**Sanika Kadam**

**2124UCSF1112**

**ASSIGNMENT NO.6**

1. **Introduction**

Audio Player Program: Write a program to play audio files. Include a user interface with play, pause, and stop buttons. For example, create a music player that allows users to select and play their favorite songs from the device.

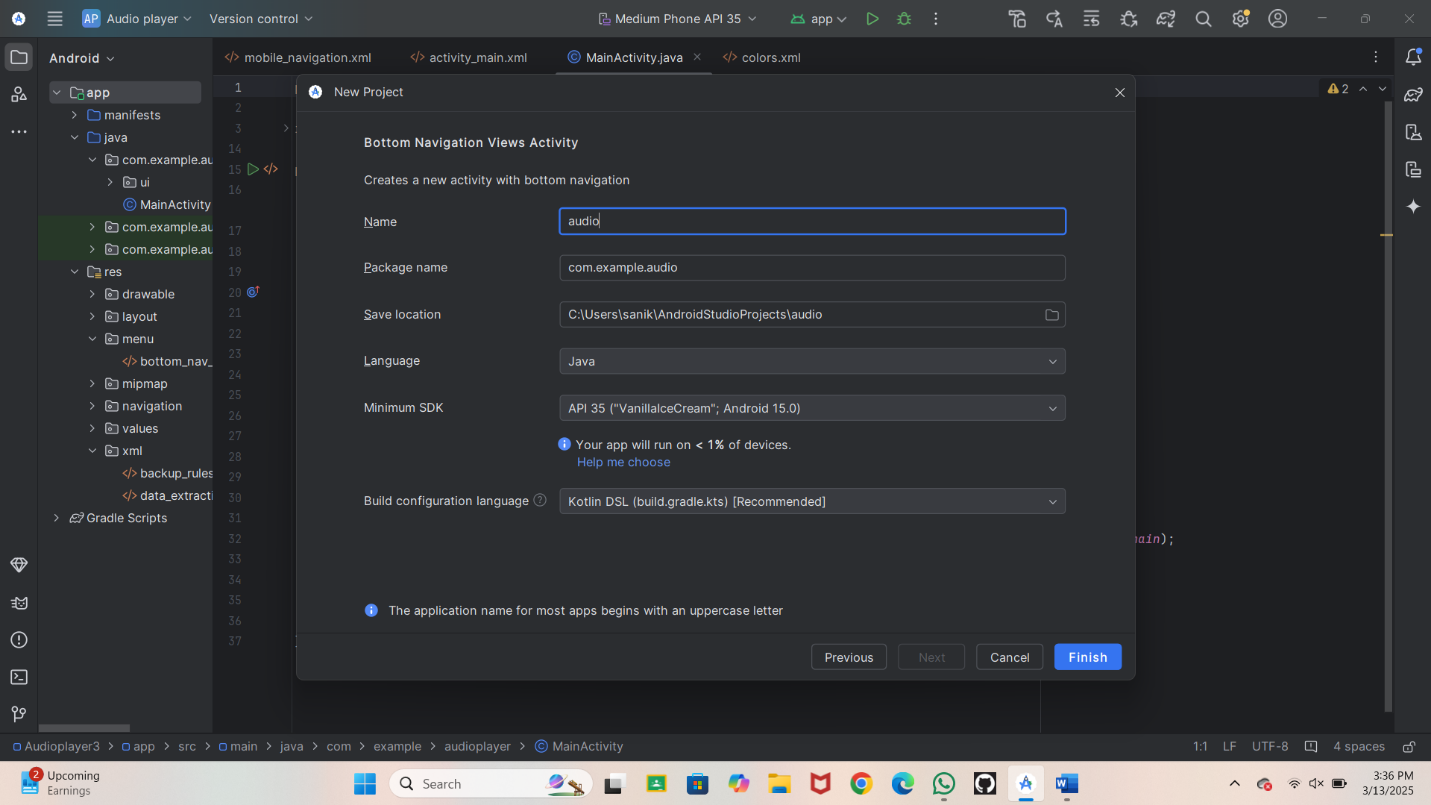
**2. Tools & Technologies Used**

* Android Studio
* Language-Java

**3. Procedure & Steps**

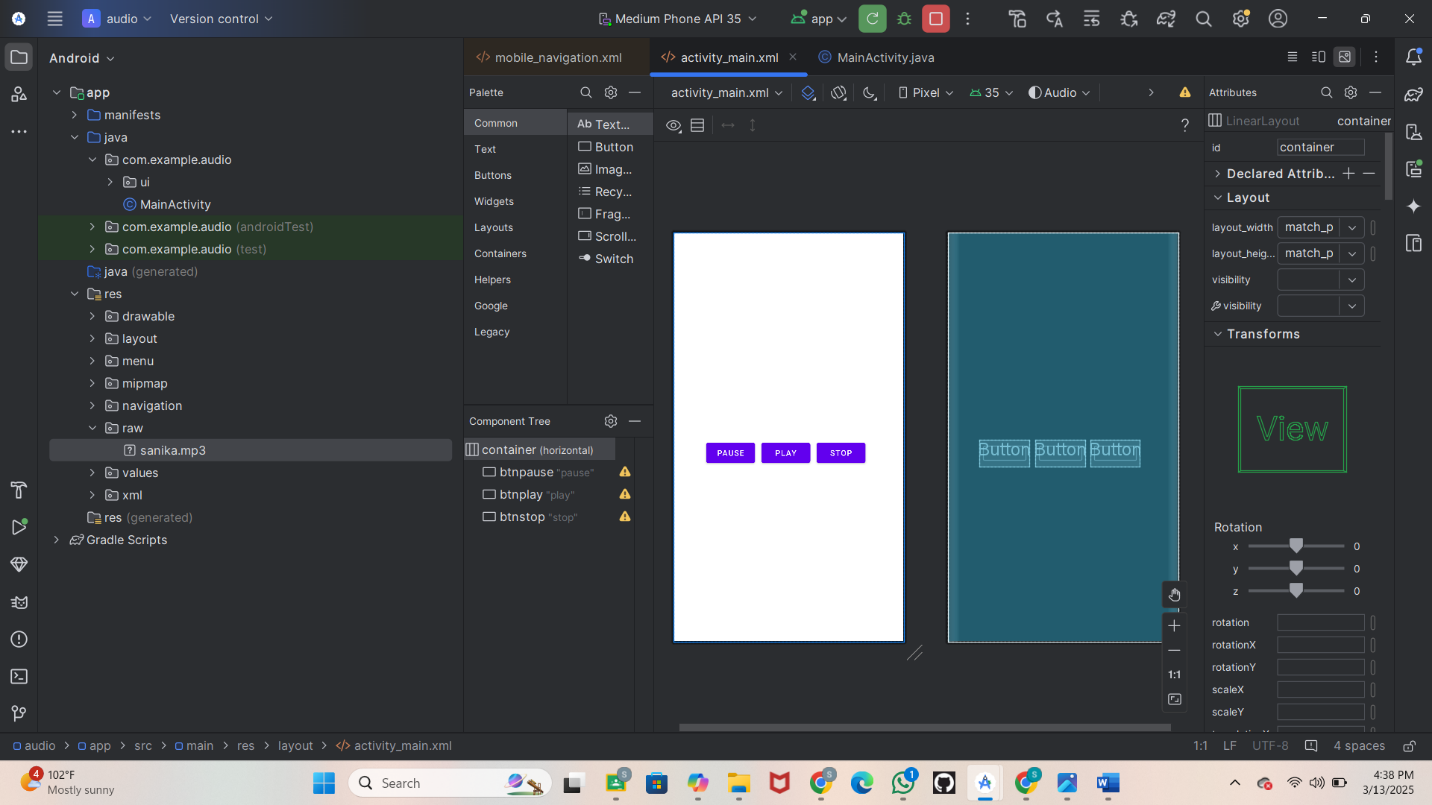
**Step 1: Create a New Project**

* Open Android Studio and create a new project.
* Select second page activity by app-activity-page you selected
* Set the project name(Audio) and package name.(com.example.audio)
* Select the programming language (Java).

**Screenshot:** ****

**Step 2: Designing the UI**

* Open activity\_main.xml and design the layout using XML.
* Add UI components such as TextView, EditText, Button, etc.



**Step 3: Writing the Code**

* Open MainActivity.java or MainActivity.kt.
* Implement functionality such as button clicks, form validation, etc.
* Use necessary Android components like Intents, RecyclerView, Fragments, etc.

Activity main xml

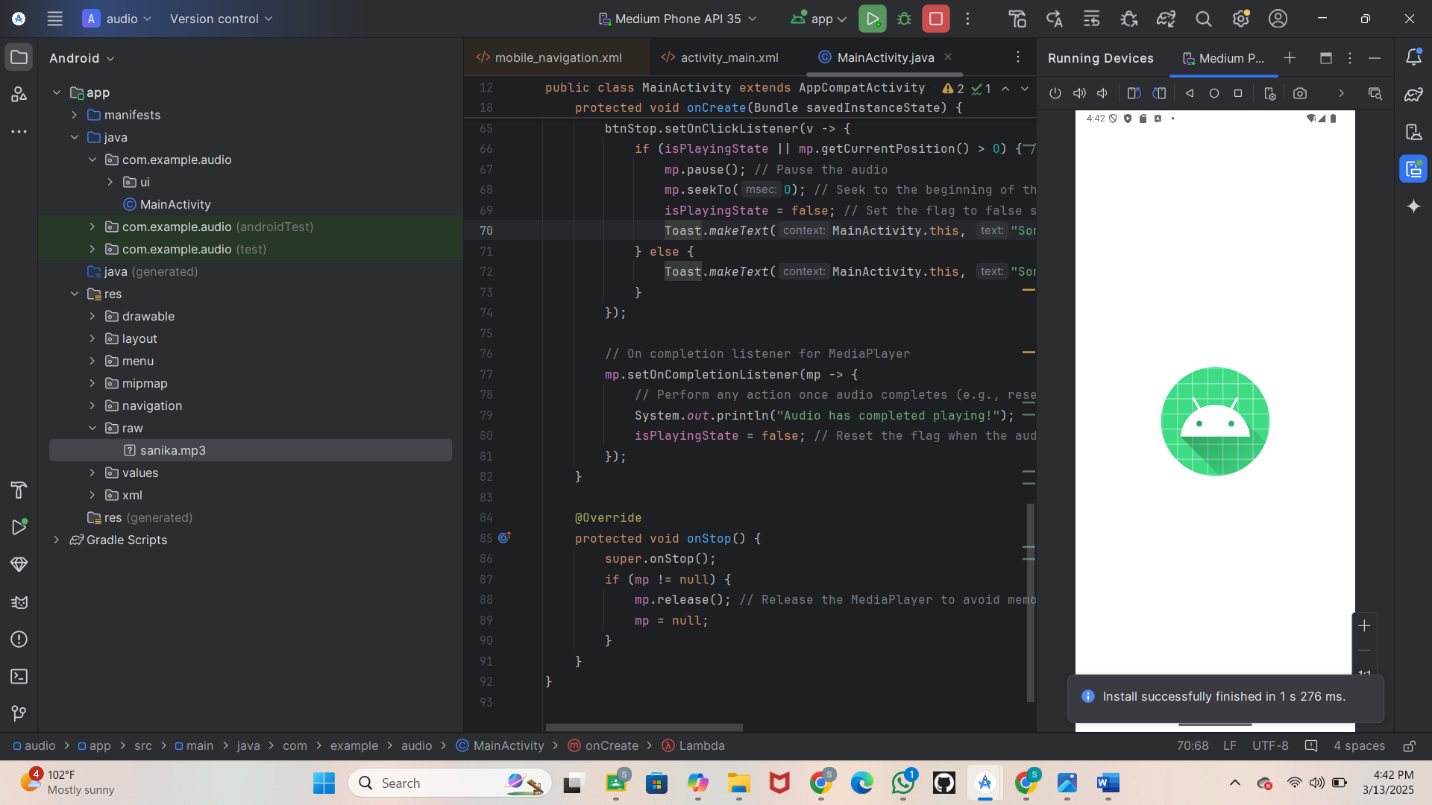
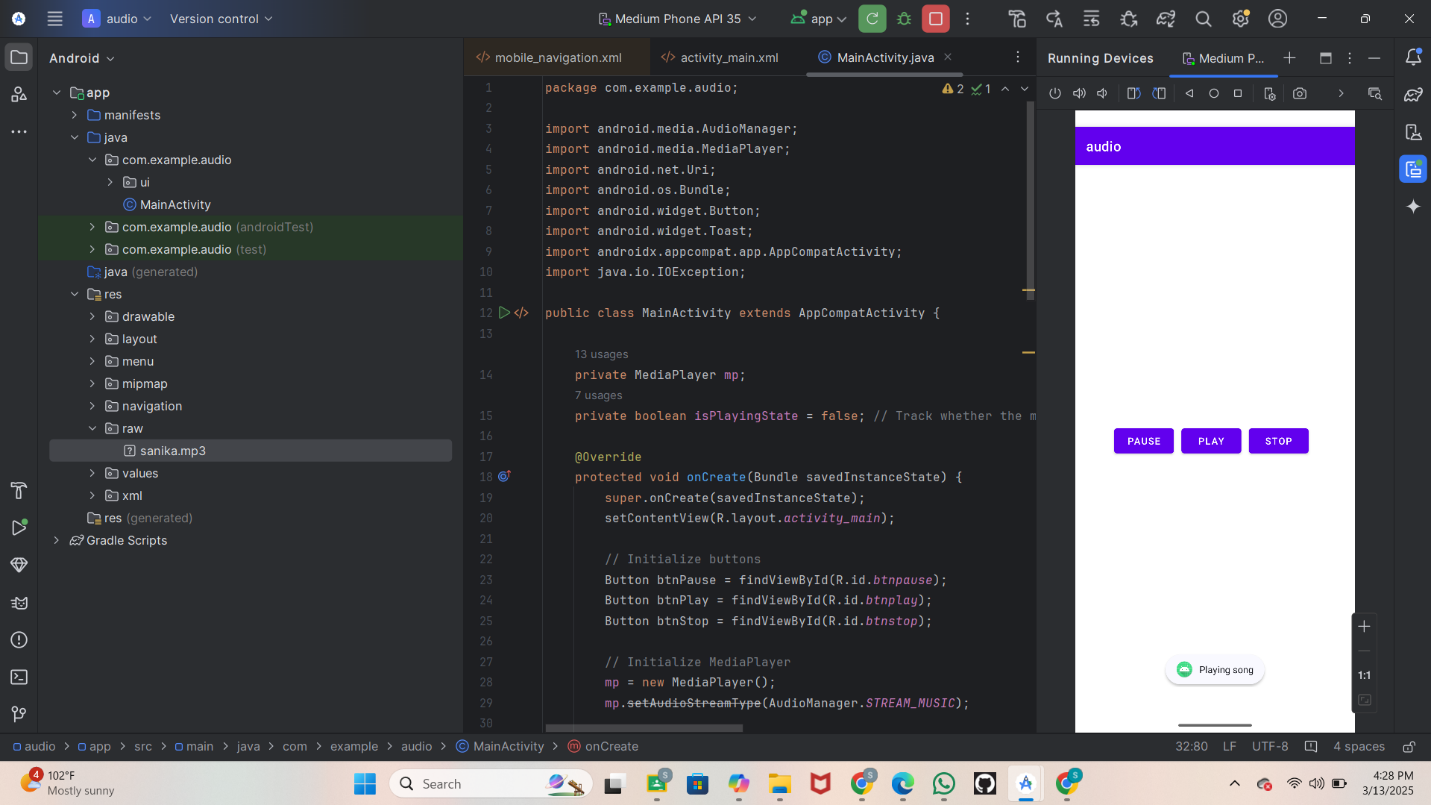
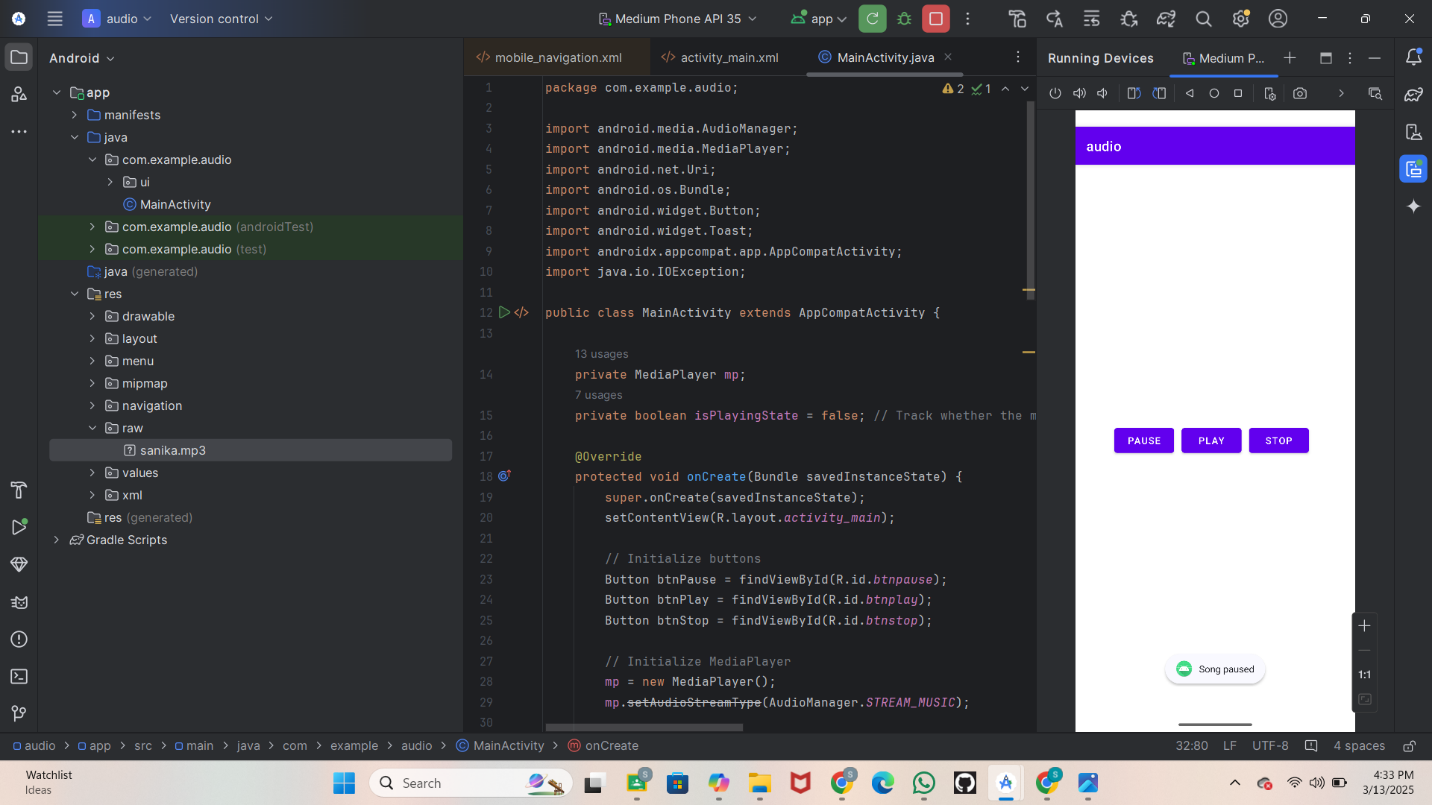
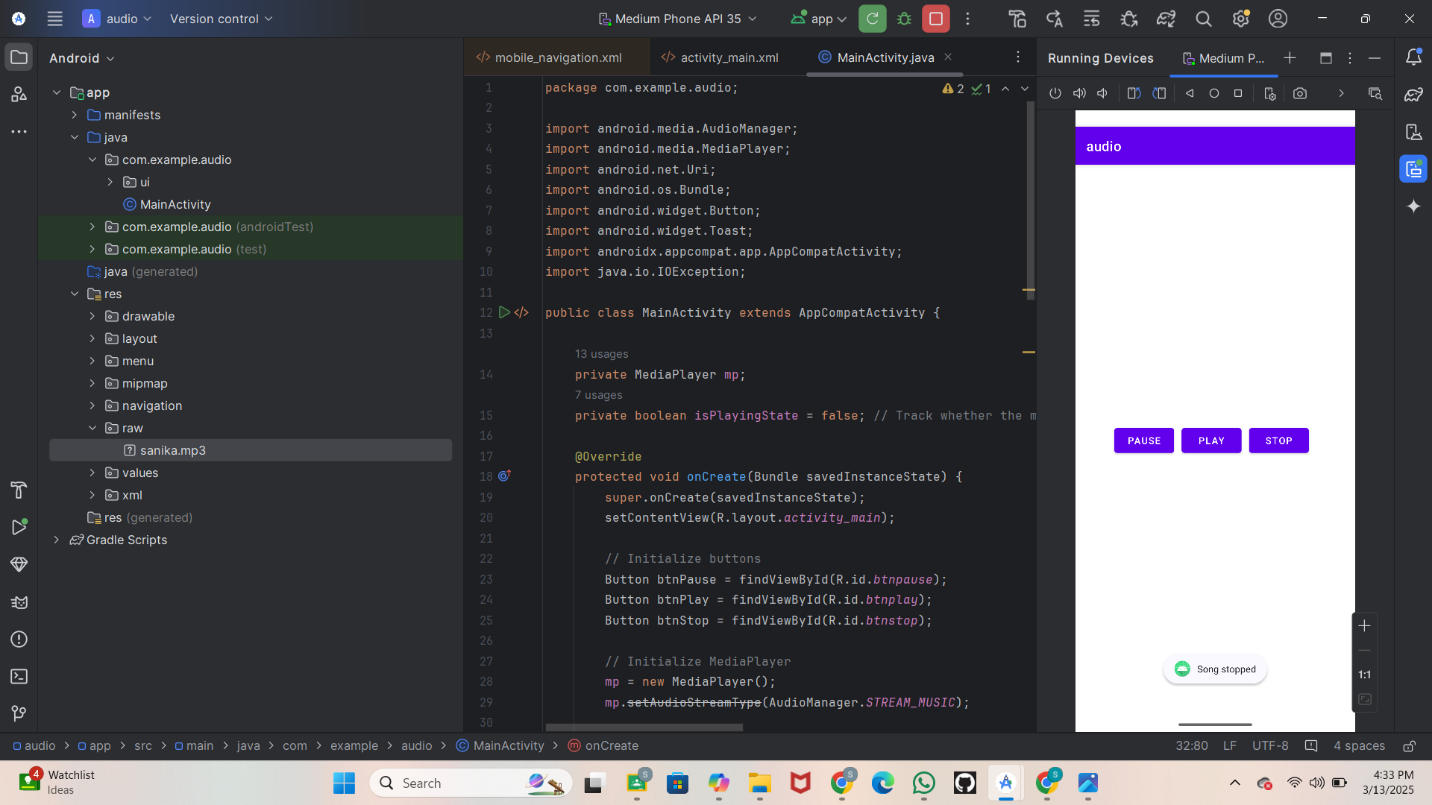
<?xml version="1.0" encoding="utf-8"?><LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:id="@+id/container"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:paddingTop="?attr/actionBarSize"  
 tools:context=".MainActivity"  
 android:gravity="center">  
  
  
 <Button  
 android:id="@+id/btnpause"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="pause"  
 android:layout\_marginRight="11dp"  
 android:layout\_marginEnd="11dp"/>  
  
 <Button  
 android:id="@+id/btnplay"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginEnd="11dp"  
 android:layout\_marginRight="11dp"  
 android:text="play" />  
  
 <Button  
 android:id="@+id/btnstop"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="stop"  
 android:layout\_marginRight="11dp"  
 android:layout\_marginEnd="11dp"/>  
  
</LinearLayout>

**Step**

Then write a code in main activity JAVA

package com.example.audio;  
  
import android.media.AudioManager;  
import android.media.MediaPlayer;  
import android.net.Uri;  
import android.os.Bundle;  
import android.widget.Button;  
import android.widget.Toast;  
import androidx.appcompat.app.AppCompatActivity;  
import java.io.IOException;  
  
public class MainActivity extends AppCompatActivity {  
  
 private MediaPlayer mp;  
 private boolean isPlayingState = false; // Track whether the media is playing or paused  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 // Initialize buttons  
 Button btnPause = findViewById(R.id.*btnpause*);  
 Button btnPlay = findViewById(R.id.*btnplay*);  
 Button btnStop = findViewById(R.id.*btnstop*);  
  
 // Initialize MediaPlayer  
 mp = new MediaPlayer();  
 mp.setAudioStreamType(AudioManager.*STREAM\_MUSIC*);  
  
 // Correct path to audio resource  
 String apathy = "android.resource://" + getPackageName() + "/raw/sanika";  
 Uri audioURI = Uri.*parse*(apathy);  
  
 try {  
 mp.setDataSource(this, audioURI); // Correct usage of context  
 mp.prepare(); // Prepare the MediaPlayer  
 } catch (IOException e) {  
 e.printStackTrace(); // Handle exception if something goes wrong  
 }  
  
 // Play button listener  
 btnPlay.setOnClickListener(v -> {  
 if (!isPlayingState) {  
 mp.start(); // Start playing the audio  
 isPlayingState = true; // Set the flag to true since the audio is now playing  
 Toast.*makeText*(MainActivity.this, "Playing song", Toast.*LENGTH\_SHORT*).show(); // Show message  
 } else {  
 Toast.*makeText*(MainActivity.this, "Song is already playing", Toast.*LENGTH\_SHORT*).show(); // Show message if already playing  
 }  
 });  
  
 // Pause button listener  
 btnPause.setOnClickListener(v -> {  
 if (isPlayingState) {  
 mp.pause(); // Pause the audio  
 isPlayingState = false; // Set the flag to false since the audio is now paused  
 Toast.*makeText*(MainActivity.this, "Song paused", Toast.*LENGTH\_SHORT*).show(); // Show message  
 } else {  
 Toast.*makeText*(MainActivity.this, "Song is already paused", Toast.*LENGTH\_SHORT*).show(); // Show message if already paused  
 }  
 });  
  
 // Stop button listener  
 btnStop.setOnClickListener(v -> {  
 if (isPlayingState || mp.getCurrentPosition() > 0) { // Check if the song is playing or has been started  
 mp.pause(); // Pause the audio  
 mp.seekTo(0); // Seek to the beginning of the audio  
 isPlayingState = false; // Set the flag to false since the audio is stopped  
 Toast.*makeText*(MainActivity.this, "Song stopped", Toast.*LENGTH\_SHORT*).show(); // Show message  
 } else {  
 Toast.*makeText*(MainActivity.this, "Song is already stopped", Toast.*LENGTH\_SHORT*).show(); // Show message if already stopped  
 }  
 });  
  
 // On completion listener for MediaPlayer  
 mp.setOnCompletionListener(mp -> {  
 // Perform any action once audio completes (e.g., reset, show a message, etc.)  
 System.*out*.println("Audio has completed playing!");  
 isPlayingState = false; // Reset the flag when the audio finishes  
 });  
 }  
  
 @Override  
 protected void onStop() {  
 super.onStop();  
 if (mp != null) {  
 mp.release(); // Release the MediaPlayer to avoid memory leaks  
 mp = null;  
 }  
 }  
}

**Step 4: Running the Application on Emulator**

* Click on the **Run** button in Android Studio. 
* **Step 5: Testing & Output**
* Test different functionalities of the app.
* the output results.
* Start The audio
* **Screenshot:**
* 
* Pause the Song
* Stop the song
* 

## **Step 5. Conclusion**

Creating an audio player with Play, Pause, and Stop buttons is a foundational project in Android development. It allows users to easily interact with audio files and demonstrates a basic understanding of the MediaPlayer API, UI design.