add this in manifest

```xml
<uses-permission android:name="android.permission.ACCESS_NOTIFICATION_POLICY"
/>
<uses-permission android:name="android.permission.POST_NOTIFICATIONS"/>
```

activity_main.xml

```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="16dp"
   android:background="#F0F4F8">

   <TextView
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       android:text="Data Storage App"
       android:textSize="24sp"
       android:textColor="#2C3E50"
       android:textStyle="bold"
       android:gravity="center"
       android:layout_marginBottom="20dp"/>

   <EditText
       android:id="@+id/fileNameInput"
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       android:hint="Enter Filename"
       android:padding="12dp"

       android:layout_marginBottom="10dp"/>

   <EditText
       android:id="@+id/dataInput"
       android:layout_width="match_parent"
       android:layout_height="150dp"
       android:hint="Enter Data"
       android:gravity="top"
       android:padding="12dp"
       android:layout_marginBottom="10dp"/>

   <RadioGroup
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       android:orientation="horizontal"
       android:gravity="center"
```

```
        android:layout_marginBottom="10dp">

        <RadioButton
            android:id="@+id/internalStorageRadio"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Internal Storage"
            android:layout_marginEnd="20dp"/>

        <RadioButton
            android:id="@+id/externalStorageRadio"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="External Storage"/>
    </RadioGroup>

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

        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Save"
            android:onClick="saveData"
            android:layout_marginEnd="10dp"/>

        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Load"
            android:onClick="loadData"/>
    </LinearLayout>

</LinearLayout>
```

MainActivity.java

```
package com.example.ad9_pr;

import android.annotation.SuppressLint;
import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.content.Context;
import android.os.Build;
import android.os.Bundle;
import android.os.Environment;
import android.view.View;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.NotificationCompat;
import androidx.core.app.NotificationManagerCompat;
import java.io.*;

public class MainActivity extends AppCompatActivity {

   private EditText dataInput, fileNameInput;
   private RadioButton internalStorageRadio, externalStorageRadio;
   private static final String CHANNEL_ID = "load_success_channel"; //
Notification Channel ID

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

       dataInput = findViewById(R.id.dataInput);
       fileNameInput = findViewById(R.id.fileNameInput);
       internalStorageRadio = findViewById(R.id.internalStorageRadio);
       externalStorageRadio = findViewById(R.id.externalStorageRadio);

       createNotificationChannel(); // Ensure notification channel exists
   }

   // ✅ Save Data Method - Writes data to selected storage
   public void saveData(View view) {
       String fileName = fileNameInput.getText().toString().trim();
       String data = dataInput.getText().toString().trim();

       if (fileName.isEmpty() || data.isEmpty()) {
           Toast.makeText(this, "Filename and data cannot be empty!",
Toast.LENGTH_SHORT).show();
           return;
       }

       try {
           if (internalStorageRadio.isChecked()) {
               writeInternalStorage(fileName, data);
           } else if (externalStorageRadio.isChecked()) {
               writeExternalStorage(fileName, data);
           } else {
               Toast.makeText(this, "Select storage type",
Toast.LENGTH_SHORT).show();
               return;
```

```
            }
            Toast.makeText(this, "Data saved successfully!",
Toast.LENGTH_SHORT).show();
        } catch (IOException e) {
            Toast.makeText(this, "Error saving data: " + e.getMessage(),
Toast.LENGTH_SHORT).show();
        }
    }

    // ✅ Load Data Method - Reads data from selected storage
    public void loadData(View view) {
        String fileName = fileNameInput.getText().toString().trim();

        if (fileName.isEmpty()) {
            Toast.makeText(this, "Please enter filename",
Toast.LENGTH_SHORT).show();
            return;
        }

        try {
            String data;
            if (internalStorageRadio.isChecked()) {
                File file = new File(getFilesDir(), fileName);
                if (!file.exists()) {
                    Toast.makeText(this, "File not found in Internal Storage",
Toast.LENGTH_SHORT).show();
                    return;
                }
                data = readInternalStorage(fileName);
            } else if (externalStorageRadio.isChecked()) {
                File file = new File(getExternalFilesDir(null), fileName);
                if (!file.exists()) {
                    Toast.makeText(this, "File not found in External Storage",
Toast.LENGTH_SHORT).show();
                    return;
                }
                data = readExternalStorage(fileName);
            } else {
                Toast.makeText(this, "Select storage type",
Toast.LENGTH_SHORT).show();
                return;
            }

            dataInput.setText(data); // Display retrieved data in the EditText
            showSuccessNotification(fileName); // Show notification after
loading

        } catch (Exception e) {
            Toast.makeText(this, "Error reading data: " + e.getMessage(),
Toast.LENGTH_SHORT).show();
        }
    }

    // ✅ Shows Notification After Successful Load
    private void showSuccessNotification(String fileName) {
        NotificationCompat.Builder builder = new
NotificationCompat.Builder(this, CHANNEL_ID)
                .setSmallIcon(R.drawable.ic_notification) // Ensure this icon
exists in res/drawable
                .setContentTitle("Load Successful")
                .setContentText("File '" + fileName + "' has been loaded
successfully!")
                .setPriority(NotificationCompat.PRIORITY_HIGH);

        NotificationManagerCompat notificationManager =
NotificationManagerCompat.from(this);
        if (notificationManager.areNotificationsEnabled()) {
            notificationManager.notify(1, builder.build());
        } else {
            Toast.makeText(this, "Notifications are disabled",
Toast.LENGTH_SHORT).show();
        }
    }

    // ✅ Creates Notification Channel for Android 8.0+
    private void createNotificationChannel() {
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            CharSequence name = "Load Success Channel";
            String description = "Notifies when a file is successfully loaded.";
            int importance = NotificationManager.IMPORTANCE_HIGH;
            NotificationChannel channel = new NotificationChannel(CHANNEL_ID,
name, importance);
            channel.setDescription(description);
            NotificationManager notificationManager =
getSystemService(NotificationManager.class);
            notificationManager.createNotificationChannel(channel);
        }
    }

    // ✅ Writes Data to Internal Storage
    private void writeInternalStorage(String fileName, String data) throws
IOException {
        try (FileOutputStream fos = openFileOutput(fileName,
Context.MODE_PRIVATE);
             OutputStreamWriter osw = new OutputStreamWriter(fos);
             BufferedWriter bw = new BufferedWriter(osw)) {
            bw.write(data);
        }
    }

    // ✅ Reads Data from Internal Storage
    private String readInternalStorage(String fileName) throws IOException {
        try (FileInputStream fis = openFileInput(fileName);
             BufferedReader br = new BufferedReader(new InputStreamReader(fis)))
{
            StringBuilder sb = new StringBuilder();
            String line;
            while ((line = br.readLine()) != null) {
                sb.append(line).append("\n");
            }
            return sb.toString().trim();
        }
    }

    // ✅ Writes Data to External Storage
    private void writeExternalStorage(String fileName, String data) throws
IOException {
        if (isExternalStorageWritable()) {
            File file = new File(getExternalFilesDir(null), fileName);
            try (FileOutputStream fos = new FileOutputStream(file);
                 OutputStreamWriter osw = new OutputStreamWriter(fos);
                 BufferedWriter bw = new BufferedWriter(osw)) {
                bw.write(data);
            }
        } else {
            throw new IOException("External Storage Not Writable");
        }
    }

    // ✅ Reads Data from External Storage
    private String readExternalStorage(String fileName) throws IOException {
        if (isExternalStorageReadable()) {
            File file = new File(getExternalFilesDir(null), fileName);
            try (FileInputStream fis = new FileInputStream(file);
                 BufferedReader br = new BufferedReader(new
InputStreamReader(fis))) {
                StringBuilder sb = new StringBuilder();
                String line;
                while ((line = br.readLine()) != null) {
                    sb.append(line).append("\n");
                }
                return sb.toString().trim();
            }
        } else {
            throw new IOException("External Storage Not Readable");
        }
    }

    // ✅ Check if External Storage is Writable
    private boolean isExternalStorageWritable() {
        return
Environment.MEDIA_MOUNTED.equals(Environment.getExternalStorageState());
    }

    // ✅ Check if External Storage is Readable
    private boolean isExternalStorageReadable() {
        String state = Environment.getExternalStorageState();
        return Environment.MEDIA_MOUNTED.equals(state) ||
Environment.MEDIA_MOUNTED_READ_ONLY.equals(state);
    }
}
```