GOVERNMENT POLYTECHNIC KASHIPUR

(U.S NAGAR)

Semester: - 5th

Session (2025-2026) Winter

Subject: - Android Application Development

Branch: -Information Technology

Batch (2023-2026)

Submitted To:

MR. J.C. Pandey

Head of Department

Submitted By:

Aman Diktiya

23009120017

INDEX

S. NO.	Name of the Practical	Page NO.	Issue date	Submission date	Grade	Signature
01.	To study of Installation and Configuration of Android Development Framework.					
02.	Study of Various Android Development Environment (IDE).					
03.	To design an Application in an Android Environment representing a Simple Calculator.					
04.	Develop an application for working with Menus and screen Navigation.					
05.	Develop an application demonstrating internal storage to store private data on the device memory.					
06.	Design a simple to-do list application using SQLite.					
07.	Develop an application for connecting to the internet and sending email.					
08.	Develop an application for working with graphics and animations.					
09.	Develop an application for working with location-based service.					
10.	Develop an application for working with device camera.					

Practical-1

Aim- To study of Installation and Configuration of Android Development Framework.

Android Studio is the official IDE (Integrated Development Environment) for Android app development based on JetBrains' IntelliJ IDEA software. Android Studio provides many excellent features that enhance productivity when building Android apps, such as:

- A blended environment where one can develop for all Android devices.
- Apply Changes to push code and resource changes to the running app without restarting the app.
- A flexible Gradle-based build system.
- A fast and feature-rich emulator.
- GitHub and Code template integration to assist you in developing standard app features and importing sample code.
- Extensive testing tools and frameworks.
- C++ and NDK support.
- Built-in support for Google Cloud Platform makes integrating Google Cloud Messaging and App Engine easy, and many more.
- Provides GUI tools that simplify the less interesting parts of app development.
- Easy integration with real-time database 'firebase'.

System Requirements-

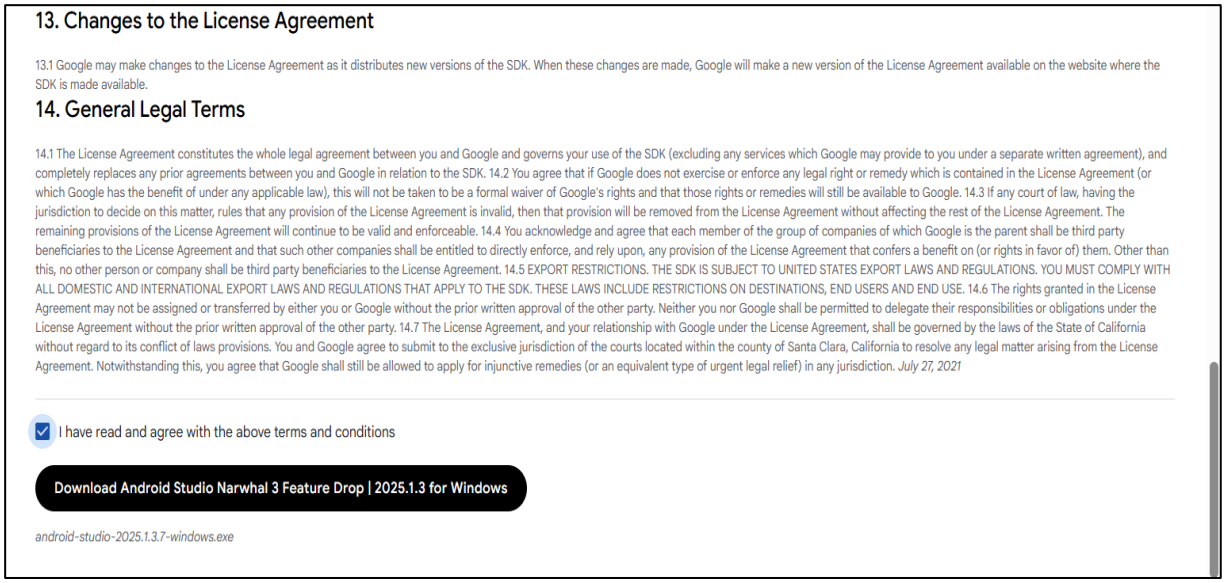
- Microsoft Windows 7/8/10 (32-bit or 64-bit).
- 4 GB RAM minimum, 8 GB RAM recommended (plus 1 GB for the Android Emulator)
- 2 GB of available disk space minimum, 4 GB recommended (500 MB for IDE plus 1.5 GB for Android SDK and emulator system image).
- 1280 x 800 minimum screen resolution.

Stepwise instructions to install Android Studio-

Step 1: Go to https://developer.android.com/studio/#downloads to get the Android Studio executable or zip file.

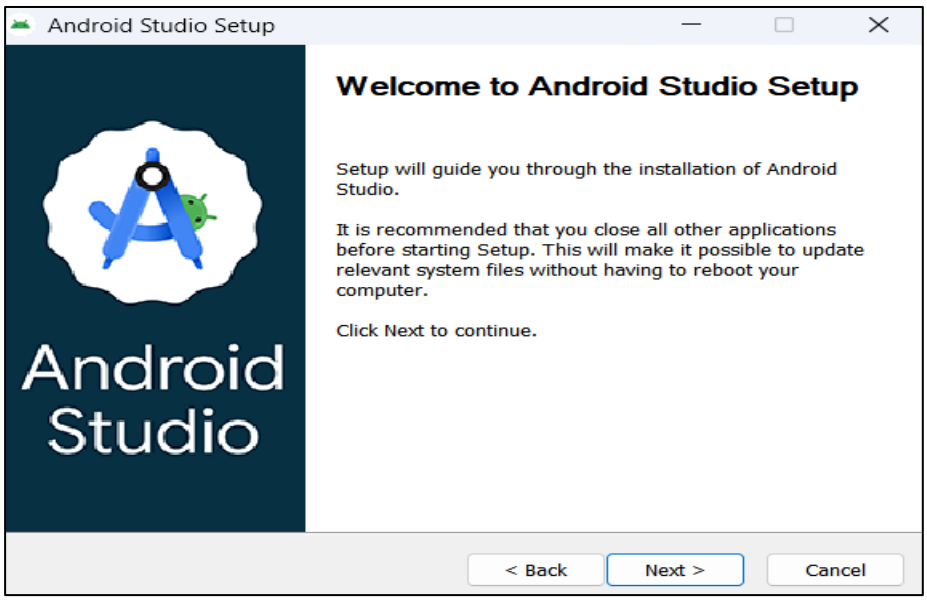
Step 2: Click on the **Download Android Studio** Button.

Click on the “I have read and agree with the above terms and conditions” checkbox followed by the download button.



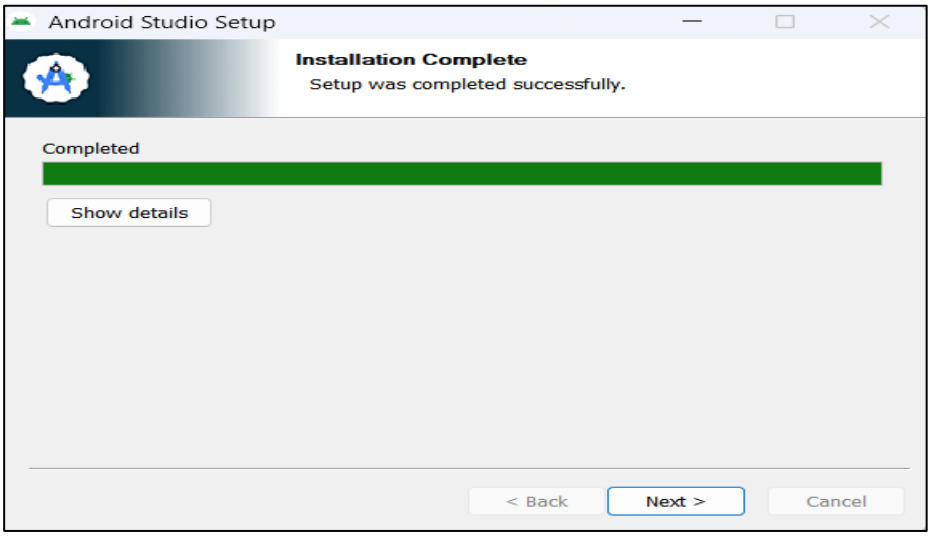
Click on the Save file button in the appeared prompt box and the file will start downloading.

Step 3: After the downloading has finished, open the file from downloads and run it. It will prompt the following dialog box.

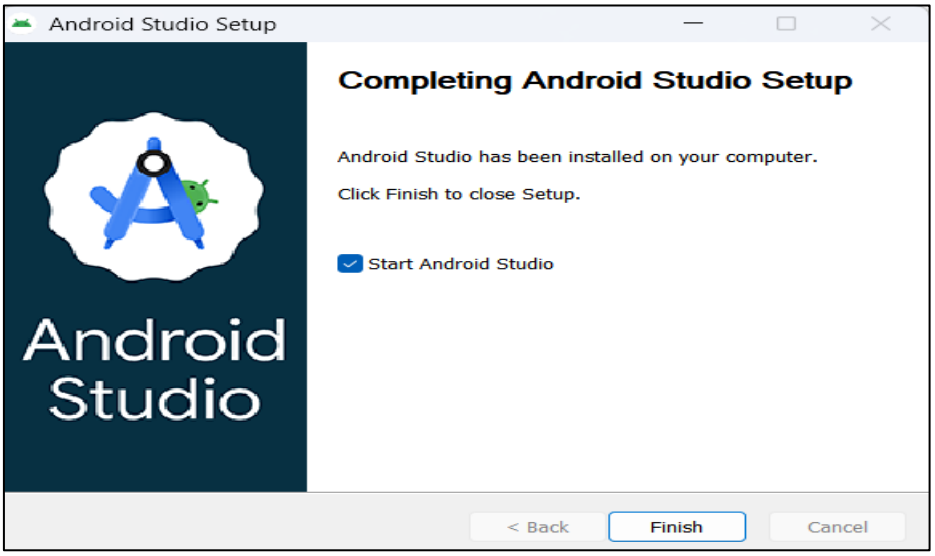


Click on next. In the next prompt, it will ask for a path for installation. Choose a path and hit next.

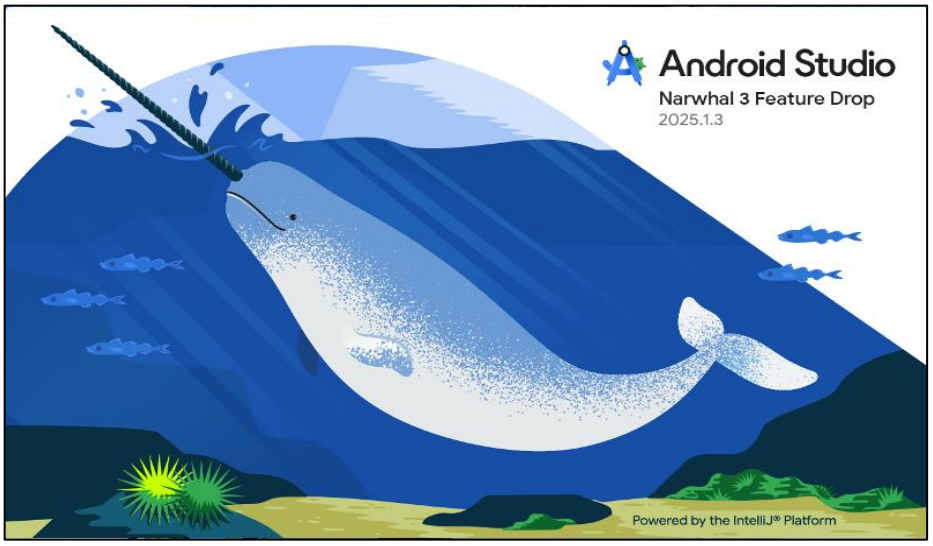
Step 4: It will start the installation, and once it is completed, it will be like the image shown below. Click on next.



Step 5: Once "**Finish**" is clicked.

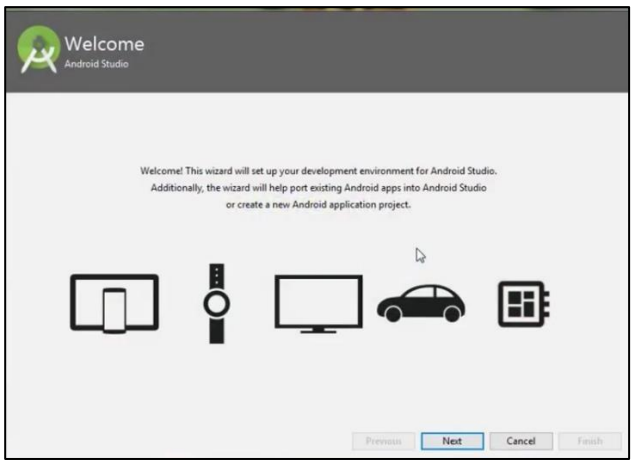


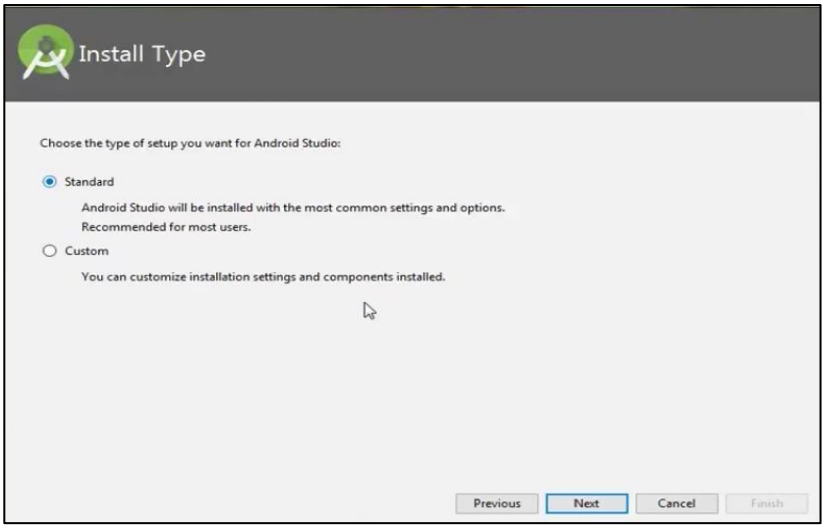
Step 6: This will start the Android Studio.



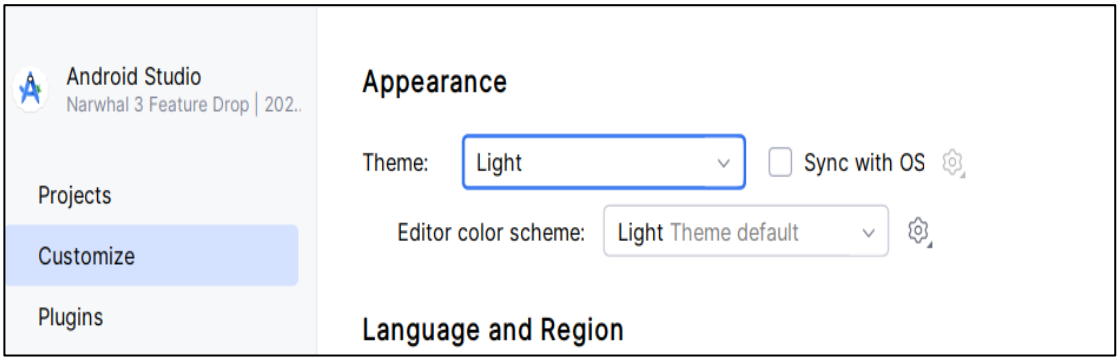
Meanwhile, it will be finding the available SDK components.

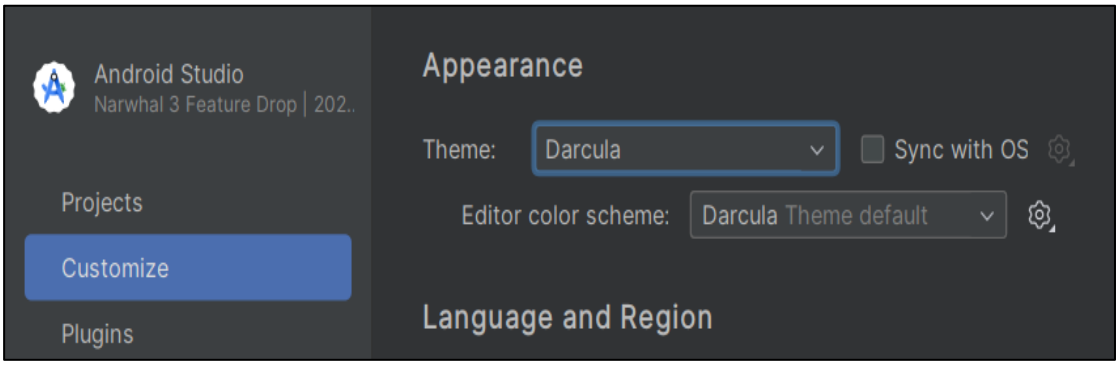
Step 7: After it has found the SDK components, it will redirect to the Welcome dialog box. Click on **Next**.





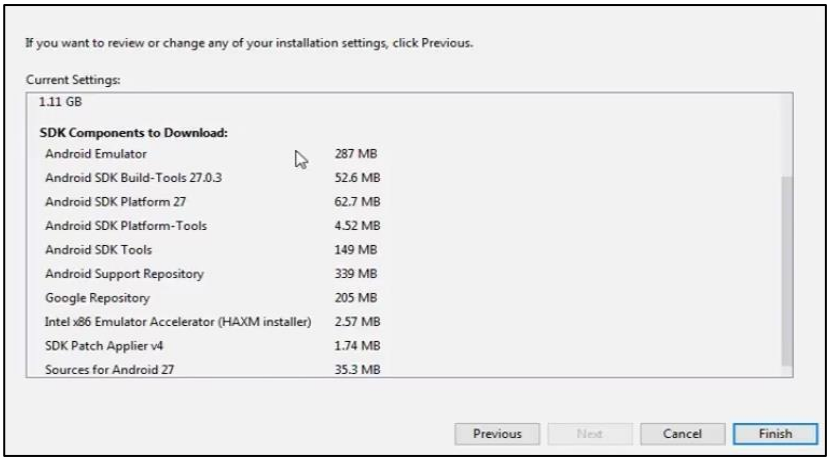
Choose Standard and click on Next. Now choose the theme, whether the **Light** theme or the **Dark** one. The light one is called
the **IntelliJ** theme whereas the dark theme is called **Dracula**. Choose as required.





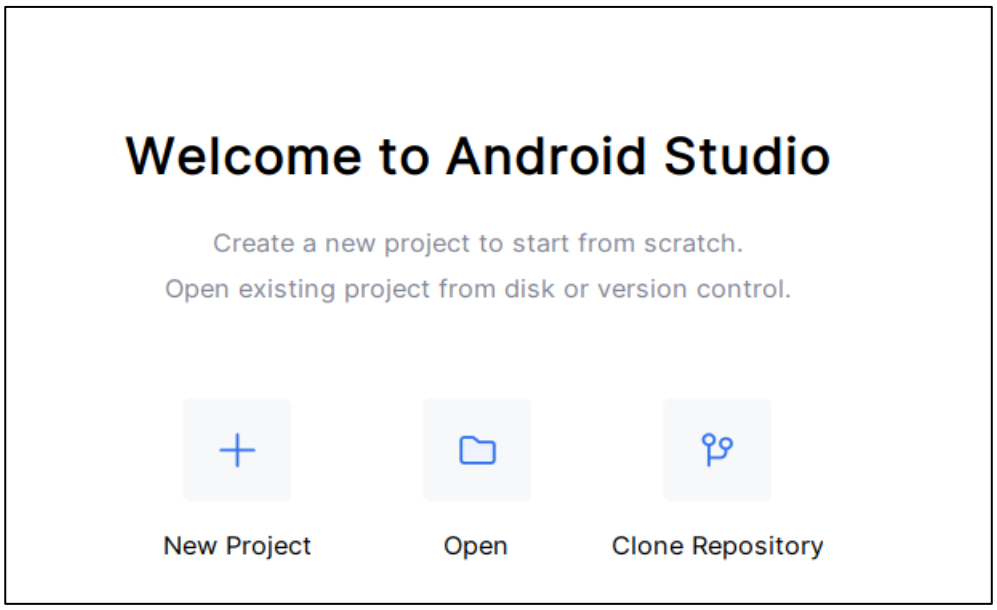
Step 8: Now it is time to download the SDK components.

Click on Finish. Components begin to download let it complete.



The Android Studio has been successfully configured. Click on the Finish button to launch it.

Step 9: Click on **Start a new Android Studio project** to build a new app.



Practical 2

Aim- Study of Various Android Development Environment (IDE).

Studying various Android Development Environments (IDEs) can help you choose the best tool for your Android app development projects. Here's a concise overview of some popular Android IDEs:

1. Android Studio:

- Official IDE for Android development, based on IntelliJ IDEA.
- Offers a rich set of tools, including a visual layout editor, code analysis, and performance profiling.
- Supports Java and Kotlin for app development.
- Provides easy integration with Android SDK and emulators.

2. Eclipse with ADT (Android Development Tools):

- Previously a popular choice but is now less recommended since official support ended.
- Supports Java for app development.
- Requires installing the ADT plugin for Android development.

3. Visual Studio with Xamarin:

- Allows cross-platform development using C# and . NET.
- Xamarin. Forms enable sharing code across Android, iOS, and Windows.
- Offers a visual designer for UI creation.

4. IntelliJ IDEA with Kotlin:

- IntelliJ IDEA is a powerful IDE, and Kotlin is a concise and expressive language for Android.
- Requires configuring Android support manually.
- Popular among Kotlin enthusiasts.

5. React Native:

- A JavaScript framework for building mobile apps with a focus on native performance.
- Supports developing for Android and iOS simultaneously.
- Uses a single codebase for cross-platform development.

6. Xcode with Flutter:

- Flutter is a UI toolkit by Google for building natively compiled applications for mobile, web, and desktop from a single codebase.
- Supports Android and iOS development.
- Uses the Dart programming language.

7. Cordova/PhoneGap:

- Allows building Android apps using HTML, CSS, and JavaScript.
- Wraps web code into a native container.
- Suitable for hybrid app development.

8. App Inventor:

- A visual programming environment for building Android apps, particularly for beginners and educators.
- Provides a blocks-based approach for app development.
- Simplifies app creation without extensive coding.

9. B4X:

- Cross-platform development using a variant of BASIC language.
- Supports Android, iOS, and other platforms.
- Offers a rapid development environment.

10. Thunkable:

- Provides a visual, drag-and-drop interface for building Android and iOS apps.
- Aimed at non-programmers and beginners.
- Supports app creation without coding.

Choosing the right Android IDE depends on your development background, programming language preference, and project requirements. Android Studio is the recommended choice for most Android developers, especially if you're using Java or Kotlin. However, other IDEs may be more suitable for specific scenarios, such as cross-platform development or visual programming.

Practical 3

Aim-To design an Application in an Android Environment representing a Simple Calculator.

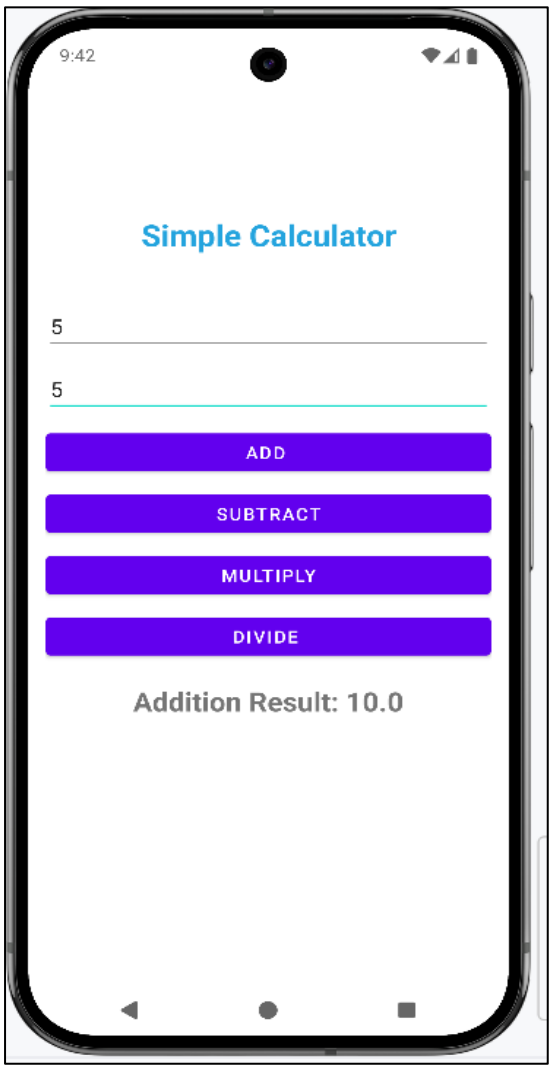
Java code-

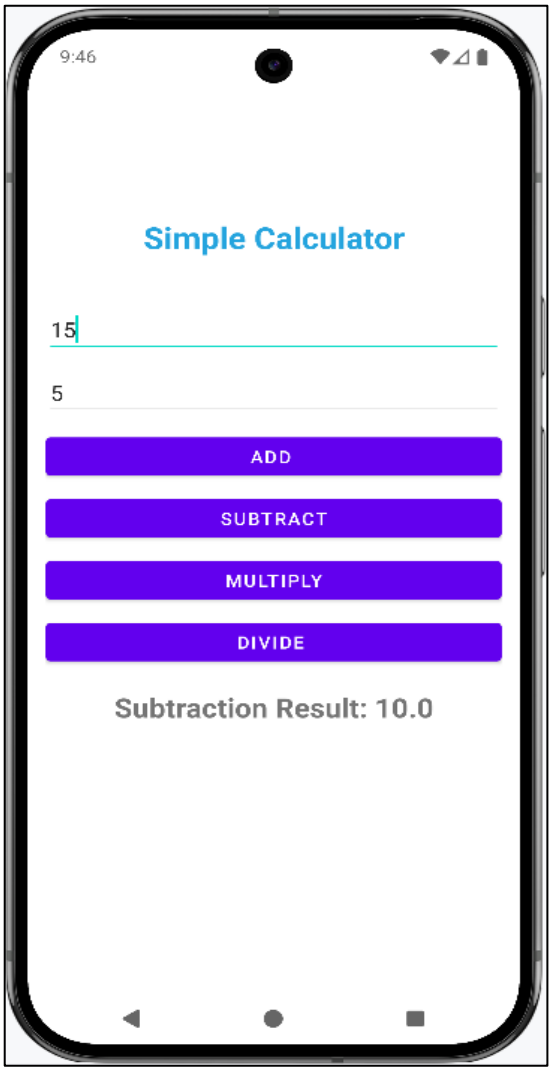
```
package com.example.simplecalculator;

import androidx.appcompat.app.AppCompatActivity;
import android.graphics.Color;
import android.os.Build;
import android.os.Bundle;
import android.view.View;
import android.widget.*;

public class MainActivity extends AppCompatActivity {

    private EditText etNumber1, etNumber2;
    private Button btnAdd, btnSubtract, btnMultiply, btnDivide;
    private TextView tvResult;

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

        // Set custom Status Bar background and icons color
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            getWindow().setStatusBarColor(Color.parseColor("#FF5722"));  // Orange background

getWindow().getDecorView().setSystemUiVisibility(View.SYSTEM_UI_FLAG_LIGHT_STATUS_BAR);
        }

        // Initialize Views
        etNumber1 = findViewById(R.id.etNumber1);
        etNumber2 = findViewById(R.id.etNumber2);
        btnAdd = findViewById(R.id.btnAdd);
        btnSubtract = findViewById(R.id.btnSubtract);
        btnMultiply = findViewById(R.id.btnMultiply);
        btnDivide = findViewById(R.id.btnDivide);
        tvResult = findViewById(R.id.tvResult);

        // Set button click listeners
```

```
    btnAdd.setOnClickListener(view -> performOperation("+"));
    btnSubtract.setOnClickListener(view -> performOperation("-"));
    btnMultiply.setOnClickListener(view -> performOperation("*"));
    btnDivide.setOnClickListener(view -> performOperation("/"));
}

private void performOperation(String operator) {
    String num1Str = etNumber1.getText().toString().trim();
    String num2Str = etNumber2.getText().toString().trim();

    if (num1Str.isEmpty() || num2Str.isEmpty()) {
        Toast.makeText(this, "Please enter both numbers", Toast.LENGTH_SHORT).show();
        return;
    }

    double num1, num2;

    try {
        num1 = Double.parseDouble(num1Str);
        num2 = Double.parseDouble(num2Str);
    } catch (NumberFormatException e) {
        Toast.makeText(this, "Invalid input! Enter valid numbers", Toast.LENGTH_SHORT).show();
        return;
    }

    double result = 0;
    String operationText = "";

    switch (operator) {
        case "+":
            result = num1 + num2;
            operationText = "Addition";
            break;

        case "-":
            result = num1 - num2;
            operationText = "Subtraction";
            break;

        case "*":
            result = num1 * num2;
            operationText = "Multiplication";
            break;

        case "/":
            if (num2 == 0) {
                Toast.makeText(this, "Cannot divide by zero", Toast.LENGTH_SHORT).show();
```

```
                return;
            }
            result = num1 / num2;
            operationText = "Division";
            break;
        }

        tvResult.setText(operationText + " Result: " + result);
    }
}
```

OUTPUT-





Practical 4

Aim- Develop an application for working with Menus and screen Navigation.

Step 1: Create a New Android Studio Project. Create an empty activity android studio project.

Step 2: Adding a dependency to the project.

Step 3: Create the menu folder under the res folder by the name navigation_menu. **res>>android resource directory>>menu**. Now click on **menu>>menu>>menu resource file.**

Invoke the following code in the **activity_main_drawer.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    tools:showIn="navigation_view">

    <group android:checkableBehavior="single">
        <item
            android:id="@+id/nav_home"
            android:icon="@drawable/ic_menu_camera"
            android:title="@string/menu_home" />
        <item
            android:id="@+id/nav_gallery"
            android:icon="@drawable/ic_menu_gallery"
            android:title="@string/menu_gallery" />
        <item
            android:id="@+id/nav_slideshow"
            android:icon="@drawable/ic_menu_slideshow"
            android:title="@string/menu_slideshow" />
    </group>
</menu>
```

Step 4: Invoke the following code in the activity_main.xml to set up the basic things required for the Navigation Drawer.

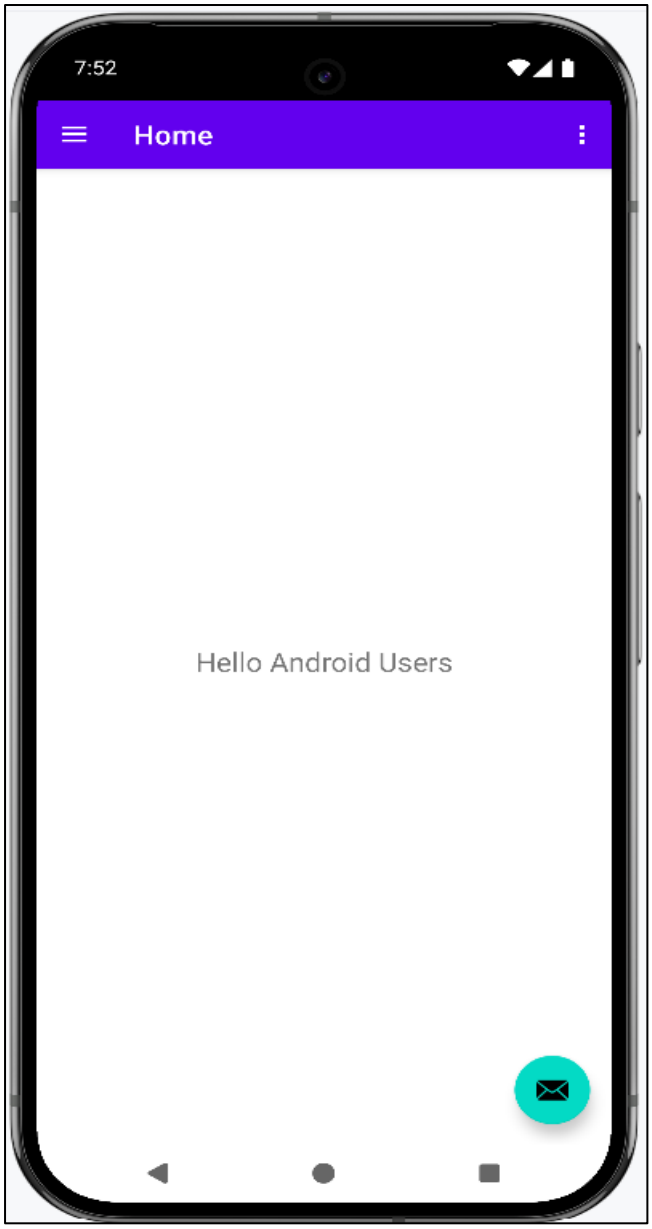
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.drawerlayout.widget.DrawerLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
```

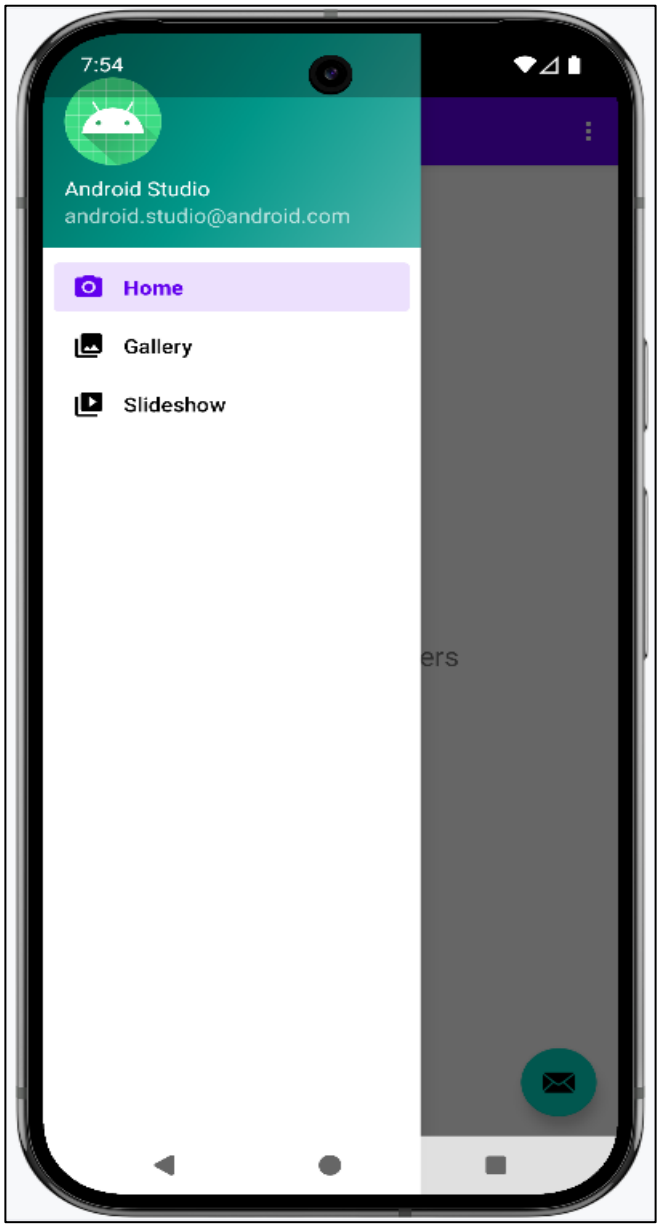
```
    tools:openDrawer="start">

    <include
        android:id="@+id/app_bar_main"
        layout="@layout/app_bar_main"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <com.google.android.material.navigation.NavigationView
        android:id="@+id/nav_view"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="start"
        android:fitsSystemWindows="true"
        app:headerLayout="@layout/nav_header_main"
        app:menu="@menu/activity_main_drawer" />
</androidx.drawerlayout.widget.DrawerLayout>
```

Step 5: Include the Open Close strings in the string.xml Invoke the following code in the app/res/values/strings.xml file.

```
<resources>
    <string name="app_name">My Application</string>
    <string name="navigation_drawer_open">Open navigation drawer</string>
    <string name="navigation_drawer_close">Close navigation drawer</string>
    <string name="nav_header_title">Android Studio</string>
    <string name="nav_header_subtitle">android.studio@android.com</string>
    <string name="nav_header_desc">Navigation header</string>
    <string name="action_settings">Settings</string>

    <string name="menu_home">Home</string>
    <string name="menu_gallery">Gallery</string>
    <string name="menu_slideshow">Slideshow</string>
</resources>
```

Output-


7:52
Home
Hello Android Users


7:54
Android Studio
android.studio@android.com
Home
Gallery
Slideshow

Practical 5

Aim- Develop an application demonstrating internal storage to store private data on the device memory.

XML code-

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="vertical"
   android:padding="24dp"
   android:layout_width="match_parent"
   android:layout_height="match_parent">

   <EditText
      android:id="@+id/etData"
      android:hint="Enter private data"
      android:layout_width="match_parent"
      android:layout_marginTop="120dp"
      android:layout_height="wrap_content"/>

   <Button
      android:id="@+id/btnSave"
      android:text="Save Data"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_marginTop="16dp"/>

   <Button
      android:id="@+id/btnLoad"
      android:text="Load Data"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_marginTop="8dp"/>

   <TextView
      android:id="@+id/tvResult"
      android:text="Data will appear here"
      android:textSize="16sp"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:layout_marginTop="24dp"/>
</LinearLayout>
```

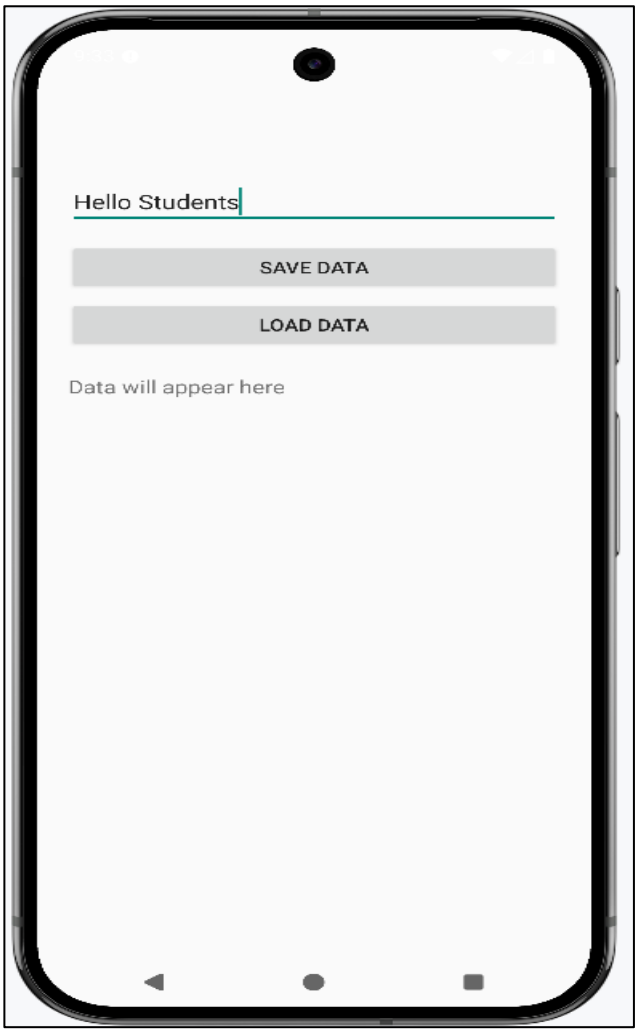
Java code-

```
package com.example.internalstorageapp

import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity
import android.widget.Button
import android.widget.EditText
import android.widget.TextView
import java.io.*

class MainActivity : AppCompatActivity() {

    private lateinit var etData: EditText
    private lateinit var btnSave: Button
    private lateinit var btnLoad: Button
    private lateinit var tvResult: TextView

    private val fileName = "private_data.txt"

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        etData = findViewById(R.id.etData)
        btnSave = findViewById(R.id.btnSave)
        btnLoad = findViewById(R.id.btnLoad)
        tvResult = findViewById(R.id.tvResult)

        btnSave.setOnClickListener { saveData() }
        btnLoad.setOnClickListener { loadData() }
    }

    private fun saveData() {
        val data = etData.text.toString()

        try {
            openFileOutput(fileName, MODE_PRIVATE).use { output ->
                output.write(data.toByteArray())
            }
            tvResult.text = "Data saved successfully."
        } catch (e: IOException) {
            e.printStackTrace()
            tvResult.text = "Error saving data."
        }
    }
```

```
    private fun loadData() {
        try {
            val fileInput = openFileInput(fileName)
            val data = fileInput.bufferedReader().useLines { lines ->
                lines.joinToString("\n")
            }
            tvResult.text = "Loaded Data:\n$data"
        } catch (e: FileNotFoundException) {
            e.printStackTrace()
            tvResult.text = "No data found."
        } catch (e: IOException) {
            e.printStackTrace()
            tvResult.text = "Error reading data."
        }
    }
}
```

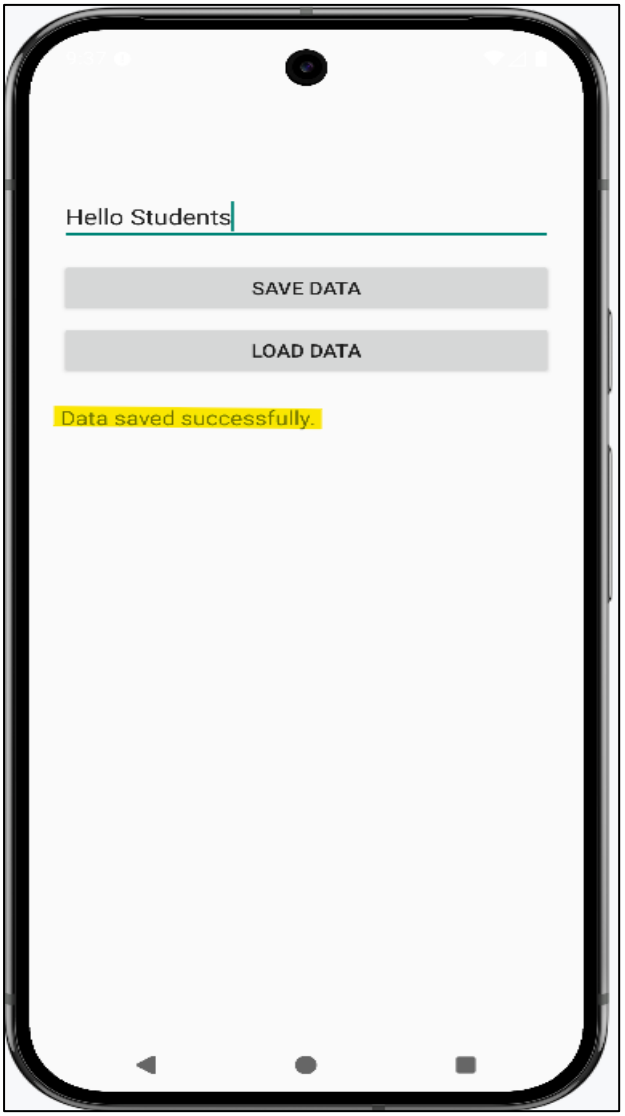
Output-

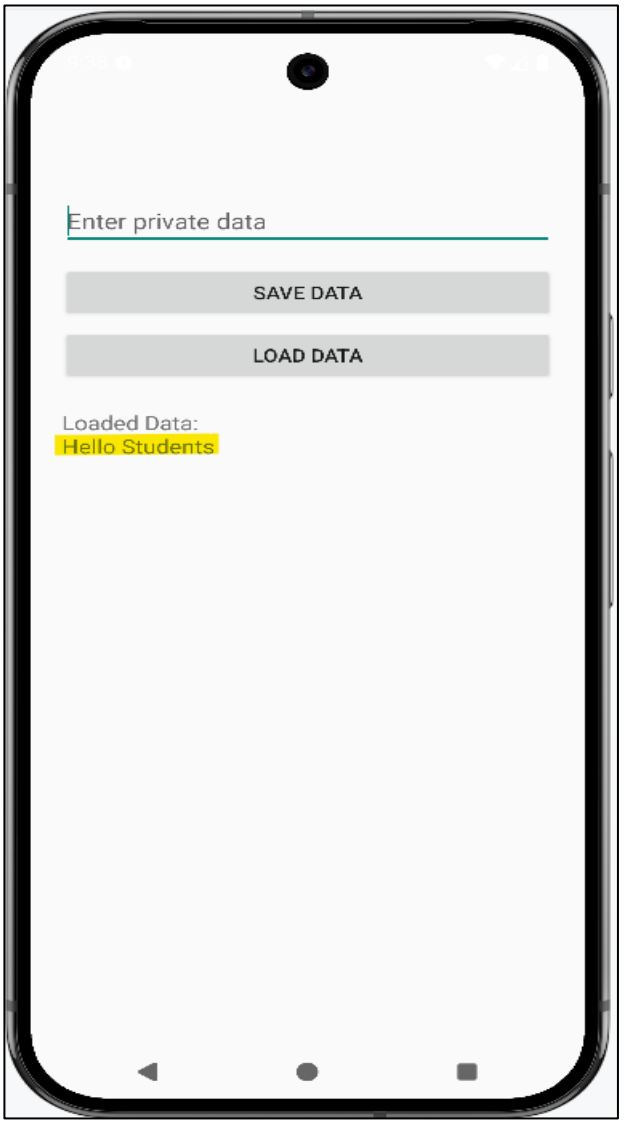
It will show this screen as output now enter any text to here.



After writing any text **press save data**, and that text will be saved in internal memory.

If you want to see what text, you have written then **press load data** and it will show previously written text.





Practical 6

Aim- Design a simple to-do list application using SQLite.

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">

   <ListView
      android:id="@+id/lvTasks"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:layout_above="@+id/fabAdd"/>

   <com.google.android.material.floatingactionbutton.FloatingActionButton
      android:id="@+id/fabAdd"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_alignParentBottom="true"
      android:layout_alignParentEnd="true"
      android:layout_margin="150dp"
      android:layout_marginTop="150dp"
      android:layout_gravity="center_horizontal"
      android:contentDescription="Add Task"
      android:src="@android:drawable/ic_input_add"/>
</RelativeLayout>
```

task_item.xml-

```
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="horizontal"
   android:padding="10dp"
   android:layout_width="match_parent"
   android:layout_height="wrap_content">

   <TextView
      android:id="@+id/tvTask"
      android:layout_width="0dp"
      android:layout_weight="1"
      android:textSize="18sp"
      android:layout_height="wrap_content"/>
```

```
    <Button
        android:id="@+id/btnComplete"
        android:text="Complete"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>

</LinearLayout>
```

MainActivity.kt-

```
package com.example.todolistapp

import android.app.AlertDialog
import android.os.Bundle
import android.widget.*
import androidx.appcompat.app.AppCompatActivity
import com.google.android.material.floatingactionbutton.FloatingActionButton

class MainActivity : AppCompatActivity() {

    private lateinit var lvTasks: ListView
    private lateinit var fabAdd: FloatingActionButton
    private lateinit var dbHelper: DBHelper
    private lateinit var taskList: ArrayList<String>
    private lateinit var adapter: ArrayAdapter<String>

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        lvTasks = findViewById(R.id.lvTasks)
        fabAdd = findViewById(R.id.fabAdd)
        dbHelper = DBHelper(this)
        taskList = dbHelper.getAllTasks()

        adapter = object : ArrayAdapter<String>(this, R.layout.task_item, R.id.tvTask, taskList) {
            override fun getView(position: Int, convertView: android.view.View?, parent:
android.view.ViewGroup): android.view.View {
                val view = super.getView(position, convertView, parent)
                val btnComplete = view.findViewById<Button>(R.id.btnComplete)
                btnComplete.setOnClickListener {
                    val task = taskList[position]
                    dbHelper.deleteTask(task)
                    taskList.removeAt(position)
                    notifyDataSetChanged()
```

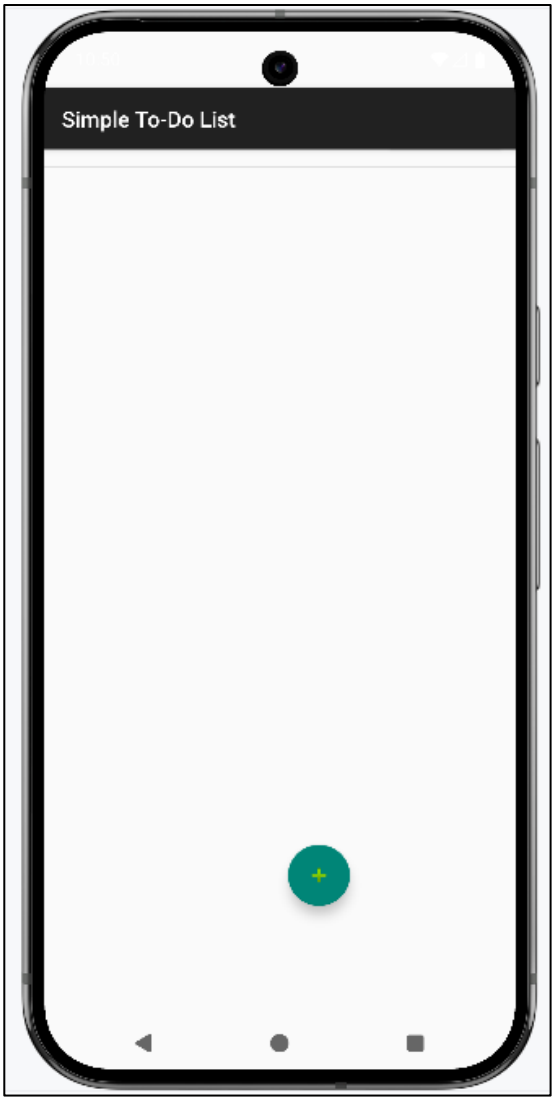
```
                Toast.makeText(this@MainActivity, "Task Completed", Toast.LENGTH_SHORT).show()
            }
            return view
        }
    }

    lvTasks.adapter = adapter

    fabAdd.setOnClickListener {
        showAddTaskDialog()
    }
}

private fun showAddTaskDialog() {
    val et = EditText(this)
    et.hint = "Describe the Todo task..."

    AlertDialog.Builder(this)
        .setTitle("Add Todo Task Item")
        .setView(et)
        .setPositiveButton("Add Task") { dialog, _ ->
            val task = et.text.toString()
            if (task.isNotEmpty()) {
                dbHelper.insertTask(task)
                taskList.add(task)
                adapter.notifyDataSetChanged()
                dialog.dismiss()
                Toast.makeText(this, "Task Added", Toast.LENGTH_SHORT).show()
            }
        }
        .setNegativeButton("Cancel", null)
        .show()
}
}
```

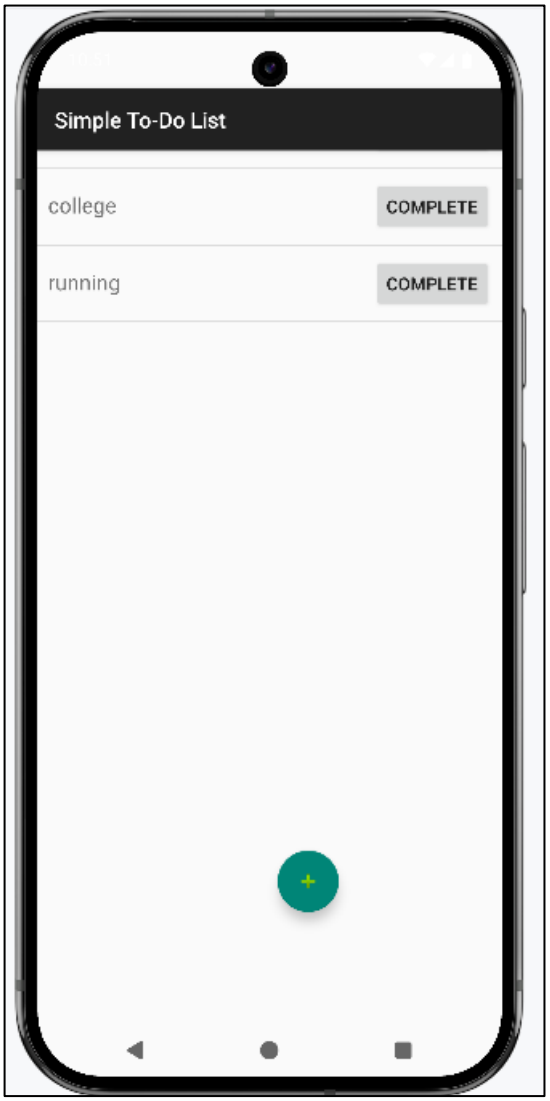
DBHelper.kt-

```
package com.example.todolistapp

import android.content.ContentValues
import android.content.Context
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteOpenHelper

class DBHelper(context: Context): SQLiteOpenHelper(context, "TodoDB", null, 1) {
```

```
    override fun onCreate(db: SQLiteDatabase?) {
        db?.execSQL("CREATE TABLE Tasks (id INTEGER PRIMARY KEY AUTOINCREMENT, task
TEXT)")
    }

    override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int, newVersion: Int) {
        db?.execSQL("DROP TABLE IF EXISTS Tasks")
        onCreate(db)
    }

    fun insertTask(task: String): Boolean {
        val db = writableDatabase
        val cv = ContentValues()
        cv.put("task", task)
        return db.insert("Tasks", null, cv) != -1L
    }

    fun deleteTask(task: String) {
        val db = writableDatabase
        db.delete("Tasks", "task=?", arrayOf(task))
    }

    fun getAllTasks(): ArrayList<String> {
        val list = ArrayList<String>()
        val cursor = readableDatabase.rawQuery("SELECT * FROM Tasks", null)
        if (cursor.moveToFirst()) {
            do {
                val task = cursor.getString(cursor.getColumnIndexOrThrow("task"))
                list.add(task)
            } while (cursor.moveToNext())
        }
        cursor.close()
        return list
    }
}
```

Output-


Simple To-Do List


Simple To-Do List
college
COMPLETE
running
COMPLETE

Practical 7

Aim- Develop an application for connecting to the internet and sending email.

Step by Step Implementation-

Step 1: Create a New Project in Android Studio.

Step 2: Next, create **activity_main.xml** file, below is the code for the activity_main.xml file :-

```
<?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"
   android:padding="16dp">

   <EditText
      android:id="@+id/etRecipient"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="Recipient Email"
      android:inputType="textEmailAddress"
      android:layout_marginBottom="12dp"
      android:layout_marginTop="80dp"/>

   <EditText
      android:id="@+id/etSubject"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="Subject"
      android:inputType="text"
      android:layout_marginBottom="12dp"/>

   <EditText
      android:id="@+id/etMessage"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="Message"
      android:inputType="textMultiLine"
      android:lines="5"
      android:gravity="top"
      android:layout_marginBottom="12dp"/>
```

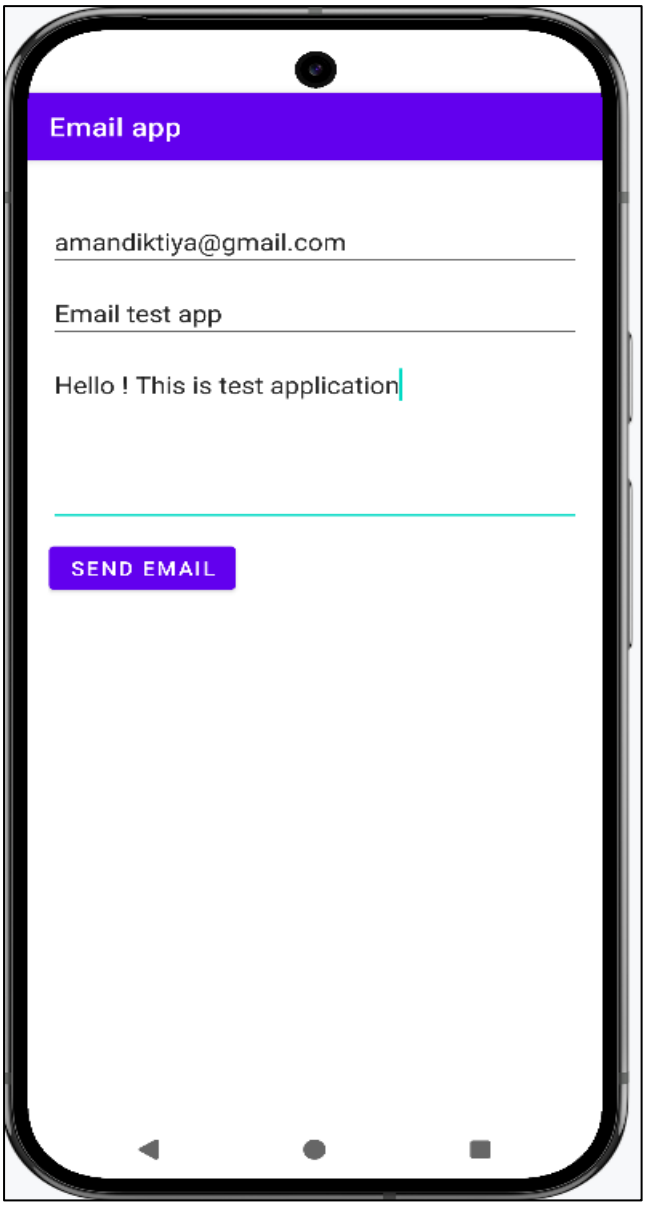
```
    <Button
        android:id="@+id/btnSendEmail"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Send Email"/>
</LinearLayout>
```

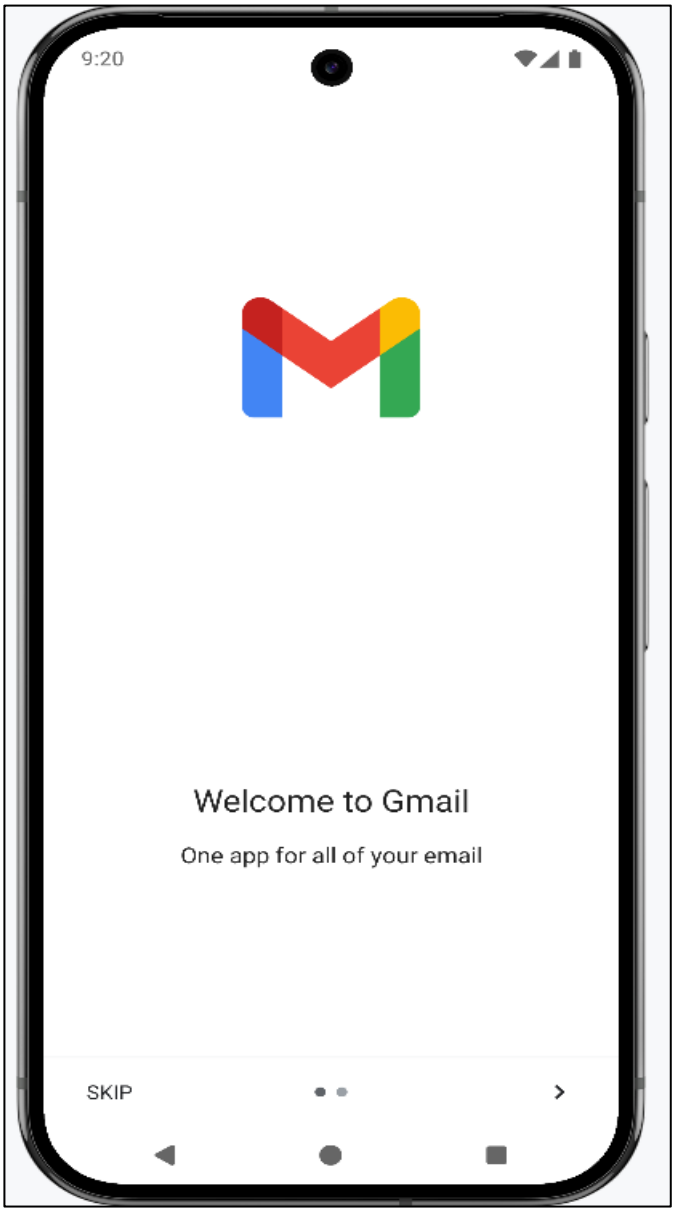
Java code-

```
package com.example.emailapp;

import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.Network;
import android.net.NetworkCapabilities;
import android.net.Uri;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    private EditText etRecipient, etSubject, etMessage;
    private Button btnSendEmail;

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

        // Initialize views
        etRecipient = findViewById(R.id.etRecipient);
        etSubject = findViewById(R.id.etSubject);
        etMessage = findViewById(R.id.etMessage);
        btnSendEmail = findViewById(R.id.btnSendEmail);

        btnSendEmail.setOnClickListener(v -> sendEmail());
    }

    private void sendEmail() {
        // Input validation
```

```
        String recipient = etRecipient.getText().toString().trim();
        String subject = etSubject.getText().toString().trim();
        String message = etMessage.getText().toString().trim();

        if (recipient.isEmpty()) {
            Toast.makeText(this, "Please enter recipient email", Toast.LENGTH_SHORT).show();
            return;
        }

        if (!isNetworkAvailable()) {
            Toast.makeText(this, "No internet connection", Toast.LENGTH_SHORT).show();
            return;
        }

        Intent emailIntent = new Intent(Intent.ACTION_SENDTO);
        emailIntent.setData(Uri.parse("mailto:" + recipient));
        emailIntent.putExtra(Intent.EXTRA_SUBJECT, subject);
        emailIntent.putExtra(Intent.EXTRA_TEXT, message);

        try {
            startActivity(Intent.createChooser(emailIntent, "Choose Email Client"));
        } catch (Exception e) {
            Toast.makeText(this, "No email client installed", Toast.LENGTH_SHORT).show();
            e.printStackTrace();
        }
    }

    private boolean isNetworkAvailable() {
        ConnectivityManager cm = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
        if (cm != null) {
            Network network = cm.getActiveNetwork();
            if (network != null) {
                NetworkCapabilities capabilities = cm.getNetworkCapabilities(network);
                return capabilities != null &&
                        (capabilities.hasTransport(NetworkCapabilities.TRANSPORT_WIFI) ||
                                capabilities.hasTransport(NetworkCapabilities.TRANSPORT_CELLULAR) ||
                                capabilities.hasTransport(NetworkCapabilities.TRANSPORT_ETHERNET));
            }
        }
        return false;
    }
}
```

Output-Firstly, write the email address to which you want to send email, its subject and body and **press send email!!** Button.





Sign up or Sign in to Gmail Account and continue further process.

Practical 8

Aim- Develop an application for working with graphics and animations.

Step 1: Create a New Project

Step 2: Working with the strings.xml file.

Strings.xml can be found from the **app > res > values > strings.xml.**

```
<resources>
   <string name="app_name">GFG App</string>
   <string name="blink">BLINK</string>
   <string name="clockwise">ROTATE</string>
   <string name="fade">FADE</string>
   <string name="move">MOVE</string>
   <string name="slide">SLIDE</string>
   <string name="zoom">ZOOM</string>
   <string name="stop_animation">STOP ANIMATION</string>
   <string name="course_rating">Course Rating</string>
   <string name="course_name">Course Name</string>
</resources>
```

Step 3: Create ImageView in the activity_main.xml along with buttons that will add animation to the view. Navigate to the app > res > layout > **activity_main.xml**

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:padding="20dp">

   <!-- Plant Image -->
   <ImageView
      android:id="@+id/imageView"
      android:layout_width="500dp"
      android:layout_height="300dp"
      android:src="@drawable/plant"
      android:layout_centerHorizontal="true"
      android:layout_marginTop="90dp" />

   <!-- 1st row of buttons -->
   <LinearLayout
      android:id="@+id/row1"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:orientation="horizontal"
      android:gravity="center"
      android:layout_below="@id/imageView"
      android:layout_marginTop="250dp">
```

```
<!-- BLINK Button at last -->

      <Button
         android:id="@+id/btnBlink"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="BLINK"
         android:backgroundTint="@android:color/holo_green_light"
         android:layout_margin="10dp"/>

<!—ROTATE Button at last -->
      <Button
         android:id="@+id/btnRotate"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="ROTATE"
         android:backgroundTint="@android:color/holo_green_light"
         android:layout_margin="10dp"/>

<!--  FADE Button at last -->
      <Button
         android:id="@+id/btnFade"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="FADE"
         android:backgroundTint="@android:color/holo_green_light"
         android:layout_margin="10dp"/>
   </LinearLayout>

   <!-- 2nd row of buttons -->
   <LinearLayout
      android:id="@+id/row2"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:orientation="horizontal"
      android:gravity="center"
      android:layout_below="@id/row1">

<!-- MOVE Button at last -->
      <Button
         android:id="@+id/btnMove"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="MOVE"
         android:backgroundTint="@android:color/holo_green_light"
         android:layout_margin="10dp"/>

<!—SLIDE Button at last -->
      <Button
         android:id="@+id/btnSlide"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
```

```
        android:text="SLIDE"
        android:backgroundTint="@android:color/holo_green_light"
        android:layout_margin="10dp"/>

<!-- ZOOM Button at last -->
      <Button
        android:id="@+id/btnZoom"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="ZOOM"
        android:backgroundTint="@android:color/holo_green_light"
        android:layout_margin="10dp"/>
   </LinearLayout>

   <!-- STOP Button at last -->
   <Button
      android:id="@+id/btnStop"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:text="STOP ANIMATION"
      android:backgroundTint="@android:color/holo_purple"
      android:textColor="@android:color/white"
      android:layout_below="@id/row2"
      android:layout_marginTop="30dp"/>
</RelativeLayout>
```

Step 5: Create 6 different types of animation for ImageView Navigate to the **app > res > Right-Click on res >> New >> Directory >>** Name your directory as “anim”. Inside this directory, we will create our animations. For creating a new anim right click on the **anim directory >>** Animation Resource file and give the name to your file.

1) Blink Animation

```
        <?xml version="1.0" encoding="utf-8"?>
        <alpha xmlns:android="http://schemas.android.com/apk/res/android"
           android:duration="300"
           android:fromAlpha="0.0"
           android:toAlpha="1.0"
           android:repeatMode="reverse"
           android:repeatCount="infinite"/>
```

2) Fade Animation

```
        <?xml version="1.0" encoding="utf-8"?>
        <alpha xmlns:android="http://schemas.android.com/apk/res/android"
            android:fromAlpha="1.0"
             android:toAlpha="0.0"
             android:duration="2000"
              android:repeatMode="reverse"
              android:repeatCount="infinite"/>
```

3) Move Animation

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

```
<translate xmlns:android="http://schemas.android.com/apk/res/android"
    android:fromXDelta="0%"
    android:toXDelta="75%"
    android:duration="1000"
    android:repeatCount="infinite"
    android:repeatMode="reverse"/>
```

4) Rotate Animation

```
<?xml version="1.0" encoding="utf-8"?>
<rotate xmlns:android="http://schemas.android.com/apk/res/android"
   android:fromDegrees="0"
   android:toDegrees="360"
  android:pivotX="50%"
    android:pivotY="50%"
    android:duration="1000"
    android:repeatCount="infinite"/>
```

5) Slide Animation

```
<?xml version="1.0" encoding="utf-8"?>
<translate xmlns:android="http://schemas.android.com/apk/res/android"
    android:fromYDelta="100%"
    android:toYDelta="0%"
    android:duration="1000"
    android:repeatMode="reverse"
    android:repeatCount="infinite"/>
```

6) Zoom Animation

```
<?xml version="1.0" encoding="utf-8"?>
<scale xmlns:android="http://schemas.android.com/apk/res/android"
    android:fromXScale="1.0"
    android:toXScale="1.5"
    android:fromYScale="1.0"
    android:toYScale="1.5"
    android:pivotX="50%"
    android:pivotY="50%"
    android:duration="1000"
    android:repeatMode="reverse"
    android:repeatCount="infinite"/>
```

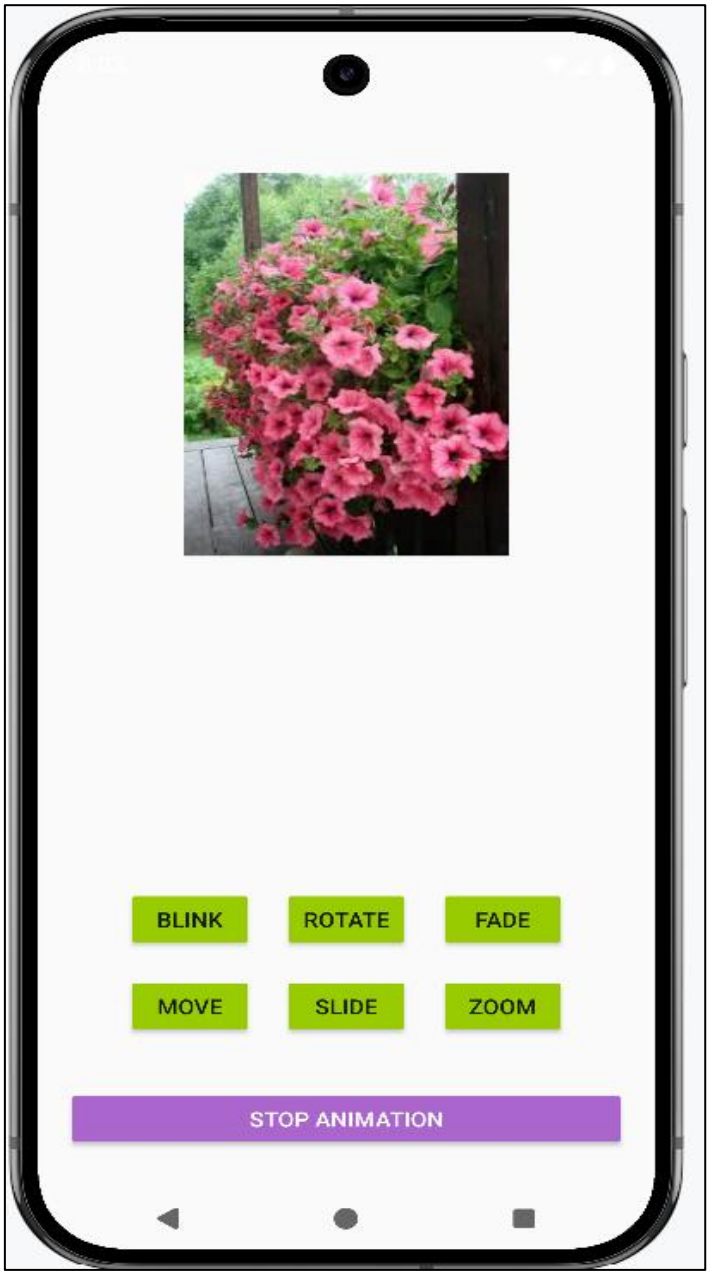
Step 6: Add animation to the ImageView by clicking a specific Button. Navigate to the app > java > your apps package name >> **MainActivity.java**

```
package com.example.graphicsanimationapp;

import android.os.Bundle;
import android.view.animation.Animation;
```

```
import android.view.animation.AnimationUtils;
import android.widget.Button;
import android.widget.ImageView;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    ImageView imageView;
    Button btnBlink, btnRotate, btnFade, btnMove, btnSlide, btnZoom, btnStop;
    Animation anim;

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

        imageView = findViewById(R.id.imageView);

    btnMove = findViewById(R.id.btnMove); // ye wahi ID hogi jo XML me hai
    imageView = findViewById(R.id.imageView);
        btnBlink = findViewById(R.id.btnBlink);
        btnRotate = findViewById(R.id.btnRotate);
        btnFade = findViewById(R.id.btnFade);
        btnSlide = findViewById(R.id.btnSlide);
        btnZoom = findViewById(R.id.btnZoom);
        btnStop = findViewById(R.id.btnStop);

        btnBlink.setOnClickListener(v -> {
            anim = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.blink_animation);
            imageView.startAnimation(anim);
        });

        btnRotate.setOnClickListener(v -> {
            anim = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.rotate_animation);
            imageView.startAnimation(anim);
        });

        btnFade.setOnClickListener(v -> {
            anim = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.fade_animation);
            imageView.startAnimation(anim);
        });

        btnSlide.setOnClickListener(v -> {
            anim = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.slide_animation);
            imageView.startAnimation(anim);
        });
```

```
        btnZoom.setOnClickListener(v -> {
            anim = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.zoom_animation);
            imageView.startAnimation(anim);
        });

        btnMove.setOnClickListener(v -> {
            anim = AnimationUtils.loadAnimation(getApplicationContext(), R.anim.move_animation);
            imageView.startAnimation(anim);
        });


        btnStop.setOnClickListener(v -> {
            imageView.clearAnimation();  // Stop any running animation
        });
    }
}
```

Output-



Practical 9

Aim- Develop an application for working with location-based service.

Step 1: Acquiring Permissions. We need to add all these permissions in the **AndroidManifest.xml**

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
  <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
  <uses-permission android:name="android.permission.INTERNET"/>
  <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

Step 2: Designing the layout-

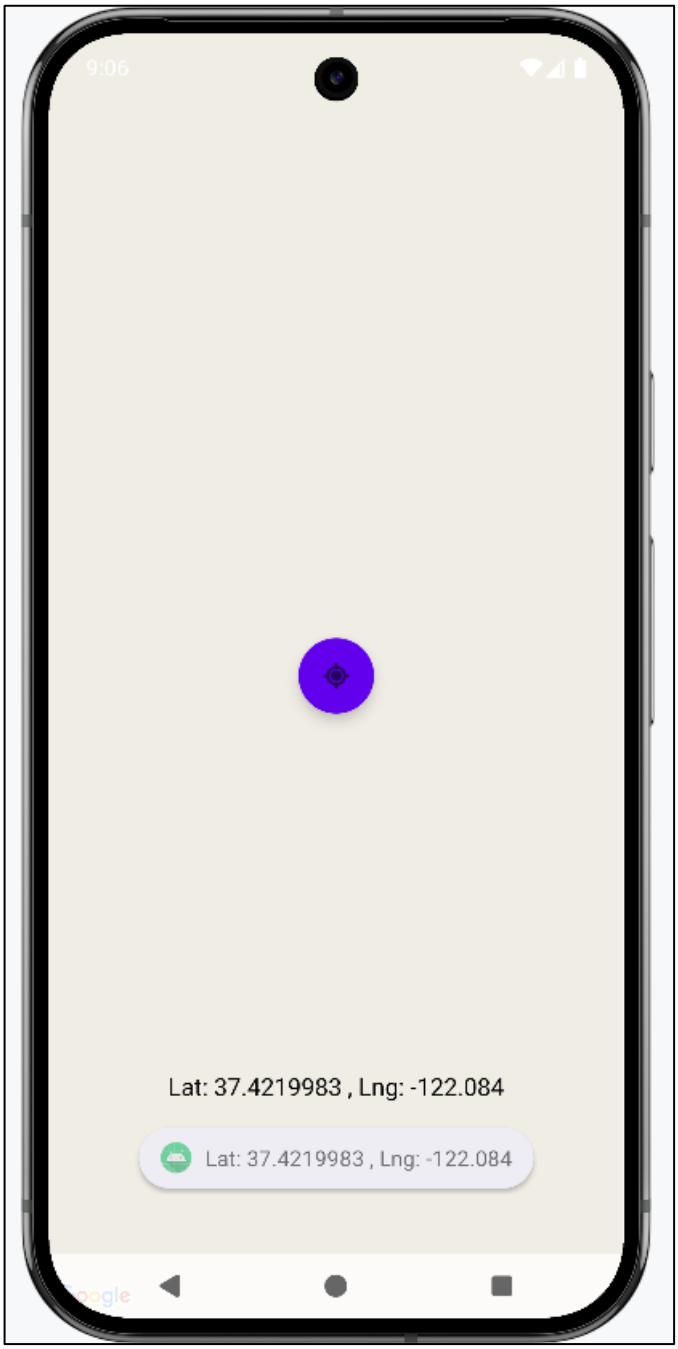
```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  android:layout_width="match_parent"
  android:layout_height="match_parent">

  <!-- Google Map Fragment -->
  <fragment
    android:id="@+id/map"
    android:name="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />

  <TextView
    android:id="@+id/tvLocation"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Location will appear here"
    android:textSize="16sp"
    android:textColor="@android:color/black"
    android:layout_centerHorizontal="true"
    android:layout_alignParentBottom="true"
    android:layout_marginBottom="160dp"/>

  <com.google.android.material.floatingactionbutton.FloatingActionButton
    android:id="@+id/btnLocation"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@android:drawable/ic_menu_mylocation"
    android:contentDescription="Show My Location"
    app:backgroundTint="@color/purple_500"
    android:layout_centerInParent="true" />

</RelativeLayout>
```

Step 3: java code is given below-

```
package com.example.locationapp;
import android.Manifest;
import android.content.pm.PackageManager;
import android.location.Location;
import android.os.Bundle;
import android.widget.TextView;
import android.widget.Toast;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;

import com.google.android.gms.location.FusedLocationProviderClient;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.material.floatingactionbutton.FloatingActionButton;

public class MainActivity extends AppCompatActivity implements OnMapReadyCallback {

    private GoogleMap mMap;
    private FusedLocationProviderClient fusedLocationClient;
    private FloatingActionButton btnLocation;
    private TextView tvLocation;

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

        // Map fragment
        SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
                .findFragmentById(R.id.map);
        if (mapFragment != null) {
            mapFragment.getMapAsync(this);
        }

        fusedLocationClient = LocationServices.getFusedLocationProviderClient(this);
```

```
        btnLocation = findViewById(R.id.btnLocation);
        tvLocation = findViewById(R.id.tvLocation);

        btnLocation.setOnClickListener(v -> getMyLocation());
    }

    @Override
    public void onMapReady(@NonNull GoogleMap googleMap) {
        mMap = googleMap;
    }

    private void getMyLocation() {
        if (ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions(this,
                    new String[]{Manifest.permission.ACCESS_FINE_LOCATION}, 1);
            return;
        }

        fusedLocationClient.getLastLocation().addOnSuccessListener(this, location -> {
            if (location != null && mMap != null) {
                LatLng myLatLng = new LatLng(location.getLatitude(), location.getLongitude());
                mMap.clear();
                mMap.addMarker(new MarkerOptions().position(myLatLng).title("You are here"));
                mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(myLatLng, 16f));

                String msg = "Lat: " + location.getLatitude() + " , Lng: " + location.getLongitude();
                Toast.makeText(this, msg, Toast.LENGTH_LONG).show();
                tvLocation.setText(msg);  /
            }
        });
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull
int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        if (requestCode == 1 && grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
            getMyLocation();
        }
    }
}
```

Output-



Practical 10

Aim- Develop an application for working with device camera.

XML Code-

```
<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:background="#4CAF50" tools:context=".MainActivity">

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_marginBottom="100dp"
        android:layout_centerHorizontal="true" android:text="Take a
        Photo" >
    </Button>

    <ImageView android:id="@+id/imageView1"
        android:layout_width="200dp"
        android:layout_height="match_parent"
        android:layout_above="@+id/button1"
        android:layout_alignParentTop="true"
        android:layout_marginLeft="100dp"
        android:src="@drawable/photo" >
    </ImageView>
</RelativeLayout>
```

Java Code-

```
package com.example.simplecamera; import
android.app.Activity;
import android.content.Intent;
import android.graphics.Bitmap; import
android.os.Bundle; import android.view.Menu;
import android.view.View; import
android.widget.Button;
import android.widget.ImageView;
```

```
public class MainActivity extends Activity {

    private static final int CAMERA_REQUEST = 1888; ImageView imageView;
    public void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        imageView = (ImageView) this.findViewById(R.id.imageView1); Button photoButton
        = (Button) this.findViewById(R.id.button1); photoButton.setOnClickListener(new
        View.OnClickListener() {


        @Override

        public void onClick(View v) { Intent
            cameraIntent = new
Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE);
            startActivityForResult(cameraIntent, CAMERA_REQUEST);
         }

        });

       }

    protected void onActivityResult(int requestCode, int resultCode, Intent data) { if (requestCode ==
     CAMERA_REQUEST) {
      Bitmap photo = (Bitmap) data.getExtras().get("data"); imageView.setImageBitmap(photo);
     }

  }

}
```

Output-