**STOPWATCH**

**user interface with start, stop, and reset buttons, and implement logic to measure time and update the UI accordingly**

**Step 1: Set up the Android Project**

1. Open Android Studio.
2. Click on "Start a new Android Studio project."
3. Choose "Empty Activity" as the template and click "Next."
4. Configure your project details (e.g., name, package name, location) and click "Finish."

**Step 2: Design the User Interface**

Open the **activity\_main.xml** layout file in the **res/layout** folder, and replace the default XML code with the following layout for the stopwatch:

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

<RelativeLayout

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

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:padding="16dp"

tools:context=".MainActivity">

<Chronometer

android:id="@+id/chronometer"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textSize="48sp"

android:layout\_centerInParent="true"

android:layout\_marginBottom="32dp"/>

<Button

android:id="@+id/startButton"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Start"

android:layout\_below="@+id/chronometer"

android:layout\_marginTop="16dp"

android:layout\_marginEnd="16dp"

android:layout\_alignParentEnd="true"/>

<Button

android:id="@+id/stopButton"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Stop"

android:layout\_below="@+id/chronometer"

android:layout\_marginTop="16dp"

android:layout\_marginStart="16dp"

android:layout\_toStartOf="@+id/startButton"/>

</RelativeLayout>

**Step 3: Implement App Logic**

Open the **MainActivity.java** file in the **java/com.example.yourpackage** folder and write the code to handle button clicks and start/stop the stopwatch:

package com.example.myapplication;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;

import android.os.SystemClock;

import android.view.View;

import android.widget.Button;

import android.widget.Chronometer;

public class MainActivity extends AppCompatActivity {

private Chronometer chronometer;

private Button startButton, stopButton;

private boolean isRunning = false;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

chronometer = findViewById(R.id.chronometer);

startButton = findViewById(R.id.startButton);

stopButton = findViewById(R.id.stopButton);

startButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

if (!isRunning) {

chronometer.setBase(SystemClock.elapsedRealtime() - chronometer.getBase());

chronometer.start();

isRunning = true;

startButton.setEnabled(false);

stopButton.setEnabled(true);

}

}

});

stopButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View view) {

if (isRunning) {

chronometer.stop();

isRunning = false;

startButton.setEnabled(true);

stopButton.setEnabled(false);

}

}

});

}

}

**Step 4: Run the App**

1. Connect an Android device or start an emulator.
2. Click the "Run" button in Android Studio to build and run the app on the selected device or emulator.
3. The app should launch, and you can start and stop the stopwatch by clicking the "Start" and "Stop" buttons, respectively. The elapsed time will be displayed on the screen.