);

    NotificationCompat.Action replyAction =
        new NotificationCompat.Action.Builder(
            R.drawable.bg, "MARK READ", pendingIntent)
            .setSemanticAction(NotificationCompat.Action.SEMANTIC_ACTION_MARK_AS_READ)
            .build();
    NotificationCompat.Builder separateBuilder = builder;
    separateBuilder.addAction(replyAction);

    new NotificationCompat.MessagingStyle(bot)

```

```

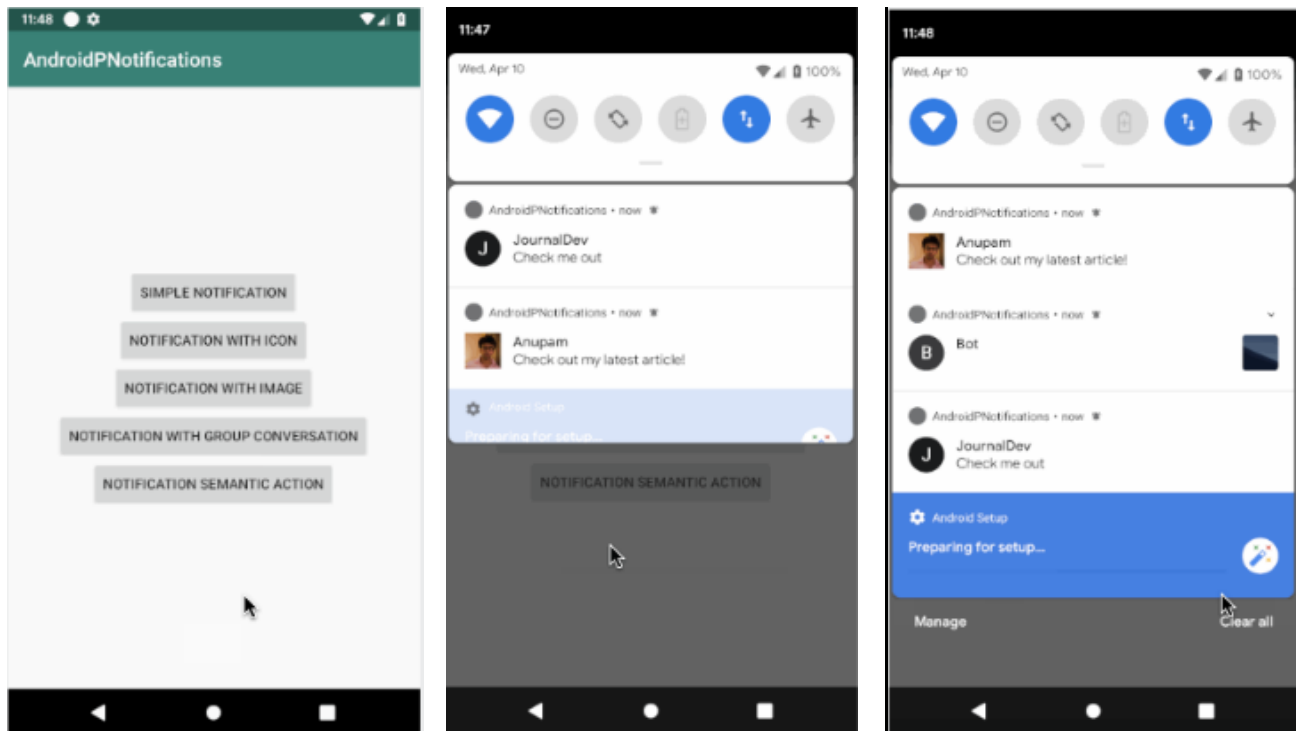
.addMessage("Hi. How are you?", new Date().getTime(), anupam)
.addMessage(message)
.addMessage("Does this image look good?", new Date().getTime(), bot)
.addMessage("Looks good!", new Date().getTime(), jd)
.setGroupConversation(true)
.setConversationTitle("Sample Conversation")
.setBuilder(separateBuilder);

notificationManager.notify(5, separateBuilder.build());
}
@Override
protected void onResume() {
    super.onResume();

    if(getIntent()!=null && getIntent().getExtras()!=null)
    {
        String value = getIntent().getStringExtra("hi");
        Toast.makeText(getApplicationContext(),value,Toast.LENGTH_LONG).show();
    }
}
}
}

```

Output:



Result:

Thus Android Application that makes use of notification manager is developed and executed successfully.

Date:

Aim:

To develop an Android Application that implements Multi threading.

Procedure:**Creating a New project:**

- Open Android Studio and then click on **File -> New -> New project**.
- Then type the Application name as **"exno6"** and click Next.
- Then **select the Minimum SDK** as shown below and click Next.
- Then **select the Empty Activity** and click Next.
- Finally click **Finish**.
- It will take some time to build and load the project.
- After completion it will look as given below.

Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml**.
- Now click on Text as shown below.
- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

<ImageView
    android:id="@+id/imageView"
    android:layout_width="250dp"
    android:layout_height="250dp"
    android:layout_margin="50dp"
    android:layout_gravity="center" />

<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:layout_gravity="center"
    android:text="Load Image 1" />
```

```
<Button
    android:id="@+id/button2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:layout_gravity="center"
    android:text="Load image 2" />
```

```
</LinearLayout>
```

- Now click on Design and your application will look as given below.
- So now the designing part is completed.

Java Coding for the Android Application:

- Click on **app -> java -> com.example.exno6 -> MainActivity**.
- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

```
package com.example.exno6;

import android.os.Bundle;
//import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity
{
    ImageView img;
    Button bt1, bt2;
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        bt1 = (Button) findViewById(R.id.button);
        bt2 = (Button) findViewById(R.id.button2);
        img = (ImageView) findViewById(R.id.imageView);
    }
}
```

```

bt1.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View v)
    {
        new Thread(new Runnable()
        {
            @Override
            public void run()
            {
                img.post(new Runnable()
                {
                    @Override
                    public void run()
                    {
                        img.setImageResource(R.drawable.india1);
                    }
                });
            }
        }).start();
    }
});

```

```

bt2.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View v)
    {
        new Thread(new Runnable()
        {
            @Override
            public void run()
            {
                img.post(new Runnable()
                {
                    @Override
                    public void run()
                    {
                        img.setImageResource(R.drawable.indiaz);
                    }
                });
            }
        });
    }
}

```

```
    }).start();  
  }  
};  
}
```

- So now the Coding part is also completed.
- Now run the application to see the output.

Note:

- Before running the application, copy the images given below and paste it in **"app -> res -> drawable"** by pressing **"right click mouse button on drawable"** and selecting the **"Paste"** option.

Output:



Result:

Thus Android Application that implements Multi threading is developed and executed successfully.

Date:

Aim:

To develop an Android Application that uses GPS location information.

Procedure:**Creating a New project:**

- Open Android Studio and then click on **File -> New -> New project**.
- Then type the Application name as **"exno7"** and click Next.
- Then **select the Minimum SDK** as shown below and click Next.
- Then **select the Empty Activity** and click Next.
- Finally click **Finish**.
- It will take some time to build and load the project.
- After completion it will look as given below.

Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml**.
- Now click on Text as shown below.
- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

```
<?xml version = "1.0" encoding = "utf-8"?>
<LinearLayout xmlns:android = "http://schemas.android.com/apk/res/android"
    android:layout_width = "fill_parent"
    android:layout_height = "fill_parent"
    android:orientation = "vertical" >
```

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

```
</LinearLayout>
```

- Now click on Design and your application will look as given below.
- So now the designing part is completed.

Following will be the content of res/values/strings.xml to define two new constants –

```
<?xml version = "1.0" encoding = "utf-8"?>
<resources>
<string name = "app_name">Tutorialspoint</string>
</resources>
```

Adding permissions in Manifest for the Android Application:

- Click on **app -> manifests -> AndroidManifest.xml**.

Code for AndroidManifest.xml:

```
<?xml version = "1.0" encoding = "utf-8"?>
<manifest xmlns:android = "http://schemas.android.com/apk/res/android"
package = "com.example.tutorialspoint7.myapplication">
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name = "android.permission.INTERNET" />
<application
android:allowBackup = "true"
android:icon = "@mipmap/ic_launcher"
android:label = "@string/app_name"
android:supportsRtl = "true"
android:theme = "@style/AppTheme">

<activity android:name = ".MainActivity">
<intent-filter>
<action android:name = "android.intent.action.MAIN" />
<category android:name = "android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>

</manifest>
```

Java Coding for the Android Application:

- Click on **app -> java -> com.example.exno7 -> MainActivity**.
- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

```
packagecom.example.exno7;

import android.Manifest;
import android.app.Activity;
import android.os.Bundle;
import android.support.v4.app.ActivityCompat;
import android.test.mock.MockPackageManager;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

public class MainActivity extends Activity {

    Button btnShowLocation;
    private static final int REQUEST_CODE_PERMISSION = 2;
    String mPermission = Manifest.permission.ACCESS_FINE_LOCATION;

    // GPSTracker class
    GPSTracker gps;

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

        try {
            if (ActivityCompat.checkSelfPermission(this, mPermission)
                != MockPackageManager.PERMISSION_GRANTED) {

                ActivityCompat.requestPermissions(this, new String[]{mPermission},
                    REQUEST_CODE_PERMISSION);

                // If any permission above not allowed by user, this condition will
                // execute every time, else your else part will work
            }
        } catch (Exception e) {
            e.printStackTrace();
        }

        btnShowLocation = (Button) findViewById(R.id.button);
```

```

// show location button click event
btnShowLocation.setOnClickListener(new View.OnClickListener() {

    @Override
    public void onClick(View arg0) {
        // create class object
        gps = new GPSTracker(MainActivity.this);

        // check if GPS enabled
        if(gps.canGetLocation()){

            double latitude = gps.getLatitude();
            double longitude = gps.getLongitude();

            //\n is for new line
            Toast.makeText(getApplicationContext(), "Your Location is - \nLat: "
                + latitude + "\nLong: " + longitude, Toast.LENGTH_LONG).show();
        }else{
            // can't get location
            // GPS or Network is not enabled
            // Ask user to enable GPS/network in settings
            gps.showSettingsAlert();
        }

    }

});
}
}

```

- Following is the content of the modified main activity file **GPSTracker.java**.

Code for GPDTracker.Java

```

packagecom.example.exno7;
import android.app.AlertDialog;
import android.app.Service;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.os.IBinder;
import android.provider.Settings;

```

```

import android.util.Log;
public class GPSTracker extends Service implements LocationListener {

    private final Context mContext;

    // flag for GPS status
    boolean isGPSEnabled = false;

    // flag for network status
    boolean isNetworkEnabled = false;

    // flag for GPS status
    boolean canGetLocation = false;

    Location location; // location
    double latitude; // latitude
    double longitude; // longitude

    // The minimum distance to change Updates in meters
    private static final long MIN_DISTANCE_CHANGE_FOR_UPDATES = 10; // 10 meters

    // The minimum time between updates in milliseconds
    private static final long MIN_TIME_BW_UPDATES = 1000 * 60 * 1; // 1 minute

    // Declaring a Location Manager
    protected LocationManager locationManager;

    public GPSTracker(Context context) {
        this.mContext = context;
        getLocation();
    }

    public Location getLocation() {
        try {
            locationManager = (LocationManager) mContext.getSystemService(LOCATION_SERVICE);

            // getting GPS status
            isGPSEnabled = locationManager.isProviderEnabled(LocationManager.GPS_PROVIDER);

            // getting network status
            isNetworkEnabled = locationManager
                .isProviderEnabled(LocationManager.NETWORK_PROVIDER);

```

```

if (!isGPSEnabled && !isNetworkEnabled) {
    // no network provider is enabled
} else {
    this.canGetLocation = true;
    // First get location from Network Provider
    if (isNetworkEnabled) {
        locationManager.requestLocationUpdates(
            LocationManager.NETWORK_PROVIDER,
            MIN_TIME_BW_UPDATES,
            MIN_DISTANCE_CHANGE_FOR_UPDATES, this);

        Log.d("Network", "Network");
        if (locationManager != null) {
            location = locationManager
                .getLastKnownLocation(LocationManager.NETWORK_PROVIDER);

            if (location != null) {
                latitude = location.getLatitude();
                longitude = location.getLongitude();
            }
        }
    }

    // if GPS Enabled get lat/long using GPS Services
    if (isGPSEnabled) {
        if (location == null) {
            locationManager.requestLocationUpdates(
                LocationManager.GPS_PROVIDER,
                MIN_TIME_BW_UPDATES,
                MIN_DISTANCE_CHANGE_FOR_UPDATES, this);

            Log.d("GPS Enabled", "GPS Enabled");
            if (locationManager != null) {
                location = locationManager
                    .getLastKnownLocation(LocationManager.GPS_PROVIDER);

                if (location != null) {
                    latitude = location.getLatitude();
                    longitude = location.getLongitude();
                }
            }
        }
    }
}

```

```

    }
    }
}

} catch (Exception e) {
    e.printStackTrace();
}

return location;
}

/**
 * Stop using GPS listener
 * Calling this function will stop using GPS in your app
 * */

public void stopUsingGPS(){
    if(locationManager != null){
        locationManager.removeUpdates(GPSTracker.this);
    }
}

/**
 * Function to get latitude
 * */

public double getLatitude(){
    if(location != null){
        latitude = location.getLatitude();
    }

    // return latitude
    return latitude;
}

/**
 * Function to get longitude
 * */

public double getLongitude(){
    if(location != null){
        longitude = location.getLongitude();
    }
}

```

```

}

// return longitude
return longitude;
}

/**
 * Function to check GPS/wifi enabled
 * @return boolean
 * */

public boolean canGetLocation() {
    return this.canGetLocation;
}

/**
 * Function to show settings alert dialog
 * On pressing Settings button will launch Settings Options
 * */

public void showSettingsAlert(){
    AlertDialog.Builder alertDialog = new AlertDialog.Builder(mContext);

    // Setting Dialog Title
    alertDialog.setTitle("GPS is settings");

    // Setting Dialog Message
    alertDialog.setMessage("GPS is not enabled. Do you want to go to settings menu?");

    // On pressing Settings button
    alertDialog.setPositiveButton("Settings", new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog,int which) {
            Intent intent = new Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS);
            mContext.startActivity(intent);
        }
    });

    // on pressing cancel button
    alertDialog.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog, int which) {
            dialog.cancel();
        }
    });
}

```

```
});

// Showing Alert Message
AlertDialog.show();
}

@Override
public void onLocationChanged(Location location) {
}

@Override
public void onProviderDisabled(String provider) {
}

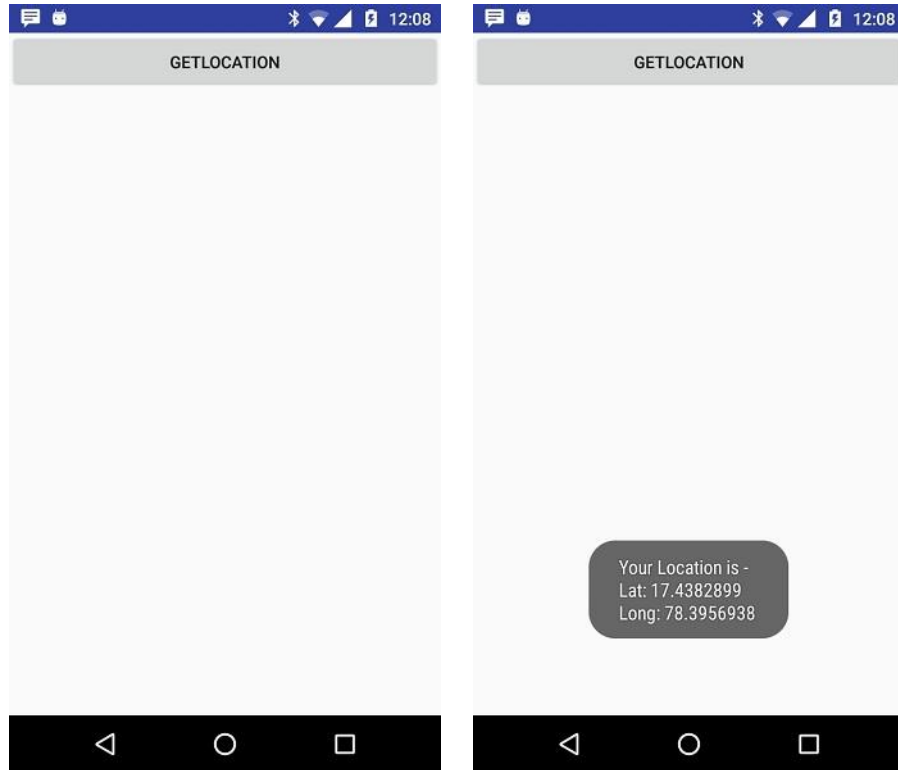
@Override
public void onProviderEnabled(String provider) {
}

@Override
public void onStatusChanged(String provider, int status, Bundle extras) {
}

@Override
public IBinder onBind(Intent arg0) {
    return null;
}
}
```

- So now the Coding part is also completed.
- Now run the application to see the output.

Output:



Result:

Thus Android Application that implements GPS Location Information is developed and executed successfully.

Date:

Aim:

To develop an Android Application that writes data to the SD Card.

Procedure:**Creating a New project:**

- Open Android Studio and then click on **File -> New -> New project**.
- Then type the Application name as **"exno8"** and click Next.
- Then **select the Minimum SDK** as shown below and click Next.
- Then **select the Empty Activity** and click Next.
- Finally click **Finish**.
- It will take some time to build and load the project.
- After completion it will look as given below.

Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml**.
- Now click on Text as shown below.
- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="20dp"
    android:orientation="vertical">

<EditText
    android:id="@+id/editText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:singleLine="true"
    android:textSize="30dp" />

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

```
android:layout_margin="10dp"
android:text="Write Data"
android:textSize="30dp" />
```

```
<Button
```

```
    android:id="@+id/button2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:text="Read data"
    android:textSize="30dp" />
```

```
<Button
```

```
    android:id="@+id/button3"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    android:text="Clear"
    android:textSize="30dp" />
```

```
</LinearLayout>
```

- Now click on Design and your application will look as given below.
- So now the designing part is completed.

Adding permissions in Manifest for the Android Application:

- Click on **app -> manifests -> AndroidManifest.xml**.
- Now include the `WRITE_EXTERNAL_STORAGE` permissions in the `AndroidManifest.xml` file as shown below.

Code for AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.exno8" >

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

<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
```

```

    android:supportsRtl="true"
    android:theme="@style/AppTheme" >
<activity android:name=".MainActivity" >
<intent-filter>
<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
</manifest>

```

- So now the Permissions are added in the Manifest.

Java Coding for the Android Application:

- Click on **app** -> **java** -> **com.example.exno8** -> **MainActivity**.
- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

```

package com.example.exno8;

import android.os.Bundle;
//import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStreamReader;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity
{
    EditText e1;
    Button write,read,clear;
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {

```

```

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

e1= (EditText) findViewById(R.id.editText);
write= (Button) findViewById(R.id.button);
read= (Button) findViewById(R.id.button2);
clear= (Button) findViewById(R.id.button3);

write.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View v)
    {
        String message=e1.getText().toString();
        try
        {
            File f=new File("/sdcard/myfile.txt");
            f.createNewFile();
            FileOutputStream fout=new FileOutputStream(f);
            fout.write(message.getBytes());
            fout.close();
            Toast.makeText(getApplicationContext(),"Data Written in SDCARD",Toast.LENGTH_LONG).show();
        }
        catch (Exception e)
        {
            Toast.makeText(getApplicationContext(),e.getMessage(),Toast.LENGTH_LONG).show();
        }
    }
});

read.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View v)
    {
        String message;
        String buf = "";
        try
        {
            File f = new File("/sdcard/myfile.txt");
            FileInputStream fin = new FileInputStream(f);
            BufferedReader br = new BufferedReader(new InputStreamReader(fin));

```

```

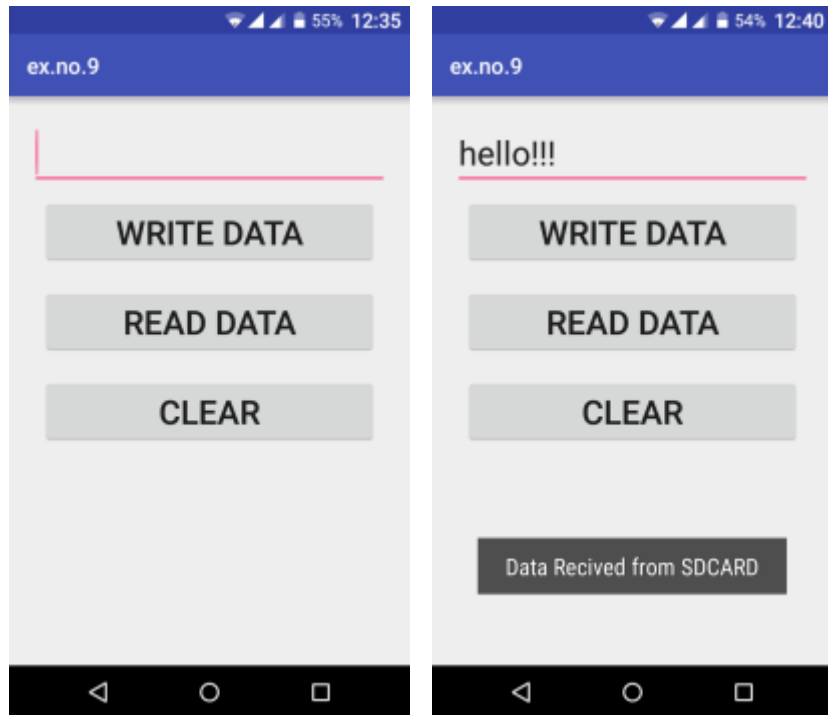
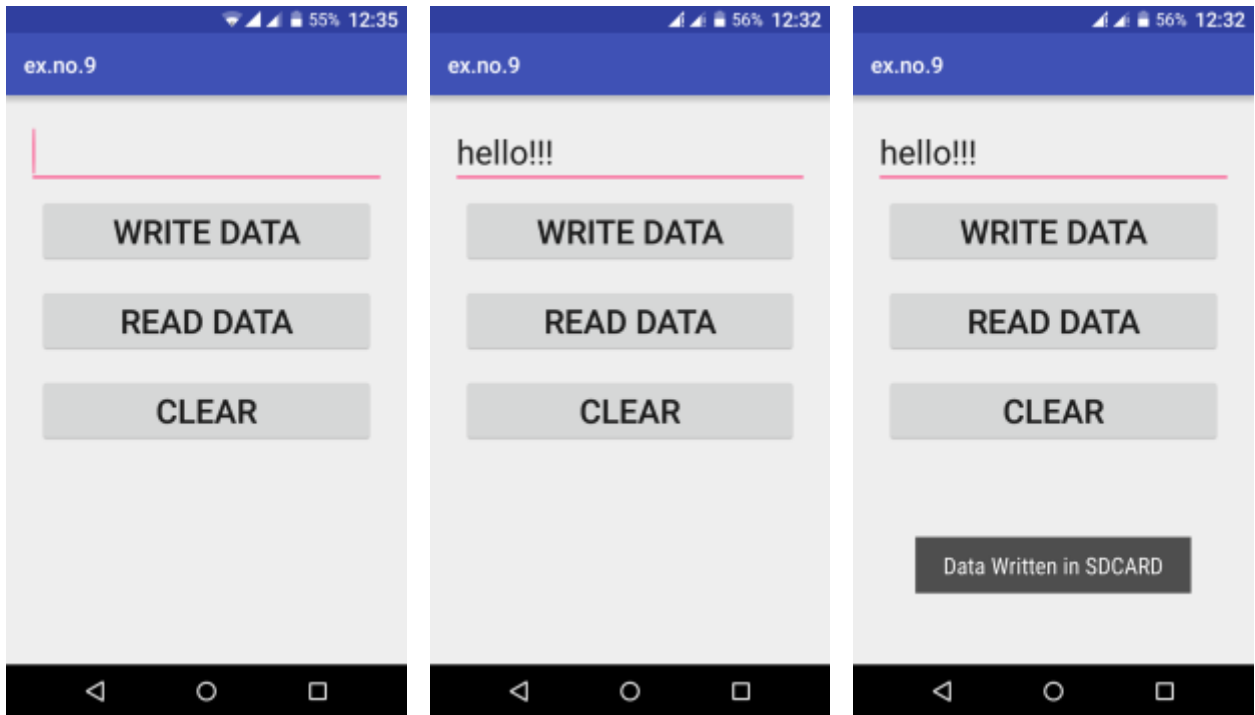
        while ((message = br.readLine()) != null)
        {
            buf += message;
        }
        e1.setText(buf);
        br.close();
        fin.close();
        Toast.makeText(getBaseContext(),"Data Recived from SDCARD",Toast.LENGTH_LONG).show();
    }
    catch (Exception e)
    {
        Toast.makeText(getBaseContext(), e.getMessage(), Toast.LENGTH_LONG).show();
    }
    }
    });

clear.setOnClickListener(new View.OnClickListener()
{
    @Override
    public void onClick(View v)
    {
        e1.setText("");
    }
});
}
}

```

- So now the Coding part is also completed.
- Now run the application to see the output.

Output:



Result:

Thus Android Application that writes data to the SD Card is developed and executed successfully.

Date:

Aim:

To develop an Android Application that creates an alert upon receiving a message.

Procedure:**Creating a New project:**

- Open Android Studio and then click on **File -> New -> New project**.
- Then type the Application name as "**ex.nog**" and click Next.
- Then **select the Minimum SDK** as shown below and click Next.
- Then **select the Empty Activity** and click Next.
- Finally click **Finish**.
- It will take some time to build and load the project.
- After completion it will look as given below.

Creating Second Activity for the Android Application:

- Click on **File -> New -> Activity -> Empty Activity**.
- Type the Activity Name as SecondActivity and click Finish button.
- Thus Second Activity For the application is created.

Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml**.
- Now click on Text as shown below.
- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="10dp"
    android:orientation="vertical">

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Message"
    android:textSize="30sp" />
```

```
<EditText
    android:id="@+id/editText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:singleLine="true"
    android:textSize="30sp" />
```

```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_margin="30dp"
    android:layout_gravity="center"
    android:text="Notify"
    android:textSize="30sp"/>
```

```
</LinearLayout>
```

- Now click on Design and your application will look as given below.
- So now the designing part is completed.

Java Coding for the Android Application:

- Click on **app -> java -> com.example.exnog -> MainActivity**.
- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

```
package com.example.exnog;

import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Intent;
import android.os.Bundle;
//import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity
{
```



```

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

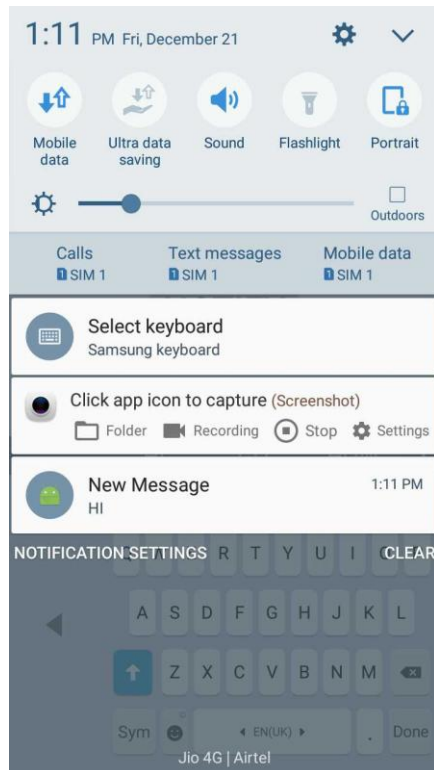
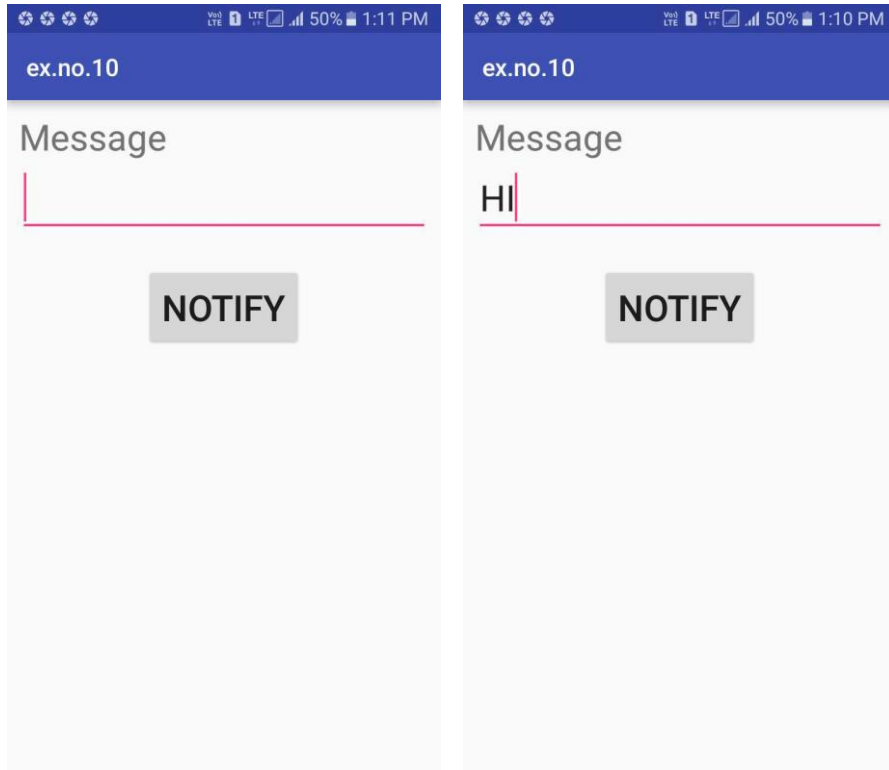
    notify= (Button) findViewById(R.id.button);
    e= (EditText) findViewById(R.id.editText);

    notify.setOnClickListener(new View.OnClickListener()
    {
        @Override
        public void onClick(View v)
        {
            Intent intent = new Intent(MainActivity.this, SecondActivity.class);
            PendingIntent pending = PendingIntent.getActivity(MainActivity.this, 0, intent, 0);
            Notification noti = new Notification.Builder(MainActivity.this).setContentTitle("New
Message").setContentText(e.getText().toString()).setSmallIcon(R.mipmap.ic_launcher).setContentIntent(pe
nding).build();
            NotificationManager manager = (NotificationManager)
getSystemService(NOTIFICATION_SERVICE);
            noti.flags |= Notification.FLAG_AUTO_CANCEL;
            manager.notify(0, noti);
        }
    });
}
}

```

- So now the coding part is also completed.
- Now run the application to see the output.

Output:



Result:

Thus Android Application that creates an alert upon receiving a message is developed and executed successfully.

Date:

Aim:

To develop an Android Application that makes use of RSS Feed.

Procedure:**Creating a New project:**

- Open Android Studio and then click on **File -> New -> New project**.
- Then type the Application name as "**exno10**" and click Next.
- Then **select the Minimum SDK** as shown below and click Next.
- Then select the **Empty Activity** and click Next.
- Finally click **Finish**.
- It will take some time to build and load the project.
- After completion it will look as given below.

Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml**.
- Now click on Text as shown below.
- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
```

```
<ListView
    android:id="@+id/listView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />
```

```
</LinearLayout>
```

- Now click on Design and your application will look as given below.
- So now the designing part is completed.

Adding permissions in Manifest for the Android Application:

- Click on **app -> manifests -> AndroidManifest.xml**.
- Now include the INTERNET permissions in the AndroidManifest.xml file as shown below.

Code for AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.exno10" >
<uses-permission android:name="android.permission.INTERNET"/>

<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:supportsRtl="true"
    android:theme="@style/AppTheme" >
<activity android:name=".MainActivity" >
<intent-filter>
<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>

</manifest>
```

- So now the Permissions are added in the Manifest.

Java Coding for the Android Application:

- Click on app -> java -> com.example.exno10 -> MainActivity.
- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

```
package com.example.exno10;

import android.app.ListActivity;
import android.content.Intent;
import android.net.Uri;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
```

```

import android.widget.ListView;
import org.xmlpull.v1.XmlPullParser;
import org.xmlpull.v1.XmlPullParserException;
import org.xmlpull.v1.XmlPullParserFactory;
import java.io.IOException;
import java.io.InputStream;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.ArrayList;
import java.util.List;

public class MainActivity extends ListActivity
{
    List headlines;
    List links;
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        new MyAsyncTask().execute();
    }
    class MyAsyncTask extends AsyncTask<Object,Void,ArrayAdapter>
    {
        @Override
        protected ArrayAdapter doInBackground(Object[] params)
        {
            headlines = new ArrayList();
            links = new ArrayList();
            try
            {
                URL url = new URL("https://codingconnect.net/feed");
                XmlPullParserFactory factory = XmlPullParserFactory.newInstance();
                factory.setNamespaceAware(false);
                XmlPullParser xpp = factory.newPullParser();

                // We will get the XML from an input stream
                xpp.setInput(getInputStream(url), "UTF_8");
                boolean insideItem = false;

```

```

// Returns the type of current event: START_TAG, END_TAG, etc..
int eventType = xpp.getEventType();
while (eventType != XmlPullParser.END_DOCUMENT)
{
    if (eventType == XmlPullParser.START_TAG)
    {
        if (xpp.getName().equalsIgnoreCase("item"))
        {
            insideltem = true;
        }
        else if (xpp.getName().equalsIgnoreCase("title"))
        {
            if (insideltem)
                headlines.add(xpp.nextText()); //extract the headline
        }
        else if (xpp.getName().equalsIgnoreCase("link"))
        {
            if (insideltem)
                links.add(xpp.nextText()); //extract the link of article
        }
    }
    else if (eventType == XmlPullParser.END_TAG && xpp.getName().equalsIgnoreCase("item"))
    {
        insideltem = false;
    }
    eventType = xpp.next(); //move to next element
}

}
catch (MalformedURLException e)
{
    e.printStackTrace();
}
catch (XmlPullParserException e)
{
    e.printStackTrace();
}
catch (IOException e)
{

```

```

        e.printStackTrace();
    }
    return null;
}
protected void onPostExecute(ArrayAdapter adapter)
{
    adapter = new ArrayAdapter(MainActivity.this, android.R.layout.simple_list_item_1, headlines);
    setListAdapter(adapter);
}
}
@Override
protected void onItemClick(ListView l, View v, int position, long id)
{
    Uri uri = Uri.parse((links.get(position)).toString());
    Intent intent = new Intent(Intent.ACTION_VIEW, uri);
    startActivity(intent);
}
public InputStream getInputStream(URL url)
{
    try
    {
        return url.openConnection().getInputStream();
    }
    catch (IOException e)
    {
        return null;
    }
}
}
}

```

- So now the Coding part is also completed.
- Now run the application to see the output.

Output:



Result:

Thus Android Application that makes use of RSS Feed is developed and executed successfully.

Date:

Aim:

To develop an Android Application to send an Email.

Procedure:**Creating a New project:**

- Open Android Studio and then click on **File -> New -> New project**.
- Then type the Application name as "**exno11**" and click Next.
- Then **select the Minimum SDK** as shown below and click Next.
- Then **select the Empty Activity** and click Next.
- Finally click **Finish**.
- It will take some time to build and load the project.
- After completion it will look as given below.

Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml**.
- Now click on Text as shown below.
- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="20dp"
    android:paddingRight="20dp"
    android:orientation="vertical" >
    <EditText
        android:id="@+id/txtTo"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="To"/>
    <EditText
        android:id="@+id/txtSub"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Subject"/>
    <EditText
```

```

    android:id="@+id/txtMsg"
    android:layout_width="match_parent"
    android:layout_height="odp"
    android:layout_weight="1"
    android:gravity="top"
    android:hint="Message"/>
<Button
    android:layout_width="100dp"
    android:layout_height="wrap_content"
    android:layout_gravity="right"
    android:text="Send"
    android:id="@+id/btnSend"/>
</LinearLayout>

```

Adding permissions in Manifest for the Android Application:

- Click on **app -> manifests -> AndroidManifest.xml**.
- Now include the INTERNET permissions in the AndroidManifest.xml file as shown below.

Code for AndroidManifest.xml:

```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.example.exno11" >
    <uses-permission android:name="android.permission.INTERNET" />
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme"
        tools:ignore="GoogleAppIndexingWarning">

        <activity
            android:name="com.example.exno11.MainActivity"
            android:label="@string/app_name">

            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />

```

```

    <action android:name="android.intent.action.SEND"/>
    <category android:name="android.intent.category.DEFAULT"/>
    <data android:mimeType="message/rfc822"/>
  </intent-filter>
</activity>
</application>
</manifest>

```

- So now the Permissions are added in the Manifest.

Java Coding for the Android Application:

- Click on **app -> java -> com.example.exno10 -> MainActivity**.
- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

```

package com.example.exno11;

import android.content.Intent;
//import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import androidx.appcompat.app.AppCompatActivity;
public class MainActivity extends AppCompatActivity {

    private EditText eTo;
    private EditText eSubject;
    private EditText eMsg;
    private Button btn;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        eTo = (EditText)findViewById(R.id.txtTo);
        eSubject = (EditText)findViewById(R.id.txtSub);
        eMsg = (EditText)findViewById(R.id.txtMsg);
        btn = (Button)findViewById(R.id.btnSend);
        btn.setOnClickListener(new View.OnClickListener() {
            @Override

```

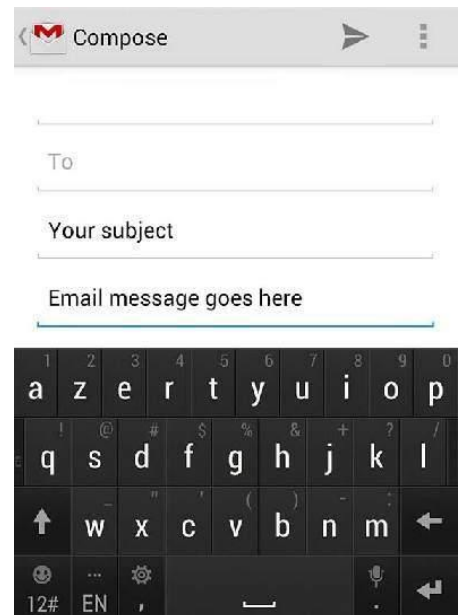
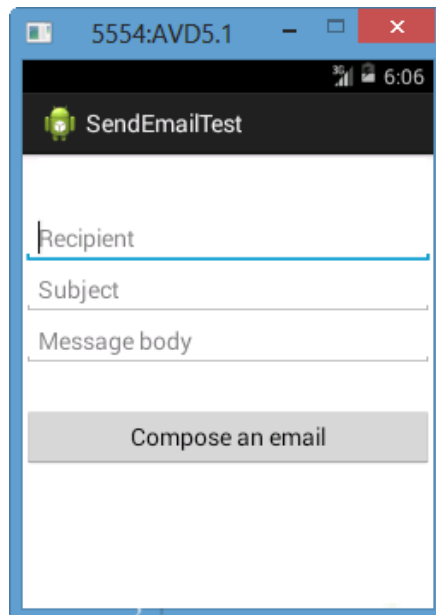
```

public void onClick(View v) {
    Intent it = new Intent(Intent.ACTION_SEND);
    it.putExtra(Intent.EXTRA_EMAIL, new String[]{eTo.getText().toString()});
    it.putExtra(Intent.EXTRA_SUBJECT,eSubject.getText().toString());
    it.putExtra(Intent.EXTRA_TEXT,eMsg.getText());
    it.setType("message/rfc822");
    startActivity(Intent.createChooser(it,"Choose Mail App"));
}
});
}
}

```

- So now the Coding part is also completed.
- Now run the application to see the output.

Output:



Result:

Thus Android Application for sending an email is developed and executed successfully.

Date:

Aim:

To develop a Simple Android Application for Native Calculator.

Procedure:**Creating a New project:**

- Open Android Studio and then click on **File -> New -> New project**.
- Then type the Application name as "**exno12**" and click Next.
- Then **select the Minimum SDK** as shown below and click Next.
- Then **select the Empty Activity** and click Next.
- Finally click **Finish**.
- It will take some time to build and load the project.
- After completion it will look as given below.

Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml**.
- Now click on Text as shown below.
- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<LinearLayout
```

```
    xmlns:android="http://schemas.android.com/apk/res/android"
```

```
    android:orientation="vertical"
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="match_parent"
```

```
    android:layout_margin="20dp">
```

```
<LinearLayout
```

```
    android:id="@+id/linearlayout1"
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="wrap_content"
```

```
    android:layout_margin="20dp">
```

```
<EditText
```

```
    android:id="@+id/edittext1"
```

```
    android:layout_width="match_parent"
```

```
    android:layout_height="wrap_content"
```

```
android:layout_weight="1"
android:inputType="numberDecimal"
android:textSize="20sp" />
```

```
<EditText
```

```
    android:id="@+id/editText2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:inputType="numberDecimal"
    android:textSize="20sp" />
```

```
</LinearLayout>
```

```
<LinearLayout
```

```
    android:id="@+id/linearLayout2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="20dp">
```

```
<Button
```

```
    android:id="@+id/Add"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:text="+"
    android:textSize="30sp"/>
```

```
<Button
```

```
    android:id="@+id/Sub"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_weight="1"
    android:text="-"
    android:textSize="30sp"/>
```

```
<Button
```

```
    android:id="@+id/Mul"
    android:layout_width="match_parent"
```

```
android:layout_height="wrap_content"
android:layout_weight="1"
android:text="*"
android:textSize="30sp"/>
```

<Button

```
android:id="@+id/Div"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_weight="1"
android:text="/"
android:textSize="30sp"/>
```

</LinearLayout>

<TextView

```
android:id="@+id/textView"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginTop="50dp"
android:text="Answer is"
android:textSize="30sp"
android:gravity="center"/>
```

</LinearLayout>

- Now click on Design and your application will look as given below.
- So now the designing part is completed.

Java Coding for the Android Application:

- Click on **app -> java -> com.example.exno12 -> MainActivity**.
- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

```
package com.example.exno12;
import android.os.Bundle;
//import android.support.v7.app.AppCompatActivity;
import android.text.TextUtils;
import android.view.View;
import android.view.View.OnClickListener;
```

```

import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;
public class MainActivity extends AppCompatActivity implements OnClickListener
{
    //Defining the Views
    EditText Num1;
    EditText Num2;
    Button Add;
    Button Sub;
    Button Mul;
    Button Div;
    TextView Result;

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

        //Referring the Views
        Num1 = (EditText) findViewById(R.id.editText1);
        Num2 = (EditText) findViewById(R.id.editText2);
        Add = (Button) findViewById(R.id.Add);
        Sub = (Button) findViewById(R.id.Sub);
        Mul = (Button) findViewById(R.id.Mul);
        Div = (Button) findViewById(R.id.Div);
        Result = (TextView) findViewById(R.id.textView);

        // set a listener
        Add.setOnClickListener(this);
        Sub.setOnClickListener(this);
        Mul.setOnClickListener(this);
        Div.setOnClickListener(this);
    }
    @Override
    public void onClick (View v)
    {

```



```

float num1 = 0;
float num2 = 0;
float result = 0;
String oper = "";
// check if the fields are empty
if (TextUtils.isEmpty(Num1.getText().toString()) || TextUtils.isEmpty(Num2.getText().toString()))
    return;

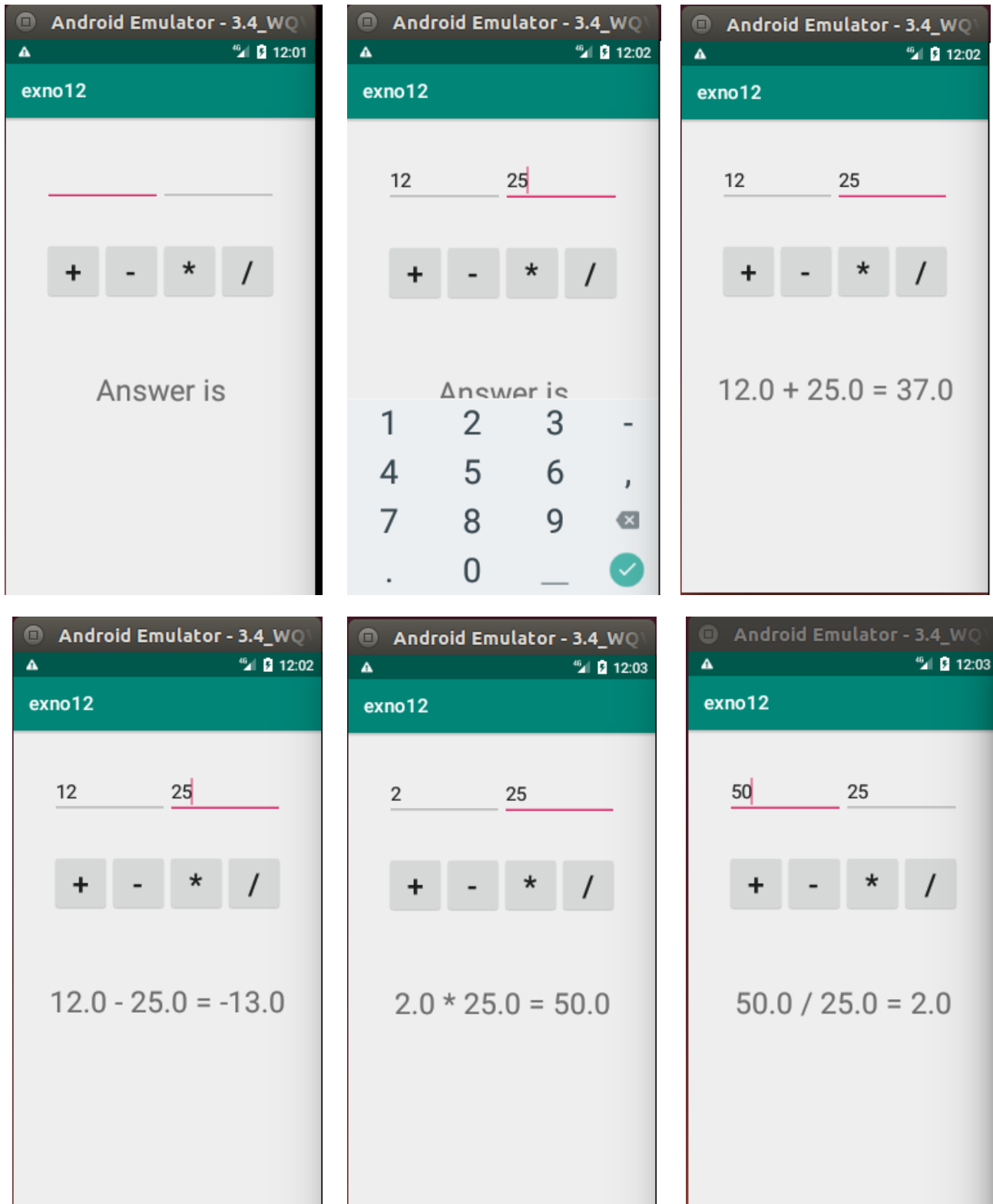
// read EditText and fill variables with numbers
num1 = Float.parseFloat(Num1.getText().toString());
num2 = Float.parseFloat(Num2.getText().toString());

// defines the button that has been clicked and performs the corresponding operation
// write operation into oper, we will use it later for output
switch (v.getId())
{
    case R.id.Add:
        oper = "+";
        result = num1 + num2;
        break;
    case R.id.Sub:
        oper = "-";
        result = num1 - num2;
        break;
    case R.id.Mul:
        oper = "*";
        result = num1 * num2;
        break;
    case R.id.Div:
        oper = "/";
        result = num1 / num2;
        break;
    default:
        break;
}
// form the output line
Result.setText(num1 + " " + oper + " " + num2 + " = " + result);
}
}

```

- So now the Coding part is also completed.
- Now run the application to see the output.

Output:



Result:

Thus a Simple Android Application for Native Calculator is developed and executed successfully.

Additional Exercises

Ex. No. 01

Android Application that creates Alarm Clock

Date:

Aim:

To develop a Android Application that creates Alarm Clock.

Procedure:

Creating a New project:

- Open Android Studio and then click on **File -> New -> New project**.
- Then type the Application name as "**exn013**" and click Next.
- Then **select the Minimum SDK** as shown below and click Next.
- Then **select the Empty Activity** and click Next.
- Finally click **Finish**.
- It will take some time to build and load the project.
- After completion it will look as given below.

Creating Second Activity for the Android Application:

- Click on **File -> New -> Activity -> Empty Activity**.
- Type the **Activity Name as AlarmReceiver** and click **Finish** button.
- Thus **Second Activity** for the application is created.

Designing layout for the Android Application:

- Click on **app -> res -> layout -> activity_main.xml**.
- Now click on Text as shown below.
- Then delete the code which is there and type the code as given below.

Code for Activity_main.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
```

```
<TimePicker
    android:id="@+id/timePicker"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center" />
```

```
<ToggleButton
    android:id="@+id/toggleButton"
```

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_gravity="center"
android:layout_margin="20dp"
android:checked="false"
android:onClick="OnToggleClicked" />
```

```
</LinearLayout>
```

- Now click on Design and your application will look as given below.
- So now the designing part is completed.

Changes in Manifest for the Android Application:

- Click on **app -> manifests -> AndroidManifest.xml**.
- Now change the activity tag to receiver tag in the AndroidManifest.xml file as shown below.

Code for AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.exn013" >

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme" >
        <activity android:name=".MainActivity" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <receiver android:name=".AlarmReceiver" >
            </receiver>
        </application>

    </manifest>
```

- So now the changes are done in the Manifest.

Java Coding for the Android Application:

Java Coding for Main Activity:

- Click on **app -> java -> com.example.exno13 -> MainActivity**.
- Then delete the code which is there and type the code as given below.

Code for MainActivity.java:

```
package com.example.exno13;
import android.app.AlarmManager;
import android.app.PendingIntent;
import android.content.Intent;
import android.os.Bundle;
//import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.TimePicker;
import android.widget.Toast;
import android.widget.ToggleButton;

import java.util.Calendar;
import androidx.appcompat.app.AppCompatActivity;
public class MainActivity extends AppCompatActivity
{
    TimePicker alarmTimePicker;
    PendingIntent pendingIntent;
    AlarmManager alarmManager;

    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        alarmTimePicker = (TimePicker) findViewById(R.id.timePicker);
        alarmManager = (AlarmManager) getSystemService(ALARM_SERVICE);
    }
    public void OnToggleClicked(View view)
    {
        long time;
        if (((ToggleButton) view).isChecked())
        {
            Toast.makeText(MainActivity.this, "ALARM ON", Toast.LENGTH_SHORT).show();
            Calendar calendar = Calendar.getInstance();
            calendar.set(Calendar.HOUR_OF_DAY, alarmTimePicker.getCurrentHour());
```

```

calendar.set(Calendar.MINUTE, alarmTimePicker.getCurrentMinute());
Intent intent = new Intent(this, AlarmReceiver.class);
pendingIntent = PendingIntent.getBroadcast(this, 0, intent, 0);

time=(calendar.getTimeInMillis()-(calendar.getTimeInMillis()%60000));
if(System.currentTimeMillis(>time)
{
    if (calendar.AM_PM == 0)
        time = time + (1000*60*60*12);
    else
        time = time + (1000*60*60*24);
}
alarmManager.setRepeating(AlarmManager.RTC_WAKEUP, time, 10000, pendingIntent);
}
else
{
    alarmManager.cancel(pendingIntent);
    Toast.makeText(MainActivity.this, "ALARM OFF", Toast.LENGTH_SHORT).show();
}
}

```

Java Coding for Alarm Receiver:

- Click on **app** -> **java** -> **com.example.exno13** -> **AlarmReceiver**.
- Then delete the code which is there and type the code as given below.

Code for AlarmReceiver.java:

```

package com.example.exno13;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.media.Ringtone;
import android.media.RingtoneManager;
import android.net.Uri;
import android.widget.Toast;

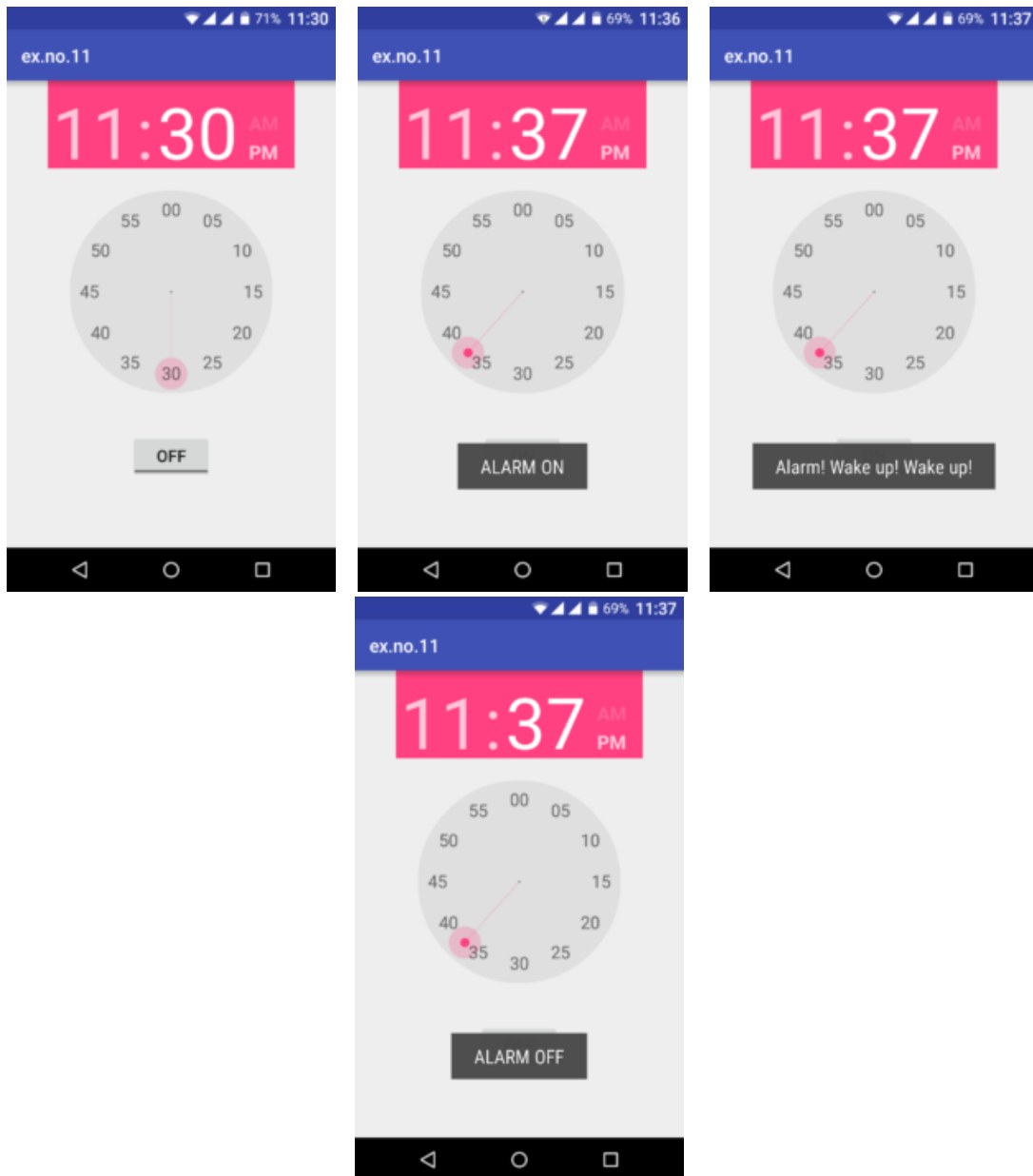
public class AlarmReceiver extends BroadcastReceiver
{
    @Override
    public void onReceive(Context context, Intent intent)
    {
        Toast.makeText(context, "Alarm! Wake up! Wake up!", Toast.LENGTH_LONG).show();
        Uri alarmUri = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_ALARM);
    }
}

```

```
if (alarmUri == null)
{
    alarmUri = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
}
Ringtone ringtone = RingtoneManager.getRingtone(context, alarmUri);
ringtone.play();
}
}
```

- So now the Coding part of Alarm Receiver is also completed.
- Now run the application to see the output.

Output:



Result:

Thus Android Application that creates Alarm Clock is developed and executed successfully.