BATCH AND ROLL NO:
EXPERIMENT NO.8
TITLE: Design a mobile application for media player.
DATE OF PERFORMANCE:
DATE OF SUBMISSION:

Title: Design a mobile application for media player.

Requirements:
1 Android studio

Theory:

Introduction

In the realm of mobile application development, creating a media player application provides an avenue to deliver immersive and interactive experiences for users. This lab focuses on the design and implementation of a mobile media player application, empowering users to enjoy audio or video content seamlessly. The integration of a media player not only enhances the entertainment aspect of an application but also showcases the utilization of multimedia capabilities in modern mobile devices.

Objective of the Lab: The primary objective of this lab is to guide you through the process of designing a mobile application equipped with a media player component. By the end of this lab, you should be proficient in implementing features such as playing, pausing, and controlling media playback. Additionally, you will explore aspects like handling media files, implementing user controls, and providing a seamless and engaging media playback experience.

Components of the Application:

1. **Media Player Component:**
    - The media player component serves as the core element responsible for handling and controlling audio or video playback.
    - It includes functionalities such as play, pause, stop, forward, and rewind, contributing to a user-friendly and feature-rich media experience.

Lab Prerequisites:

- Basic understanding of mobile application development concepts.
- Familiarity with the chosen development environment (e.g., Android Studio, Xcode).
- Prior knowledge of programming languages such as Java or Kotlin (for Android) or Swift (for iOS).

Steps:

Step 1: Set Up Your Development Environment

- Ensure that you have Android Studio installed and configured on your machine.

Step 2: Create a New Project

- Open Android Studio and create a new project.
- Choose an appropriate project template, such as "Empty Activity" or "Basic Activity."

Step 3: Design the Main Activity Layout

- Open the XML layout file associated with your main activity (e.g., activity_main.xml).
- Design the layout with relevant UI elements, such as buttons for play, pause, stop, and a SeekBar for progress tracking.

Step 4: Implement the Java Code

- Open the Java file associated with your main activity (e.g., MainActivity.java)
- Implement the logic for initializing the media player, handling button clicks, and updating the SeekBar.

Step 5: Test Your Application

- Run your application on an emulator or a physical device.
- Verify that the media player buttons function correctly, and the SeekBar updates as the media plays.

XML Code:

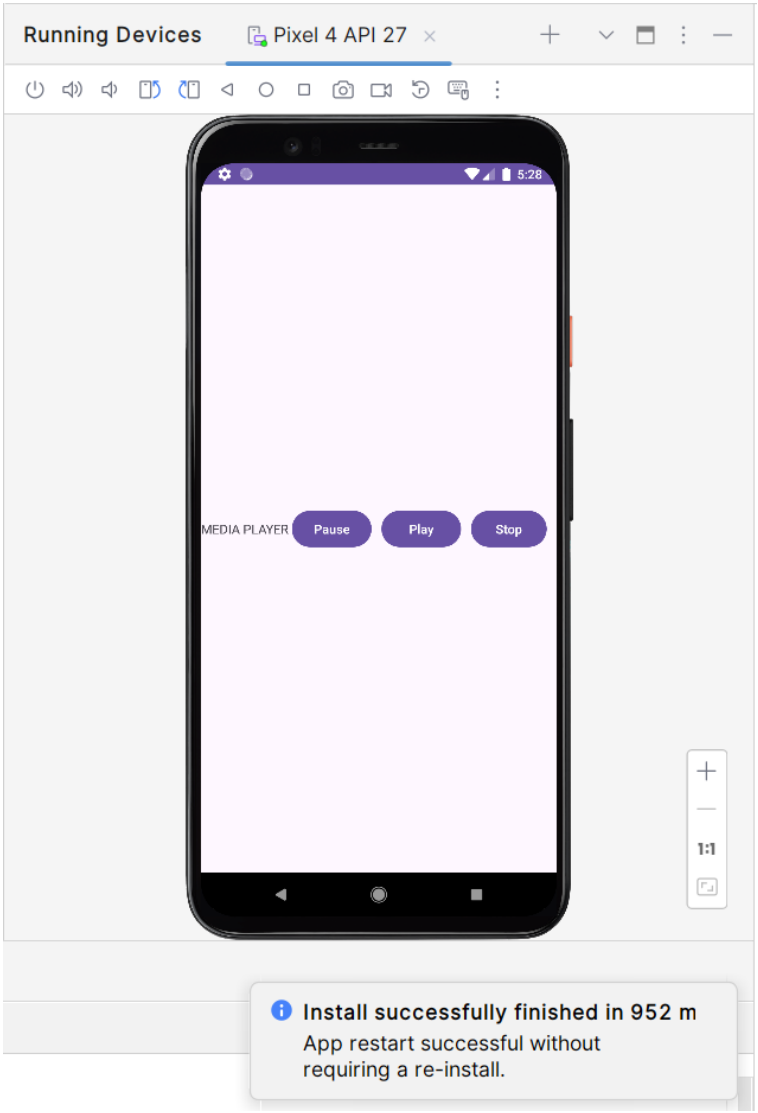
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="MEDIA PLAYER "
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"



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

    <Button
        android:id="@+id/btnPause"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginRight="11dp"
        android:text="Pause" />

    <Button
        android:id="@+id/btnPlay"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginRight="11dp"
        android:text="Play" />

    <Button
        android:id="@+id/btnStop"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginRight="11dp"
        android:text="Stop" />

</LinearLayout>
```

Java Code:

```
package com.example.media_player;

import androidx.appcompat.app.AppCompatActivity;

import android.media.AudioManager;
import android.media.MediaPlayer;
```

```
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import java.io.IOException;

public class MainActivity extends AppCompatActivity {

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

        Button btnPause, btnPlay, btnStop;

        btnPause = findViewById(R.id.btnPause);
        btnPlay = findViewById(R.id.btnPlay);
        btnStop = findViewById(R.id.btnStop);
        MediaPlayer mp = new MediaPlayer();
        mp.setAudioStreamType(AudioManager.STREAM_MUSIC);

String aPath ="android.resource://"+getPackageName()+"/raw/zaalim_badshah";
        Uri audioURI = Uri.parse(aPath);
        try {
            mp.setDataSource(this, audioURI);
            mp.prepare();
        } catch (IOException e) {
            throw new RuntimeException(e);
        }

        btnPause.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                mp.pause();

            }
        });

        btnPlay.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                mp.start();

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

                mp.pause();
                mp.seekTo(0);

            }
        });
    }
}
```

Output:



Conclusion:

……

……

……

……

……

……