|  |
| --- |
| **BATCH AND ROLL NO:** |
| **EXPERIMENT NO.9** |
| **TITLE:** Design a mobile app to store data using internal or external storage. |
| **DATE OF PERFORMANCE:** |
| **DATE OF SUBMISSION:** |

**Title:** Design a mobile app to store data using internal or external storage.

**Requirements:**

1 Android studio

**Theory:**

**Introduction**

In the landscape of mobile application development, the efficient storage and retrieval of data play a pivotal role in creating robust and functional applications. This lab focuses on designing a mobile application capable of storing and retrieving data, offering users a seamless and personalized experience. The choice between internal and external storage methods allows developers to tailor solutions based on the nature and volume of data, showcasing the versatility of storage options available in mobile development.

**Objective of the Lab:** The primary objective of this lab is to guide you through the process of designing a mobile application that incorporates data storage functionalities. By the end of this lab, you should be proficient in implementing mechanisms for storing data persistently, whether it be on the device's internal storage or external storage (e.g., SD card). This involves managing data input, storage, and retrieval, contributing to a comprehensive understanding of data handling within mobile applications.

**Components of the Application:**

1. **Data Storage Mechanism:**
   * The application will utilize either internal or external storage to store and retrieve data persistently.
   * Internal storage is often used for storing private app-specific data, while external storage allows for the storage of data that can be accessed by other applications or the user.

**Lab Prerequisites:**

* Basic understanding of mobile application development concepts.
* Familiarity with the chosen development environment (e.g., Android Studio).

**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 User Interface (Optional)**

* Open the XML layout file associated with your main activity (e.g., activity\_main.xml).
* Design the layout with relevant UI elements, such as text fields for data input and buttons for data storage and retrieval.

### **Step 4: Implement Data Storage Logic**

#### Internal Storage:

* For internal storage, you can use the FileOutputStream and FileInputStream classes to write and read data, respectively.

#### External Storage:

* For external storage, ensure you have the necessary permissions in the AndroidManifest.xml, and use the Environment.getExternalStorageDirectory() to get the path.

### **Step 5: Test Your Application**

* Run your application on an emulator or a physical device.
* Verify that data is stored and retrieved correctly based on the chosen storage method.

**XML 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">

    <uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE"

        android:maxSdkVersion="32" />

    <uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE"

        android:maxSdkVersion="32"

        tools:ignore="ScopedStorage" />

    <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.Exp\_9"

        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>

</manifest>

**Activity\_main.xml**

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

<LinearLayout 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:orientation="vertical"

    android:padding="16dp">

    <!-- EditText with increased padding and a minimum height -->

    <EditText

        android:id="@+id/editText"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content"

        android:layout\_marginBottom="16dp"

        android:hint="Enter data"

        android:inputType="text"

        android:minHeight="48dp"

        android:padding="16dp"

       android:textSize="16sp"

        tools:ignore="HardcodedText"

        android:autofillHints="" /> <!-- Adjust text size if needed -->

    <!-- Save to Internal Storage Button -->

    <Button

        android:id="@+id/saveInternalButton"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content"

        android:text="Save to Internal Storage"

        android:minHeight="48dp"

        tools:ignore="HardcodedText" /> <!-- Ensure button touch target is large enough -->

    <!-- Save to External Storage Button -->

    <Button

        android:id="@+id/saveExternalButton"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content"

        android:text="Save to External Storage"

        android:minHeight="48dp"

        tools:ignore="HardcodedText" /> <!-- Ensure button touch target is large enough -->

    <!-- Load Data Button -->

    <Button

        android:id="@+id/loadButton"

        android:layout\_width="match\_parent"

        android:layout\_height="wrap\_content"

        android:text="Load Data"

        android:minHeight="48dp"

        tools:ignore="HardcodedText" /> <!-- Ensure button touch target is large enough -->

</LinearLayout>

**Java Code:**

**Activity\_Main**

package com.example.exp\_9;

import android.Manifest;

import android.content.pm.PackageManager;

import android.os.Bundle;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

import androidx.annotation.NonNull;

import androidx.appcompat.app.AppCompatActivity;

import androidx.core.app.ActivityCompat;

import androidx.core.content.ContextCompat;

import java.io.File;

import java.io.IOException;

public class MainActivity extends AppCompatActivity {

    private EditText editText;

    private static final int PERMISSION\_REQUEST\_CODE = 100;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

        editText = findViewById(R.id.editText);

        Button saveInternalButton = findViewById(R.id.saveInternalButton);

        Button saveExternalButton = findViewById(R.id.saveExternalButton);

        Button loadButton = findViewById(R.id.loadButton);

        saveInternalButton.setOnClickListener(v -> {

            String data = editText.getText().toString();

            saveDataInternal(data);

        });

        saveExternalButton.setOnClickListener(v -> {

            if (ContextCompat.checkSelfPermission(MainActivity.this, Manifest.permission.WRITE\_EXTERNAL\_STORAGE) != PackageManager.PERMISSION\_GRANTED) {

                ActivityCompat.requestPermissions(MainActivity.this, new String[]{Manifest.permission.WRITE\_EXTERNAL\_STORAGE}, PERMISSION\_REQUEST\_CODE);

            } else {

                String data = editText.getText().toString();

                saveDataExternal(data);

            }

        });

        loadButton.setOnClickListener(v -> loadData());

    }

    // Save data to internal storage

    private void saveDataInternal(String data) {

        try {

            File file = new File(getFilesDir(), "data.txt");

            FileStorageHelper.writeToFile(file, data);

            Toast.makeText(this, "Data saved to internal storage!", Toast.LENGTH\_SHORT).show();

        } catch (IOException e) {

            e.printStackTrace();

            Toast.makeText(this, "Error saving data.", Toast.LENGTH\_SHORT).show();

        }

    }

    // Save data to external storage

    private void saveDataExternal(String data) {

        if (android.os.Environment.getExternalStorageState().equals(android.os.Environment.MEDIA\_MOUNTED)) {

            File file = new File(getExternalFilesDir(null), "data.txt");

            try {

                FileStorageHelper.writeToFile(file, data);

                Toast.makeText(this, "Data saved to external storage!", Toast.LENGTH\_SHORT).show();

            } catch (IOException e) {

                e.printStackTrace();

                Toast.makeText(this, "Error saving data.", Toast.LENGTH\_SHORT).show();

            }

        } else {

            Toast.makeText(this, "External storage is not available.", Toast.LENGTH\_SHORT).show();

        }

    }

    // Load data from internal storage

    private void loadData() {

        File file = new File(getFilesDir(), "data.txt");

        String data = FileStorageHelper.readFromFile(file);

        editText.setText(data);

    }

    @Override

    public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions, @NonNull int[] grantResults) {

        super.onRequestPermissionsResult(requestCode, permissions, grantResults);

        if (requestCode == PERMISSION\_REQUEST\_CODE) {

            if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION\_GRANTED) {

                String data = editText.getText().toString();

                saveDataExternal(data);

            } else {

                Toast.makeText(this, "Permission denied to write to external storage.", Toast.LENGTH\_SHORT).show();

            }

        }

    }

}

**Internal\_Stroage**

package com.example.exp\_9;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

public class FileStorageHelper {

    // Write data to a file

    public static void writeToFile(File file, String data) throws IOException {

        try (FileOutputStream fos = new FileOutputStream(file)) {

            fos.write(data.getBytes());

        }

    }

    // Read data from a file

    public static String readFromFile(File file) {

        FileInputStream fis = null;

        StringBuilder stringBuilder = new StringBuilder();

        try {

            fis = new FileInputStream(file);

            int character;

            while ((character = fis.read()) != -1) {

                stringBuilder.append((char) character);

            }

        } catch (IOException e) {

            e.printStackTrace();

        } finally {

            try {

                if (fis != null) {

                    fis.close();

                }

            } catch (IOException e) {

                e.printStackTrace();

            }

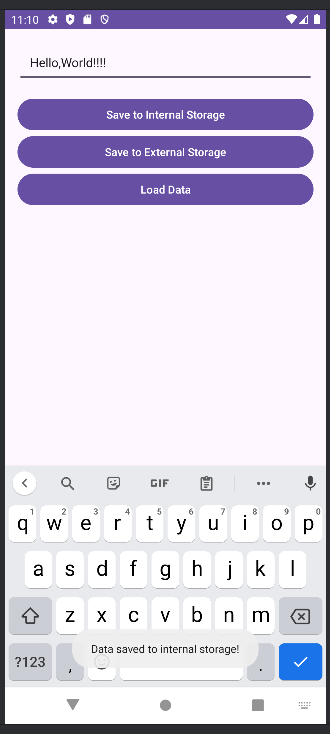
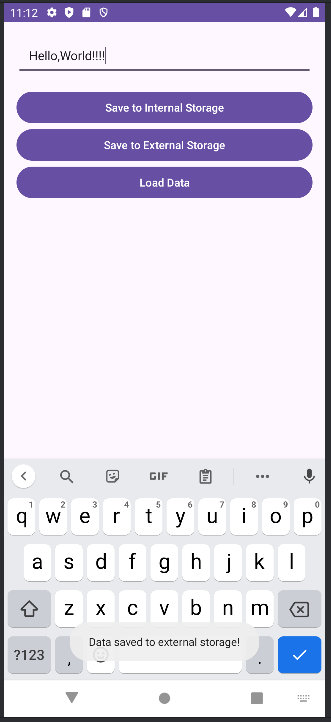
        }

        return stringBuilder.toString();

    }

}

**Output:**

#### Conclusion:

#### ………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………