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 -->
```

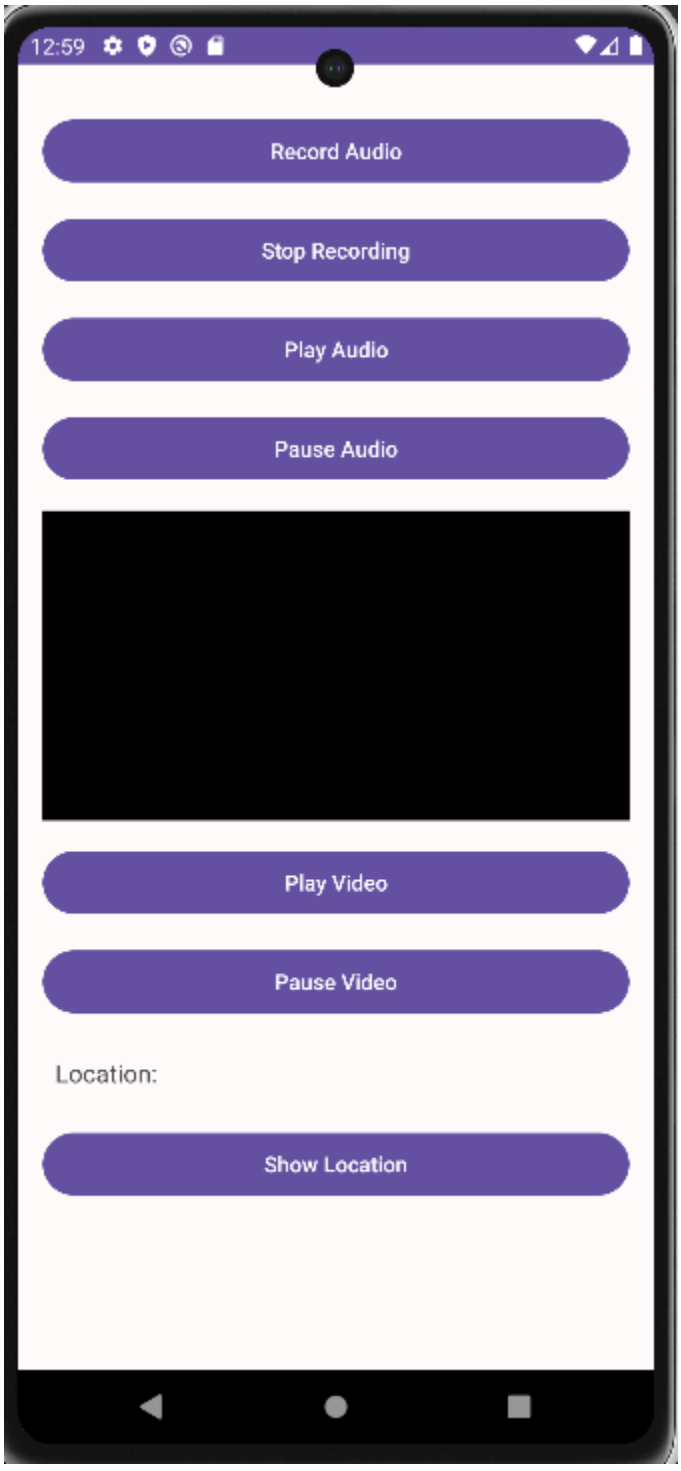
```
<Button
    android:id="@+id/playButton"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Play Audio"
    android:layout_marginTop="16dp"/>

<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"/>
```

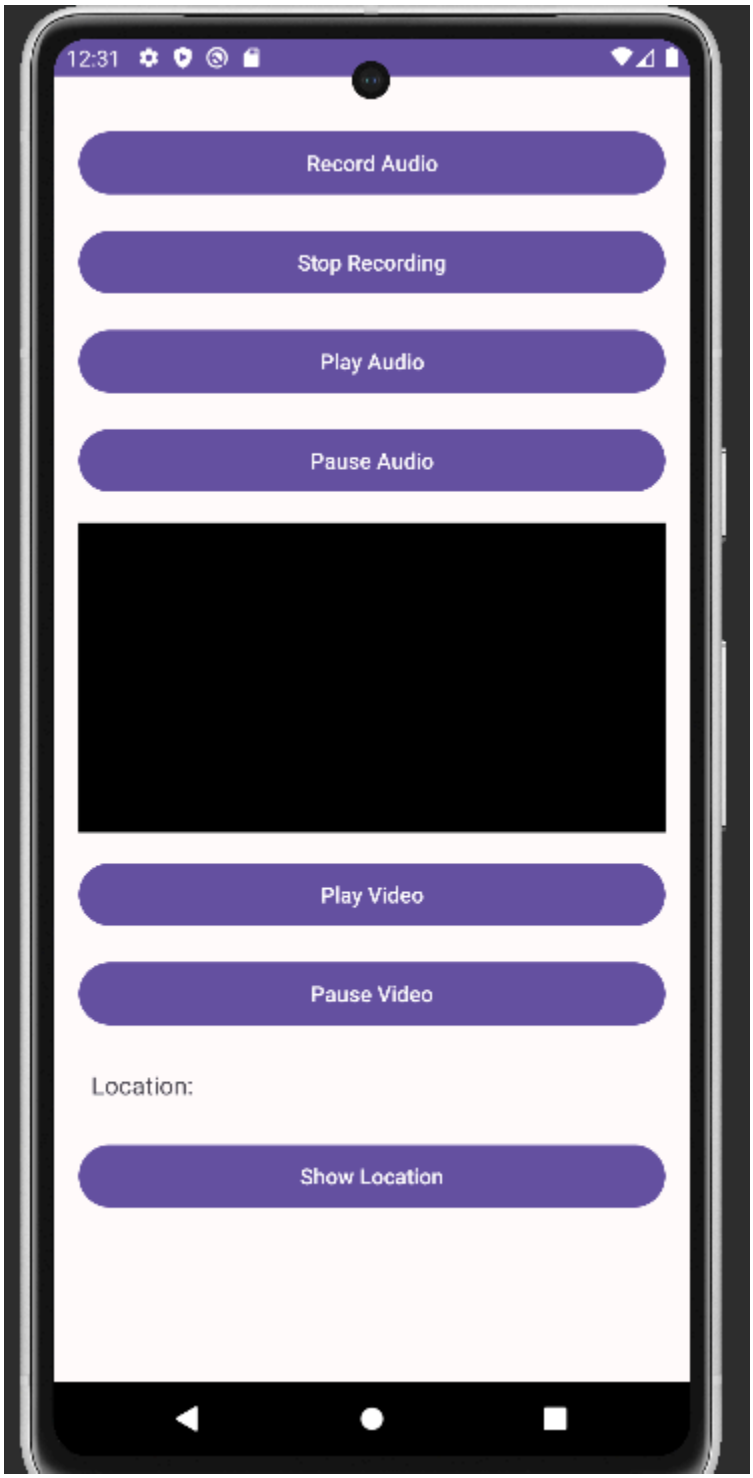
```
    <!-- Button to Show Location -->
    <Button
        android:id="@+id/showLocationButton"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Show Location"
        android:layout_marginTop="16dp"/>
</LinearLayout>
```

Output

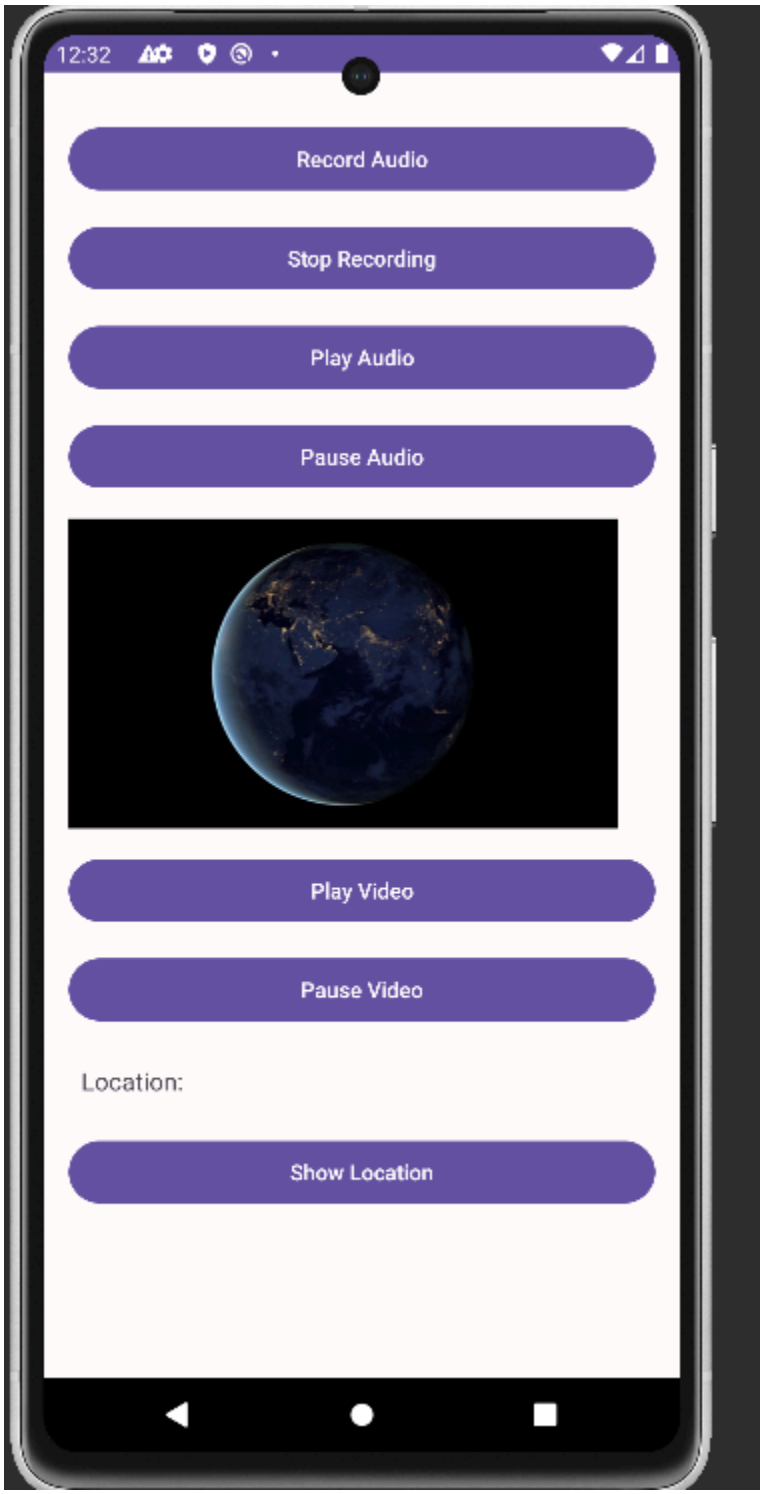
Record Audio


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

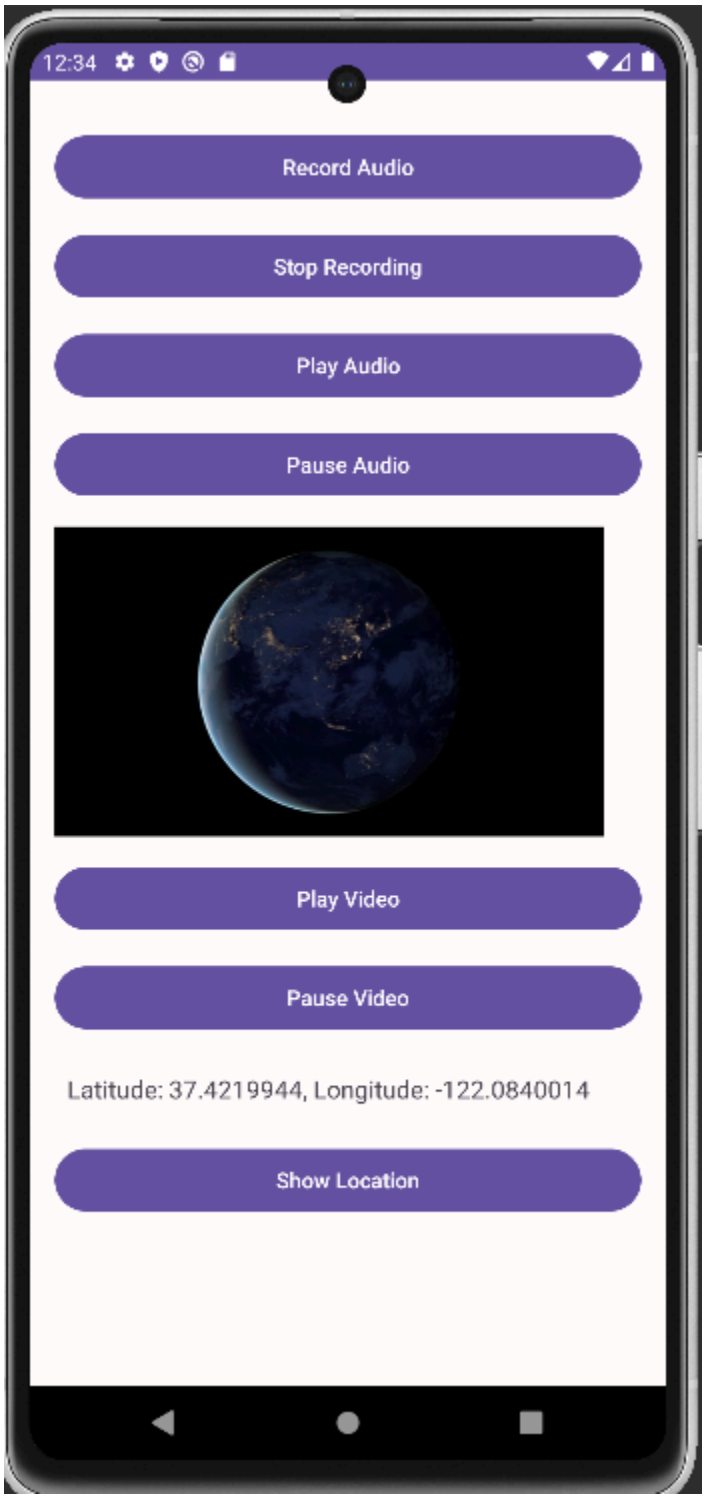
Stop Audio


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.