Roll No. 45 Exam Seat No._____

VIVEKANANDEDUCATION SOCIETY'S INSTITUTE OF TECHNOLOGY

HashuAdvani Memorial Complex, Collector's Colony, R. C. Marg,
Chembur, Mumbai – 400074. Contact No. 02261532532

Since 1962

CERTIFICATE

Certified that Mr./Miss Pushkar Prasad Sane

of SY MCA / A has

satisfactorily completed a course of the necessary experiments in

Skill based Lab Mobile Computing Lab under my supervision

in the Institute of Technology in the academic year 2024 - 2025.

Principal Head of Department

Lab In-charge Subject Teacher

V.E.S. Institute of Technology, Collector Colony,
Chembur, Mumbai, Maharashtra 400047
Department of M.C.A

INDEX

Sr. No.	Contents	Date of Preparation	Date of Submission	Marks	Faculty Sign
1	A. Study about Android platform, the layers of android, Four kinds of android components, understanding the android Manifest.xml file, Android Activity Life cycle. B. Android program using various UI Components. (EditText, RatingBar, CheckBox, RadioButton, Button etc). Get the values from the components and display them on the next activity. C. Create a login form. For a successful login, display the welcome page, and in case of failure display an alert box indicating an error message and attempts made. Disable the submit button after 3 wrong attempts and display the alert message.	18-08-2024	24-08-2024		

2	To study different types of layouts and Toast class.	04-09-2024	05-09-2024		
3	Android program based on intent.	05-09-2024	07-10-2024		
4	Android program to perform CRUD operation using SQLite DB (create table students with fields roll no, name, email-Id, course, perform add, update and delete record operations).	06-10-2024	07-10-2024		
5	To implement file I/O and Shared Preferences.	04-10-2024	05-10-2024		
6	To perform the animation on an image and to apply various filters on an image.	10-09-2024	11-10-2024		
7	Android program to work with google maps and location and GPS.	17-09-2024	19-10-2024		
8	To Write a program to record and play audio and video.	20-10-2024	20-10-2024		
9	Android program based on Rest API.	24-10-2024	25-10-2024		
10	Flutter program using layout, widget and state management.	26-10-2024	26-10-2024		

Final Grade	**Instructor Signature**

Name of Student: Pushkar Sane		
Roll Number: 45	**Lab Assignment Number: 1A**	
Title of Lab Assignment: Study about Android platform, the layers of android, Four kinds of android components, understanding the android Manifest.xml file, Android Activity Life cycle.		
DOP: 18-08-2024	**DOS: 24-08-2024**	
CO Mapped: **CO1**	**PO Mapped:** **PO3, PO5, PO7, PO12, PSO1, PSO2**	**Signature:**

Practical 1A

Aim: Study about Android platform, the layers of android, Four kinds of android components, understanding the android Manifest.xml file, Android Activity Life cycle.

Theory:

Android is an open source and Linux-based operating system for mobile devices such as smartphones and tablet computers. Android was developed by the Open Handset Alliance, led by Google, and other companies.

Android programming is based on Java programming language so if you have basic understanding of Java programming then it will be easier to learn Android application development.

Layers: Android operating system is a stack of software components which is roughly divided into five sections and four main layers as shown below in the architecture diagram.

XMPP Service-Extensible Messaging and Presence Protocol is an open communication protocol designed for instant messaging, presence information, and contact list maintenance.

Surface manager responsible for managing access to the display subsystem. SGL(Scalable Graphics Library) and OpenGL ES both cross-language, cross-platform application program interface (API) are used for 2D and 3D computer graphics. SQLite provides database support. FreeType provides font support.

Web-Kit This open source web browser engine provides all the functionality to display web content and to simplify page loading.

SSL (Secure Sockets Layer) is security technology to establish an encrypted link between a web server and a web browser.

Like Java Virtual Machine (JVM), Dalvik Virtual Machine (DVM) is a register-based virtual machine and specially designed and optimized for android to ensure that a device can run multiple instances efficiently.

Four kinds of android components

Activities: An activity represents a single screen with a user interface,in-short Activity performs action on the screen. For example, an email application might have one activity that shows a list of new emails, another activity to compose an email, and another activity for reading emails. If an application has more than one activity, then one of them should be marked as the activity that is presented when the application is launched.

An activity is implemented as a subclass of Activity class as follows:

```
public class MainActivity extends Activity { }
```

Services: A service is a component that runs in the background to perform long-running operations. For example, a service might play music in the background while the user is in a different application, or it might fetch data over the network without blocking user interaction with an activity.

A service is implemented as a subclass of Service class as follows:

```
public class MyService extends Service { }
```

Broadcast: Receivers Broadcast Receivers simply respond to broadcast messages from other applications or from the system. For example, applications can also initiate broadcasts to let other applications know that some data has been downloaded to the device and is available for them to use, so this is the broadcast receiver who will intercept this communication and will initiate appropriate action. A broadcast receiver is implemented as a subclass of BroadcastReceiverclass and each message is broadcaster as an Intent object.

```
public class MyReceiver extends BroadcastReceiver {
        public void onReceive(context, intent){ }
}
```

Content Providers: A content provider component supplies data from one application to others on request. Such requests are handled by the methods of the ContentResolverclass. The data may be stored in the file system, the database or somewhere else entirely. A content provider is implemented as a subclass of ContentProvider class and must implement a standard set of APIs that enable other applications to perform transactions.

```
public class MyContentProvider extends ContentProvider {
        public void onCreate(){ }
}
```

Sr. No.	Components
1	**Activities**: They dictate the UI and handle the user interaction to the smartphone screen.
2	**Services**: They handle background processing associated with an application.
3	**Broadcast Receivers**: They handle communication between Android OS and applications.
4	**Content Providers**: They handle data and database management issues.

Understanding the android Manifest.xml file

Whatever component you develop as a part of your application, you must declare all its components in a manifest.xml which resides at the root of the application project directory. This file works as an interface between Android OS and your application, so if you do not declare your component in this file, then it will not be considered by the OS. For example, a default manifest file will look like as following file −

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

Here <application>...</application> tags enclosed the components related to the application. Attribute android:icon will point to the application icon available under res/drawable-hdpi. The application uses the image named ic_launcher.png located in the drawable folders.

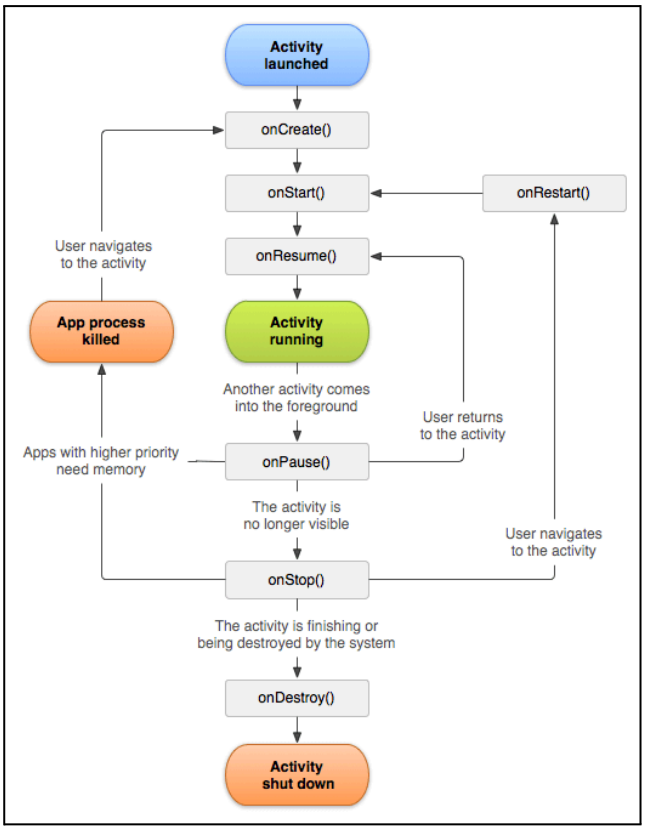
The <activity> tag is used to specify an activity and android:name attribute specifies the fully qualified class name of the Activity subclass and the android:label attributes specifies a string to use as the label for the activity. You can specify multiple activities using <activity> tags.

The action for the intent filter is named android.intent.action.MAIN to indicate that this activity serves as the entry point for the application. The category for the intent-filter is named android.intent.category.LAUNCHER to indicate that the application can be launched from the device's launcher icon.

The @string refers to the strings.xml file explained below. Hence, @string/app_name refers to the app_name string defined in the strings.xml file, which is "HelloWorld". Similarly, other strings get populated in the application.e that the application can be launched from the device's launcher icon.

Android Activity Life cycle.

An activity represents a single screen with a user interface just like a window or frame of Java. Android activity is the subclass of ContextThemeWrapper class.



The Activity class defines the following call backs i.e. events. You don't need to implement all the callbacks methods.

Sr. No.	Components and Description
1	**onCreate()**: This is the first callback and called when the activity is first created.
2	**onStart()**: This callback is called when the activity becomes visible to the user.
3	**onResume()**: This is called when the user starts interacting with the application.
4	**onPause()**: The paused activity does not receive user input and cannot execute any code and is called when the current activity is being paused and the previous activity is being resumed.
5	**onStop()**: This callback is called when the activity is no longer visible.
6	**onDestroy()**: This callback is called before the activity is destroyed by the system.
7	**onRestart()**: This callback is called when the activity restarts after stopping it.

Name of Student: Pushkar Sane		
Roll Number: 45	**Lab Assignment Number: 1B**	
Title of Lab Assignment: Android program using various UI Components.(EditText, RatingBar, CheckBox, RadioButton, Button etc). Get the values from the components and display them on the next activity.		
DOP: 18-08-2024	**DOS: 24-08-2024**	
CO Mapped: **CO1**	**PO Mapped:** **PO3, PO5, PO7, PO12, PSO1, PSO2**	**Signature:**

Practical 1B

Aim: Android program using various UI components and different layouts and views. Android program using various UI Components. (EditText, RatingBar, CheckBox, RadioButton, Button etc). Get the values from the components and display them on the next activity.

Code:

MainActivity.java

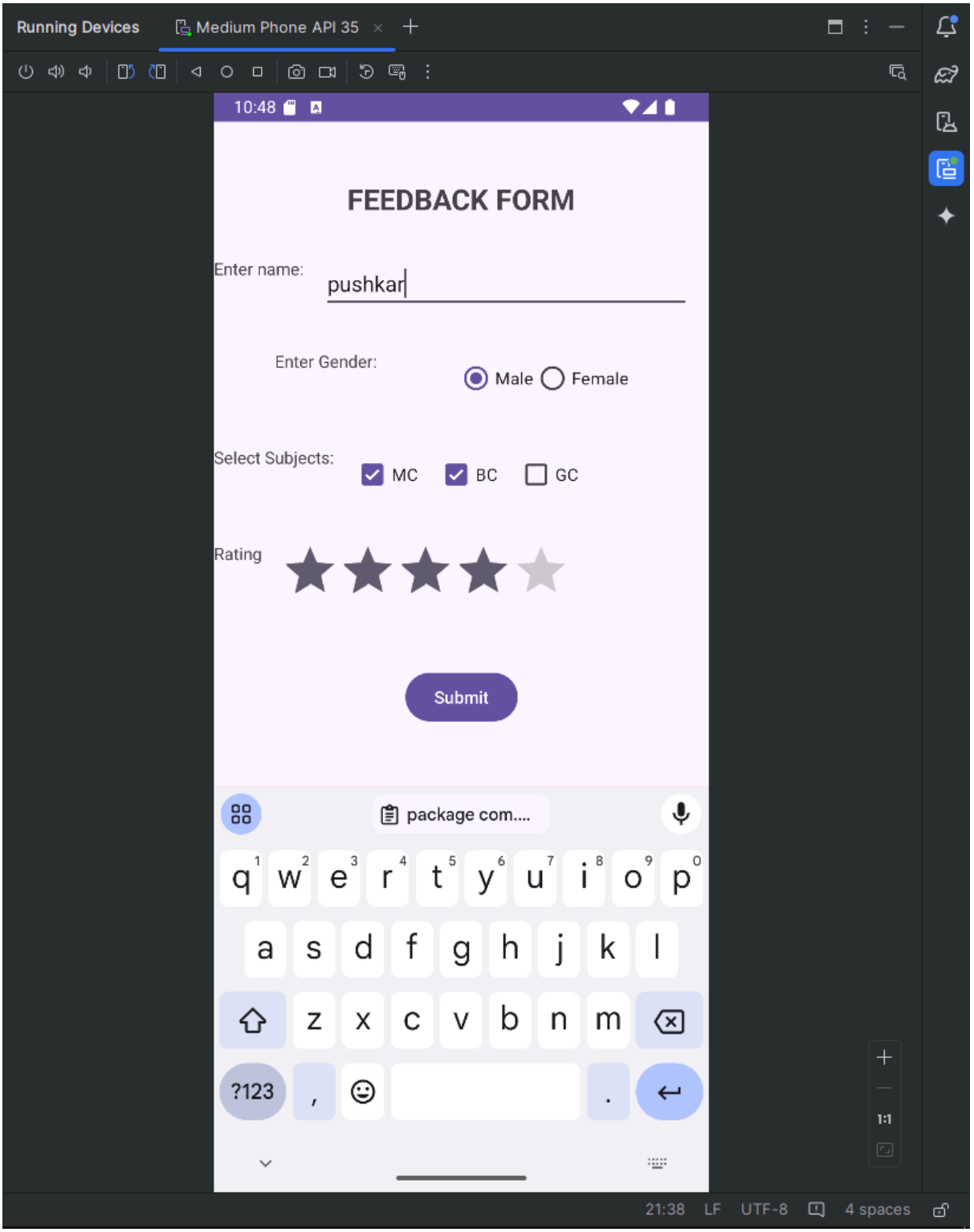
```
package com.example.myapplication;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.RatingBar;

public class MainActivity extends AppCompatActivity {
    EditText editTextName;
    RadioButton radioMale, radioFemale;
    CheckBox checkBoxMC, checkBoxBC, checkBoxGC;
    RatingBar ratingBar;
    Button submitButton;

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

        // Initialize the components
        editTextName = findViewById(R.id.editTextText3);
        radioMale = findViewById(R.id.radioButton3);
```

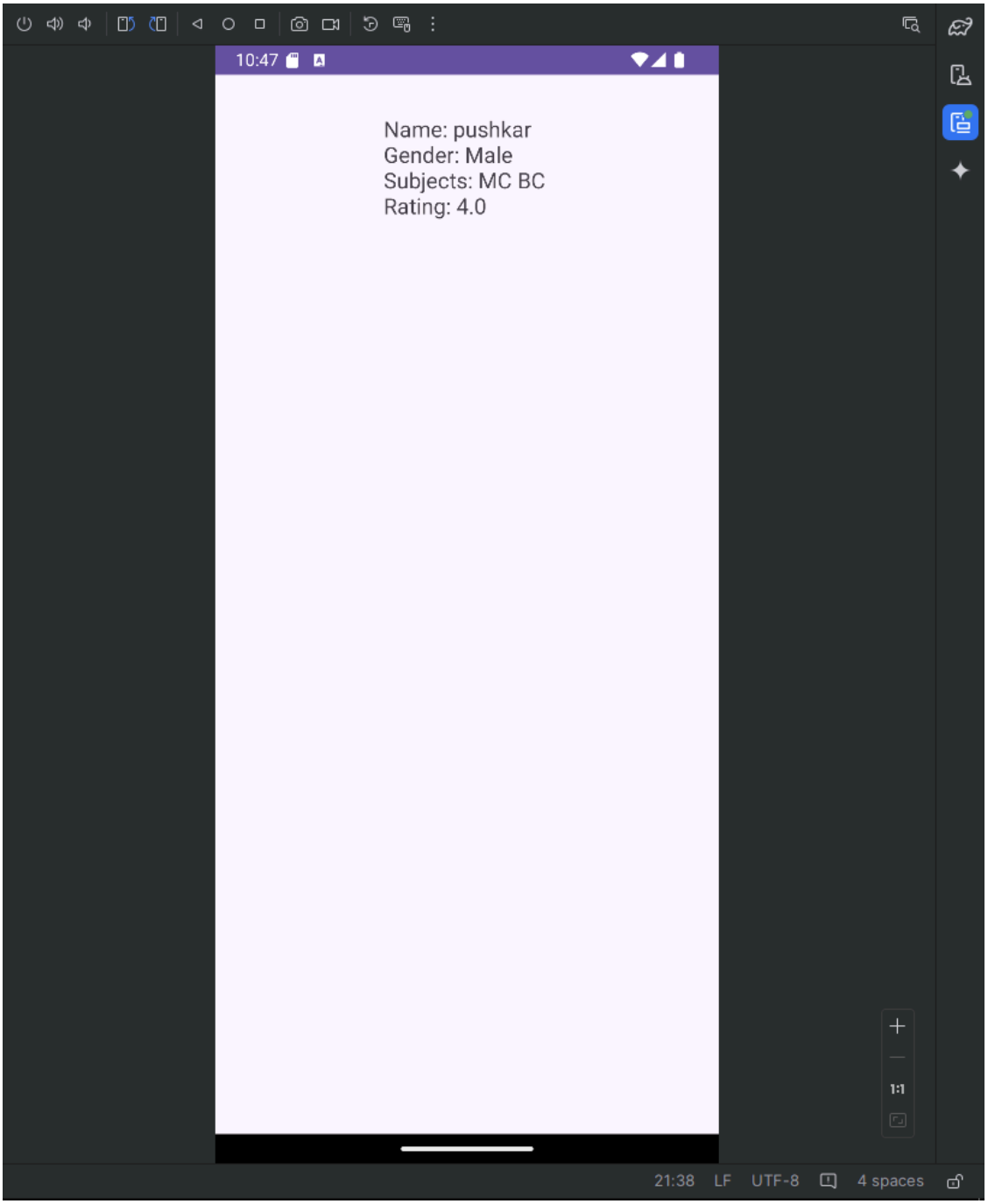
```
        radioFemale = findViewById(R.id.radioButton4);
        checkBoxMC = findViewById(R.id.checkBox);
        checkBoxBC = findViewById(R.id.checkBox2);
        checkBoxGC = findViewById(R.id.checkBox3);
        ratingBar = findViewById(R.id.ratingBar);
        submitButton = findViewById(R.id.button);

        // Set onClickListener for the button
        submitButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Get the values from the components
                String name = editTextName.getText().toString();
                String gender = radioMale.isChecked() ? "Male" : "Female";
                String subjects = "";
                if (checkBoxMC.isChecked()) subjects += "MC ";
                if (checkBoxBC.isChecked()) subjects += "BC ";
                if (checkBoxGC.isChecked()) subjects += "GC ";
                float rating = ratingBar.getRating();

                // Create an intent to go to the next activity
                Intent intent = new Intent(MainActivity.this, NextActivity.class);
                // Pass the collected data to the next activity
                intent.putExtra("name", name);
                intent.putExtra("gender", gender);
                intent.putExtra("subjects", subjects);
                intent.putExtra("rating", rating);
                startActivity(intent);
            }
        });
    }
}
```

NextActivity.java

```
package com.example.myapplication;

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

public class NextActivity extends AppCompatActivity {

    TextView textViewSummary;

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

        textViewSummary = findViewById(R.id.textViewSummary);

        // Retrieve the data passed from MainActivity
        String name = getIntent().getStringExtra("name");
        String gender = getIntent().getStringExtra("gender");
        String subjects = getIntent().getStringExtra("subjects");
        float rating = getIntent().getFloatExtra("rating", 0);

        // Display the retrieved data
        String summary = "Name: " + name + "\n" +
                "Gender: " + gender + "\n" +
                "Subjects: " + subjects + "\n" +
                "Rating: " + rating;

        textViewSummary.setText(summary);
    }
}
```

Output:


Running Devices
Medium Phone API 35
10:48
FEEDBACK FORM
Enter name:
pushkar
Enter Gender:
Male
Female
Select Subjects:
MC
BC
GC
Rating
Submit
package com....
21:38
LF
UTF-8
4 spaces



Conclusion:

Implementing UI in Android using Android Studio involves designing layouts with XML and coding interactions in Java. The process includes creating activities and defining their UI elements, such as buttons, text fields, and checkboxes. To enable navigation between activities, you use Intent objects to start new activities and pass data between them. This seamless transition allows for dynamic user experiences, where data collected from user inputs in one activity can be efficiently transferred and utilized in another, enhancing the overall functionality and interactivity of the app.

Name of Student: Pushkar Sane		
Roll Number: 45	**Lab Assignment Number: 1**	
Title of Lab Assignment: Create a login form. For a successful login, display the welcome page, and in case of failure display an alert box indicating an error message and attempts made. Disable the submit button after 3 wrong attempts and display the alert message indicating the same.		
DOP: 18-08-2024	**DOS: 24-08-2024**	
CO Mapped: **CO1**	**PO Mapped:** **PO3, PO5, PO7, PO12, PSO1, PSO2**	**Signature:**

Practical 1C

Aim: Create a login form. For a successful login, display the welcome page, and in case of failure display an alert box indicating an error message and attempts made. Disable the submit button after 3 wrong attempts and display the alert message indicating the same.

Code:

MainActivity.java

```
package com.example.practical1c;

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

public class MainActivity extends AppCompatActivity {
    private EditText usernameEditText;
    private EditText passwordEditText;
    private Button loginButton;
    private int attemptCount = 0;
    private static final int MAX_ATTEMPTS = 3;

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

        usernameEditText = findViewById(R.id.username);
        passwordEditText = findViewById(R.id.password);
        loginButton = findViewById(R.id.login_button);

        loginButton.setOnClickListener(new View.OnClickListener() {
            @Override
```

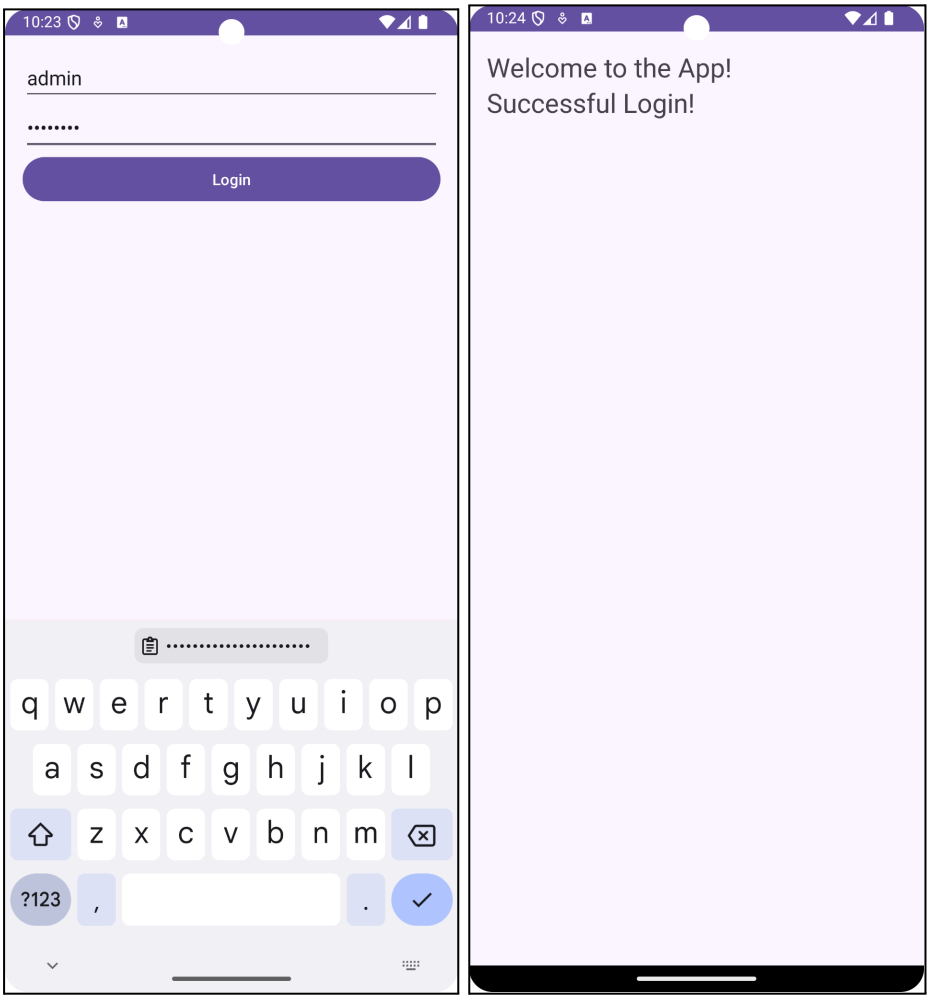
```
        public void onClick(View v) {
            handleLogin();
        }
    });
}

private void handleLogin() {
    String username = usernameEditText.getText().toString();
    String password = passwordEditText.getText().toString();

    if (username.equals("admin") && password.equals("password")) {
        // Successful login
        Intent intent = new Intent(MainActivity.this, WelcomeActivity.class);
        startActivity(intent);
        finish();
    } else {
        // Failed login
        attemptCount++;
        if (attemptCount >= MAX_ATTEMPTS) {
            loginButton.setEnabled(false);
                Toast.makeText(MainActivity.this, "Too many attempts. Please try again later.",
Toast.LENGTH_LONG).show();
        } else {
                        Toast.makeText(MainActivity.this, "Login failed. Attempts left: " +
(MAX_ATTEMPTS - attemptCount), Toast.LENGTH_SHORT).show();
        }
    }
}
```

WelcomeActivity.java

```
package com.example.practical1c;
import android.os.Bundle;
import androidx.appcompat.app.AppCompatActivity;

public class WelcomeActivity extends AppCompatActivity {
```

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

Output:



Conclusion:

In conclusion, creating a login form in Android Studio involves designing an interface that handles user authentication by displaying a welcome page upon successful login. For failed login attempts, an alert box shows an error message and the number of attempts made. To enhance security, the submit button is disabled after three incorrect attempts, accompanied by an alert message informing the user of the same. This approach ensures a user-friendly and secure login process.

Name of Student: Pushkar Sane		
Roll Number: 45	**Lab Assignment Number: 2**	
Title of Lab Assignment: To study different types of layouts and Toast class.		
DOP: 04-09-2024	**DOS: 05-09-2024**	
CO Mapped: **CO1, CO2**	**PO Mapped:** **PO1, PO2, PO3, PSO1**	**Signature:**

Practical No. 2

Aim: To study different types of layouts and Toast class.

1. Design a Registration form to show the working of different layouts.
2. Create an application to design a simple calculator to perform addition, subtraction, multiplication and division. Show message for divide by zero error using Toast.
3. Create an application for Unit Conversion.

Theory:

Layouts in Android

Layouts in Android define how UI elements are arranged on the screen. They are crucial for creating user interfaces. Here are some common types of layouts:

1. LinearLayout
    - Orientation: Can be vertical or horizontal.
    - Usage: Arranges child elements in a single row or column.
    - Example: A vertical LinearLayout might stack buttons one on top of the other.
2. RelativeLayout
    - Positioning: Allows you to position elements relative to each other or to the parent container.
    - Usage: Useful when you need flexible positioning of elements.
    - Example: Positioning a button below a text view or aligning an image to the right of a text view.
3. ConstraintLayout
    - Design: Allows you to create complex layouts with a flat view hierarchy by defining constraints between views.
    - Usage: Provides more flexibility and better performance compared to nested layouts.
    - Example: Positioning elements in a grid or aligning them relative to each other or the parent.
4. FrameLayout
    - Stacking: Displays child elements stacked on top of each other.
    - Usage: Useful for overlapping views or when only one child view is expected.
    - Example: Overlaying a loading spinner over other UI components.

5. GridLayout
    - Structure: Organizes child views into a grid.
    - Usage: Useful for creating forms or tables where elements need to align in rows and columns.
    - Example: A calendar view where each cell represents a day of the month.
6. TableLayout
    - Arrangement: Arranges child views into rows and columns, similar to a table.
    - Usage: Useful for organizing data in a tabular format.
    - Example: Creating a layout for a calculator with buttons arranged in rows and columns.

Toast Class in Android

Toast is a class in Android used for displaying brief messages to the user. These messages pop up on the screen for a short duration and then automatically disappear. They are typically used for feedback or notifications that do not require user interaction.

- Usage: Toast is used to provide simple feedback to the user, such as confirming an action or displaying a small piece of information.
- Methods:
    - Toast.makeText(Context context, CharSequence text, int duration)
        - Parameters:
            - context: The context in which the Toast should appear.
            - text: The message to be displayed.
            - duration: The duration for which the Toast should be visible (Toast.LENGTH_SHORT or Toast.LENGTH_LONG).
    - show()
        - Usage: Displays the Toast message on the screen.

Code:

1. **Registration form**

   MainActivity.java

```
package com.example.registrationform;
import android.os.Bundle;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
```

```
        super.onCreate(savedInstanceState);
        // Uncomment the layout you want to test
        setContentView(R.layout.linearlayout); // LinearLayout
        // setContentView(R.layout.relativelayout); // RelativeLayout
        // setContentView(R.layout.constraintlayout); // ConstraintLayout
        // setContentView(R.layout.gridlayout); // GridLayout
    }
}
```

Constraint Layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp">

    <EditText
        android:id="@+id/username"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:hint="Username"
        android:inputType="text"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent" />

    <EditText
        android:id="@+id/email"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:hint="Email"
        android:inputType="textEmailAddress"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/username"
```

```
        app:layout_constraintVertical_chainStyle="packed" />

    <EditText
        android:id="@+id/password"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:hint="Password"
        android:inputType="textPassword"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/email" />

    <Button
        android:id="@+id/registerButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Register"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/password"
        app:layout_constraintVertical_bias="0.1" />

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

LinearLayout.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"
    android:padding="16dp">

    <EditText
        android:id="@+id/username"
        android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
        android:hint="Username"
        android:inputType="text" />

    <EditText
        android:id="@+id/email"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Email"
        android:inputType="textEmailAddress" />

    <EditText
        android:id="@+id/password"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Password"
        android:inputType="textPassword" />

    <Button
        android:id="@+id/registerButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Register"
        android:layout_gravity="center_horizontal" />

</LinearLayout>
```

GridLayout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<GridLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp"
    android:columnCount="1"
    android:rowCount="5"
    android:orientation="vertical">
```

```
    <EditText
        android:id="@+id/username"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Username"
        android:inputType="text" />

    <EditText
        android:id="@+id/email"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Email"
        android:inputType="textEmailAddress" />

    <EditText
        android:id="@+id/password"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Password"
        android:inputType="textPassword" />

    <Button
        android:id="@+id/registerButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Register"
        android:layout_row="4"
        android:layout_gravity="center_horizontal" />
</GridLayout>
```

RelativeLayout.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"
    android:padding="16dp">
```

```
    <EditText
        android:id="@+id/username"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Username"
        android:inputType="text" />

    <EditText
        android:id="@+id/email"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Email"
        android:inputType="textEmailAddress"
        android:layout_below="@id/username"
        android:layout_marginTop="16dp" />

    <EditText
        android:id="@+id/password"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Password"
        android:inputType="textPassword"
        android:layout_below="@id/email"
        android:layout_marginTop="16dp" />

    <Button
        android:id="@+id/registerButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Register"
        android:layout_below="@id/password"
        android:layout_marginTop="24dp"
        android:layout_centerHorizontal="true" />

</RelativeLayout>
```

2. **Calculator**

MainActivity.java

```
package com.example.calculator;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

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

        EditText input1 = findViewById(R.id.input1);
        EditText input2 = findViewById(R.id.input2);
        Button addButton = findViewById(R.id.addButton);
        Button subtractButton = findViewById(R.id.subtractButton);
        Button multiplyButton = findViewById(R.id.multiplyButton);
        Button divideButton = findViewById(R.id.divideButton);
        TextView resultView = findViewById(R.id.resultView);

        addButton.setOnClickListener(v -> performOperation("add", input1, input2,
resultView));
        subtractButton.setOnClickListener(v -> performOperation("subtract", input1,
input2, resultView));
        multiplyButton.setOnClickListener(v -> performOperation("multiply", input1,
input2, resultView));
        divideButton.setOnClickListener(v -> performOperation("divide", input1, input2,
resultView));
    }
```

```
    private void performOperation(String operation, EditText input1, EditText input2,
TextView resultView) {
        try {
            double num1 = Double.parseDouble(input1.getText().toString());
            double num2 = Double.parseDouble(input2.getText().toString());
            double result = 0;

            switch (operation) {
                case "add":
                    result = num1 + num2;
                    break;
                case "subtract":
                    result = num1 - num2;
                    break;
                case "multiply":
                    result = num1 * num2;
                    break;
                case "divide":
                    if (num2 == 0) {
                        Toast.makeText(this, "Cannot divide by zero",
Toast.LENGTH_SHORT).show();
                        return;
                    }
                    result = num1 / num2;
                    break;
            }

            resultView.setText(String.valueOf(result));
        } catch (NumberFormatException e) {
            Toast.makeText(this, "Invalid input", Toast.LENGTH_SHORT).show();
        }
    }
}
```

Main_Activity.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"
    android:padding="16dp">

    <EditText
        android:id="@+id/input1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:hint="Enter number"
        android:inputType="numberDecimal"
        android:layout_marginBottom="16dp"
        android:padding="8dp"/>

    <EditText
        android:id="@+id/input2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/input1"
        android:layout_centerHorizontal="true"
        android:hint="Enter number"
        android:inputType="numberDecimal"
        android:layout_marginBottom="16dp"
        android:padding="8dp"/>
    <Button
        android:id="@+id/addButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/input2"
        android:layout_centerHorizontal="true"
        android:text="Add" />

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

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/addButton"
        android:layout_centerHorizontal="true"
        android:text="Subtract" />

    <Button
        android:id="@+id/multiplyButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/subtractButton"
        android:layout_centerHorizontal="true"
        android:text="Multiply" />

    <Button
        android:id="@+id/divideButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/multiplyButton"
        android:layout_centerHorizontal="true"
        android:text="Divide" />

    <TextView
        android:id="@+id/resultView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/divideButton"
        android:layout_centerHorizontal="true"
        android:text="Result"
        android:textSize="18sp"
        android:layout_marginTop="16dp" />
</RelativeLayout>
```

3. **Unit Conversion**

MainActivity.java

```
package com.example.unitconverter;

import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;
import java.util.HashMap;
import java.util.Map;

public class MainActivity extends AppCompatActivity {

    private Spinner conversionTypeSpinner;
    private Spinner fromUnitSpinner;
    private Spinner toUnitSpinner;
    private EditText inputValue;
    private Button convertButton;
    private TextView resultTextView;

    private Map<String, String[]> unitMap;

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

        conversionTypeSpinner = findViewById(R.id.conversionTypeSpinner);
        fromUnitSpinner = findViewById(R.id.fromUnitSpinner);
        toUnitSpinner = findViewById(R.id.toUnitSpinner);
        inputValue = findViewById(R.id.inputValue);
```

```
        convertButton = findViewById(R.id.convertButton);
        resultTextView = findViewById(R.id.resultTextView);

        setupUnitMap();
        setupConversionTypeSpinner();
        setupConvertButton();
    }

    private void setupUnitMap() {
        unitMap = new HashMap<>();
        unitMap.put("Length", getResources().getStringArray(R.array.length_units));
        unitMap.put("Weight", getResources().getStringArray(R.array.weight_units));
                                                        unitMap.put("Temperature",
getResources().getStringArray(R.array.temperature_units));
    }

    private void setupConversionTypeSpinner() {
                                        ArrayAdapter<CharSequence>      adapter      =
ArrayAdapter.createFromResource(this,
                R.array.conversion_types, android.R.layout.simple_spinner_item);

adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_ite
m);
        conversionTypeSpinner.setAdapter(adapter);

                                    conversionTypeSpinner.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
            @Override
              public void onItemSelected(AdapterView<?> parent, View view, int position,
long id) {
                String selectedType = parent.getItemAtPosition(position).toString();
                updateUnitSpinners(selectedType);
            }

            @Override
            public void onNothingSelected(AdapterView<?> parent) {
```

```
            }
        });
    }

    private void updateUnitSpinners(String type) {
        ArrayAdapter<String> adapter = new ArrayAdapter<>(this,
                android.R.layout.simple_spinner_item, unitMap.get(type));

adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_ite
m);
        fromUnitSpinner.setAdapter(adapter);
        toUnitSpinner.setAdapter(adapter);
    }

    private void setupConvertButton() {
        convertButton.setOnClickListener(v -> {
            String fromUnit = fromUnitSpinner.getSelectedItem().toString();
            String toUnit = toUnitSpinner.getSelectedItem().toString();
            double value = Double.parseDouble(inputValue.getText().toString());

            double result = performConversion(fromUnit, toUnit, value);
            resultTextView.setText(String.valueOf(result));
        });
    }

    private double performConversion(String fromUnit, String toUnit, double value) {
        if (fromUnit.equals(toUnit)) {
            return value;
        }

        // Length Conversion
        if (fromUnit.equals("Meter")) {
            if (toUnit.equals("Kilometer")) return value / 1000;
            if (toUnit.equals("Centimeter")) return value * 100;
            if (toUnit.equals("Millimeter")) return value * 1000;
            if (toUnit.equals("Inch")) return value * 39.3701;
```

```
    if (toUnit.equals("Foot")) return value * 3.28084;
}
if (fromUnit.equals("Kilometer")) {
    if (toUnit.equals("Meter")) return value * 1000;
    if (toUnit.equals("Centimeter")) return value * 100000;
    if (toUnit.equals("Millimeter")) return value * 1000000;
    if (toUnit.equals("Inch")) return value * 39370.1;
    if (toUnit.equals("Foot")) return value * 3280.84;
}
if (fromUnit.equals("Centimeter")) {
    if (toUnit.equals("Meter")) return value / 100;
    if (toUnit.equals("Kilometer")) return value / 100000;
    if (toUnit.equals("Millimeter")) return value * 10;
    if (toUnit.equals("Inch")) return value * 0.393701;
    if (toUnit.equals("Foot")) return value * 0.0328084;
}
if (fromUnit.equals("Millimeter")) {
    if (toUnit.equals("Meter")) return value / 1000;
    if (toUnit.equals("Kilometer")) return value / 1000000;
    if (toUnit.equals("Centimeter")) return value / 10;
    if (toUnit.equals("Inch")) return value * 0.0393701;
    if (toUnit.equals("Foot")) return value * 0.00328084;
}
if (fromUnit.equals("Inch")) {
    if (toUnit.equals("Meter")) return value / 39.3701;
    if (toUnit.equals("Kilometer")) return value / 39370.1;
    if (toUnit.equals("Centimeter")) return value * 2.54;
    if (toUnit.equals("Millimeter")) return value * 25.4;
    if (toUnit.equals("Foot")) return value / 12;
}
if (fromUnit.equals("Foot")) {
    if (toUnit.equals("Meter")) return value / 3.28084;
    if (toUnit.equals("Kilometer")) return value / 3280.84;
    if (toUnit.equals("Centimeter")) return value * 30.48;
    if (toUnit.equals("Millimeter")) return value * 304.8;
    if (toUnit.equals("Inch")) return value * 12;
```

```
    }

    // Weight Conversion
    if (fromUnit.equals("Gram")) {
        if (toUnit.equals("Kilogram")) return value / 1000;
        if (toUnit.equals("Pound")) return value * 0.00220462;
        if (toUnit.equals("Ounce")) return value * 0.035274;
    }
    if (fromUnit.equals("Kilogram")) {
        if (toUnit.equals("Gram")) return value * 1000;
        if (toUnit.equals("Pound")) return value * 2.20462;
        if (toUnit.equals("Ounce")) return value * 35.274;
    }
    if (fromUnit.equals("Pound")) {
        if (toUnit.equals("Gram")) return value * 453.592;
        if (toUnit.equals("Kilogram")) return value * 0.453592;
        if (toUnit.equals("Ounce")) return value * 16;
    }
    if (fromUnit.equals("Ounce")) {
        if (toUnit.equals("Gram")) return value * 28.3495;
        if (toUnit.equals("Kilogram")) return value * 0.0283495;
        if (toUnit.equals("Pound")) return value / 16;
    }

    // Temperature Conversion
    if (fromUnit.equals("Celsius")) {
        if (toUnit.equals("Fahrenheit")) return value * 9/5 + 32;
        if (toUnit.equals("Kelvin")) return value + 273.15;
    }
    if (fromUnit.equals("Fahrenheit")) {
        if (toUnit.equals("Celsius")) return (value - 32) * 5/9;
        if (toUnit.equals("Kelvin")) return (value - 32) * 5/9 + 273.15;
    }
    if (fromUnit.equals("Kelvin")) {
        if (toUnit.equals("Celsius")) return value - 273.15;
        if (toUnit.equals("Fahrenheit")) return (value - 273.15) * 9/5 + 32;
```

```
        }

        return value; // Default case, should not happen
    }
}
```

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"
    android:padding="16dp">

    <Spinner
        android:id="@+id/conversionTypeSpinner"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:entries="@array/conversion_types" />

    <Spinner
        android:id="@+id/fromUnitSpinner"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/conversionTypeSpinner"
        android:layout_marginTop="16dp" />

    <Spinner
        android:id="@+id/toUnitSpinner"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/fromUnitSpinner"
        android:layout_marginTop="16dp" />

    <EditText
        android:id="@+id/inputValue"
        android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
        android:layout_below="@id/toUnitSpinner"
        android:layout_marginTop="16dp"
        android:hint="Enter value"
        android:inputType="numberDecimal" />

    <Button
        android:id="@+id/convertButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Convert"
        android:layout_below="@id/inputValue"
        android:layout_marginTop="16dp"
        android:layout_alignParentEnd="true" />

    <TextView
        android:id="@+id/resultTextView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/convertButton"
        android:layout_marginTop="16dp"
        android:textSize="18sp"
        android:text="Result will be shown here" />
</RelativeLayout>
```

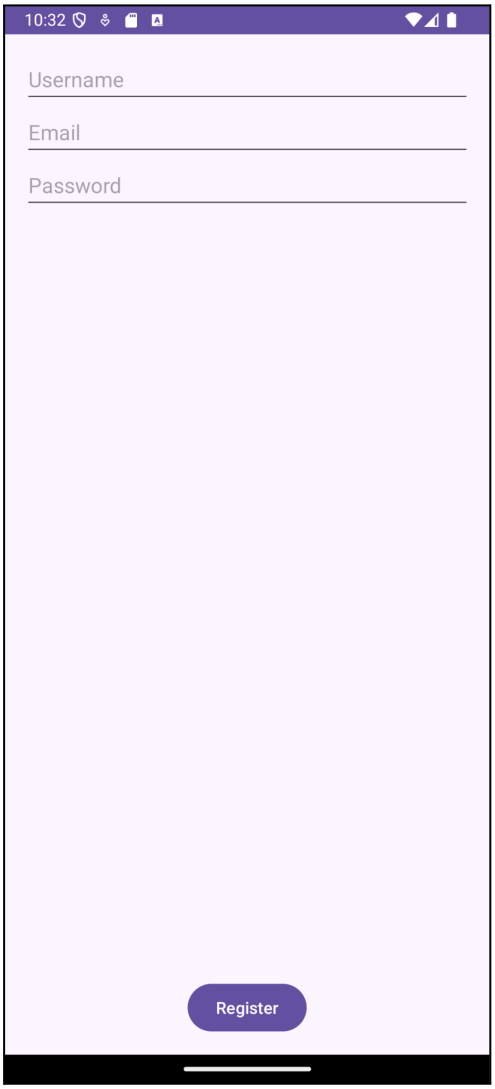
strings.xml

```
<resources>
    <string name="app_name">Unit Conversion App</string>
    <!-- Conversion Types -->
    <string-array name="conversion_types">
        <item>Length</item>
        <item>Weight</item>
        <item>Temperature</item>
    </string-array>

    <!-- Length Units -->
```

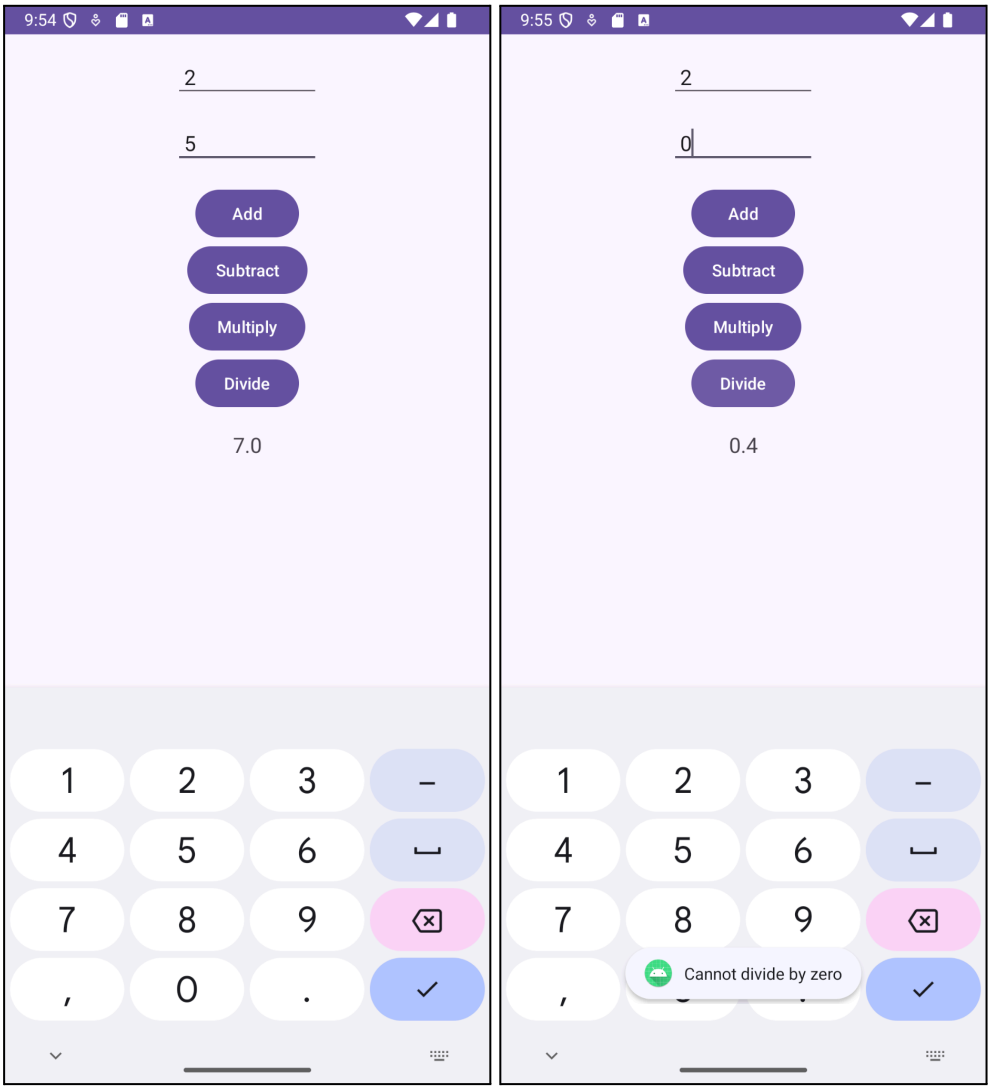
```
    <string-array name="length_units">
        <item>Meter</item>
        <item>Kilometer</item>
        <item>Centimeter</item>
        <item>Millimeter</item>
        <item>Inch</item>
        <item>Foot</item>
    </string-array>

    <!-- Weight Units -->
    <string-array name="weight_units">
        <item>Gram</item>
        <item>Kilogram</item>
        <item>Pound</item>
        <item>Ounce</item>
    </string-array>
    <!-- Temperature Units -->
    <string-array name="temperature_units">
        <item>Celsius</item>
        <item>Fahrenheit</item>
        <item>Kelvin</item>
    </string-array>
</resources>
```

Output:

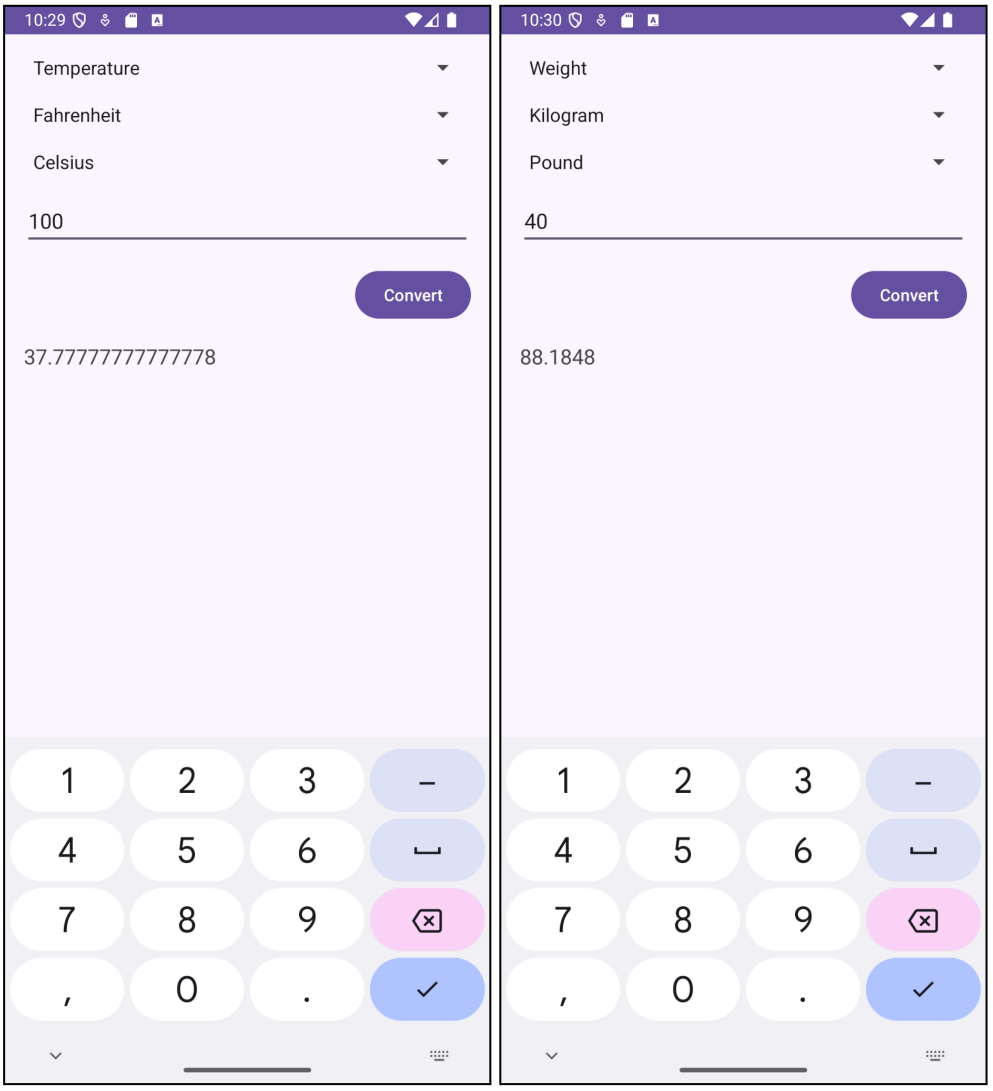
1. **Registration Form**



2. Calculator



3. Unit Conversion



Conclusion:

Through these practical exercises, we developed essential skills in Android application development, including creating a simple calculator with arithmetic operations and error handling, designing a registration form with various layout managers, and implementing navigation between different layouts. We also built a unit conversion app with comprehensive logic for length, weight, and temperature conversions.

Name of Student: Pushkar Sane		
Roll Number: 45	**Lab Assignment Number: 3**	
Title of Lab Assignment: Android program based on intent.		
DOP: 05-09-2024	**DOS: 12-09-2024**	
CO Mapped: **CO2**	**PO Mapped:** **PO2, PO3, PSO1**	**Signature:**

Practical No. 3

Aim: Android program based on intent.

Theory:

Intents in Android: Intents are an essential component in Android, used to communicate between components like activities, services, and broadcast receivers. An intent is an abstract description of an operation to be performed, facilitating messaging between different components of the Android operating system.

There are two types of intents:

1. Explicit Intents:
   Explicit intents are used to launch a specific component (e.g., a particular activity or service). The developer explicitly specifies the component that should handle the intent using the component's class name.
2. Implicit Intents:
   Implicit intents do not name a specific component. Instead, they declare a general action to perform, allowing any app that can handle the intent to process it. The system matches the intent with a suitable component using intent filters defined in the app's manifest file.

Key Concepts of Intents:

- Intent Filters:
  Intent filters are used to declare the capabilities of a component in the AndroidManifest.xml file. These filters specify the types of intents the component can handle. For example, an activity might declare an intent filter for the ACTION_SEND action, indicating that it can share data.
- Bundle:
  A Bundle object is often used to pass data between activities when starting an activity with an intent. This can include data types like strings, integers, and more.
- Extras:
  Intent extras are used to pass additional information to the receiving component. These extras can be retrieved from the intent using methods like getStringExtra().

Common Use Cases:

- Starting a new activity:
  Launching another activity within the app or an external activity (e.g., opening a web browser).
- Starting a service:
  Initiating background operations like downloading data or playing music.
- Broadcasting messages:
  Sending system-wide messages that can be received by multiple broadcast receivers.
- Communicating between fragments:
  While fragments typically communicate through the hosting activity, intents can also be used for fragment-to-fragment communication in more complex scenarios.

Code:

MainActivity.java

```
package com.example.intentprogram;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    private EditText etLink, etLocation, etMessage;

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

        etLink = findViewById(R.id.et_link);
        etLocation = findViewById(R.id.et_location);
        etMessage = findViewById(R.id.et_message);
```

```
        Button btnOpenLink = findViewById(R.id.btn_open_link);
        Button btnOpenMap = findViewById(R.id.btn_open_map);
        Button btnShareText = findViewById(R.id.btn_share_text);

        // Open Website Link
        btnOpenLink.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String url = etLink.getText().toString();
                if (!url.isEmpty()) {
                    Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse(url));
                    startActivity(intent);
                } else {
                                        Toast.makeText(MainActivity.this, "Please enter a link",
Toast.LENGTH_SHORT).show();
                }
            }
        });

        // Open Google Map
        btnOpenMap.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String location = etLocation.getText().toString();
                if (!location.isEmpty()) {
                    Uri geoUri = Uri.parse("geo:0,0?q=" + location);
                    Intent intent = new Intent(Intent.ACTION_VIEW, geoUri);
                    intent.setPackage("com.google.android.apps.maps");
                    startActivity(intent);
                } else {
                                    Toast.makeText(MainActivity.this, "Please enter a location",
Toast.LENGTH_SHORT).show();
                }
            }
        });
```

```
        // Share Text
        btnShareText.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String message = etMessage.getText().toString();
                if (!message.isEmpty()) {
                    Intent intent = new Intent(Intent.ACTION_SEND);
                    intent.setType("text/plain");
                    intent.putExtra(Intent.EXTRA_TEXT, message);
                    startActivity(Intent.createChooser(intent, "Share via"));
                } else {
                                    Toast.makeText(MainActivity.this, "Please enter a message",
Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}
```

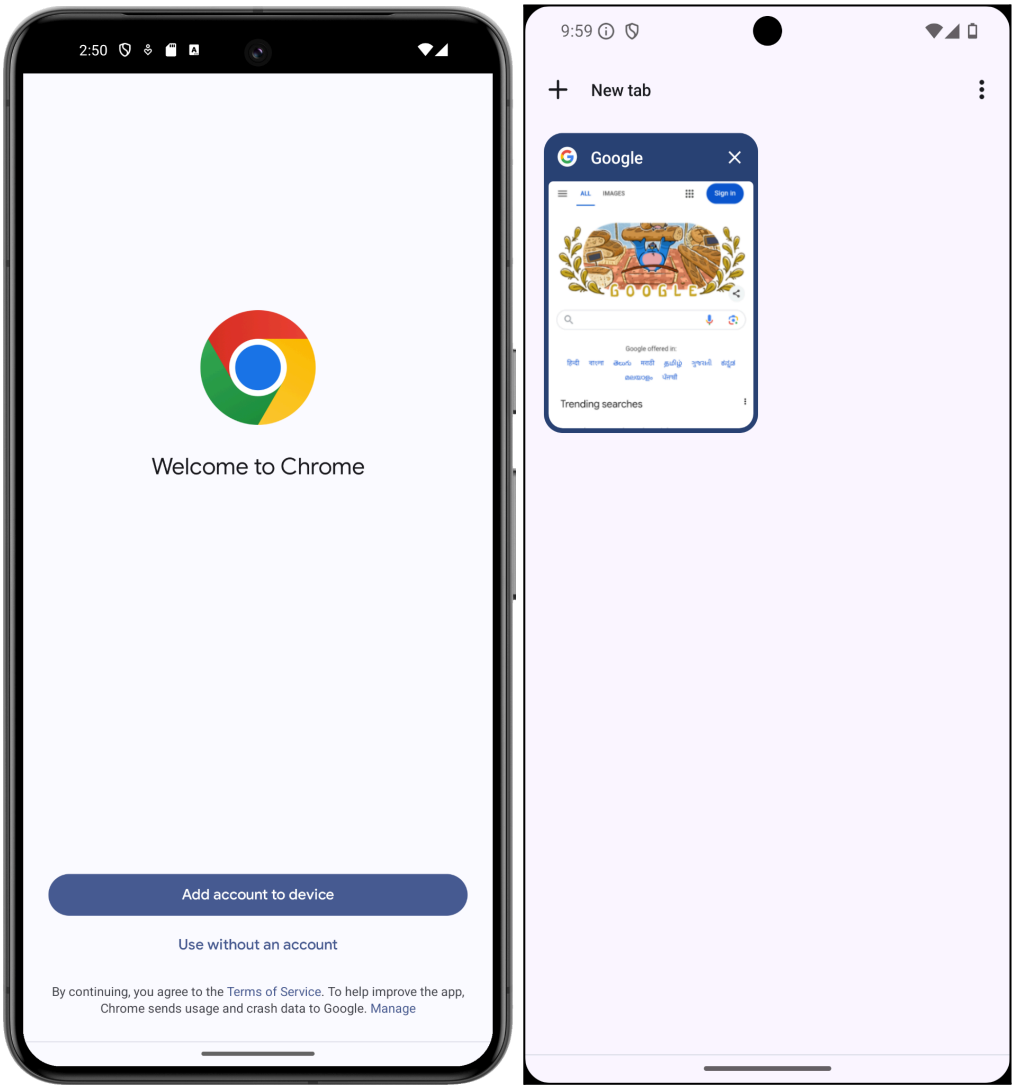
Activity_main.xml

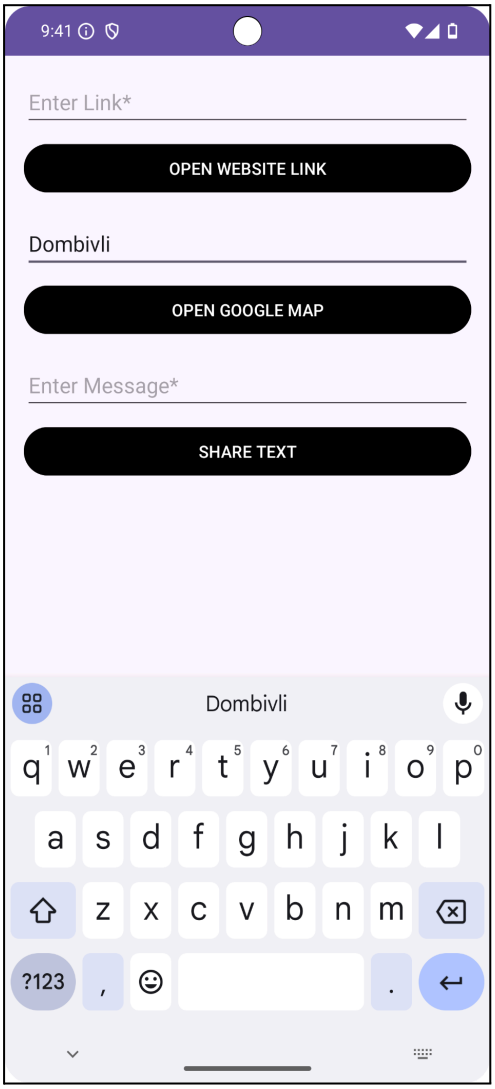
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp">

    <EditText
        android:id="@+id/et_link"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter Link*"
        android:layout_marginBottom="8dp" />

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

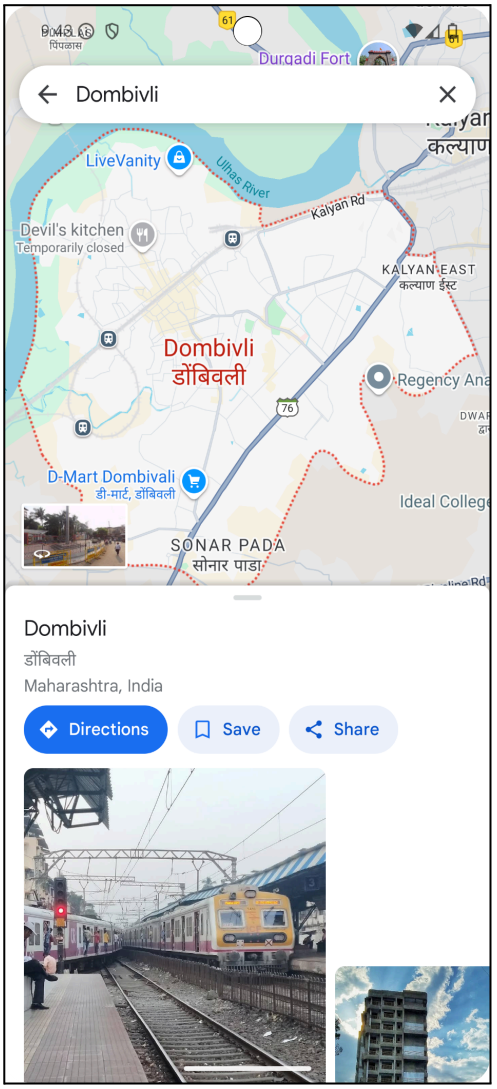
```
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@id/et_link"
    android:layout_marginBottom="16dp"
    android:backgroundTint="#000000"
    android:text="OPEN WEBSITE LINK" />

<EditText
    android:id="@+id/et_location"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Enter Location*"
    android:layout_below="@id/btn_open_link"
    android:layout_marginBottom="8dp" />

<Button
    android:id="@+id/btn_open_map"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@id/et_location"
    android:layout_marginBottom="16dp"
    android:backgroundTint="#000000"
    android:text="OPEN GOOGLE MAP" />

<EditText
    android:id="@+id/et_message"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Enter Message*"
    android:layout_below="@id/btn_open_map"
    android:layout_marginBottom="8dp" />

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

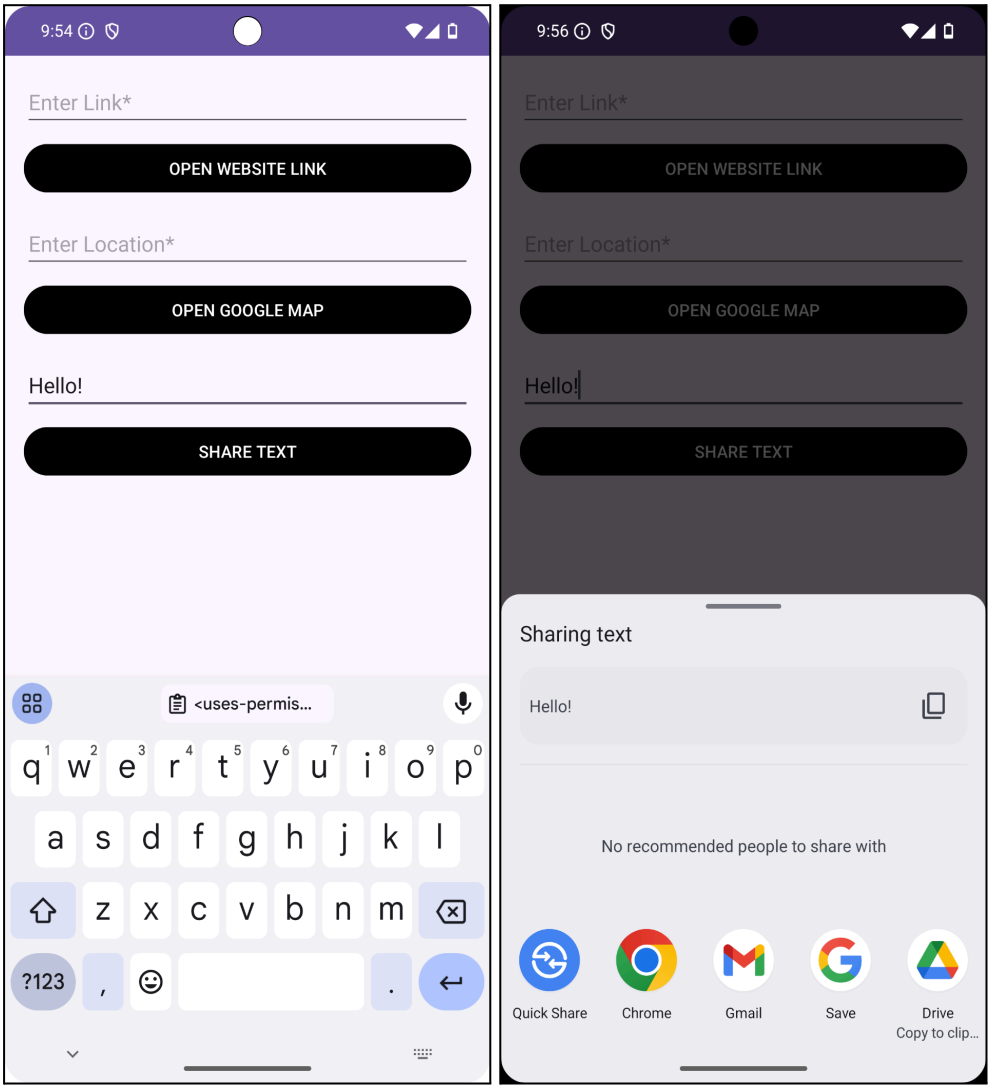
```
    android:layout_below="@id/et_message"
    android:backgroundTint="#000000"
    android:text="SHARE TEXT" />

</RelativeLayout>
```

Output:




9:41
Enter Link*
OPEN WEBSITE LINK
Dombivli
OPEN GOOGLE MAP
Enter Message*
SHARE TEXT
Dombivli


Dombivli
LiveVanity
Ulhas River
Kalyan Rd
Devil's kitchen
Temporarily closed
KALYAN EAST
कल्याण ईस्ट
Dombivli
डोंबिवली
D-Mart Dombivali
डी-मार्ट, डोंबिवली
SONAR PADA
सोनार पाडा
Dombivli
डोंबिवली
Maharashtra, India
Directions
Save
Share



Conclusion:

Android program based on intents is implemented successfully.

Name of Student: Pushkar Sane		
Roll Number: 45	**Lab Assignment Number: 4**	
Title of Lab Assignment: Android program to perform CRUD operation using SQLite DB (create table students with fields roll no, name, email-Id, course, perform add, update and delete record operations).		
DOP: 06-10-2024	**DOS: 07-10-2024**	
CO Mapped:	**PO Mapped:**	**Signature:**

Practical No. 4

Aim: Android program to perform CRUD operation using SQLite DB (create table students with fields roll no, name, email-Id, course, perform add, update and delete record operations).

Theory:

SQLite is a lightweight, embedded relational database management system that is built into Android. It is widely used for local data storage in mobile applications because of its simplicity, reliability, and efficiency.

CRUD Operations

CRUD stands for Create, Read, Update, and Delete, representing the four basic operations for managing data in a database. These operations are fundamental for any database application, and SQLite provides easy methods to implement them.

1. Create: This operation involves adding new records to the database. In SQLite, this is typically done using the INSERT statement. In an Android application, the SQLiteDatabase.insert() method is used for this purpose.
2. Read: Reading or retrieving data is accomplished using the SELECT statement. In Android, this can be done with the rawQuery() method of SQLiteDatabase, allowing you to execute SQL queries and retrieve results in the form of a Cursor object.
3. Update: This operation modifies existing records in the database. In SQLite, this is achieved through the UPDATE statement, and in Android, you can use the SQLiteDatabase.update() method to perform updates.
4. Delete: The delete operation removes records from the database using the DELETE statement. In Android, you can use the SQLiteDatabase.delete() method to remove specific entries based on given criteria.

Implementation in Android

To work with SQLite in an Android application:

1. Database Helper: Create a subclass of SQLiteOpenHelper to manage database creation and version management. This class typically contains methods for CRUD operations.
2. Database Schema: Define the database schema (tables and fields) in the onCreate() method of the helper class. This includes SQL statements to create tables.

3. CRUD Methods: Implement methods for each CRUD operation, providing simple interfaces for adding, reading, updating, and deleting records.

Advantages of Using SQLite

1. Lightweight: SQLite has a small footprint and is suitable for mobile devices.
2. Self-contained: It does not require a separate server process, making it easy to integrate into applications.
3. ACID Compliance: SQLite transactions are atomic, consistent, isolated, and durable, ensuring data integrity.
4. Cross-platform: It can be used across different platforms and is widely supported.

Code:

DatabaseHelper.java

```
package com.example.mc5;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
public class DatabaseHelper extends SQLiteOpenHelper {
    private static final String DATABASE_NAME = "students.db";
    private static final String TABLE_NAME = "students";
    private static final String COL_1 = "rollno";
    private static final String COL_2 = "name";
    private static final String COL_3 = "emailId";
    private static final String COL_4 = "course";

    public DatabaseHelper(Context context) {
        super(context, DATABASE_NAME, null, 1);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL("CREATE TABLE " + TABLE_NAME + " (rollno INTEGER PRIMARY KEY,
name TEXT, emailId TEXT, course TEXT)");
    }
```

```
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
        onCreate(db);
    }

    public boolean insertData(int rollno, String name, String emailId, String course) {
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues contentValues = new ContentValues();
        contentValues.put(COL_1, rollno);
        contentValues.put(COL_2, name);
        contentValues.put(COL_3, emailId);
        contentValues.put(COL_4, course);
        long result = db.insert(TABLE_NAME, null, contentValues);
        return result != -1; // returns true if data inserted
    }

    public Cursor getAllData() {
        SQLiteDatabase db = this.getWritableDatabase();
        return db.rawQuery("SELECT * FROM " + TABLE_NAME, null);
    }

    public boolean updateData(int rollno, String name, String emailId, String course) {
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues contentValues = new ContentValues();
        contentValues.put(COL_1, rollno);
        contentValues.put(COL_2, name);
        contentValues.put(COL_3, emailId);
        contentValues.put(COL_4, course);
        db.update(TABLE_NAME, contentValues, "rollno = ?", new
String[]{String.valueOf(rollno)});
        return true;
    }

    public Integer deleteData(int rollno) {
```

```
        SQLiteDatabase db = this.getWritableDatabase();
        return db.delete(TABLE_NAME, "rollno = ?", new String[]{String.valueOf(rollno)});
    }
}
```

ViewDataActivity.java

```
package com.example.mc5;
import android.database.Cursor;
import android.os.Bundle;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;

public class ViewDataActivity extends AppCompatActivity {
    DatabaseHelper myDb;
    TextView textViewData;

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

        myDb = new DatabaseHelper(this);
        textViewData = findViewById(R.id.textViewData);

        displayData();
    }

    private void displayData() {
        Cursor res = myDb.getAllData();
        if (res.getCount() == 0) {
            textViewData.setText("No Data Found");
            return;
        }

        StringBuilder stringBuffer = new StringBuilder();
        while (res.moveToNext()) {
```

```
            stringBuffer.append("Roll No: ").append(res.getString(0)).append("\n");
            stringBuffer.append("Name: ").append(res.getString(1)).append("\n");
            stringBuffer.append("Email ID: ").append(res.getString(2)).append("\n");
            stringBuffer.append("Course: ").append(res.getString(3)).append("\n\n");
        }

        textViewData.setText(stringBuffer.toString());
        res.close(); // Always close the cursor
    }
}
```

MainActivity.java

```
package com.example.mc5;
import android.content.Intent;
import android.database.Cursor;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    DatabaseHelper myDb;
    EditText editRollNo, editName, editEmailId, editCourse;
    Button btnAddData, btnViewData, btnUpdateData, btnDeleteData;

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

        myDb = new DatabaseHelper(this);
        editRollNo = findViewById(R.id.editTextRollNo);
        editName = findViewById(R.id.editTextName);
        editEmailId = findViewById(R.id.editTextEmailId);
```

```
        editCourse = findViewById(R.id.editTextCourse);
        btnAddData = findViewById(R.id.buttonAdd);
        btnViewData = findViewById(R.id.buttonView);
        btnUpdateData = findViewById(R.id.buttonUpdate);
        btnDeleteData = findViewById(R.id.buttonDelete);

        addData();
        viewData();
        updateData();
        deleteData();
    }

    public void addData() {
        btnAddData.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String rollno = editRollNo.getText().toString();
                String name = editName.getText().toString();
                String emailId = editEmailId.getText().toString();
                String course = editCourse.getText().toString();

                if (rollno.isEmpty() || name.isEmpty() || emailId.isEmpty() || course.isEmpty()) {
                    Toast.makeText(MainActivity.this, "Please fill all fields",
Toast.LENGTH_SHORT).show();
                    return;
                }

                try {
                    int rollNumberInt = Integer.parseInt(rollno);
                    boolean isInserted = myDb.insertData(rollNumberInt, name, emailId, course);
                    if (isInserted) {
                        Toast.makeText(MainActivity.this, "Data Inserted",
Toast.LENGTH_SHORT).show();
                        clearInputs();
                    } else {
```

```
                    Toast.makeText(MainActivity.this, "Data Not Inserted",
Toast.LENGTH_SHORT).show();
                }
            } catch (NumberFormatException e) {
                Toast.makeText(MainActivity.this, "Invalid Roll Number",
Toast.LENGTH_SHORT).show();
            }
        }
    });
}

public void viewData() {
    btnViewData.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent intent = new Intent(MainActivity.this, ViewDataActivity.class);
            startActivity(intent);
        }
    });
}

public void updateData() {
    btnUpdateData.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            String rollno = editRollNo.getText().toString();
            String name = editName.getText().toString();
            String emailId = editEmailId.getText().toString();
            String course = editCourse.getText().toString();

            if (rollno.isEmpty() || name.isEmpty() || emailId.isEmpty() || course.isEmpty()) {
                Toast.makeText(MainActivity.this, "Please fill all fields",
Toast.LENGTH_SHORT).show();
                return;
            }
```

```
            try {
                int rollNumberInt = Integer.parseInt(rollno);
                boolean isUpdated = myDb.updateData(rollNumberInt, name, emailId, course);
                if (isUpdated) {
                    Toast.makeText(MainActivity.this, "Data Updated",
Toast.LENGTH_SHORT).show();
                } else {
                    Toast.makeText(MainActivity.this, "Data Not Updated",
Toast.LENGTH_SHORT).show();
                }
            } catch (NumberFormatException e) {
                Toast.makeText(MainActivity.this, "Invalid Roll Number",
Toast.LENGTH_SHORT).show();
            }
        }
    });
  }

  public void deleteData() {
    btnDeleteData.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        String rollno = editRollNo.getText().toString();

        if (rollno.isEmpty()) {
            Toast.makeText(MainActivity.this, "Please enter a Roll Number",
Toast.LENGTH_SHORT).show();
            return;
        }

        try {
            Integer deletedRows = myDb.deleteData(Integer.parseInt(rollno));
            if (deletedRows > 0) {
                Toast.makeText(MainActivity.this, "Data Deleted",
Toast.LENGTH_SHORT).show();
            } else {
```

```
                    Toast.makeText(MainActivity.this, "Data Not Deleted",
Toast.LENGTH_SHORT).show();
                }
            } catch (NumberFormatException e) {
                Toast.makeText(MainActivity.this, "Invalid Roll Number",
Toast.LENGTH_SHORT).show();
            }
        }
    });
  }

  private void clearInputs() {
    editRollNo.setText("");
    editName.setText("");
    editEmailId.setText("");
    editCourse.setText("");
  }
}
```

viewactivity.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"
  android:padding="16dp">

  <TextView
    android:id="@+id/textViewData"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textSize="16sp" />
</LinearLayout>
```

activity_main.xml

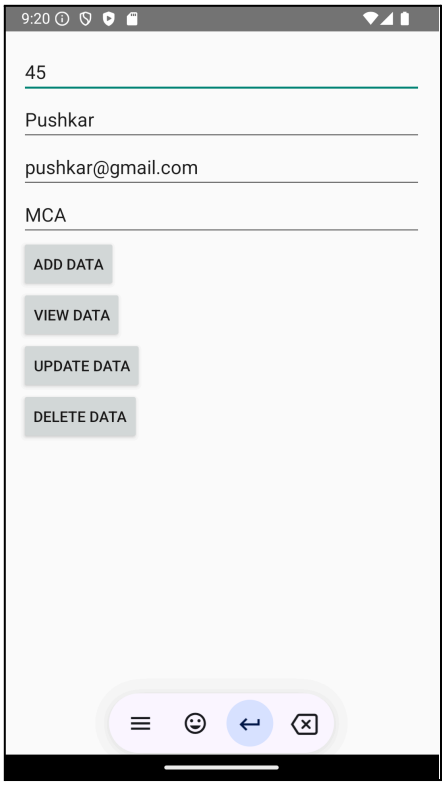
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp">

    <EditText
        android:id="@+id/editTextRollNo"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Roll No" />

    <EditText
        android:id="@+id/editTextName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Name" />

    <EditText
        android:id="@+id/editTextEmailId"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Email ID" />
    <EditText
        android:id="@+id/editTextCourse"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Course" />

    <Button
        android:id="@+id/buttonAdd"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Add Data" />
```

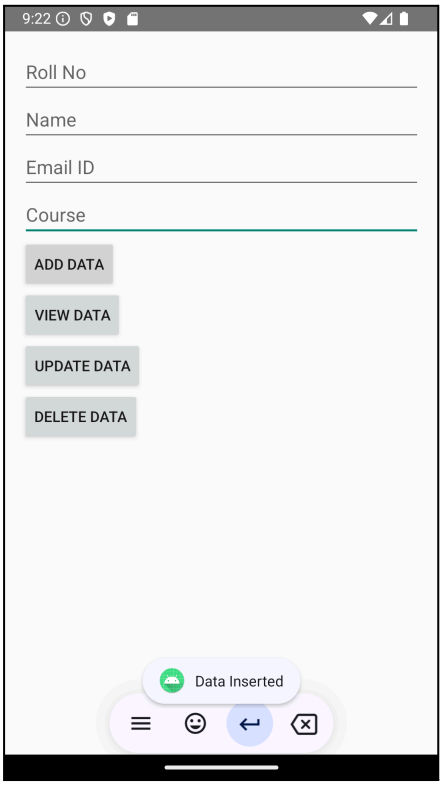
```
    <Button
        android:id="@+id/buttonView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="View Data" />

    <Button
        android:id="@+id/buttonUpdate"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Update Data" />

    <Button
        android:id="@+id/buttonDelete"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Delete Data" />
</LinearLayout>
```

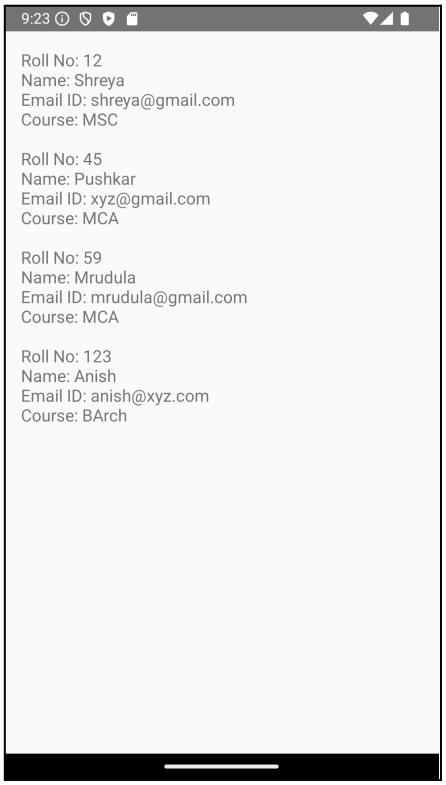
AndroidManifest.xml (Add)

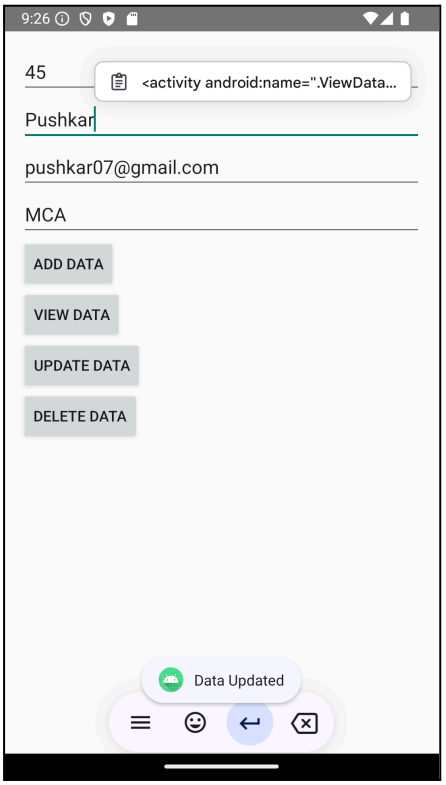
```
 <activity android:name=".ViewDataActivity" />
```

Output:


9:20
45
Pushkar
pushkar@gmail.com
MCA
ADD DATA
VIEW DATA
UPDATE DATA
DELETE DATA


9:22
Roll No
Name
Email ID
Course
ADD DATA
VIEW DATA
UPDATE DATA
DELETE DATA
Data Inserted


9:23
Roll No: 12
Name: Shreya
Email ID: shreya@gmail.com
Course: MSC
Roll No: 45
Name: Pushkar
Email ID: xyz@gmail.com
Course: MCA
Roll No: 59
Name: Mrudula
Email ID: mrudula@gmail.com
Course: MCA
Roll No: 123
Name: Anish
Email ID: anish@xyz.com
Course: BArch


9:26
45
<activity android:name=".ViewData...
Pushkar
pushkar07@gmail.com
MCA
ADD DATA
VIEW DATA
UPDATE DATA
DELETE DATA
Data Updated

Conclusion:

SQLite serves as a powerful tool for data management in Android applications, allowing developers to create robust apps that can efficiently store and retrieve data. Understanding how to implement CRUD operations is essential for building any data-driven application.

Name of Student: Pushkar Sane		
Roll Number: 45	**Lab Assignment Number: 5**	
Title of Lab Assignment: To implement file I/O and Shared Preferences.		
DOP: 04-10-2024	**DOS: 05-10-2024**	
CO Mapped: **CO2, CO3**	**PO Mapped:** **PO2, PO3, PO5, PSO1, PSO2**	**Signature:**

Practical No. 5

Aim: To implement file I/O and Shared Preferences.

1. Program to create a file in a directory and perform following file operations, Write into a file, Read from a file, Delete a file
2. Create a new project and create a login Activity. In this create a login UI asking user email and password with an option of remember me checkbox. Also a button displaying Sign In or Register using shared preferences.

Theory:

In Android, Input/Output (I/O) refers to the operations that involve reading from or writing to external sources such as files, databases, or networks. There are several ways to handle I/O operations in Android:

1. File I/O
    a. Internal Storage: Files are stored in the device's internal storage, and they are private to the application by default.
    b. External Storage: Files can be stored in external storage (SD cards, etc.), and they may be accessible by other apps depending on permissions.

    To perform file operations:
    1. openFileInput() and openFileOutput() for internal storage.
    2. getExternalStorageDirectory() for external storage.
2. Networking I/O:
    a. This involves fetching data from the internet or local networks.
    b. Android provides libraries like HttpURLConnection, OkHttp, and Retrofit for performing network operations.
3. Database I/O:
    a. Android supports local databases using SQLite and Room.
    b. Operations include reading/writing to the database using SQL queries.
4. Content Providers:
    a. Content providers enable apps to share data with other apps.
    b. It can be used for accessing data from contacts, media, etc.

Shared Preferences in Android:

SharedPreferences is a lightweight mechanism used to store small amounts of key-value data that persists across user sessions. It is mainly used to store simple data like user settings, preferences, or app states (e.g., login status).

Key points about Shared Preferences:

1. Stored as an XML file in the app's private internal storage.
2. Data persists even when the app is closed or the device is restarted.
3. It is not designed for large datasets or complex data structures.

SharedPreferences is ideal for small sets of primitive data types such as int, boolean, String, etc.

Code & Output:

1. **Read, Write, Delete**

   MainActivity.java

```
package com.example.practical5a;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.os.Environment;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;

public class MainActivity extends AppCompatActivity {
    private EditText editTextFileName;
    private EditText editTextFileContent;
    private Button buttonWrite;
    private Button buttonRead;
    private Button buttonDelete;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        editTextFileName = findViewById(R.id.editTextFileName);
        editTextFileContent = findViewById(R.id.editTextFileContent);
```

```
        buttonWrite = findViewById(R.id.buttonWrite);
        buttonRead = findViewById(R.id.buttonRead);
        buttonDelete = findViewById(R.id.buttonDelete);

        buttonWrite.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String fileName = editTextFileName.getText().toString();
                String fileContent = editTextFileContent.getText().toString();
                writeToFile(fileName, fileContent);
            }
        });

        buttonRead.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String fileName = editTextFileName.getText().toString();
                readFromFile(fileName);
            }
        });

        buttonDelete.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String fileName = editTextFileName.getText().toString();
                deleteCustomFile(fileName);
            }
        });
    }

    private void writeToFile(String fileName, String fileContent) {
        File documentsDirectory = new
File(Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DO
CUMENTS), "");

        if (!documentsDirectory.exists()) {
```

```
            documentsDirectory.mkdirs();
        }

        File file = new File(documentsDirectory, fileName);

        try {
            FileOutputStream fos = new FileOutputStream(file);
            fos.write(fileContent.getBytes());
            fos.close();
            Toast.makeText(this, "File saved at: " + file.getAbsolutePath(),
Toast.LENGTH_SHORT).show();
        } catch (IOException e) {
            e.printStackTrace();
            Toast.makeText(this, "Error saving file", Toast.LENGTH_SHORT).show();
        }
    }

    private void readFromFile(String fileName) {
        File file = new
File(Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DO
CUMENTS), fileName);
        if (file.exists()) {
            try {
                FileInputStream fis = new FileInputStream(file);
                int c;
                StringBuilder fileContent = new StringBuilder();
                while ((c = fis.read()) != -1) {
                    fileContent.append((char) c);
                }
                fis.close();
                Toast.makeText(this, "File Content: \n" + fileContent.toString(),
Toast.LENGTH_LONG).show();
            } catch (IOException e) {
                e.printStackTrace();
                Toast.makeText(this, "Error reading file", Toast.LENGTH_SHORT).show();
            }
```

```
        } else {
            Toast.makeText(this, "File not found", Toast.LENGTH_SHORT).show();
        }
    }

    private void deleteCustomFile(String fileName) {
        File file = new
File(Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DO
CUMENTS), fileName);
        boolean deleted = file.delete();
        if (deleted) {
            Toast.makeText(this, "File deleted", Toast.LENGTH_SHORT).show();
        } else {
            Toast.makeText(this, "File not found", Toast.LENGTH_SHORT).show();
        }
    }
}
```

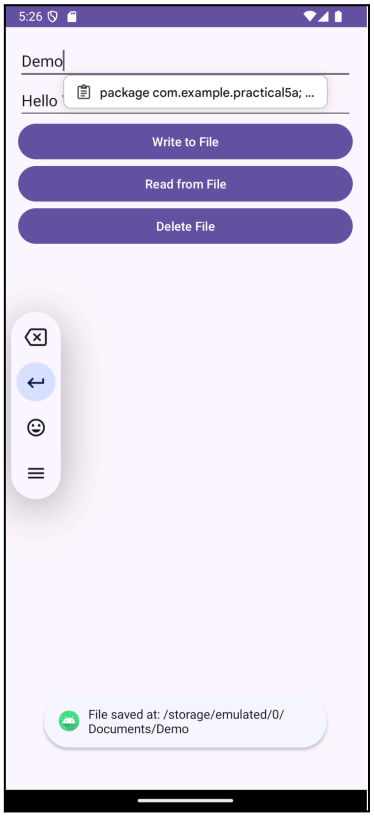
Main_Activity.xml

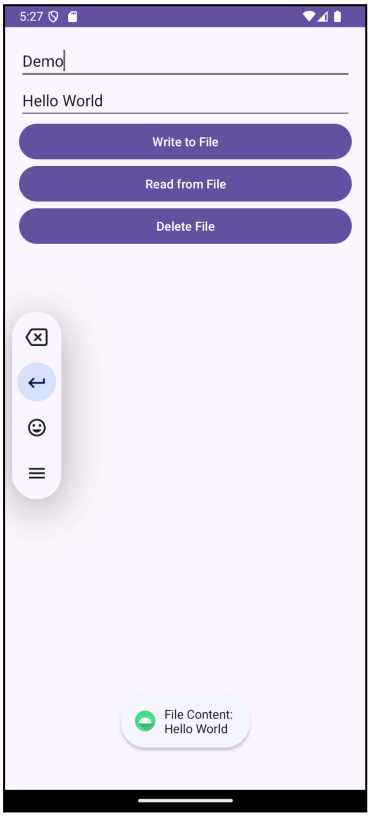
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp">

    <EditText
        android:id="@+id/editTextFileName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="File Name" />

    <EditText
        android:id="@+id/editTextFileContent"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
```

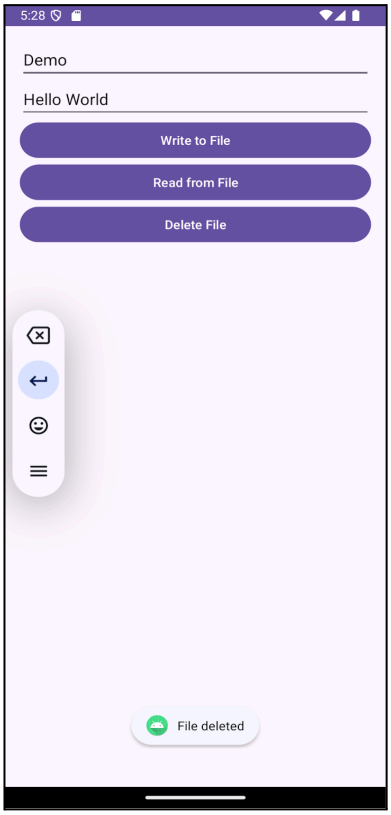
```
        android:hint="File Content" />
    <Button
        android:id="@+id/buttonWrite"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Write to File" />

    <Button
        android:id="@+id/buttonRead"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Read from File" />

    <Button
        android:id="@+id/buttonDelete"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Delete File" />
</LinearLayout>
```

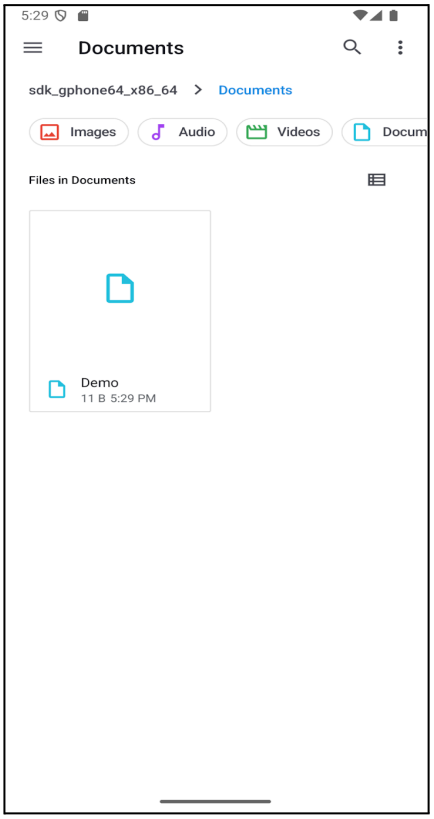
AndroidManifest.xml

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


5:26
Demo
package com.example.practical5a; ...
Hello
Write to File
Read from File
Delete File
File saved at: /storage/emulated/0/
Documents/Demo


5:27
Demo
Hello World
Write to File
Read from File
Delete File
File Content:
Hello World


5:28
Demo
Hello World
Write to File
Read from File
Delete File
File deleted


5:29
Documents
sdk_gphone64_x86_64
Documents
Images
Audio
Videos
Docum
Files in Documents
Demo
11 B 5:29 PM

2. **Login Activity**

MainActivity.java

```
package com.example.practical5b;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Context;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    private EditText editTextEmail;
    private EditText editTextPassword;
    private CheckBox checkBoxRememberMe;
    private Button buttonSignInOrRegister;
    private SharedPreferences sharedPreferences;

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

        editTextEmail = findViewById(R.id.editTextEmail);
        editTextPassword = findViewById(R.id.editTextPassword);
        checkBoxRememberMe = findViewById(R.id.checkBoxRememberMe);
        buttonSignInOrRegister = findViewById(R.id.buttonSignInOrRegister);

        sharedPreferences = getSharedPreferences("LoginPrefs",
Context.MODE_PRIVATE);
        loadSavedCredentials();

        buttonSignInOrRegister.setOnClickListener(new View.OnClickListener() {
            @Override
```

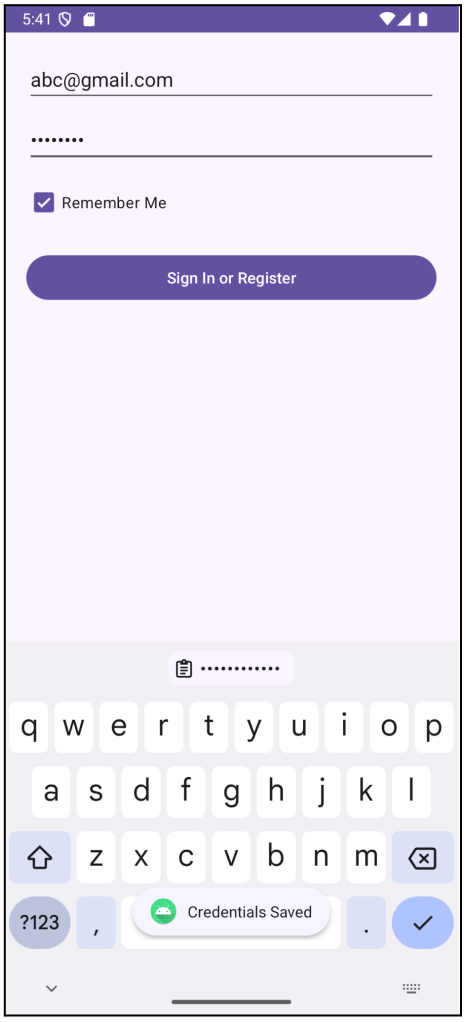
```
            public void onClick(View v) {
                String email = editTextEmail.getText().toString();
                String password = editTextPassword.getText().toString();

                if (email.isEmpty() || password.isEmpty()) {
                    Toast.makeText(LoginActivity.this, "Please enter both email and
password", Toast.LENGTH_SHORT).show();
                } else {
                    if (checkBoxRememberMe.isChecked()) {
                        saveCredentials(email, password);
                    } else {
                        clearSavedCredentials();
                    }
                    Toast.makeText(LoginActivity.this, "Welcome, " + email + "!",
Toast.LENGTH_SHORT).show();
                }
            }
        });
    }

    private void saveCredentials(String email, String password) {
        SharedPreferences.Editor editor = sharedPreferences.edit();
        editor.putString("Email", email);
        editor.putString("Password", password);
        editor.putBoolean("RememberMe", true);
        editor.apply();
        Toast.makeText(this, "Credentials Saved", Toast.LENGTH_SHORT).show();
    }

    private void loadSavedCredentials() {
        String savedEmail = sharedPreferences.getString("Email", "");
        String savedPassword = sharedPreferences.getString("Password", "");
        boolean rememberMe = sharedPreferences.getBoolean("RememberMe", false);

        if (rememberMe) {
            editTextEmail.setText(savedEmail);
```

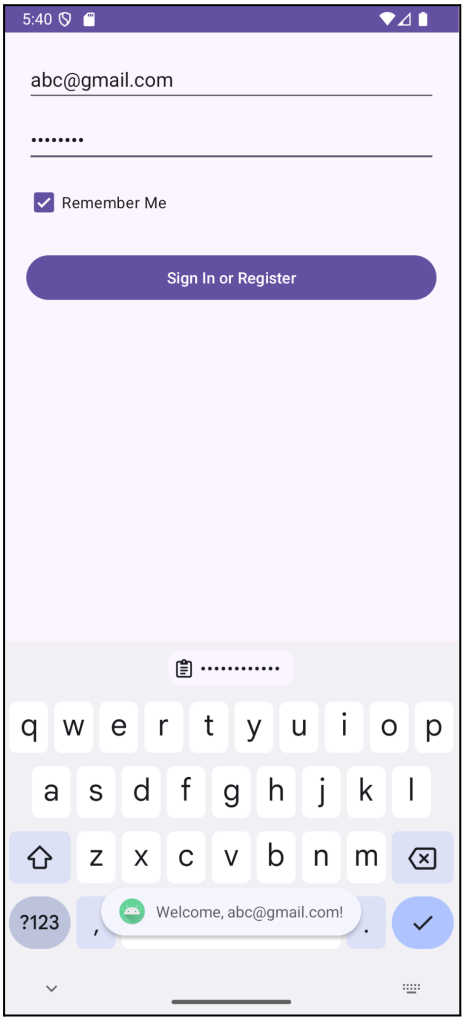
```
            editTextPassword.setText(savedPassword);
            checkBoxRememberMe.setChecked(true);
        }
    }

    private void clearSavedCredentials() {
        SharedPreferences.Editor editor = sharedPreferences.edit();
        editor.remove("Email");
        editor.remove("Password");
        editor.remove("RememberMe");
        editor.apply();
    }
}
```

Main_Activity.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"
    android:padding="20dp">

    <EditText
        android:id="@+id/editTextEmail"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter Email"
        android:inputType="textEmailAddress"
        android:layout_marginBottom="10dp" />

    <EditText
        android:id="@+id/editTextPassword"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter Password"
```

```
        android:inputType="textPassword"
        android:layout_marginBottom="10dp" />

    <CheckBox
        android:id="@+id/checkBoxRememberMe"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Remember Me"
        android:layout_marginBottom="20dp" />

    <Button
        android:id="@+id/buttonSignInOrRegister"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Sign In or Register" />

</LinearLayout>
```





Conclusion:

Successfully performed file operations with file I/O and created a simple login system using SharedPreferences for storing user credentials.

Name of Student: Pushkar Sane		
Roll Number: 45	**Lab Assignment Number: 6**	
Title of Lab Assignment: To perform the animation on an image and to apply various filters on an image.		
DOP: 10-09-2024	**DOS: 11-10-2024**	
CO Mapped: **CO2, CO4**	**PO Mapped:** **PO2, PO3, PO5, PSO1, PSO2**	**Signature:**

Practical No. 6

Aim: To perform the animation on an image and to apply various filters on an image.

1. Perform the following animation on the image:
    a. Move.
    b. Rotate.
    c. Expand.
2. Apply the following effects on the image:
    a. Brightness.
    b. Darkness.
    c. Grayscale.

Theory:

Steps to Add an Image to the Drawable Folder in Android Studio

1. Switch to Project View:
    a. In Android Studio, look for the Project panel on the top-left.
    b. Change the view from **Android** to **Project**.
2. Navigate to Drawable Folder:
    a. In the Project View, expand the folder structure as follows:
    app > src > main > res > drawable
3. Open Drawable Folder in File Explorer:
    a. Right-click on the drawable folder.
    b. Select Open in **Explorer** (Windows) or Reveal in **Finder** (macOS).
4. Add the Image: Once the drawable folder opens in your file explorer, copy your flower.jpg image file into this folder.

Code:

MainActivity.java

```
package com.example.practical6;
import android.graphics.ColorMatrix;
import android.graphics.ColorMatrixColorFilter;
import android.os.Bundle;
import android.view.animation.ScaleAnimation;
import android.view.animation.TranslateAnimation;
import android.view.animation.RotateAnimation;
import android.view.View;
```

```
import android.widget.Button;
import android.widget.ImageView;
import androidx.appcompat.app.AppCompatActivity;
public class MainActivity extends AppCompatActivity {
    private ImageView imageView;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        imageView = findViewById(R.id.imageView);
        Button btnMove = findViewById(R.id.btnMove);
        Button btnRotate = findViewById(R.id.btnRotate);
        Button btnExpand = findViewById(R.id.btnExpand);
        Button btnBrightness = findViewById(R.id.btnBrightness);
        Button btnDarkness = findViewById(R.id.btnDarkness);
        Button btnGrayscale = findViewById(R.id.btnGrayscale);

        // Animation: Move
        btnMove.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                TranslateAnimation translate = new TranslateAnimation(0, 300, 0, 300);
                translate.setDuration(1000);
                translate.setFillAfter(true);
                imageView.startAnimation(translate);
            }
        });

        // Animation: Rotate
        btnRotate.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                RotateAnimation rotate = new RotateAnimation(0, 360,
                        imageView.getWidth() / 2, imageView.getHeight() / 2);
                rotate.setDuration(1000);
```

```
        rotate.setFillAfter(true);
        imageView.startAnimation(rotate);
    }
});

// Animation: Expand
btnExpand.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        ScaleAnimation scale = new ScaleAnimation(1f, 2f, 1f, 2f,
                imageView.getWidth() / 2, imageView.getHeight() / 2);
        scale.setDuration(1000);
        scale.setFillAfter(true);
        imageView.startAnimation(scale);
    }
});

// Filter: Brightness
btnBrightness.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        ColorMatrix matrix = new ColorMatrix();
        matrix.set(new float[]{
                1.5f, 0, 0, 0, 0,
                0, 1.5f, 0, 0, 0,
                0, 0, 1.5f, 0, 0,
                0, 0, 0, 1, 0
        });
        imageView.setColorFilter(new ColorMatrixColorFilter(matrix));
    }
});

// Filter: Darkness
btnDarkness.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
```

```
            ColorMatrix matrix = new ColorMatrix();
            matrix.set(new float[]{
                    0.5f, 0, 0, 0, 0,
                    0, 0.5f, 0, 0, 0,
                    0, 0, 0.5f, 0, 0,
                    0, 0, 0, 1, 0
            });
            imageView.setColorFilter(new ColorMatrixColorFilter(matrix));
        }
    });

    // Filter: Grayscale
    btnGrayscale.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            ColorMatrix matrix = new ColorMatrix();
            matrix.setSaturation(0);
            imageView.setColorFilter(new ColorMatrixColorFilter(matrix));
        }
    });
  }
}
```

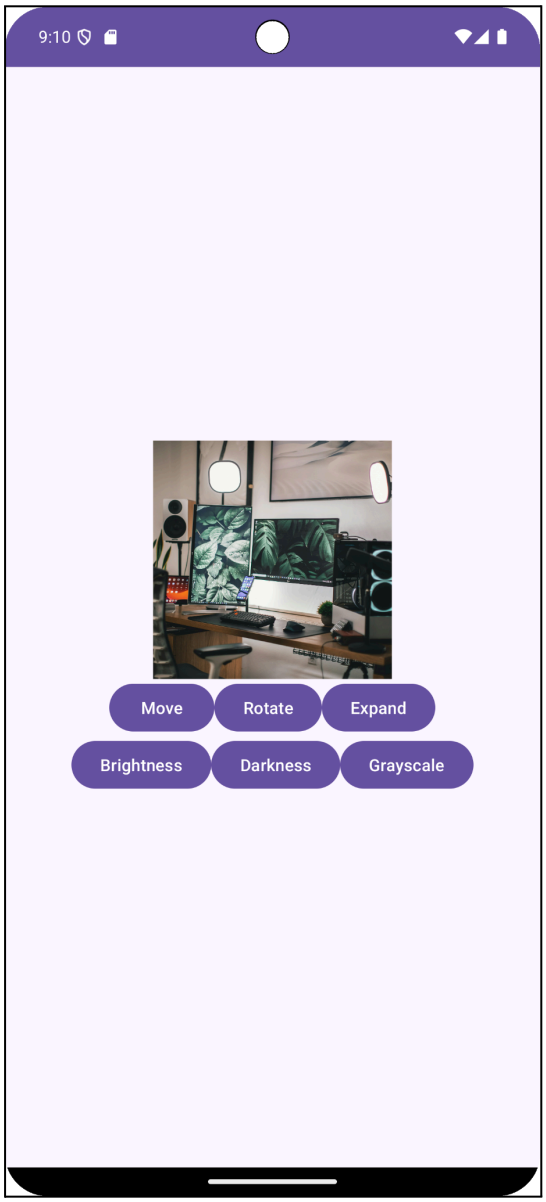
main_activity.xml

```
<?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"
  android:gravity="center">
  <ImageView
    android:id="@+id/imageView"
    android:layout_width="200dp"
    android:layout_height="200dp"
    android:src="@drawable/image"
```

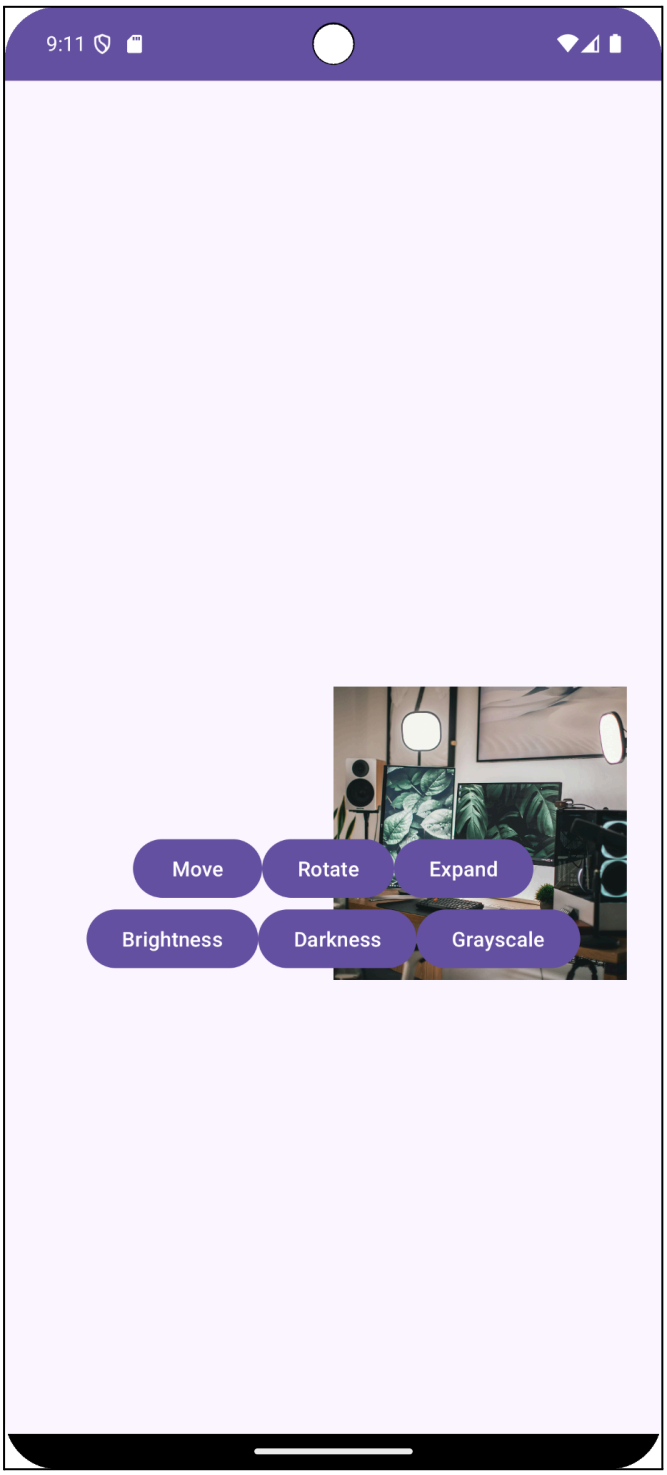
```
    android:scaleType="centerCrop"/>
  <LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <!-- Buttons for animations -->
    <Button
      android:id="@+id/btnMove"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Move"/>
    <Button
      android:id="@+id/btnRotate"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Rotate"/>
    <Button
      android:id="@+id/btnExpand"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Expand"/>
  </LinearLayout>
  <LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
    <!-- Buttons for filters -->
    <Button
      android:id="@+id/btnBrightness"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Brightness"/>
    <Button
      android:id="@+id/btnDarkness"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
```

```
        android:text="Darkness"/>
    <Button
        android:id="@+id/btnGrayscale"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Grayscale"/>
  </LinearLayout>
</LinearLayout>
```

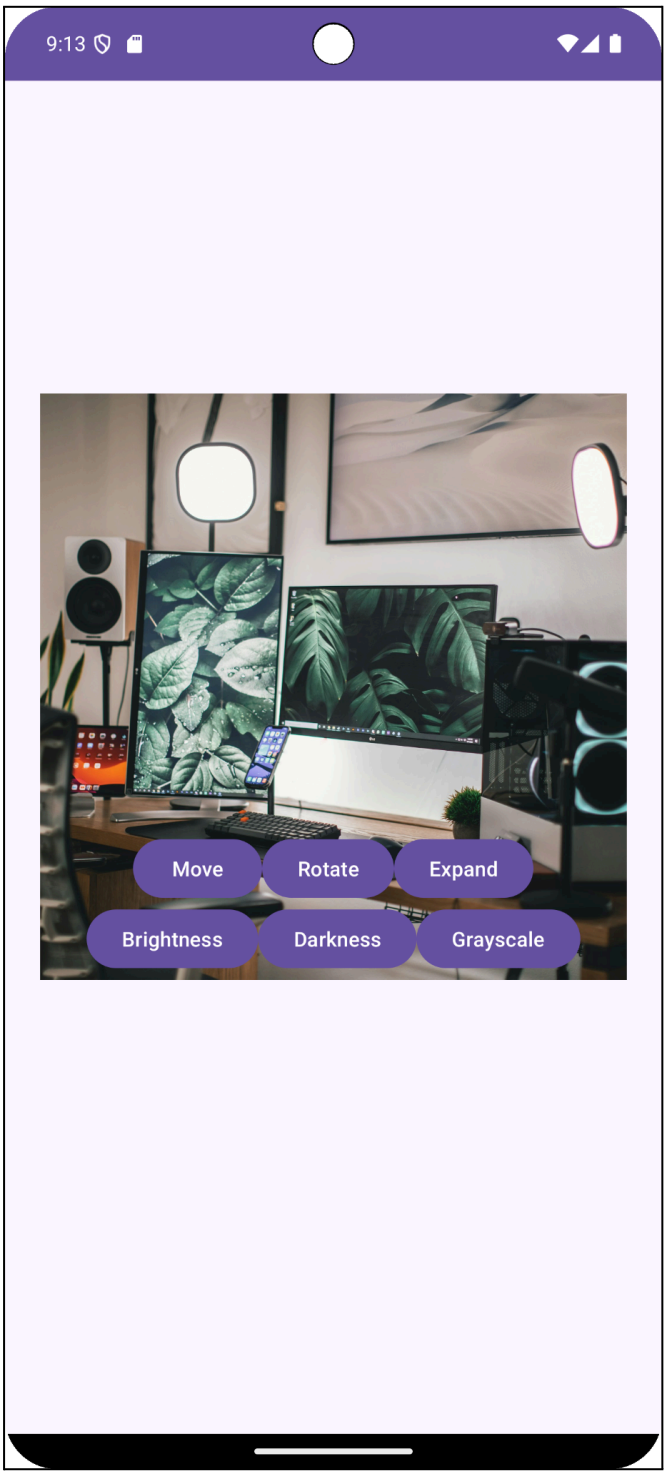
Output:



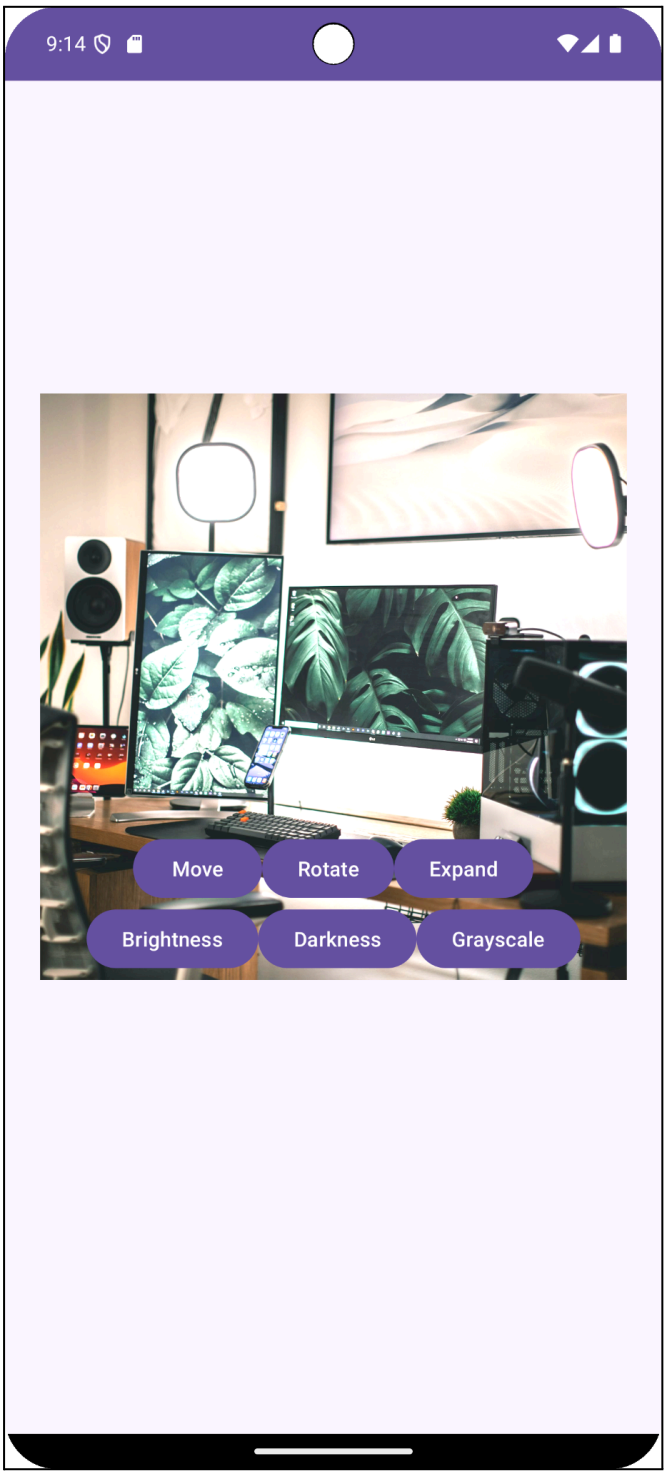
Move:



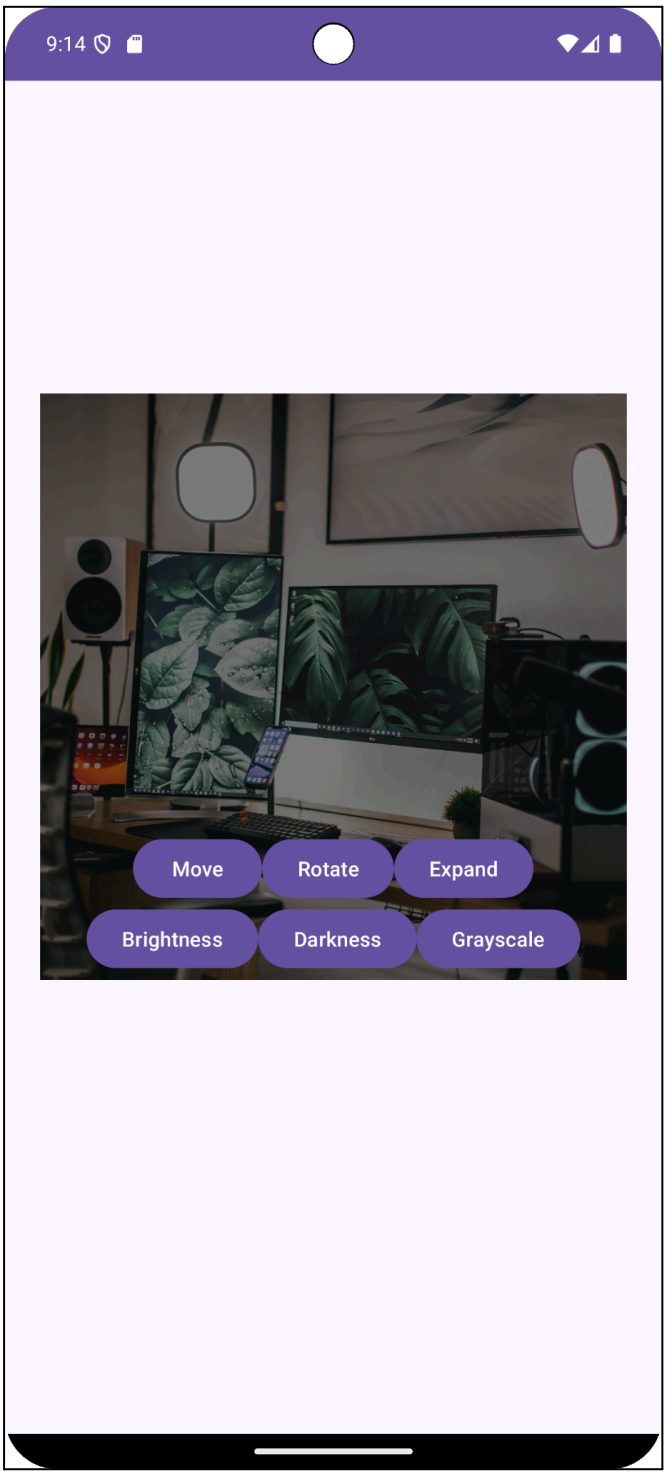
Expand:



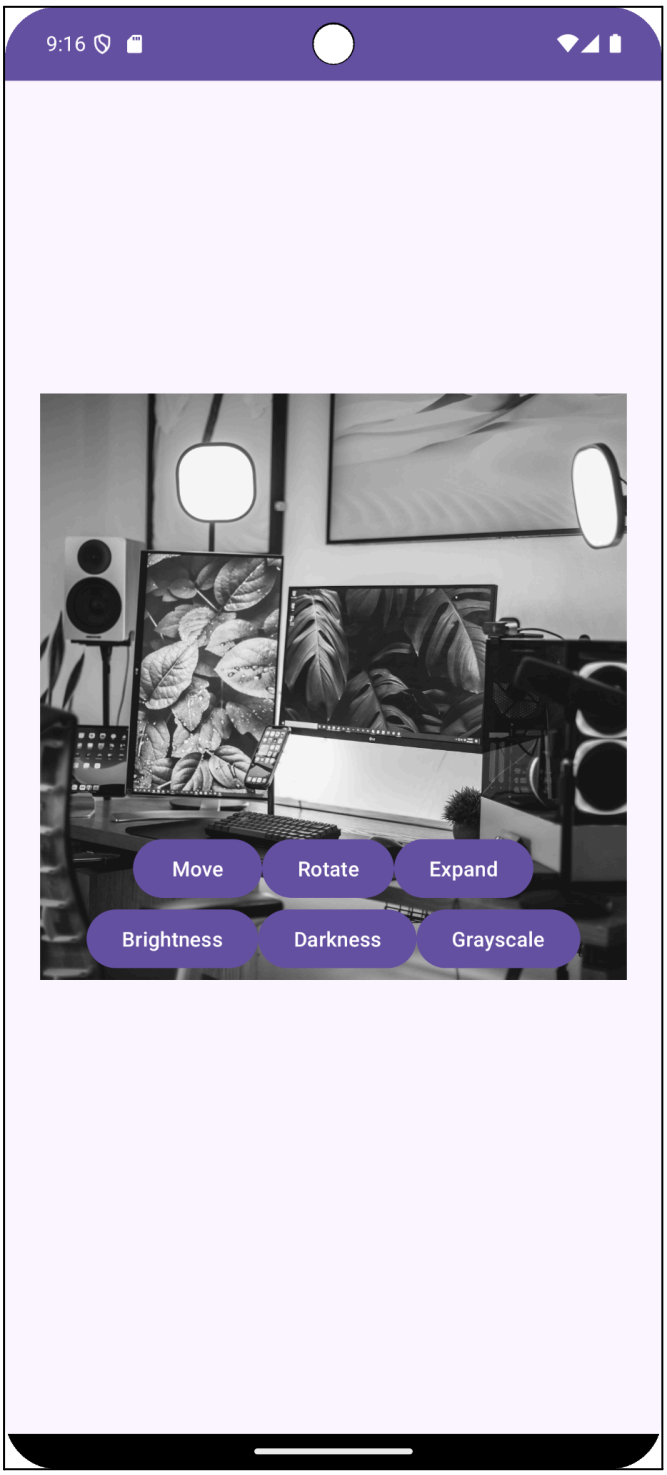
Brightness:



Darkness:



GreyScale:



Conclusion:

Successfully demonstrated implementation of animation on the image and applying effects on the image.

Name of Student: Pushkar Sane		
Roll Number: 45	**Lab Assignment Number: 7**	
Title of Lab Assignment: Android program to work with google maps and location and GPS		
DOP: 17-09-2024	**DOS: 19-10-2024**	
CO Mapped: **CO2, CO5**	**PO Mapped:** **PO2, PO3, PO5, PSO1, PSO2**	**Signature:**

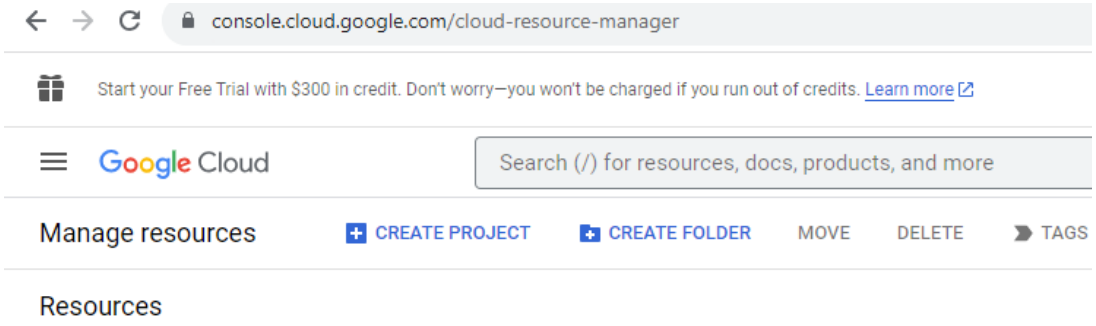
PRACTICAL 7

Aim: Android program to work with google maps and location "Add marker "method to be used in the application students are creating.

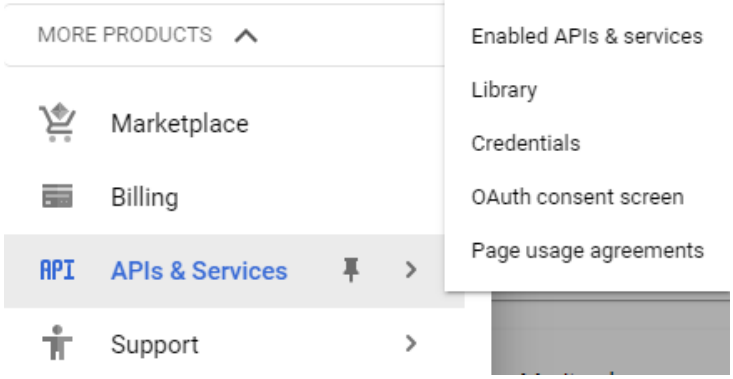
Theory:
Google Maps API: An API key is needed to access the Google Maps servers. This key is free, and you can use it with any of your applications. If you haven't created project, you can follow the below steps to get started:

Step 1: Open Google developer console and sign in with your gmail account: https://console.developers.google.com/project

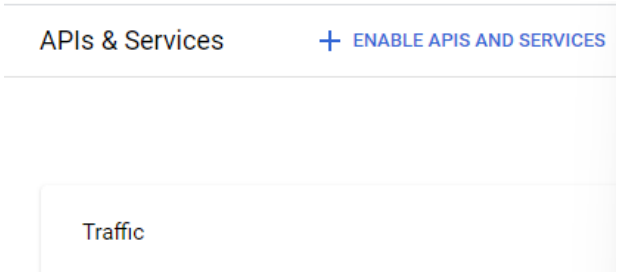
Step 2: Now create a new project. You can create a new project by clicking on the Create Project button and giving the name to your project.



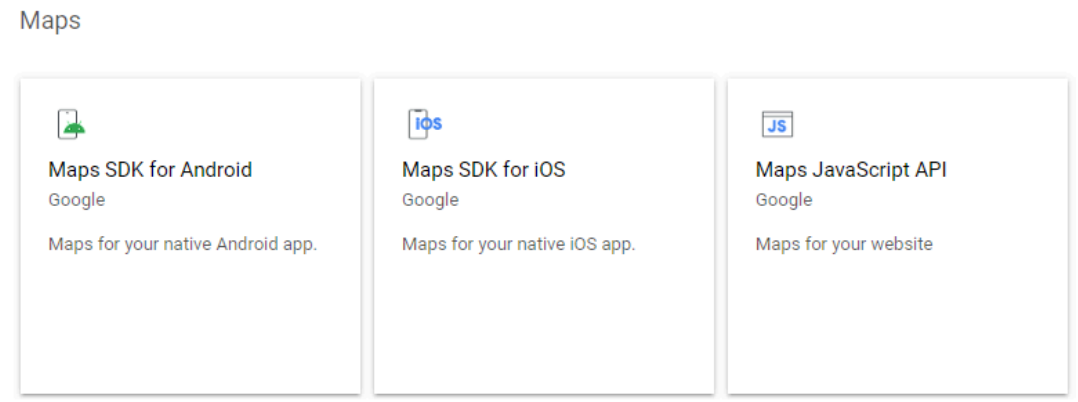
Step 3: Now click on APIs & Services and open Dashboard from it



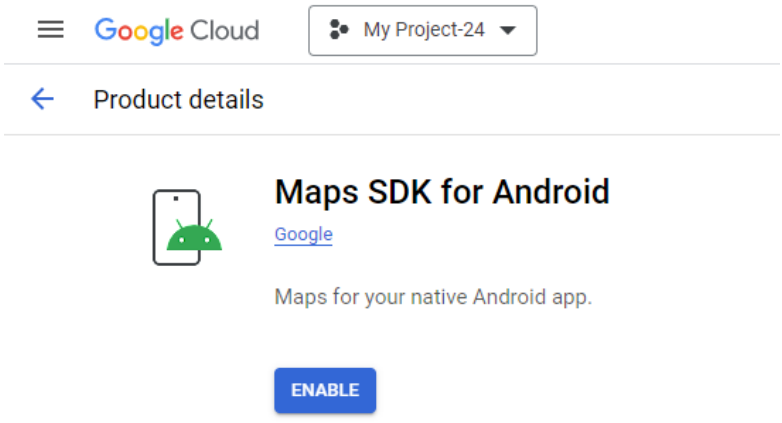
Step 4: In this open Enable APIS AND SERVICES



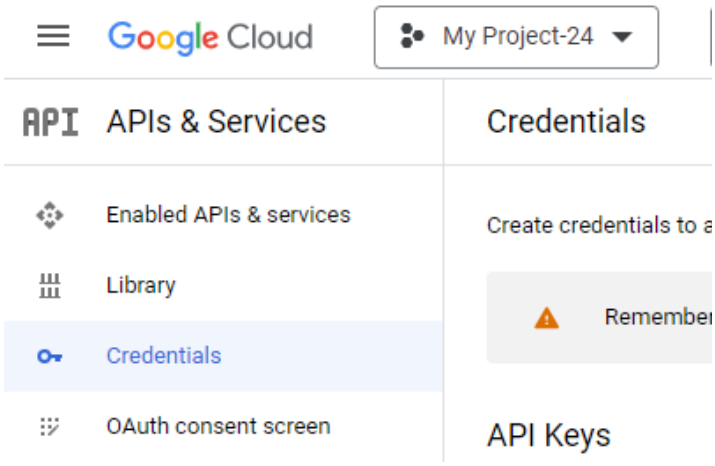
Step 5: Now open Map SDK for Android.



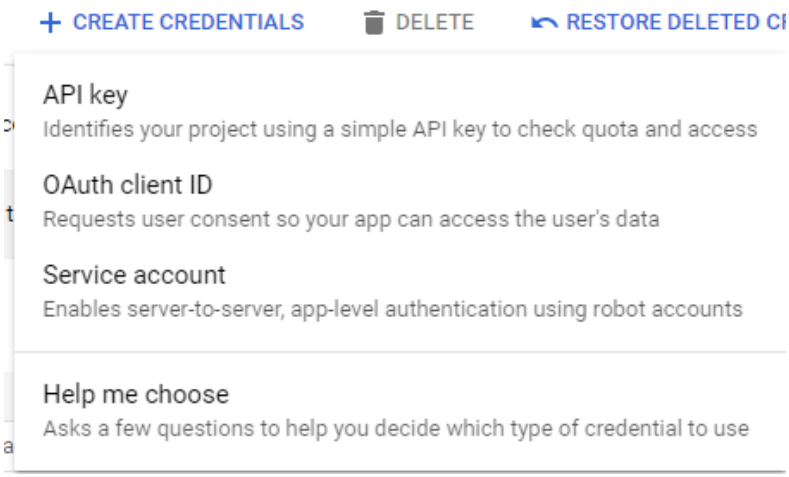
Step 6: Now enable the Google Maps Android API



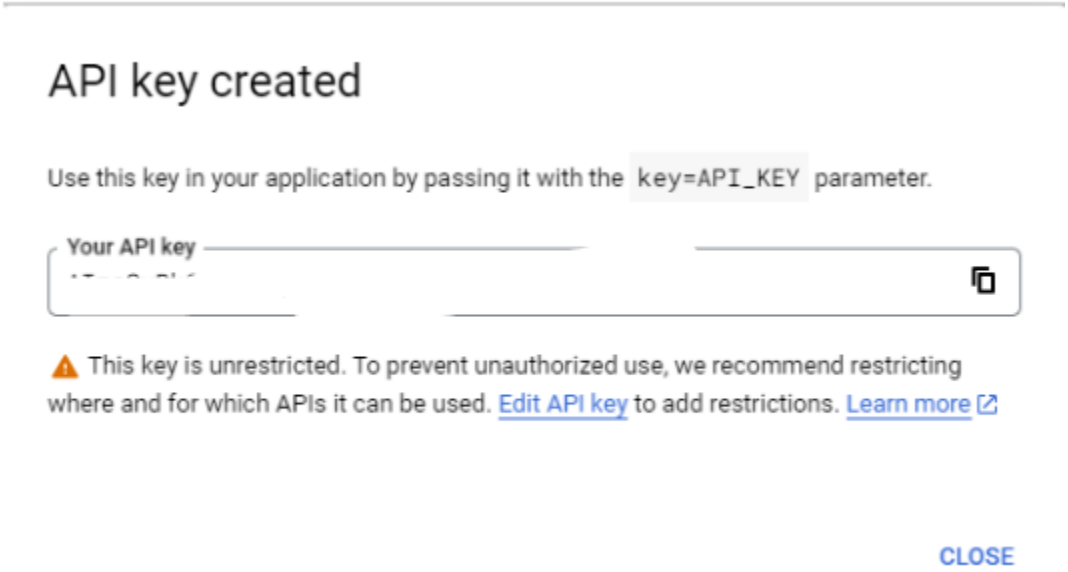
Step 7: Now go to Credentials



Step 8: Here click on Create credentials and choose API key



Step 9: Now your API key will be generated. Copy it and save it somewhere as we will need it when implementing Google Map in our Android project.



Strings.xml:

```
<resources>
  <string name="app_name">My Application_prac7</string>
  <string name="map_key"
translatable="false">AIzaSyBkQ7SsgYqle37tPs0BYLpXddFu6m0uCuk</string>
</resources>
```

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">
  <meta-data android:name="com.google.android.geo.API_KEY"
    android:value="@string/map_key"/>

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

```
  <application
     android:allowBackup="true"
     android:dataExtractionRules="@xml/data_extraction_rules"
     android:fullBackupContent="@xml/backup_rules"
     android:icon="@mipmap/ic_launcher"
     android:label="@string/app_name"
     android:roundIcon="@mipmap/ic_launcher_round"
     android:supportsRtl="true"
     android:theme="@style/Theme.MyApplication_prac7"
     tools:targetApi="31">
     <activity
        android:name=".MainActivity"
        android:exported="true">
        <intent-filter>
           <action android:name="android.intent.action.MAIN" />
           <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
        <meta-data
           android:name="android.app.lib_name"
           android:value="" />
     </activity>
  </application>
</manifest>
```

Dependencies:

```
implementation 'com.google.android.libraries.maps:maps:3.1.0-beta'
implementation 'com.google.android.gms:play-services-maps:18.0.0'
implementation 'com.google.android.gms:play-services-location:18.0.0'
implementation 'com.google.maps.android:android-maps-utils:2.3.0'
```

activity_main.xml:

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

```
    android:id="@+id/google_map"
    android:name="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:ignore="MissingClass" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

MainActivity.java:

```
package com.example.myapplication_prac7;

import androidx.appcompat.app.AppCompatActivity;
import android.Manifest;
import android.annotation.SuppressLint;
import android.content.pm.PackageManager;
import android.location.Location;
import android.os.Bundle;
import android.widget.Toast;
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.gms.tasks.OnSuccessListener;
import com.google.android.gms.tasks.Task;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import androidx.core.app.ActivityCompat;

public class MainActivity extends AppCompatActivity implements OnMapReadyCallback{
    Location currentLocation;
    GoogleMap gMap;
    FusedLocationProviderClient fusedLocationProviderClient;
    private static final int REQUEST_CODE = 101;

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

```
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    fusedLocationProviderClient = LocationServices.getFusedLocationProviderClient(this);
    fetchlastLocation();
  }
  private void fetchlastLocation() {
    if
(ActivityCompat.checkSelfPermission(this,Manifest.permission.ACCESS_FINE_LOCATION) !=
        PackageManager.PERMISSION_GRANTED &&
        ActivityCompat.checkSelfPermission(this,
            Manifest.permission.ACCESS_COARSE_LOCATION) !=
            PackageManager.PERMISSION_GRANTED) {
      ActivityCompat.requestPermissions(this, new String[]
          {Manifest.permission.ACCESS_FINE_LOCATION},REQUEST_CODE);
      return;
    }

    Task<Location> task = fusedLocationProviderClient.getLastLocation();
    task.addOnSuccessListener(new OnSuccessListener<Location>() {
      @Override
      public void onSuccess(Location location) {
        if(location != null){
          currentLocation = location;
          Toast.makeText(getApplicationContext(),currentLocation.getLatitude()
+""+currentLocation.getLongitude(), Toast.LENGTH_SHORT).show();
          SupportMapFragment supportMapFragment = (SupportMapFragment)
getSupportFragmentManager().findFragmentById(R.id.google_map);
          supportMapFragment.getMapAsync(MainActivity.this);
        }
      }
    });
  }
  @Override
  public void onMapReady(@NonNull GoogleMap googleMap) {
    double srcLat = currentLocation.getLatitude();
    double srcLong = currentLocation.getLongitude();
    LatLng latLng = new
LatLng(currentLocation.getLatitude(),currentLocation.getLongitude());
    MarkerOptions markerOptions = new MarkerOptions().position(latLng).title(srcLat + ":" +
srcLong);
```

```
    googleMap.animateCamera(CameraUpdateFactory.newLatLng(latLng));
    googleMap.animateCamera(CameraUpdateFactory.newLatLngZoom(latLng,10));
    googleMap.addMarker(markerOptions);
    gMap = googleMap;
    gMap.setOnMapClickListener(new GoogleMap.OnMapClickListener() {
      int count = -1;
      @Override
      public void onMapClick(@NonNull LatLng latLng) {
        MarkerOptions markerOptions1 = new MarkerOptions();
        markerOptions1.position(latLng);
        markerOptions1.title(latLng.latitude+ ":" + latLng.longitude);
        count++;
        if(count % 2 != 0){
          gMap.clear();
        }
        gMap.animateCamera(CameraUpdateFactory.newLatLng(latLng));
        gMap.animateCamera(CameraUpdateFactory.newLatLngZoom(latLng,10));
        gMap.addMarker(markerOptions1);
        double res = distance(srcLat,latLng.latitude,srcLong,latLng.longitude);
        Toast.makeText(getApplicationContext(), "Distance = "+ res,
Toast.LENGTH_SHORT).show();
      }
    });
  }
  public static double distance(double lat1, double lat2, double
      lon1,double lon2)
  {
    lon1 = Math.toRadians(lon1);
    lon2 = Math.toRadians(lon2);
    lat1 = Math.toRadians(lat1);
    lat2 = Math.toRadians(lat2);
    double dlon = lon2 - lon1;
    double dlat = lat2 - lat1;
    double a = Math.pow(Math.sin(dlat / 2), 2) + Math.cos(lat1) * Math.cos(lat2) *
Math.pow(Math.sin(dlon / 2),2);
    double c = 2 * Math.asin(Math.sqrt(a));
    double r = 6371;

    return(c * r);
  }
```

```
  @SuppressLint("MissingSuperCall")
  @Override
  public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
@NonNull int[] grantResults) {
      switch (requestCode) {
         case REQUEST_CODE:
            if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED);
            fetchlastLocation();
            break;
      }
  }
}
```

Output:

Conclusion:I have successfully implemented an android program to work with google maps and location .

Name of Student: Pushkar Sane		
Roll Number: 45	**Lab Assignment Number: 8**	
Title of Lab Assignment: To Write a program to record and play audio and video.		
DOP: 20-10-2024	**DOS: 20-10-2024**	
CO Mapped: **CO5**	**PO Mapped:** **PO1, PO3, PO5, PSO2**	**Signature:**

Practical No. 8

Aim: To Write a program to record and play audio and video.
A) Record an audio and play
B) Play a video in Videoview.

1. Recording Audio

Theory: The `MediaRecorder` class is used for recording audio in Android. It provides an API to configure the audio source, format, and encoder. The recording is saved as a file in a specified location on the device. Android 10 and above requires runtime permission handling for recording audio, so the app must request permissions at runtime.

Steps:

- Create a `MediaRecorder` object and set the audio source to `MediaRecorder.AudioSource.MIC` for recording from the microphone.
- Set the output format using `MediaRecorder.OutputFormat.THREE_GPP`.
- Set the audio encoder as `MediaRecorder.AudioEncoder.AMR_NB` for the `.3gp` format.
- Prepare the `MediaRecorder` and start recording.
- The recorded audio is saved to the external storage of the device.

---

2. Playing Recorded Audio

Theory: The `MediaPlayer` class is used for playing media files such as audio and video. After recording the audio, the same file can be played using the `MediaPlayer`. The `setDataSource()` method is used to specify the location of the file, and the `prepare()` method prepares the player for playback.

Steps:

- Create a `MediaPlayer` object and set the data source to the path of the recorded audio file.
- Call `prepare()` to load the file and start playing using the `start()` method.

---

3. Playing Video

Theory: A `VideoView` widget is used to display and play video files in Android. It can play video files from different sources such as local resources or from a URL. Here, the video is played from the local raw resources folder.

Steps:

- Use a `VideoView` to play the video from the raw resource folder using a `Uri`.
- Start the video using the `start()` method.
- Pause the video using the `pause()` method when required.

---

4. Displaying Current Location

Theory: The `FusedLocationProviderClient` is part of the Google Play services location APIs and is used to retrieve the device's last known location. It combines signals from various location providers (GPS, Wi-Fi, cell networks, etc.) to provide location data with high accuracy. Android requires runtime permissions to access fine or coarse location data.

Steps:

- Check for location permissions using `ActivityCompat.checkSelfPermission()`.
- Use the `FusedLocationProviderClient` to get the last known location of the device.
- Once the location is retrieved, display the latitude and longitude in a `TextView`.

---

5. Handling Permissions

Theory: Starting from Android 6.0 (API 23), dangerous permissions such as recording audio and accessing location must be requested at runtime. These permissions are defined in the `AndroidManifest.xml` and are requested in the code if they are not already granted.

Steps:

- Check if the necessary permissions (like `RECORD_AUDIO` and `ACCESS_FINE_LOCATION`) are granted.
- If not, request permissions using `ActivityCompat.requestPermissions()`.
- Handle the permission request result in `onRequestPermissionsResult()` to proceed with the functionality.

Code-

AndroidManifest.xml

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

  <application
    android:allowBackup="true"
    android:dataExtractionRules="@xml/data_extraction_rules"
    android:fullBackupContent="@xml/backup_rules"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/Theme.Practical8App"
    tools:targetApi="31">
    <activity
```

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

<!-- Permissions -->
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION" />
</manifest>
```

MainActivity.java

```
package com.example.practical8app;


import android.Manifest;
import android.content.pm.PackageManager;
import android.location.Location;
import android.media.MediaPlayer;
import android.media.MediaRecorder;
```

```
import android.net.Uri;
import android.os.Bundle;
import android.widget.Button;
import android.widget.TextView;
import android.widget.VideoView;
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.tasks.OnSuccessListener;
import java.io.IOException;

public class MainActivity extends AppCompatActivity {

  private MediaRecorder myAudioRecorder;
  private String outputFile;
  private MediaPlayer mediaPlayer;
  private FusedLocationProviderClient fusedLocationClient;
  private TextView locationText;
  private boolean isRecording = false;

  @Override
```

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

    // Initialize components
    Button recordButton = findViewById(R.id.recordButton);
    Button stopRecordButton = findViewById(R.id.stopRecordButton);
    Button playButton = findViewById(R.id.playButton);
    Button playVideoButton = findViewById(R.id.playVideoButton);
    Button pauseVideoButton = findViewById(R.id.pauseVideoButton);
    Button showLocationButton = findViewById(R.id.showLocationButton);
    VideoView simpleVideoView = findViewById(R.id.simpleVideoView);
    locationText = findViewById(R.id.locationText);
    fusedLocationClient = LocationServices.getFusedLocationProviderClient(this);

    // Set the output file path for audio recording
    outputFile = getExternalFilesDir(null).getAbsolutePath() + "/myrecording.3gp";

    // Check and request audio recording permission
    if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.RECORD_AUDIO) !=
PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.RECORD_AUDIO}, 200);
```

```
    }

    // Audio Recording
    recordButton.setOnClickListener(v -> startRecording());

    // Stop Audio Recording
    stopRecordButton.setOnClickListener(v -> stopRecording());

    // Play Audio
    playButton.setOnClickListener(v -> playAudio());

    // Play Video
    playVideoButton.setOnClickListener(v -> playVideo(simpleVideoView));

    // Pause Video
    pauseVideoButton.setOnClickListener(v -> pauseVideo(simpleVideoView));

    // Show Location
    showLocationButton.setOnClickListener(v -> displayLocation());
}

// Method for starting audio recording
private void startRecording() {
```

```
        if (!isRecording) {
            try {
                myAudioRecorder = new MediaRecorder();
                myAudioRecorder.setAudioSource(MediaRecorder.AudioSource.MIC);
myAudioRecorder.setOutputFormat(MediaRecorder.OutputFormat.THREE_GPP);
                myAudioRecorder.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);
                myAudioRecorder.setOutputFile(outputFile);

                myAudioRecorder.prepare();
                myAudioRecorder.start();
                isRecording = true;
            } catch (IOException e) {
                e.printStackTrace();
            } catch (IllegalStateException e) {
                e.printStackTrace();
            }
        }
    }

    // Method for stopping audio recording
    private void stopRecording() {
        if (isRecording) {
```

```
            myAudioRecorder.stop();
            myAudioRecorder.release();
            myAudioRecorder = null;
            isRecording = false;
        }
    }

    // Method for playing the recorded audio
    private void playAudio() {
        mediaPlayer = new MediaPlayer();
        try {
            mediaPlayer.setDataSource(outputFile);
            mediaPlayer.prepare();
            mediaPlayer.start();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    // Method to play video
    private void playVideo(VideoView videoView) {
        Uri videoUri = Uri.parse("android.resource://" + getPackageName() + "/" +
R.raw.samplevideo);
```

```
        videoView.setVideoURI(videoUri);
        videoView.start();
    }

    // Method to pause video
    private void pauseVideo(VideoView videoView) {
        if (videoView.isPlaying()) {
            videoView.pause();
        }
    }

    // Method to get current location
    private void displayLocation() {
        if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED &&
                ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.ACCESS_FINE_LOCATION}, 101);
            return;
        }

        fusedLocationClient.getLastLocation()
```

```
            .addOnSuccessListener(this, location -> {
                if (location != null) {
                    double latitude = location.getLatitude();
                    double longitude = location.getLongitude();
                    locationText.setText("Latitude: " + latitude + ", Longitude: " + longitude);
                }
            });
    }

    // Handle permission results
    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        if (requestCode == 101 && grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
            displayLocation();
        } else if (requestCode == 200 && grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
            startRecording();
        }
    }
}
```

activity_main.xml

```
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:padding="16dp">

  <!-- Record and Stop Record Audio Buttons -->
  <Button
    android:id="@+id/recordButton"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Record Audio"
    android:layout_marginTop="16dp"/>

  <Button
    android:id="@+id/stopRecordButton"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Stop Recording"
    android:layout_marginTop="16dp"/>

  <!-- Play and Pause Audio Buttons -->
```

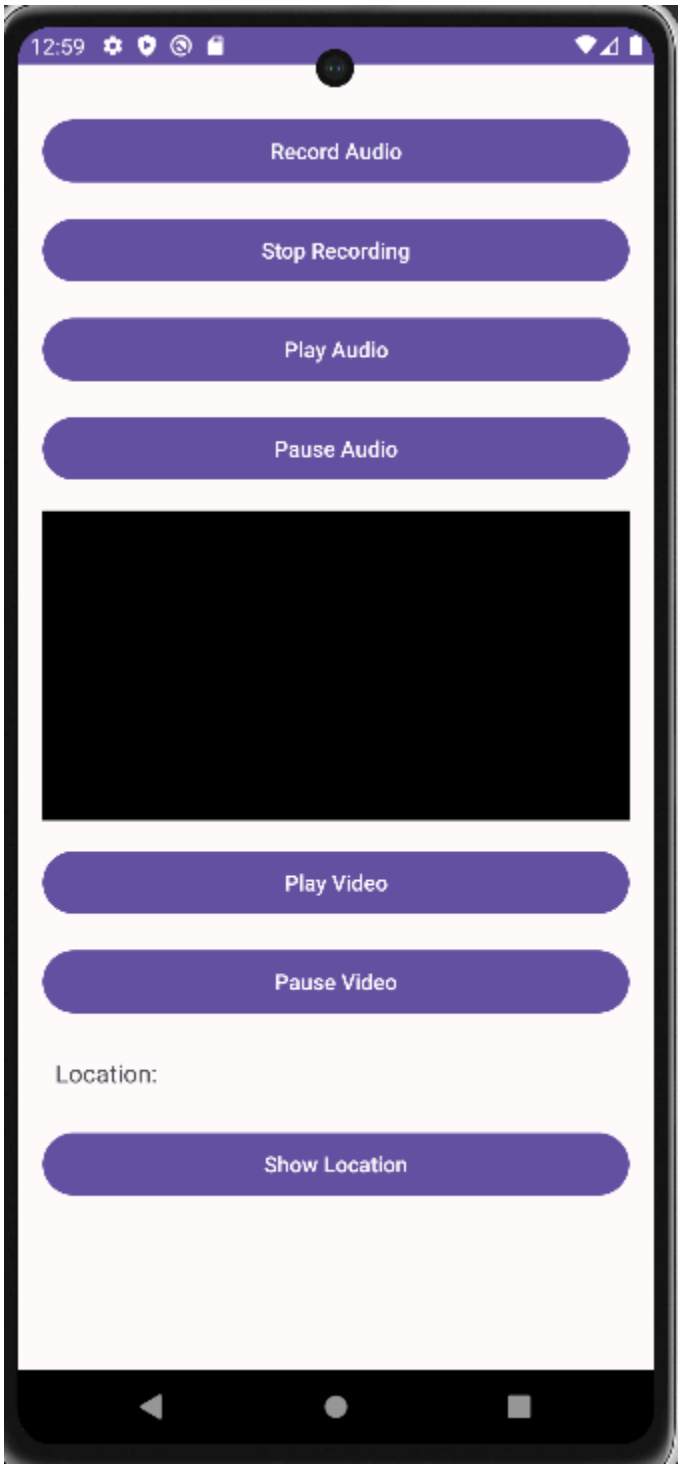
```
<Button
    android:id="@+id/playButton"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Play Audio"
    android:layout_marginTop="16dp"/>

<Button
    android:id="@+id/pauseAudioButton"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Pause Audio"
    android:layout_marginTop="16dp"/>

<!-- VideoView for Playing Video -->
<VideoView
    android:id="@+id/simpleVideoView"
    android:layout_width="match_parent"
    android:layout_height="200dp"
    android:layout_marginTop="16dp"/>

<!-- Play and Pause Video Buttons -->
<Button
```

```
    android:id="@+id/playVideoButton"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Play Video"
    android:layout_marginTop="16dp"/>

<Button
    android:id="@+id/pauseVideoButton"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Pause Video"
    android:layout_marginTop="16dp"/>

<!-- TextView to Display Location -->
<TextView
    android:id="@+id/locationText"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Location: "
    android:padding="8dp"
    android:textSize="16sp"
    android:layout_marginTop="16dp"/>
```

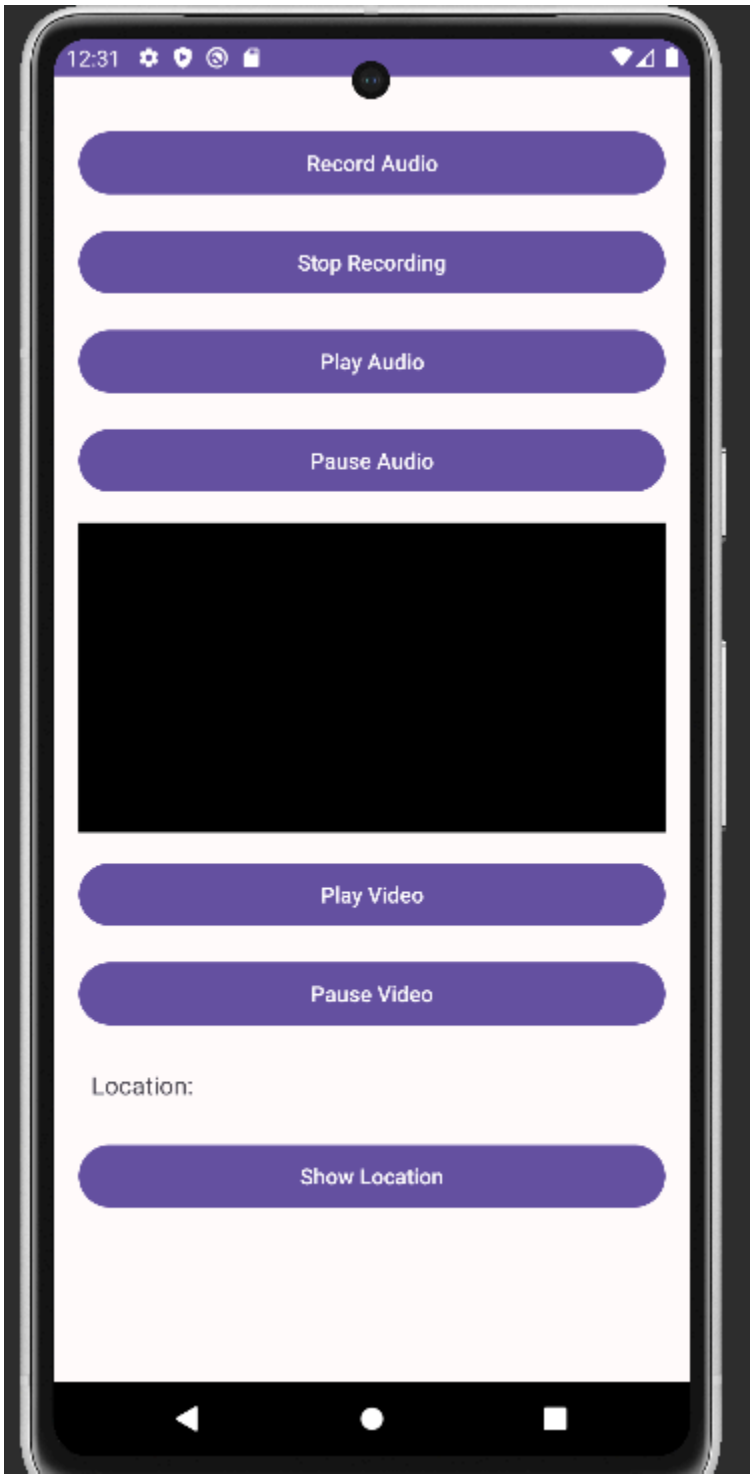
```
    <!-- Button to Show Location -->
    <Button
        android:id="@+id/showLocationButton"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Show Location"
        android:layout_marginTop="16dp"/>
</LinearLayout>
```

Output

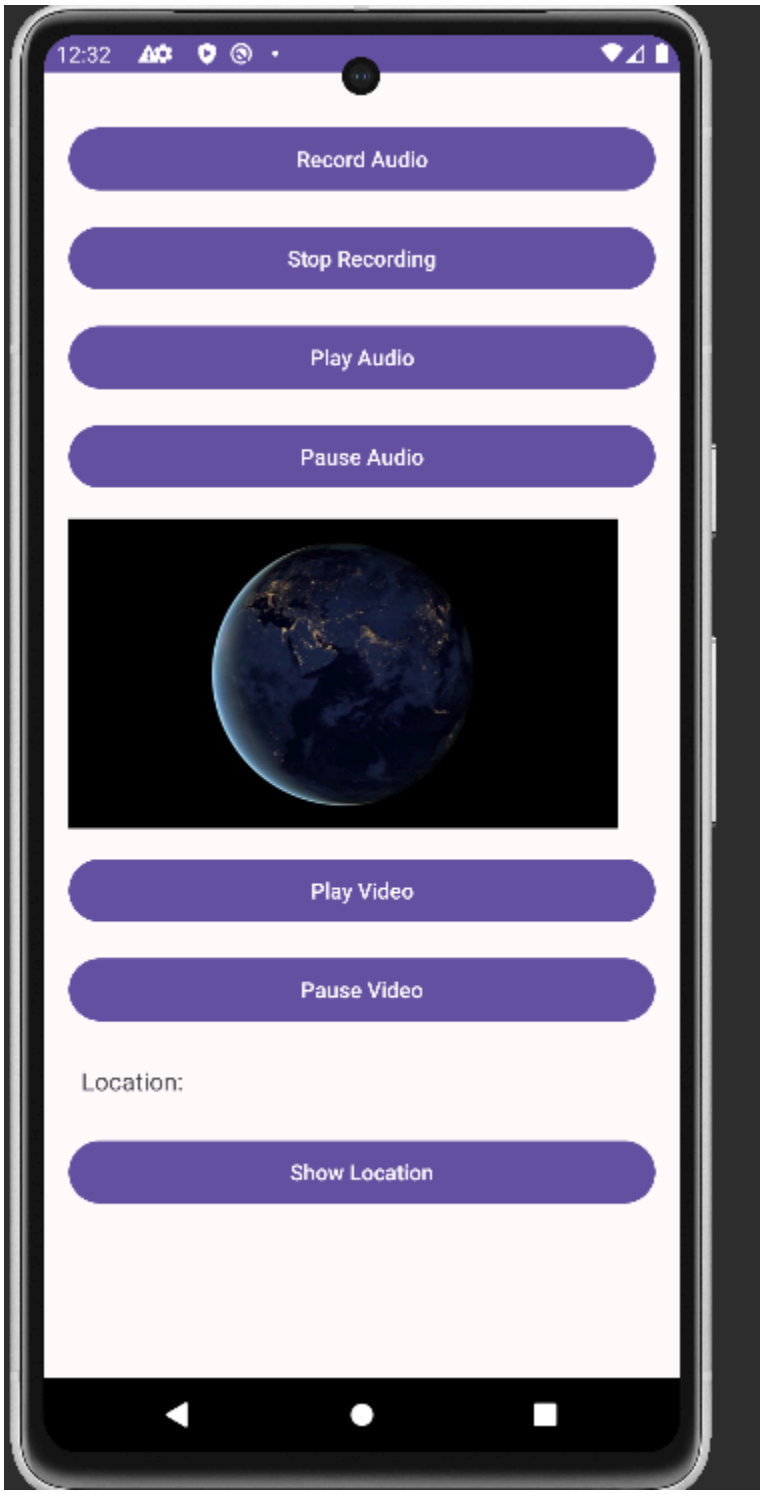
Record Audio


12:59
Record Audio
Stop Recording
Play Audio
Pause Audio
Play Video
Pause Video
Location:
Show Location

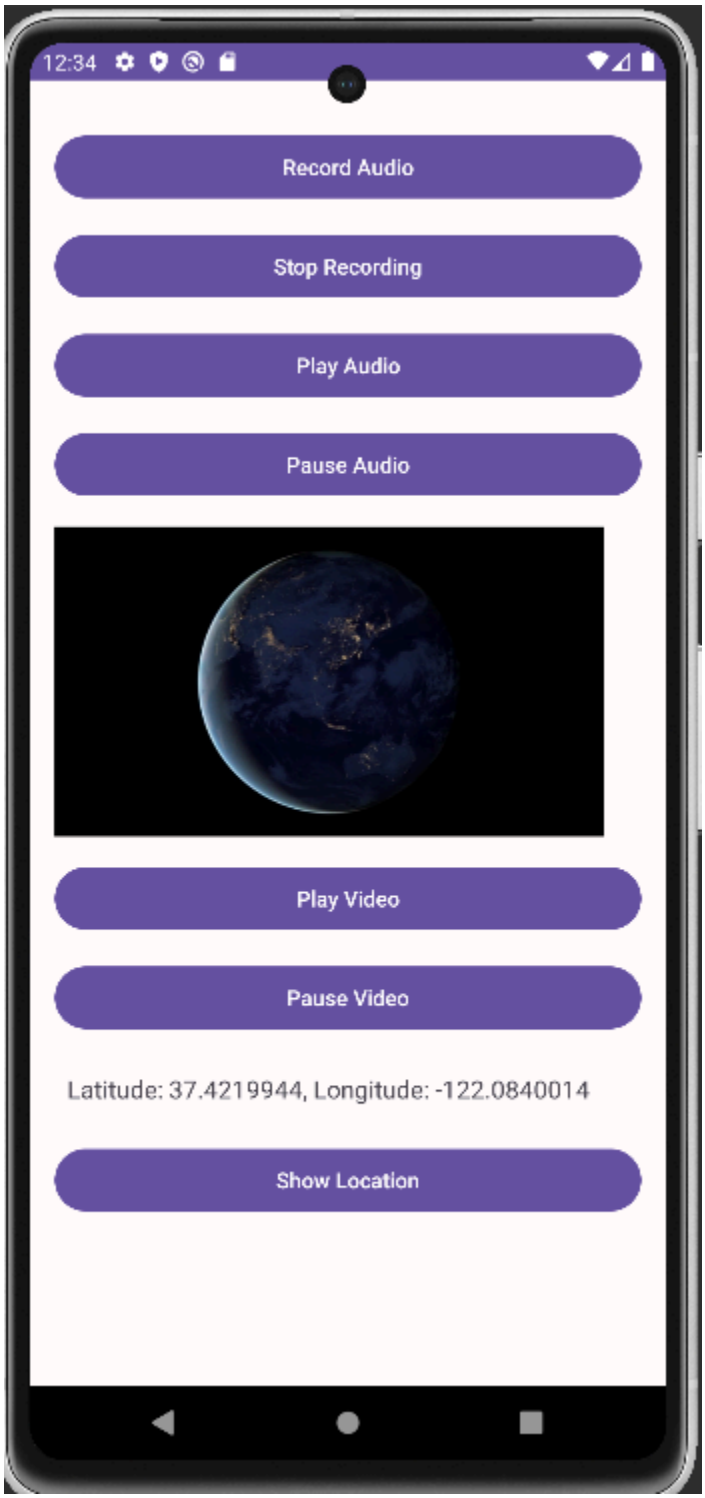
Stop Audio


12:31
Record Audio
Stop Recording
Play Audio
Pause Audio
Play Video
Pause Video
Location:
Show Location

Start Video



Show location


12:34
Record Audio
Stop Recording
Play Audio
Pause Audio
Play Video
Pause Video
Latitude: 37.4219944, Longitude: -122.0840014
Show Location

Conclusion

This demonstrates how to handle audio recording, playback, video playing, and location services in Android. It highlights key concepts such as `MediaRecorder`, `MediaPlayer`, `VideoView`, and `FusedLocationProviderClient` and ensures the proper management of runtime permissions for sensitive actions like recording and accessing location.

Name of Student: Pushkar Sane		
Roll Number: 45	**Lab Assignment Number: 9**	
Title of Lab Assignment: Android program based on Rest API.		
DOP: 24-10-2024	**DOS: 25-10-2024**	
CO Mapped: **CO2, CO5**	**PO Mapped:** **PO2, PO3, PO5, PSO2**	**Signature:**

Practical No. 9

Aim: Android program based on Rest API.

1. Create a basic application that allows you to download HTML from a given web page using HttpURLConnection.
2. Create an application to parse the data using JSON Object methods and set it in the Text View's. (Employee name and salary stored in JSON format

Theory:

1. **HttpURLConnection Class in Android:** The HttpURLConnection class in Android facilitates communication with REST APIs over HTTP. This class handles network requests and responses, making it ideal for interacting with web services.

   Key methods of HttpURLConnection:

   1. openConnection(): Opens a connection to the specified URL.
   2. setRequestMethod(): Sets the HTTP method (GET, POST, etc.) for the request.
   3. getInputStream(): Retrieves the input stream from the server response.
   4. getOutputStream(): Writes data to the server (used in POST/PUT requests).
   5. getResponseCode(): Checks the HTTP response status (e.g., 200 for success).
   6. disconnect(): Closes the connection once the task is complete.

   Steps to Use HttpURLConnection:

   1. Establish the Connection: Create a URL object and call openConnection() to establish the connection.
   2. Configure the Request: Set properties like request method (GET or POST), headers, timeouts, etc.
   3. Send Data (if applicable): For POST requests, write data to the server using OutputStream.
   4. Read Response: Use InputStream to read the response from the server. For HTML data, it can be processed line by line using BufferedReader.
   5. Disconnect: Close the connection to free resources using disconnect().

2. **JSON (JavaScript Object Notation)**

   JSON is a lightweight data-interchange format that is easy to read and write for humans and easy to parse and generate for machines. It is widely used in REST APIs to exchange data between the client and server.

- Structure of JSON:
  - JSONObject: A collection of key/value pairs. The keys are strings, and the values can be strings, numbers, booleans, arrays, or other JSON objects.
  - JSONArray: An ordered list of values, which can be of any JSON type (e.g., strings, numbers, or objects).

Code:

MainActivity.java

```
package com.example.practical9;
import androidx.appcompat.app.AppCompatActivity;
import android.app.ProgressDialog;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;

public class MainActivity extends AppCompatActivity {
    EditText urlInput;
    TextView htmlContent;
    Button fetchHtmlButton;
    ProgressDialog progressDialog;

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

        urlInput = findViewById(R.id.urlInput);
        htmlContent = findViewById(R.id.htmlContent);
```

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

    fetchHtmlButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            String urlString = urlInput.getText().toString();
            new FetchHtmlTask().execute(urlString);
        }
    });
}

// AsyncTask to fetch HTML content
private class FetchHtmlTask extends AsyncTask<String, String, String> {
    @Override
    protected void onPreExecute() {
        super.onPreExecute();
        progressDialog = new ProgressDialog(MainActivity.this);
        progressDialog.setMessage("Fetching HTML...");
        progressDialog.setCancelable(false);
        progressDialog.show();
    }

    @Override
    protected String doInBackground(String... params) {
        String urlString = params[0];
        String result = "";
        try {
            URL url = new URL(urlString);
            HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();
            InputStream in = urlConnection.getInputStream();
            InputStreamReader isw = new InputStreamReader(in);

            int data = isw.read();
            while (data != -1) {
                result += (char) data;
                data = isw.read();
```

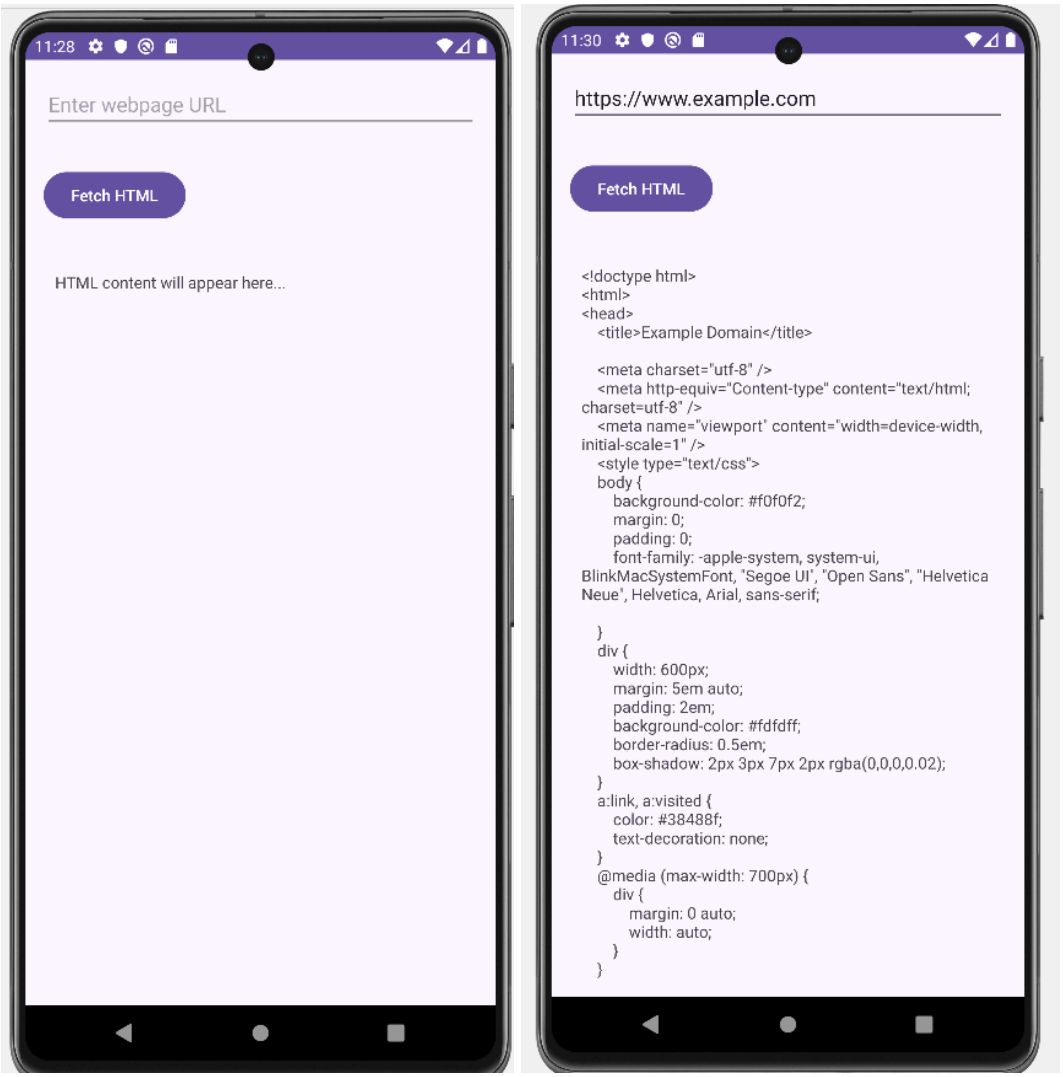
```
            }

            return result;
        } catch (Exception e) {
            e.printStackTrace();
            return "Error fetching HTML";
        }
    }

    @Override
    protected void onPostExecute(String result) {
        super.onPostExecute(result);
        progressDialog.dismiss();
        htmlContent.setText(result);
    }
  }
}
```

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">

    <!-- Input for URL -->
    <EditText
        android:id="@+id/urlInput"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter webpage URL"
        android:inputType="textUri"
        android:layout_margin="16dp" />

    <!-- Button to Fetch HTML -->
    <Button
        android:id="@+id/fetchHtmlButton"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Fetch HTML"
        android:layout_below="@id/urlInput"
        android:layout_margin="16dp" />

    <!-- TextView to display the fetched HTML content -->
    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/fetchHtmlButton"
        android:layout_margin="16dp">

        <TextView
            android:id="@+id/htmlContent"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="HTML content will appear here..."
            android:textIsSelectable="true"
            android:padding="10dp" />
    </ScrollView>
</RelativeLayout>
```

Output:


11:28
Enter webpage URL
Fetch HTML
HTML content will appear here...
11:30
https://www.example.com
Fetch HTML
<!doctype html>
<html>
<head>
<title>Example Domain</title>
<meta charset="utf-8" />
<meta http-equiv="Content-type" content="text/html; charset=utf-8" />
<meta name="viewport" content="width=device-width, initial-scale=1" />
<style type="text/css">
body {
background-color: #f0f0f2;
margin: 0;
padding: 0;
font-family: -apple-system, system-ui, BlinkMacSystemFont, 'Segoe UI', "Open Sans", "Helvetica Neue', Helvetica, Arial, sans-serif;
}
div {
width: 600px;
margin: 5em auto;
padding: 2em;
background-color: #fdfdff;
border-radius: 0.5em;
box-shadow: 2px 3px 7px 2px rgba(0,0,0,0.02);
}
a:link, a:visited {
color: #38488f;
text-decoration: none;
}
@media (max-width: 700px) {
div {
margin: 0 auto;
width: auto;
}
}

3. **REST API Overview**

REST (Representational State Transfer) is a set of architectural principles used for designing web services that allow communication over the internet. REST APIs use standard HTTP methods to perform operations on resources, where each resource is identified by a URI (Uniform Resource Identifier). RESTful services are widely used because of their simplicity, flexibility, and ability to use lightweight data formats such as JSON or XML.

- HTTP Methods in REST API:
  - GET: Retrieves data from the server (e.g., fetching employee details).
  - POST: Sends data to the server to create a new resource (e.g., adding a new employee).
  - PUT: Updates an existing resource (e.g., updating employee salary).
  - DELETE: Removes a resource from the server (e.g., deleting an employee record).
- Advantages of REST API:
  - Stateless: Each request from the client to the server must contain all the information the server needs to fulfill the request, making it stateless and scalable.
  - Cacheable: Responses can be cached to improve performance.
  - Platform Independent: REST API can be used across different programming languages and platforms.
  - Lightweight: REST primarily uses JSON, which is less verbose than XML, making data transfer faster and more efficient.

Code:

MainActivity.java

```
package com.example.practical9;

import androidx.appcompat.app.AppCompatActivity;
import android.app.ProgressDialog;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import java.io.InputStream;
```

```
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;
import org.json.JSONArray;
import org.json.JSONObject;

public class MainActivity extends AppCompatActivity {
    String myUrl = "https://dummyjson.com/c/7bb9-58b5-4d5f-8584";
    TextView resultsTextView;
    Button displayData;
    ProgressDialog progressDialog;

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

        resultsTextView = findViewById(R.id.results);
        displayData = findViewById(R.id.displayData);

        displayData.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                new MyAsyncTasks().execute();
            }
        });
    }

    private class MyAsyncTasks extends AsyncTask<String, String, String> {
        @Override
        protected void onPreExecute() {
            super.onPreExecute();
            progressDialog = new ProgressDialog(MainActivity.this);
            progressDialog.setMessage("Fetching data...");
            progressDialog.setCancelable(false);
            progressDialog.show();
```

```
    }

    @Override
    protected String doInBackground(String... params) {
        String result = "";
        try {
            URL url = new URL(myUrl);
            HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();
            InputStream in = urlConnection.getInputStream();
            InputStreamReader isw = new InputStreamReader(in);

            int data = isw.read();
            while (data != -1) {
                result += (char) data;
                data = isw.read();
            }

            return result;
        } catch (Exception e) {
            e.printStackTrace();
            return null;
        }
    }

    @Override
    protected void onPostExecute(String result) {
        super.onPostExecute(result);
        progressDialog.dismiss();

        try {
            // Parse the JSON result
            JSONObject jsonObject = new JSONObject(result);

            // Extract the "Names" array
            JSONArray namesArray = jsonObject.getJSONArray("Names");
            // Build a string with the names
```

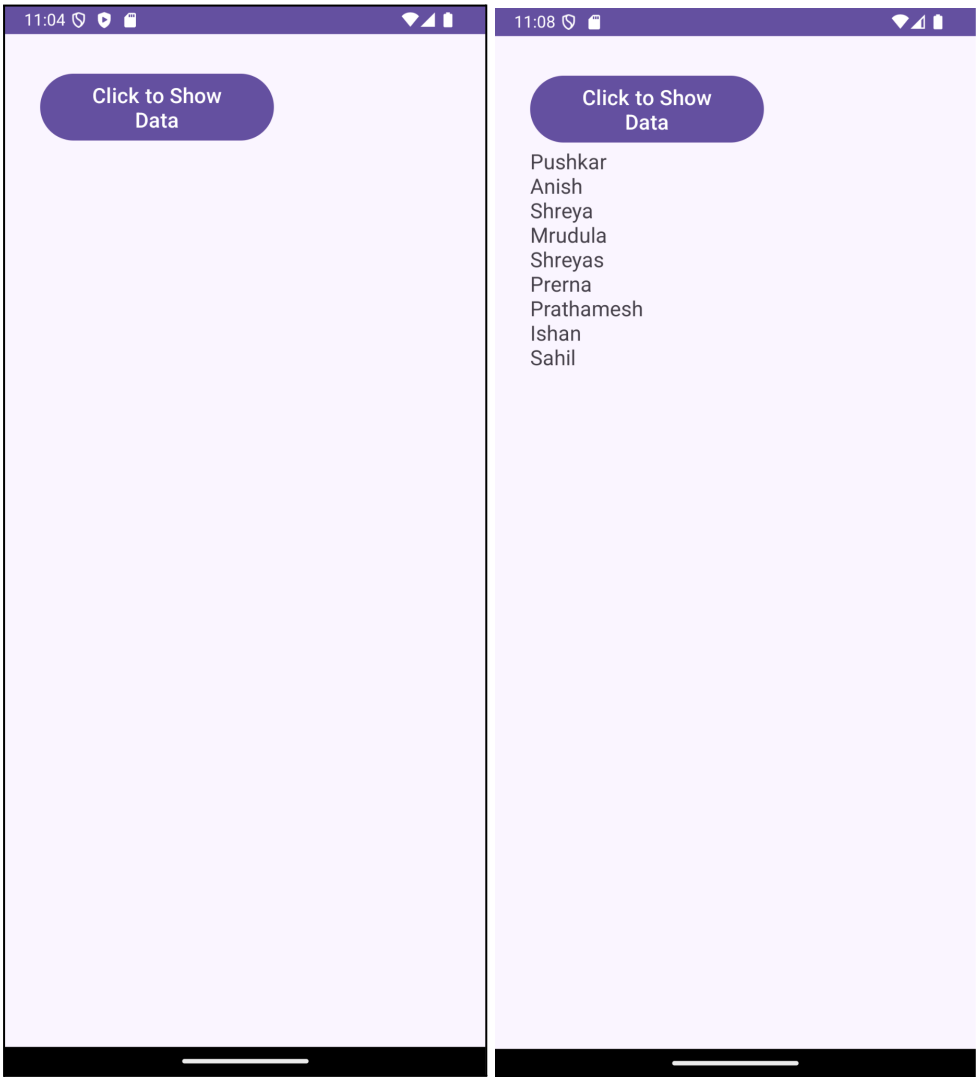
```
            StringBuilder names = new StringBuilder();
            for (int i = 0; i < namesArray.length(); i++) {
                names.append(namesArray.getString(i)).append("\n");
            }

            // Set the extracted names in the TextView
            resultsTextView.setVisibility(View.VISIBLE);
            resultsTextView.setText(names.toString());

        } catch (Exception e) {
            e.printStackTrace();
            resultsTextView.setText("Error parsing JSON.");
        }
    }
  }
}
```

activity_main.xml

```
<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="30dp">

    <Button
        android:id="@+id/displayData"
        android:layout_width="200dp"
        android:layout_height="wrap_content"
        android:text="Click to Show Data"
        android:textSize="18sp" />

    <TextView
        android:id="@+id/results"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="18sp"
```

```
    android:visibility="gone" />

</LinearLayout>
```

Output:



Conclusion:

REST APIs provide an efficient way to interact with web services using HTTP requests, with JSON as a preferred data exchange format. Using HttpURLConnection for network communication and JSON parsing techniques, developers can build Android applications that retrieve, process, and display data dynamically from remote servers. This practical introduces the foundation of working with APIs and JSON, essential for building modern Android apps.

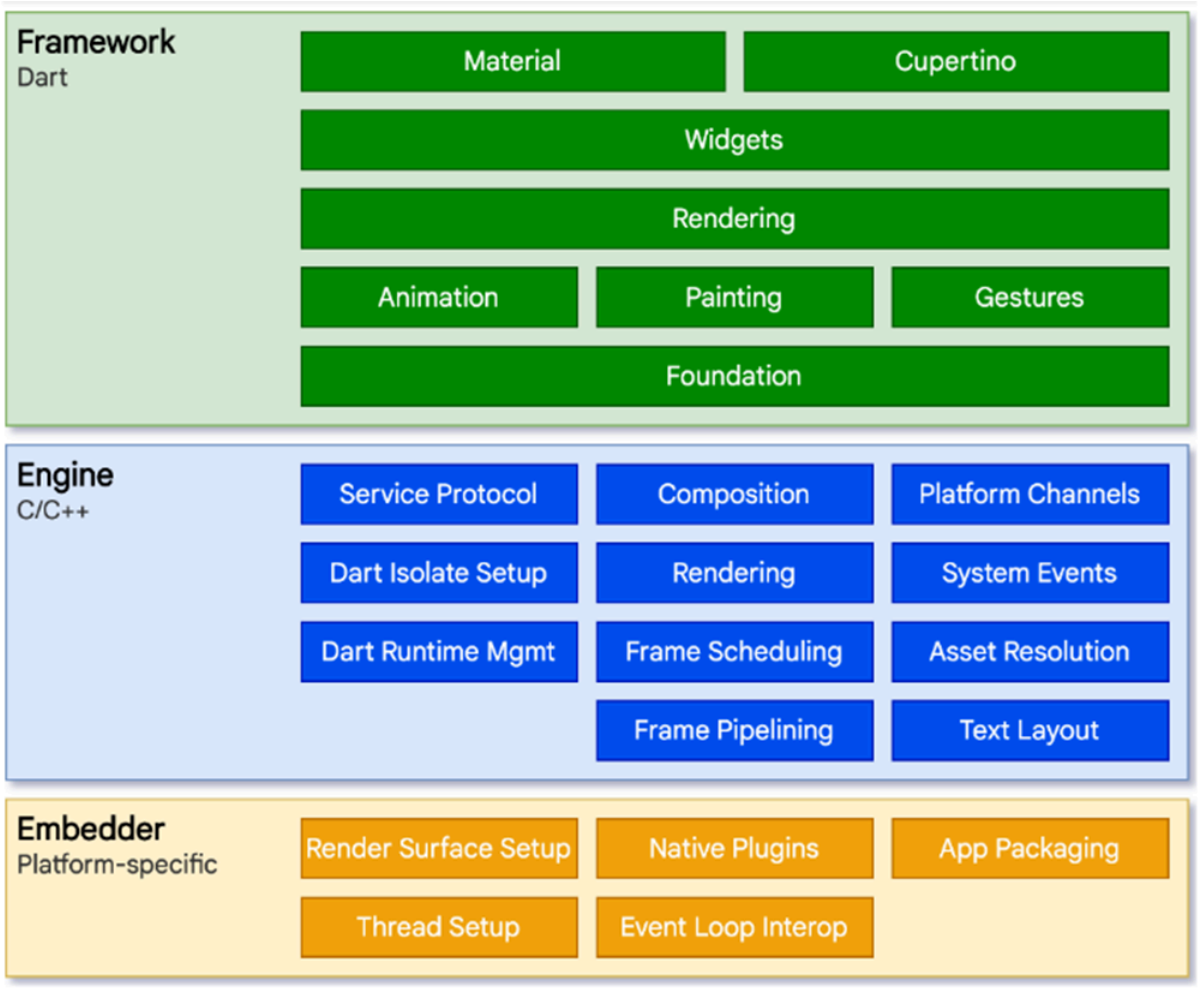
Name of Student: Pushkar Sane		
Roll Number: 45	**Lab Assignment Number: 10**	
Title of Lab Assignment: Flutter program using layout, widget and state management.		
DOP: 26-10-2024	**DOS: 26-10-2024**	
CO Mapped: **CO3, CO4**	**PO Mapped:** **PO2, PO3,PO5, PSO1, PSO2**	**Signature:**

Practical No. 10

Aim: Flutter program using layout, widget and state management.

Theory:

Flutter: Flutter is an open-source UI software development toolkit created by Google for building natively compiled applications for mobile, web, and desktop from a single codebase. Flutter provides a rich set of widgets and tools to create high-performance, expressive, and flexible user interfaces. It also follows a specific architectural pattern for structuring Flutter applications. There are two primary architectural patterns used in Flutter: Single Widget and Reactive Framework and BLoC (Business Logic Component) Architecture.



Layout in Flutter:

1. Widget Hierarchy: In Flutter, everything is a widget. Understanding how to create a hierarchy of widgets is crucial for building user interfaces. Widgets are composed into a tree structure, with a single root widget, which is typically a MaterialApp or CupertinoApp, and many child widgets that make up the user interface. Widgets can have children, and these children can be any type of widget.
2. Box Constraints: Flutter uses box constraints to determine the size of widgets. Widgets can have minimum and maximum dimensions, and their sizes are calculated based on available space and constraints. Widgets can overflow or be clipped if their content exceeds the constraints.
3. Rows and Columns: Flutter provides Row and Column widgets for arranging widgets horizontally and vertically, respectively. These widgets are commonly used to create structured layouts.
4. Containers: The Container widget is a versatile widget for styling and positioning child widgets. You can use it to set padding, margins, alignment, and background color for its children.
5. Expanded Widget: The Expanded widget is often used in combination with Row and Column widgets. It allows a child widget to take up available space within the parent widget, distributing space proportionally among multiple Expanded widgets.
6. Card Widget: The Card widget is a material design card that can be used to group related information. It provides an elegant and consistent way to present content in a structured manner.
7. Padding and Margins: Padding is used to add space inside a widget, while margins add space outside a widget. Both are important for controlling the spacing between widgets in your layout.

Layout Challenges: Designing responsive and adaptive layouts is essential. This involves handling screen sizes, orientations, and ensuring that your app looks and functions well on a variety of devices.

Widgets in Flutter:

1. **Introduction to Widgets:** Widgets are the building blocks of a Flutter app. Every visual element, from text and buttons to layout structures, is a widget. Understanding their role is fundamental.
2. **Stateless Widgets:** Stateless widgets represent components that don't change after they are initially created. They are used for displaying static content.

3. **Stateful Widgets:** Stateful widgets are used when the content needs to change over time, such as user interactions or data updates. They have an associated state object that can change during the widget's lifetime.
4. **Widget Building:** Building custom widgets is essential for creating reusable components and maintaining a clean and organized codebase. You can compose widgets by nesting them inside one another.
5. **Widget Lifecycle:** Widgets go through a life cycle of creation, updating, and potentially disposal. Understanding when these lifecycle methods (e.g., build, initState, dispose) are called is crucial for managing your widgets' behavior.
6. **Widget Libraries:** Flutter provides a rich set of built-in widgets, including those for common UI elements, navigation, and gestures. Additionally, you can use external packages to expand the set of available widgets.
7. **Custom Painting:** For advanced customization, you can use the CustomPaint widget to create custom-drawn widgets. This allows you to paint on the canvas with full control over the rendering process.

State Management in Flutter:

1. **Introduction to State Management:** State management is essential for handling and maintaining the state of your app. It involves keeping track of data, user interactions, and UI updates.
2. **Local State:** You can manage state within a widget using setState. This method is suitable for simple scenarios where a widget's state needs to be updated.
3. **Provider Package:** The provider package is a popular choice for state management in Flutter. It allows you to create and access data models throughout your app, making state management more organized and efficient.

Create a widget Product Box that contains the details of the product, such as image, name, price, and description. In the product Box widget, we use the following child widgets: Container, Row, Column, Expanded, Card, Text, Image, etc.

Code:

main.dart

```
import 'package:flutter/material.dart';
import 'package:provider/provider.dart';

void main() {
    runApp(
        ChangeNotifierProvider(
                create: (context) => ProductList(),
                child: MyApp(),
            ),
    );
}

class Product {
    final String imageUrl;
    final String name;
    final double price;
    final String description;

    Product({
            required this.imageUrl,
            required this.name,
            required this.price,
            required this.description,
    });
}

class ProductList with ChangeNotifier {
    final List<Product> _products = [
            Product(
```

```
                                                                    imageUrl:
'https://images.unsplash.com/photo-1505740420928-5e560c06d30e?auto=format&fit=crop&
q=80&w=2070&ixlib=rb-4.0.3&ixid=M3wxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx
8fA%3D%3D',
            name: 'Boat Headphone',
            price: 19.99,
                description: 'Wireless audio with Boat: clear, stylish, and convenient
headphones.',
        ),
        Product(
                                                                    imageUrl:
'https://images.unsplash.com/photo-1523275335684-37898b6baf30?auto=format&fit=crop&q
=80&w=1999&ixlib=rb-4.0.3&ixid=M3wxMjA3fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8f
A%3D%3D',
            name: 'Apple Smart Watch',
            price: 29.99,
            description: 'Wristwear for smart living: time, apps, fitness, and notifications',
        ),
  ];

  List<Product> get products => _products;
  void addProduct(Product product) {
        _products.add(product);
        notifyListeners();
  }
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
      return MaterialApp(
          home: Scaffold(
          appBar: AppBar(
              title: Text('Product Catalog'),
          ),
```

```
            body: ProductListWidget(),
          ),
        );
    }
}

class ProductListWidget extends StatelessWidget {
    @override
    Widget build(BuildContext context) {
            final productProvider = Provider.of<ProductList>(context);
            return ListView.builder(
                itemCount: productProvider.products.length,
                itemBuilder: (context, index) {
                return ProductBox(product: productProvider.products[index]);
                },
            );
    }
}

class ProductBox extends StatelessWidget {
    final Product product;
    ProductBox({required this.product});

    @override
    Widget build(BuildContext context) {
            return Card(
                margin: EdgeInsets.all(10.0),
                child: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                    Image.network(product.imageUrl),
                    Padding(
                    padding: const EdgeInsets.all(8.0),
                    child: Text(
```

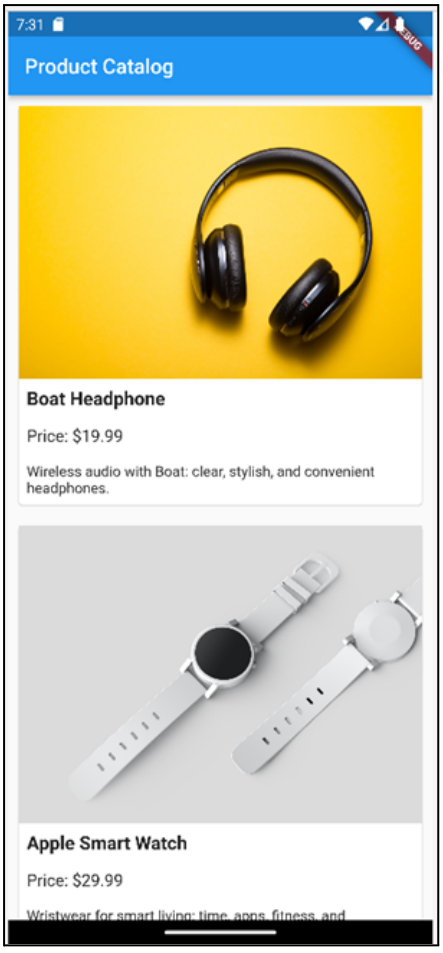
```
                    product.name,
                    style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
                ),
                ),
                Padding(
                padding: const EdgeInsets.all(8.0),
                child: Text(
                    'Price: \$${product.price.toStringAsFixed(2)}',
                    style: TextStyle(fontSize: 16),
                ),
                ),
                Padding(
                padding: const EdgeInsets.all(8.0),
                child: Text(
                    product.description,
                    style: TextStyle(fontSize: 14),
                ),
                ),
            ],
            ),
        );
    }
}
```

pubspec.yml

```
name: p_10
description: A new Flutter project.
publish_to: 'none'
version: 1.0.0+1

environment:
  sdk: '>=3.1.2 <4.0.0'

dependencies:
```

```
  flutter:
    sdk: flutter
  provider: ^5.0.0
  cupertino_icons: ^1.0.2

dev_dependencies:
  flutter_test:
            sdk: flutter
  flutter_lints: ^2.0.0

flutter:
  uses-material-design: true
```

Output:



Conclusion: I have successfully implemented a flutter application in android.