Shram Sadhana Bombay Trust Sanchlit
Arts, Commerce and Science College, Bambhori, Jalgaon
Bachelor of Computer Application (B.C.A.)

Bachelor of Computer Applications

LAB MANUAL ON
BCA 606 Lab on Android Application Development

Name:

Class:

Sem:

Roll No:

Seat No:

Name of faculty: Ms. Bhagyashri S. Patil

Academic Year: 20 - 20

Shram Sadhana Bombay Trust Sanchlit
Arts, Commerce and Science College, Bambhori, Jalgaon
Bachelor of Computer Application (B.C.A.)
Practical: 01

DOP: **DOC**:

Title: Installation and setup of java development kit (JDK), setup android SDK, setup eclipse IDE, setup android development tools (ADT) plugins, create android virtual device.

Objective: To set up the development environment for Android applications by installing the necessary tools, including Java Development Kit (JDK), Android SDK, Eclipse IDE, and Android Development Tools (ADT) plugins, and creating an Android Virtual Device (AVD).

Theory:

1. Install Java Development Kit (JDK)
Steps:

1. **Download JDK**
   - Visit Oracle's official website or install OpenJDK.
   - Choose the appropriate version (JDK 8 or later) based on your system architecture (Windows, macOS, or Linux).
2. **Install JDK**
   - Run the installer and follow the on-screen instructions.
3. **Set Environment Variables (Windows Users)**
   - Go to **Control Panel → System → Advanced system settings → Environment Variables**.
   - Under **System Variables**, add/update:
     - JAVA_HOME → Path to JDK installation directory.
     - Path → Append %JAVA_HOME%\bin.
4. **Verify Installation**
   - Open a command prompt or terminal and type:

     ```
     java -version
     javac -version
     ```
   - It should display the installed JDK version.

2. Install Android SDK
Steps:

1. **Download SDK**
   - Visit the Android Developer website.
   - Scroll down to **Command Line Tools only** and download the package for your OS.
2. **Extract and Set Up Android SDK**
   - Extract the SDK to a preferred location (e.g., C:\Android\Sdk for Windows).
   - Add the SDK tools and platform-tools to the system Path variable.
3. **Verify Installation**
   - Open a terminal or command prompt and run:

     ```
     sdkmanager --list
     ```
   - If the SDK is installed correctly, it will list available Android components.

3. Install Eclipse IDE
Steps:

1. **Download Eclipse**
    - Visit Eclipse Downloads.
    - Download **Eclipse IDE for Java Developers**.
2. **Install Eclipse**
    - Extract the downloaded package and run the eclipse.exe (Windows) or eclipse (Linux/macOS).

4. Install Android Development Tools (ADT) Plugin
Steps:

1. **Open Eclipse** and go to **Help → Eclipse Marketplace**.
2. Search for **"Android Development Tools (ADT)"** and install it.
3. Restart Eclipse after installation.

5. Create an Android Virtual Device (AVD)
Steps:

1. Open **Android SDK Manager** in Eclipse.
2. Go to **Tools → AVD Manager**.
3. Click **Create Virtual Device**.
4. Select a hardware profile (e.g., Pixel 4).
5. Choose a system image (Android version).
6. Configure the settings (RAM, screen size, etc.).
7. Click **Finish** to create the AVD.

To start the emulator, select the AVD and click **Start**.

Conclusion: By following these steps, you have successfully set up your Android development environment with Eclipse, JDK, Android SDK, ADT Plugin, and AVD. You are now ready to build and run Android applications.

Submitted By: **Checked By:**

Sign: **Ms. Bhagyashri S. Patil**

Name:

Roll No:

Shram Sadhana Bombay Trust Sanchlit
Arts, Commerce and Science College, Bambhori, Jalgaon
Bachelor of Computer Application (B.C.A.)
Practical: 02

DOP: **DOC**:

Title: Create “Hello World” application. That will display “Hello World” in the middle of the screen using TextView Widget in the red color.

Objective: To develop a simple Android application that displays "Hello World" at the center of the screen using the **TextView** widget with red text color.

Theory:

1. Create a New Android Studio Project

1. **Open Android Studio**.
2. Click **"Start a new Android Studio project"**.
3. Select **"Empty views Activity"** and click **Next**.
4. Enter the project details:
   - **Name:** HelloWorldApp
   - **Package Name:** com.example.helloworld
   - **Save Location:** Choose a folder.
   - **Language:** Java (or Kotlin if preferred).
   - **Minimum SDK:** API 21 (Android 5.0) or later.
5. Click **Finish** to create the project.

2. Modify Layout to Display "Hello World" in Red Color

Edit activity_main.xml

1. Open res/layout/activity_main.xml.
2. Replace the existing code with the following:

```
<?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:id="@+id/main"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   tools:context=".MainActivity">

<TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Hello World!"
      android:textColor="#FF0000"
      app:layout_constraintBottom_toBottomOf="parent"
      app:layout_constraintEnd_toEndOf="parent"
```

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

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

3. Modify MainActivity Java Code

Edit MainActivity.java

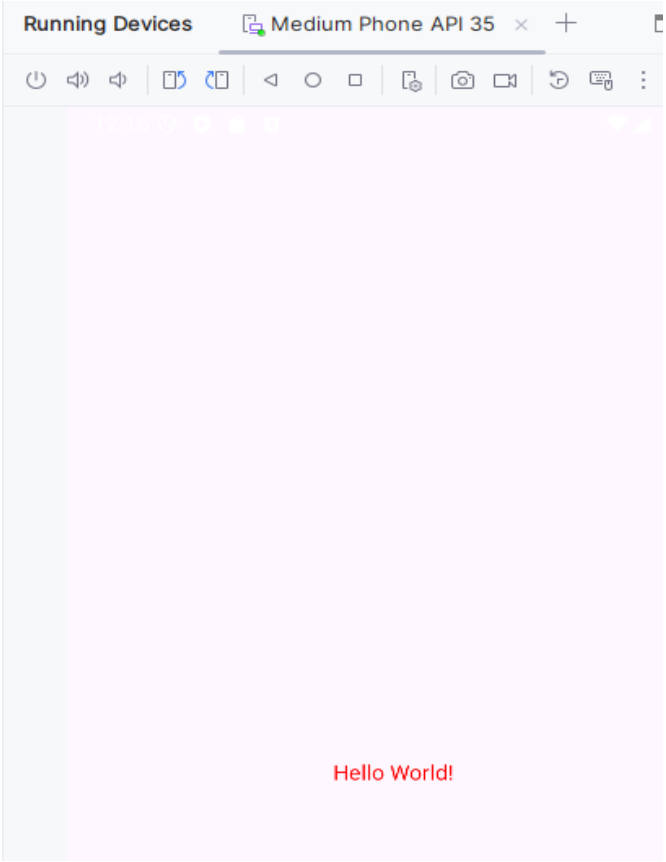
1. Open java/com/example/helloworld/MainActivity.java.
2. Update the code as follows:

```
package com.example.helloworld;

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

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

4. Run the Application

1. **Launch an Android Virtual Device (AVD)**:
    - Open **AVD Manager** (Tools → Device Manager).
    - Click **Create Device**, select a device profile (e.g., Pixel 4), and configure the settings.
    - Click **Finish** and then **Run** the emulator.
2. **Run the Project**:
    - Click **Run → Run 'app'** or press **Shift + F10**.
    - The emulator will launch, displaying "Hello World" in red at the center of the screen.

Output:



Conclusion: You have successfully created an Android application in **Android Studio** that displays "Hello World" in red color at the center of the screen using the **TextView** widget.

Submitted By:

Sign:

Name:

Roll No:

Checked By:

Ms. Bhagyashri S. Patil

Shram Sadhana Bombay Trust Sanchlit
Arts, Commerce and Science College, Bambhori, Jalgaon
Bachelor of Computer Application (B.C.A.)
Practical: 03

DOP: **DOC**:

Title: Create Registration page to demonstration of Basic widgets available in android.

Objective: To develop a **Registration Page** using various Android widgets such as **EditText, Button, RadioButton, CheckBox and TextView**. The application will allow users to enter details and submit the form.

Theory:

1. Create a New Android Studio Project

1. **Open Android Studio**.
2. Click **"Start a new Android Studio project"**.
3. Select **"Empty Activity"** and click **Next**.
4. Enter the project details:
   - **Name:** RegistrationApp
   - **Package Name:** com.example.registrationapp
   - **Language:** Java (or Kotlin if preferred).
   - **Minimum SDK:** API 21 (Android 5.0) or later.
5. Click **Finish** to create the project.

2. Design the Registration Page Layout

Modify activity_main.xml

1. Open res/layout/activity_main.xml.
2. Replace the existing code with the following:

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

<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools" android:layout_width="match_parent"
android:layout_height="match_parent" android:orientation="vertical"
tools:context=".MainActivity">

<LinearLayout android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_gravity="center"
android:orientation="vertical" android:padding="10dp" tools:ignore="MissingConstraints">

<TextView android:layout_width="match_parent" android:layout_height="wrap_content"
android:layout_marginTop="20dp" android:fontFamily="serif" android:gravity="center"
android:text="@string/registration" android:textColor="#e65100" android:textSize="40sp"
/>
```

```
<LinearLayout android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_gravity="center"
android:orientation="vertical" android:padding="2dp">

<com.google.android.material.textfield.TextInputLayout
android:layout_width="match_parent" android:layout_height="wrap_content">

<com.google.android.material.textfield.TextInputEditText android:id="@+id/nm"
android:layout_width="match_parent" android:layout_height="wrap_content"
android:layout_marginLeft="20dp" android:layout_marginTop="2dp"
android:layout_marginRight="20dp" android:hint="@string/name"

android:padding="10dp" android:textColorHint="#546E7A"

tools:ignore="TextContrastCheck,TouchTargetSizeCheck,VisualLintTextFieldSize"  />

</com.google.android.material.textfield.TextInputLayout>

<com.google.android.material.textfield.TextInputLayout
android:layout_width="match_parent" android:layout_height="wrap_content">

<com.google.android.material.textfield.TextInputEditText android:id="@+id/mob"
android:layout_width="match_parent" android:layout_height="wrap_content"
android:layout_marginLeft="20dp" android:layout_marginRight="20dp"
android:hint="Mobile"

android:padding="10dp" android:textColorHint="#546E7A"

tools:ignore="HardcodedText,TextContrastCheck,TouchTargetSizeCheck,VisualLintTextFi
eld Size" />

</com.google.android.material.textfield.TextInputLayout>

<com.google.android.material.textfield.TextInputLayout
android:layout_width="match_parent" android:layout_height="wrap_content">

<com.google.android.material.textfield.TextInputEditText android:id="@+id/email"

android:layout_width="match_parent" android:layout_height="wrap_content"
android:layout_marginLeft="20dp" android:layout_marginRight="20dp"
android:hint="@string/email" android:padding="10dp" android:textColorHint="#546E7A"

tools:ignore="TextContrastCheck,TouchTargetSizeCheck,VisualLintTexize" />

</com.google.android.material.textfield.TextInputLayout>

<com.google.android.material.textfield.TextInputLayout
android:layout_width="match_parent" android:layout_height="wrap_content">

<com.google.android.material.textfield.TextInputEditText android:id="@+id/pass"
android:layout_width="match_parent" android:layout_height="wrap_content"
android:layout_marginLeft="20dp" android:layout_marginRight="20dp"
```

```
android:hint="@string/password" android:inputType="textPassword"
android:padding="10dp"

tools:ignore="TextContrastCheck,TouchTargetSizeCheck,VisualLintTextFieldSize"  />

</com.google.android.material.textfield.TextInputLayout>

<com.google.android.material.textfield.TextInputLayout
android:layout_width="match_parent" android:layout_height="wrap_content">

<com.google.android.material.textfield.TextInputEditText android:id="@+id/rollNo"
android:layout_width="match_parent" android:layout_height="wrap_content"
android:layout_marginLeft="20dp" android:layout_marginTop="2dp"
android:layout_marginRight="20dp" android:hint="@string/roll_no"
android:padding="10dp"

tools:ignore="TextContrastCheck,TouchTargetSizeCheck,VisualLintTextFieldSize"  />

</com.google.android.material.textfield.TextInputLayout>

<com.google.android.material.textfield.TextInputLayout
android:layout_width="match_parent" android:layout_height="wrap_content">

<com.google.android.material.textfield.TextInputEditText

android:id="@+id/age" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_marginLeft="20dp"
android:layout_marginTop="2dp" android:layout_marginRight="20dp"
android:hint="@string/age" android:padding="10dp" android:textColorHint="#546E7A"

tools:ignore="TextContrastCheck,TouchTargetSizeCheck,VisualLintTextFieldSize"  />

</com.google.android.material.textfield.TextInputLayout>

<RadioGroup android:id="@+id/rd_gen"

android:layout_width="match_parent" android:layout_height="wrap_content"
android:layout_marginLeft="10dp" android:layout_marginTop="10dp"
android:layout_marginRight="10dp" android:orientation="horizontal">

<RadioButton android:id="@+id/right_male" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_marginLeft="50dp"
android:gravity="center" android:text="@string/male" android:textSize="20sp"
tools:ignore="RtlHardcoded" />


<RadioButton android:id="@+id/right_female" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_marginStart="20dp"
android:text="@string/female" android:textSize="20sp" />

</RadioGroup>

</LinearLayout>
```

```
<Button

android:id="@+id/btnsub" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_gravity="center"
android:layout_marginTop="5dp" android:layout_marginBottom="30dp"

android:backgroundTint="#e65100" android:padding="20dp"
android:text="@string/submit" android:textColor="#fff" android:textSize="20sp" />

</LinearLayout>

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

strings.xml:

```
<resources>

<string name="app_name">AAD_Pr3</string>

<string name="registration">Registration</string>

<string name="name">Name</string>

<string name="email">Email</string>

<string name="password">Password</string>

<string name="roll_no">Roll No</string>

<string name="age">Age</string>

<string name="male">Male</string>

<string name="female">Female</string>

<string name="submit">Submit</string>

</resources>
```

3. Modify Java Code for Registration Functionality
Edit MainActivity.java

1. **Open java/com/example/registrationapp/MainActivity.java.**
2. **Replace the existing code with:**

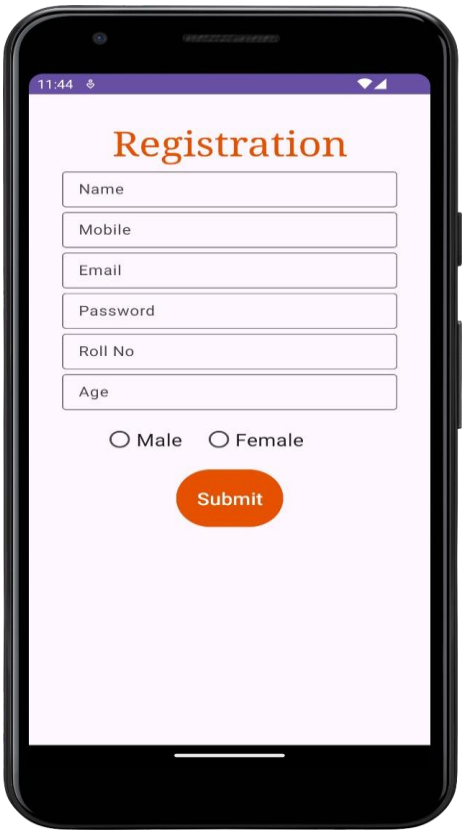
```
package com.example.aad_pr3;

import android.os.Bundle;

import  androidx.annotation.Nullable;
import  androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
```

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

4. Run the Application

1. **Launch an Android Virtual Device (AVD)**:
   - Open **AVD Manager** (Tools → Device Manager).
   - Click **Create Device**, select a device profile (e.g., Pixel 4), and configure the settings.
   - Click **Finish** and then **Run** the emulator.
2. **Run the Project**:
   - Click **Run → Run 'app'** or press **Shift + F10**.
   - The emulator will launch, displaying "Hello World" in red at the center of the screen.

Output:



Conclusion: In this project, we created a **Registration Page** in **Android Studio** using Java. The registration form demonstrates various **basic Android widgets**.

Submitted By:

Sign:

Name:

Roll No:

Checked By:

Ms. Bhagyashri S. Patil

Shram Sadhana Bombay Trust Sanchlit
Arts, Commerce and Science College, Bambhori, Jalgaon
Bachelor of Computer Application (B.C.A.)
Practical: 04

DOP: **DOC**:

Title: Create sample application with login module.(Check username and password) On successful login, Change TextView "Login Successful". And on failing login, alert user using Toast "Login fail".

Objective: The objective of this sample Android application is to create a login module that checks the entered username and password. When the user enters the correct credentials, the TextView displays "Login Successful". If the credentials are incorrect, the app alerts the user with a Toast message saying "Login Fail".

Theory:

Steps:

1. Create an Android project with an Activity.
2. Use a TextView for displaying the status, an EditText for username and password input, and a Button to trigger the login attempt.
3. Write Java code to handle the login logic and display the result.

1. XML Layout (activity_main.xml)

This layout contains two EditText fields (for username and password), a Button (for login), and a TextView (to display the result).

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#FA80CAD5">

    <TextView
        android:id="@+id/txthead"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="40dp"
        android:layout_marginRight="20dp"
        android:text="@string/login_form"
        android:textAlignment="center"
        android:textColor="#616161"
        android:textStyle="bold" />

    <EditText
        android:id="@+id/edt_email1"
        android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
        android:layout_below="@+id/txthead"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="20dp"
        android:layout_marginRight="20dp"
        android:autofillHints="true"
        android:text="@string/enter_email" tools:ignore=
    "LabelFor,TextFields,TouchTargetSizeCheck,VisualLintTextFieldSize"  />

    <EditText android:id="@+id/edt_passwd1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/edt_email1"
        android:layout_marginLeft="20dp"
        android:layout_marginTop="22dp"
        android:minHeight="48dp"
        android:text="@string/enter_password"
        android:textColorHint="#37474F"
        tools:ignore=
"Autofill,LabelFor,RtlHardcoded,SpeakableTextPresentCheck,TextFields,Visua
lLintTextFieldSize" />

    <Button
        android:id="@+id/btnLog"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/edt_passwd1"
        android:layout_marginLeft="60dp"
        android:layout_marginTop="40dp"
        android:layout_marginRight="60dp"
        android:background="#0B8B9B"
        android:text="Login" android:textColor="#fff"
        android:textSize="25sp"
        tools:ignore="HardcodedText,VisualLintButtonSize"  />
</RelativeLayout>
```

strings.xml:

```
<resources>
    <string name="app_name">AAD_Pr4</string>
    <string name="login_form">Login Form</string>
    <string name="enter_email">Enter your email</string>
    <string name="enter_password">Enter your password</string>
</resources>
```

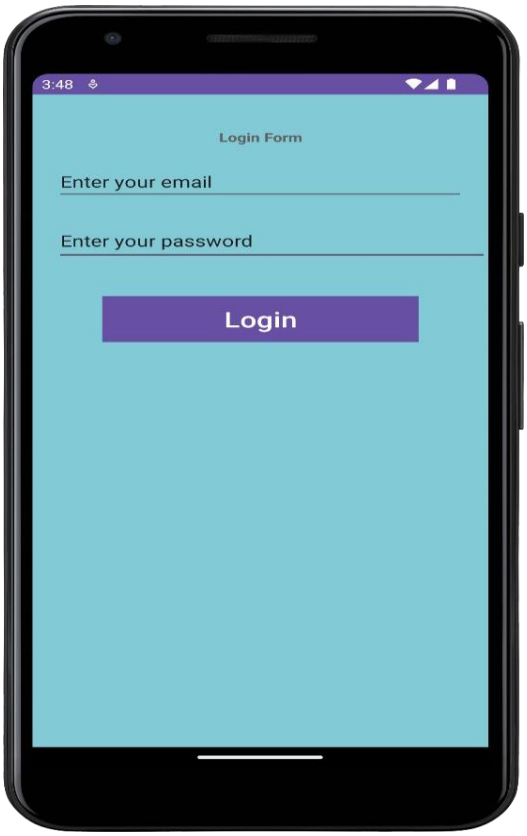
2. Java Code (MainActivity.java)

This Java code handles the login logic, checks the entered username and password, and updates the UI accordingly.

```
package com.example.aad_pr4;

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.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity implements View.OnClickListener
{
    TextView txtheadline; EditText
    edteemail1, edtpw1; Button btnlog;

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

        // Initialize UI components
        txtheadline = findViewById(R.id.txthead); edteemail1
        = findViewById(R.id.edt_email1); edtpw1 =
        findViewById(R.id.edt_passwd1); btnlog =
        findViewById(R.id.btnLog);
        // Set click listener btnlog.setOnClickListener(this);
    }

    @Override
    public void onClick(View v) {
        // Check login credentials
        if ((edteemail1.getText().toString().equals("user@gmail.com")) &&
    (edtpw1.getText().toString().equals("1234"))) {
            Toast.makeText(this, "Login Successful", Toast.LENGTH_SHORT).show();
        } else {
            Toast.makeText(this, "Login Failed", Toast.LENGTH_SHORT).show();
        }
    }
}
```

Output:



Conclusion: In this project, we created a **Registration Page** in **Android Studio** using Java. The registration form demonstrates various **basic Android widgets**.

Submitted By: **Checked By:**

Sign: **Ms. Bhagyashri S. Patil**

Name:

Roll No:

Shram Sadhana Bombay Trust Sanchlit
Arts, Commerce and Science College, Bambhori, Jalgaon
Bachelor of Computer Application (B.C.A.)
Practical: 05

DOP: **DOC**:

Title: Create an application for demonstration of Scroll view in android

Objective: The objective of this application is to demonstrate the use of ScrollView in Android. ScrollView is a layout container that allows users to scroll through a long list of content, which does not fit within the screen dimensions. This ensures better usability and accessibility in mobile applications.

Theory:

Implementation:

1. **Create a New Android Project**: Open Android Studio and create a new project with an Empty Activity.
2. **Modify the Layout File**:
   - Use a ScrollView as the root layout or wrap a content layout inside it.
   - Add multiple UI elements (TextViews, Images, Buttons, etc.) inside a LinearLayout to observe scrolling.
3. **Update MainActivity.java (or Kotlin equivalent)**: No specific logic is needed for basic scrolling functionality.
4. **Run and Test the Application**: Deploy the app on an emulator or a physical device to check the scrolling functionality.

activity_main.xml

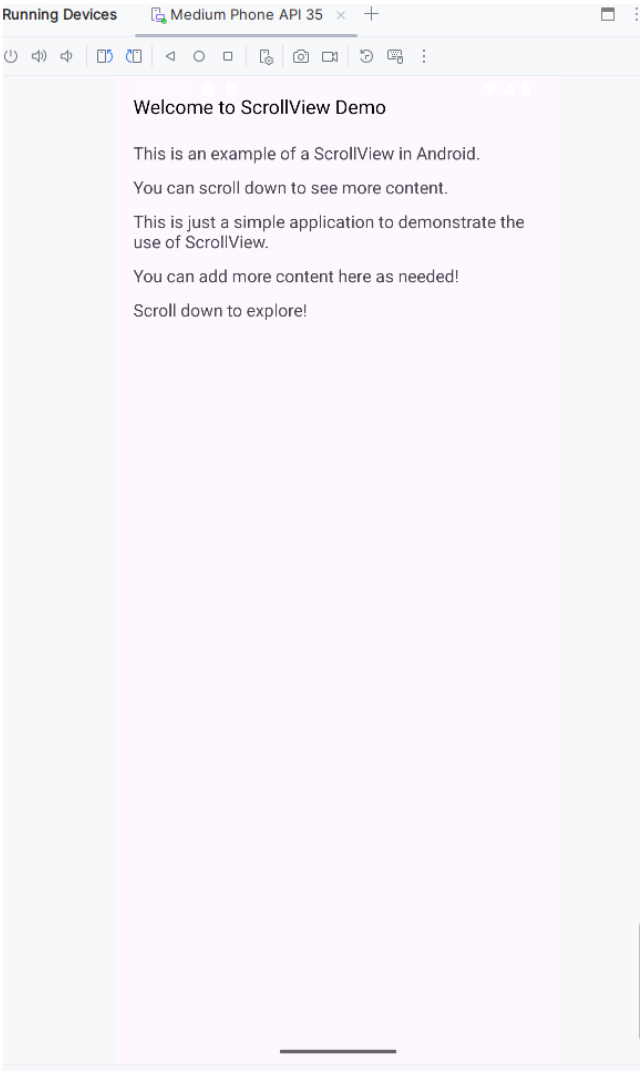
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:id="@+id/main"
   android:layout_width="match_parent"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

   <android.widget.ScrollView
      android:id="@+id/scrollView"
      android:layout_width="match_parent"
      android:layout_height="match_parent">

      <LinearLayout
         android:layout_width="match_parent"
         android:layout_height="wrap_content"
         android:orientation="vertical"
         android:padding="16dp">
```

```
        <!-- Sample content inside ScrollView -->
        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Welcome to ScrollView Demo"
            android:textSize="18sp"
            android:textColor="@android:color/black"
            android:layout_marginBottom="20dp"/>

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="This is an example of a ScrollView in Android."
            android:textSize="16sp"
            android:layout_marginBottom="10dp"/>

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="You can scroll down to see more content."
            android:textSize="16sp"
            android:layout_marginBottom="10dp"/>

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="This is just a simple application to demonstrate the use of ScrollView."
            android:textSize="16sp"
            android:layout_marginBottom="10dp"/>

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="You can add more content here as needed!"
            android:textSize="16sp"
            android:layout_marginBottom="10dp"/>

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Scroll down to explore!"
            android:textSize="16sp"
            android:layout_marginBottom="20dp"/>

    </LinearLayout>

  </android.widget.ScrollView>

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

MainActivity.java

```
package com.example.scrollview;

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


public class MainActivity extends AppCompatActivity {

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

Output:



Conclusion: The ScrollView widget in Android enables users to navigate through content that exceeds the screen size. It is an essential component in mobile applications for displaying long forms, articles, or lists. By implementing ScrollView, developers can enhance the user experience by providing smooth and intuitive scrolling functionality.

Submitted By: **Checked By:**

Sign: **Ms. Bhagyashri S. Patil**

Name:

Roll No:

Shram Sadhana Bombay Trust Sanchlit
Arts, Commerce and Science College, Bambhori, Jalgaon
Bachelor of Computer Application (B.C.A.)
Practical: 06

DOP: **DOC**:

Title: Create login application where you will have to validate username and passwords till the username and password is not validated, login button should remain disabled.

Objective: The objective of this application is to demonstrate the implementation of a login screen where the login button remains disabled until the user enters a valid username and password. This ensures proper validation before proceeding further, enhancing security and usability.

Theory:

Implementation Steps:

1. **Create a New Android Project**: Open Android Studio and start a new project with an Empty Activity.
2. **Design the Layout (activity_main.xml)**:
   - Use EditText for username and password fields.
   - Add a Button for login, which should be initially disabled.
   - Use TextWatcher to enable the button when valid input is detected.
3. **Write the Logic in MainActivity.java (or Kotlin equivalent)**:
   - Implement TextWatcher to check username and password input.
   - Enable the login button when both fields are correctly filled.
   - Add a click listener for login functionality.
4. **Run and Test the Application**: Ensure that the login button remains disabled until valid input is provided.

.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="32dp"
    android:gravity="center">

    <EditText
        android:id="@+id/username"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter Username"
        android:inputType="text"
        android:padding="16dp"/>

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

```
        android:hint="Enter Password"
        android:inputType="textPassword"
        android:padding="16dp"/>

    <Button
        android:id="@+id/loginButton"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Login"
        android:enabled="false"
        android:padding="16dp"/>

</LinearLayout>
```

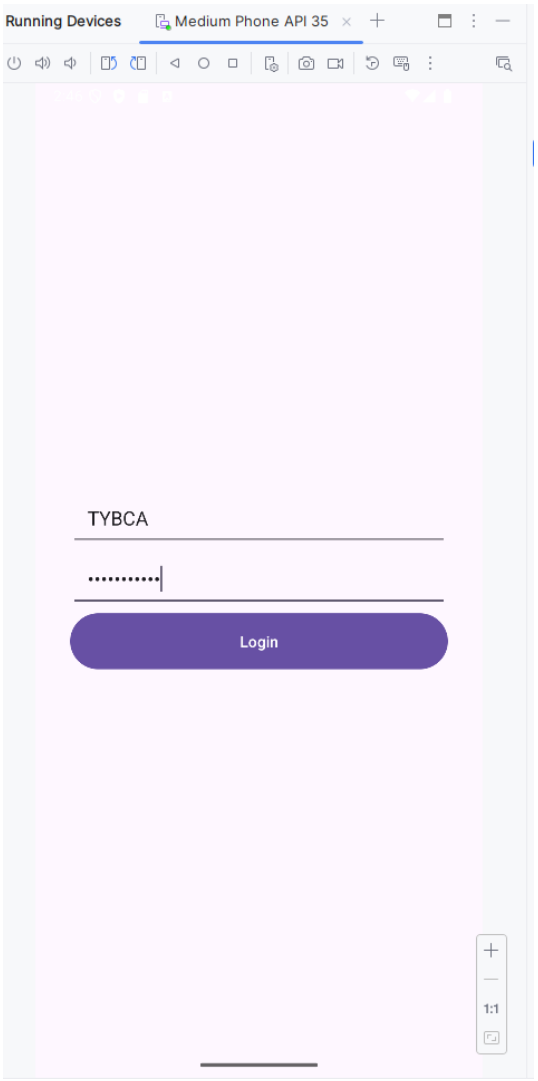
.Java

```
package com.example.validationinandroid;

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, passwordEditText;
    private Button loginButton;

    // Hardcoded correct credentials
    private static final String CORRECT_USERNAME = "TYBCA";
    private static final String CORRECT_PASSWORD = "password123";

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

        // Initialize views
        usernameEditText = findViewById(R.id.username);
        passwordEditText = findViewById(R.id.password);
        loginButton = findViewById(R.id.loginButton);

        // Add text change listeners to both fields
        usernameEditText.addTextChangedListener(new android.text.TextWatcher() {
            @Override
            public void beforeTextChanged(CharSequence charSequence, int start, int count, int
after) {}

            @Override
            public void onTextChanged(CharSequence charSequence, int start, int before, int
count) {
```

```
            validateInput();
        }

        @Override
        public void afterTextChanged(android.text.Editable editable) {}
    });

    passwordEditText.addTextChangedListener(new android.text.TextWatcher() {
        @Override
        public void beforeTextChanged(CharSequence charSequence, int start, int count, int
after) {}

        @Override
        public void onTextChanged(CharSequence charSequence, int start, int before, int
count) {
            validateInput();
        }

        @Override
        public void afterTextChanged(android.text.Editable editable) {}
    });

    // Set up login button onClickListener
    loginButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            String enteredUsername = usernameEditText.getText().toString().trim();
            String enteredPassword = passwordEditText.getText().toString().trim();

            // Validate credentials
            if (enteredUsername.equals(CORRECT_USERNAME) &&
enteredPassword.equals(CORRECT_PASSWORD)) {
                Toast.makeText(MainActivity.this, "Login Successful",
Toast.LENGTH_SHORT).show();
            } else {
                Toast.makeText(MainActivity.this, "Invalid Username or Password",
Toast.LENGTH_SHORT).show();
            }
        }
    });
}

// Method to validate input fields and enable/disable the login button
private void validateInput() {
    String enteredUsername = usernameEditText.getText().toString().trim();
    String enteredPassword = passwordEditText.getText().toString().trim();

    // If both fields are correct, enable login button
    if (enteredUsername.equals(CORRECT_USERNAME) &&
enteredPassword.equals(CORRECT_PASSWORD)) {
        loginButton.setEnabled(true);
    } else {
        loginButton.setEnabled(false);
```

```
        }
    }
}
```

Output:



Conclusion: Developing a login page in android application is a fundamental practice that helps developers understand user authentication, UI design and Data Handling.

Submitted By: **Checked By:**

Sign: **Ms. Bhagyashri S. Patil**

Name:

Roll No:

Shram Sadhana Bombay Trust Sanchlit
Arts, Commerce and Science College, Bambhori, Jalgaon
Bachelor of Computer Application (B.C.A.)
Practical: 07

DOP: **DOC**:

Title: Create an application for calculator.

Objective: To develop a basic calculator application in Android Studio using Java/Kotlin that performs arithmetic operations such as addition, subtraction, multiplication, and division. This project helps in understanding Android UI components like EditText, Button, and TextView, along with event handling.

Theory:

Implementation:
The application includes:

- **User Interface (UI):**
  - Two EditText fields for input numbers.
  - Four Button elements for operations (+, -, *, /).
  - A TextView to display the result.
- **Functionality:**
  - When the user enters two numbers and clicks an operation button, the corresponding calculation is performed.
  - The result is displayed in the TextView.
  - Proper validation is done to handle division by zero and empty inputs.

Code Implementation:

.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="20dp">

   <EditText
      android:id="@+id/etNumber1"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="Enter Number 1"
      android:inputType="numberDecimal" />

   <EditText
      android:id="@+id/etNumber2"
      android:layout_width="match_parent"
```

```
        android:layout_height="wrap_content"
        android:hint="Enter Number 2"

       android:inputType="numberDecimal" />

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:gravity="center">

        <Button
            android:id="@+id/btnAdd"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="+" />

        <Button
            android:id="@+id/btnSubtract"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="-" />

        <Button
            android:id="@+id/btnMultiply"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="*" />

        <Button
            android:id="@+id/btnDivide"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="/" />
    </LinearLayout>

    <TextView
        android:id="@+id/tvResult"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Result: "
        android:textSize="20sp"
        android:gravity="center"
        android:padding="10dp"/>
</LinearLayout>
```

java

```
package com.example.calculatorapp;


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

    @Override

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

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

        btnAdd.setOnClickListener(view -> calculate('+'));
        btnSubtract.setOnClickListener(view -> calculate('-'));
        btnMultiply.setOnClickListener(view -> calculate('*'));
        btnDivide.setOnClickListener(view -> calculate('/'));
    }

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

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

        double num1 = Double.parseDouble(num1Str);
        double num2 = Double.parseDouble(num2Str);
        double result = 0;
```

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

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

Output:

Conclusion: Developing a calculator application for android effects both practical and educational values.

Submitted By:

Sign:

Name:

Roll No:

Checked By:

Ms. Bhagyashri S. Patil

Shram Sadhana Bombay Trust Sanchlit
Arts, Commerce and Science College, Bambhori, Jalgaon
Bachelor of Computer Application (B.C.A.)
Practical: 08

DOP: **DOC**:

Title: Demonstrate use of scroll view.

Objective: To demonstrate the use of ScrollView in an Android application, allowing content to be scrollable when it exceeds the screen size. This helps in managing large forms, images, or lengthy text content.

Theory:

Implementation:

The application contains:

- **A ScrollView Layout** that enables scrolling.
- **Multiple TextViews & Buttons** arranged vertically to create a long UI.
- **Demonstration of Scrolling** when content exceeds screen space.

Code Implementation:

XML Layout (activity_main.xml)

```
<ScrollView
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent">

   <LinearLayout
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:orientation="vertical"
      android:padding="16dp">

      <TextView
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="Welcome to ScrollView Demo!"
         android:textSize="20sp"
         android:textStyle="bold"
         android:paddingBottom="10dp" />
```

```
<!-- Multiple TextViews to create a long scrollable view -->
 <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="This is a demonstration of ScrollView in Android."
    android:textSize="16sp"
    android:paddingBottom="10dp" />

 <ImageView
    android:layout_width="match_parent"
    android:layout_height="200dp"
    android:src="@mipmap/ic_launcher"
    android:scaleType="centerCrop"
    android:paddingBottom="10dp" />

 <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Scroll down to see more content..."
    android:textSize="16sp"
    android:paddingBottom="10dp" />

 <!-- Adding multiple buttons to extend the view -->
 <Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Button 1" />

 <Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Button 2" />

 <Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Button 3" />

 <Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Button 4" />

 <Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Button 5" />
```

```
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="End of ScrollView Content"
            android:textSize="16sp"
            android:textStyle="bold"
            android:paddingTop="10dp" />

    </LinearLayout>
</ScrollView>
```

Java Code (MainActivity.java)

```
package com.example.scrollviewdemo;

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

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

Output:

Conclusion: This application successfully demonstrates the use of ScrollView in Android. It allows the user to scroll through content when the screen size is insufficient to display all elements. This is particularly useful in forms, long text content, and lists.

Submitted By: **Checked By:**

Sign: **Ms. Bhagyashri S. Patil**

Name:

Roll No:

Shram Sadhana Bombay Trust Sanchlit
Arts, Commerce and Science College, Bambhori, Jalgaon
Bachelor of Computer Application (B.C.A.)
Practical: 09

DOP: **DOC**:

Title: Demonstrate use of intent in android.

Objective: To demonstrate the use of **Intent** in Android by creating an application that navigates from one activity to another using both **Explicit Intent** and **Implicit Intent**.

Theory:

Implementation:

The application contains:

- **Two Activities (MainActivity & SecondActivity)** to demonstrate explicit intent.
- **A Button** in the first activity to navigate to the second activity.
- **An EditText** to send user input from one activity to another.
- **An Implicit Intent** to open a webpage in a browser.

Code Implementation:

Step 1: Main Activity (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="16dp">

   <EditText
      android:id="@+id/etMessage"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="Enter Message" />

   <Button
      android:id="@+id/btnNavigate"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:text="Go to Second Activity" />
```

```
    <Button
      android:id="@+id/btnOpenBrowser"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:text="Open Google in Browser" />

</LinearLayout>
```

Step 2: MainActivity.java

```
package com.example.intentdemo;

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 androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    EditText etMessage;
    Button btnNavigate, btnOpenBrowser;

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

        etMessage = findViewById(R.id.etMessage);
        btnNavigate = findViewById(R.id.btnNavigate);
        btnOpenBrowser = findViewById(R.id.btnOpenBrowser);

        // Explicit Intent - Navigating to SecondActivity
        btnNavigate.setOnClickListener(view -> {
            String message = etMessage.getText().toString();
            Intent intent = new Intent(MainActivity.this, SecondActivity.class);
            intent.putExtra("MESSAGE", message);
            startActivity(intent);
        });

        // Implicit Intent - Open Browser
        btnOpenBrowser.setOnClickListener(view -> {
            Intent browserIntent = new Intent(Intent.ACTION_VIEW, Uri.parse("https://www.google.com"));
            startActivity(browserIntent);
        });
    }
}
```

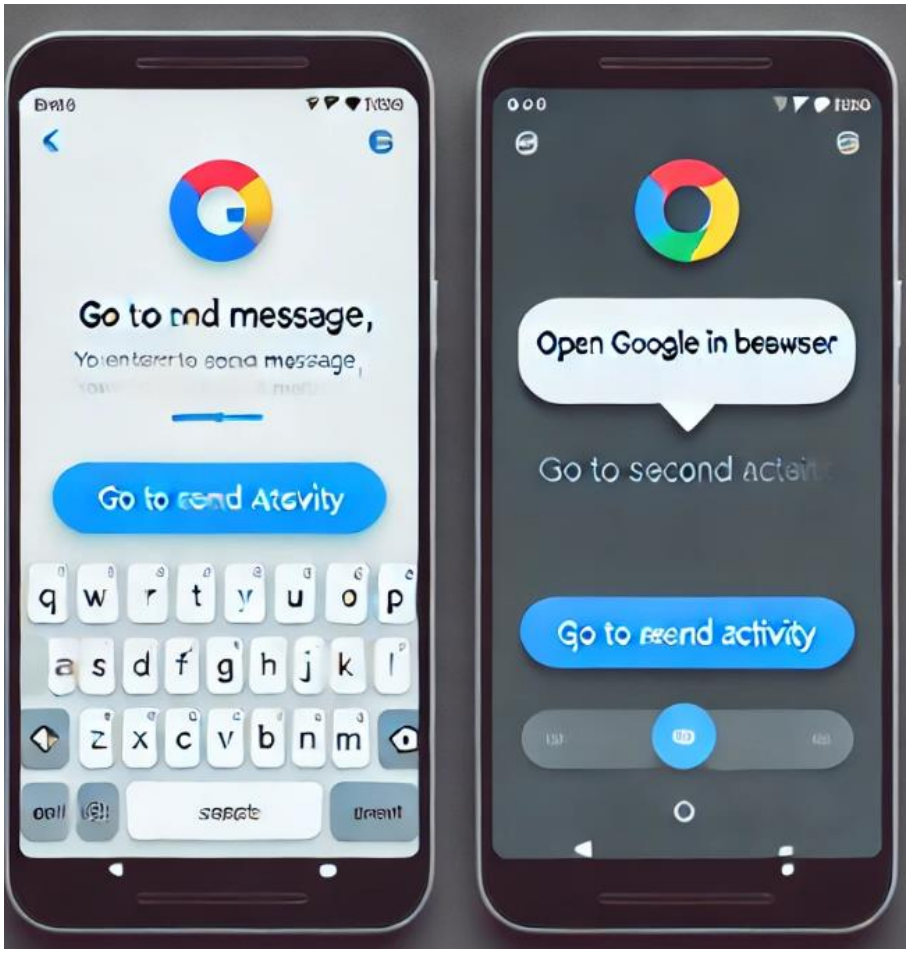
Step 3: Second Activity (activity_second.xml)

```
<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="16dp">

   <TextView
      android:id="@+id/tvReceivedMessage"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:text="Message received:"
      android:textSize="18sp"
      android:textStyle="bold" />

</LinearLayout>
```

Step 4: SecondActivity.java

```
package com.example.intentdemo;

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

public class SecondActivity extends AppCompatActivity {
   TextView tvReceivedMessage;

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

      tvReceivedMessage = findViewById(R.id.tvReceivedMessage);

      // Receiving data from Intent
      String message = getIntent().getStringExtra("MESSAGE");
      tvReceivedMessage.setText("Message received: " + message);
   }
}
```

Output:



Conclusion: This application successfully demonstrates the use of **Explicit Intent** (for switching between activities) and **Implicit Intent** (for opening a web browser). Intents play a crucial role in Android development, enabling seamless communication between activities and applications.

Submitted By: **Checked By:**

Sign: **Ms. Bhagyashri S. Patil**

Name:

Roll No:

Shram Sadhana Bombay Trust Sanchlit
Arts, Commerce and Science College, Bambhori, Jalgaon
Bachelor of Computer Application (B.C.A.)
Practical: 10

DOP: **DOC**:

Title: Create application to demonstrate menu option.

Objective: To develop an Android application that demonstrates the use of an **Options Menu**. The menu will contain different options such as **Settings, About, and Exit**, and will be displayed when the user clicks on the three-dot menu in the Action Bar.

Theory:

Implementation:

The application contains:

- **An Options Menu** that appears in the action bar.
- **Menu Items:**
  - **Settings** → Displays a toast message.
  - **About** → Opens a new activity.
  - **Exit** → Closes the application.

Code Implementation:

Step 1: Update AndroidManifest.xml

Ensure your activity has a theme that supports an Action Bar:

```
<application
   android:theme="@style/Theme.AppCompat.Light.DarkActionBar">
</application>
```

Step 2: Define the Menu Layout (res/menu/menu_main.xml)

Create a new directory **res/menu** if it doesn't exist. Then create **menu_main.xml** inside it.

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
   <item
      android:id="@+id/menu_settings"
      android:title="Settings"
      android:icon="@android:drawable/ic_menu_preferences"
      android:showAsAction="never"/>
```

```
  <item
      android:id="@+id/menu_about"
      android:title="About"
      android:icon="@android:drawable/ic_menu_info_details"
      android:showAsAction="never"/>

    <item
      android:id="@+id/menu_exit"
      android:title="Exit"
      android:icon="@android:drawable/ic_menu_close_clear_cancel"
      android:showAsAction="never"/>
</menu>
```

Step 3: MainActivity.java

```
package com.example.menudemo;

import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
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);
    }

    // Inflating the menu
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.menu_main, menu);
        return true;
    }

    // Handling menu item clicks
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.menu_settings:
                Toast.makeText(this, "Settings Clicked", Toast.LENGTH_SHORT).show();
                return true;

            case R.id.menu_about:
                Intent intent = new Intent(this, AboutActivity.class);
                startActivity(intent);
```

```
            return true;

        case R.id.menu_exit:
            finish();
            return true;

        default:
            return super.onOptionsItemSelected(item);
    }
  }
}
```

Step 4: AboutActivity.java

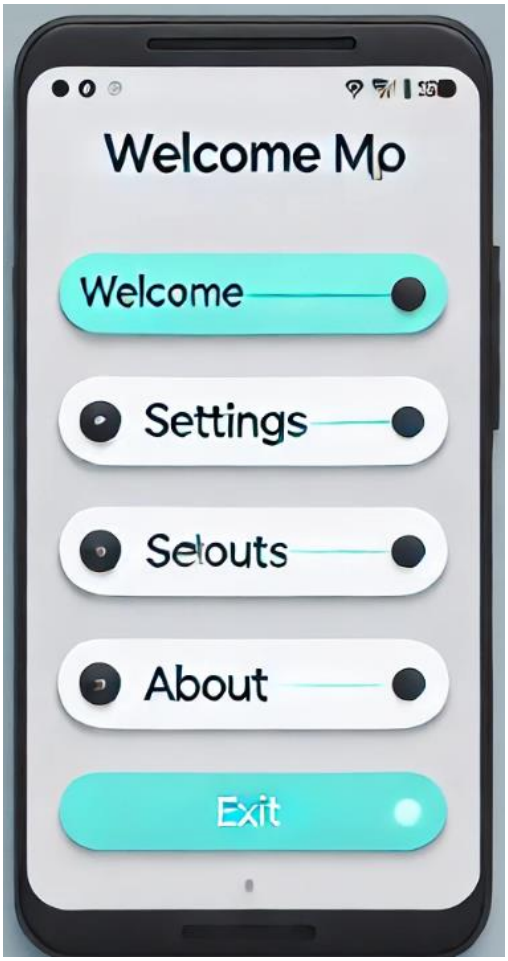
Create a new activity to display when the **About** option is clicked.

```
package com.example.menudemo;

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

public class AboutActivity extends AppCompatActivity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    TextView textView = new TextView(this);
    textView.setText("This is the About Page");
    textView.setTextSize(20);
    setContentView(textView);
  }
}
```

Step 5: Main Activity Layout (activity_main.xml)

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

  <TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Welcome to Menu Demo App"
    android:textSize="20sp"
    android:textStyle="bold" />
</LinearLayout>
```

Output:



Conclusion: This project successfully demonstrates how to implement an **Options Menu** in Android. It shows how to create and handle menu items like **Settings, About, and Exit**, providing a real-world example of navigation and user interaction.

Submitted By: **Checked By:**

Sign: **Ms. Bhagyashri S. Patil**

Name:

Roll No:

Shram Sadhana Bombay Trust Sanchlit
Arts, Commerce and Science College, Bambhori, Jalgaon
Bachelor of Computer Application (B.C.A.)
Practical: 11

DOP: **DOC**:

Title: Create application to demonstrate progress bar.

Objective: To develop an Android application that demonstrates the use of a **Progress Bar**, both **determinate** (fixed progress) and **indeterminate** (loading spinner). The application will include a **Button** to start progress and a **Handler** to simulate a loading process.

Theory:

Implementation:

The application includes:

- **A Circular Indeterminate Progress Bar** to show continuous loading.
- **A Horizontal Determinate Progress Bar** to track progress with a percentage.
- **A Button** to start and simulate progress.
- **A Handler & Runnable** to update the progress bar.

Code Implementation:

Step 1: Layout File (activity_main.xml)

```
<LinearLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:gravity="center"
   android:orientation="vertical"
   android:padding="20dp">

   <ProgressBar
      android:id="@+id/progressCircular"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:visibility="gone"
      style="?android:attr/progressBarStyleLarge"/>

   <ProgressBar
      android:id="@+id/progressHorizontal"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:visibility="gone"
```

```
        style="?android:attr/progressBarStyleHorizontal"
        android:max="100"
        android:progress="0"/>

    <TextView
        android:id="@+id/tvProgress"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Progress: 0%"
        android:padding="10dp"
        android:textSize="18sp"/>

    <Button
        android:id="@+id/btnStart"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Start Progress" />

</LinearLayout>
```

Step 2: Java Code (MainActivity.java)

```
package com.example.progressbardemo;

import android.os.Bundle;
import android.os.Handler;
import android.view.View;
import android.widget.Button;
import android.widget.ProgressBar;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    ProgressBar progressCircular, progressHorizontal;
    TextView tvProgress;
    Button btnStart;
    int progressStatus = 0;
    Handler handler = new Handler();

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

        progressCircular = findViewById(R.id.progressCircular);
        progressHorizontal = findViewById(R.id.progressHorizontal);
        tvProgress = findViewById(R.id.tvProgress);
        btnStart = findViewById(R.id.btnStart);
```

```
        btnStart.setOnClickListener(v -> startProgress());
    }

    private void startProgress() {
        progressStatus = 0;
        progressCircular.setVisibility(View.VISIBLE);
        progressHorizontal.setVisibility(View.VISIBLE);
        tvProgress.setText("Progress: 0%");

        new Thread(() -> {
            while (progressStatus < 100) {
                progressStatus += 10;
                handler.post(() -> {
                    progressHorizontal.setProgress(progressStatus);
                    tvProgress.setText("Progress: " + progressStatus + "%");
                });
                try {
                    Thread.sleep(500);
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
            handler.post(() -> {
                progressCircular.setVisibility(View.GONE);
                progressHorizontal.setVisibility(View.GONE);
                tvProgress.setText("Progress Completed!");
            });
        }).start();
    }
}
```

Output:

Conclusion: This application successfully demonstrates the use of **ProgressBar** in Android. It shows both **indeterminate progress bars** (for loading indicators) and **determinate progress bars** (for progress tracking). The implementation uses **Handler & Thread** to simulate a real-time progress update.

Submitted By: **Checked By:**

Sign: **Ms. Bhagyashri S. Patil**

Name:

Roll No: