PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE – 411043

Department of Electronics & Telecommunication Engineering

BATCH AND ROLL NO: Q5 42441
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:

activity_main.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:layout_marginVertical="150dp"
    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:background="@drawable/rounded_edittext"
        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:background="@drawable/rounded_edittext"
        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"
            style="@style/Widget.MaterialComponents.Button.OutlinedButton"/>

        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Load"
            android:onClick="loadData"
            style="@style/Widget.MaterialComponents.Button.OutlinedButton"/>
    </LinearLayout>
</LinearLayout>
```

rounded_edittex.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#FFFFFF"/>
    <corners android:radius="8dp"/>
    <stroke android:width="1dp" android:color="#CCCCCC"/>
</shape>
```

AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.expt09_42441">

    <!-- Permissions for External Storage -->
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
```

```
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.DataStorageApp">

        <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>
```

Java Code:

MainActivity.java:

```
package com.example.expt09_42441;

import android.content.Context;
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 java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStreamReader;
import java.nio.charset.StandardCharsets;

public class MainActivity extends AppCompatActivity {
    private EditText dataInput, fileNameInput;
    private RadioButton internalStorageRadio, externalStorageRadio;
```

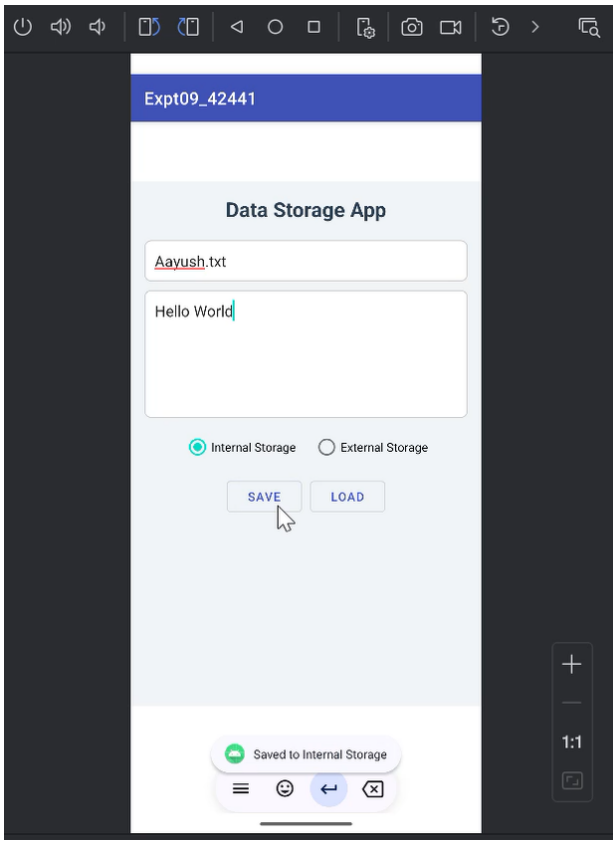
```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

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

public void saveData(View view) {
    String data = dataInput.getText().toString();
    String fileName = fileNameInput.getText().toString();

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

    try {
        if (internalStorageRadio.isChecked()) {
            saveInternalStorage(fileName, data);
        } else if (externalStorageRadio.isChecked()) {
            saveExternalStorage(fileName, data);
        }
    } catch (Exception e) {
        Toast.makeText(this, "Error saving data: " + e.getMessage(), Toast.LENGTH_SHORT).show();
    }
}

public void loadData(View view) {
    String fileName = fileNameInput.getText().toString();

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

    try {
        String data;
        if (internalStorageRadio.isChecked()) {
            data = readInternalStorage(fileName);
        } else if (externalStorageRadio.isChecked()) {
            data = readExternalStorage(fileName);
        } else {
            Toast.makeText(this, "Select storage type", Toast.LENGTH_SHORT).show();
            return;
        }
        dataInput.setText(data);
    } catch (Exception e) {
        Toast.makeText(this, "Error reading data: " + e.getMessage(), Toast.LENGTH_SHORT).show();
    }
}
```

```
    private void saveInternalStorage(String fileName, String data) {
        try (FileOutputStream fos = openFileOutput(fileName, Context.MODE_PRIVATE)) {
            fos.write(data.getBytes(StandardCharsets.UTF_8));
            Toast.makeText(this, "Saved to Internal Storage", Toast.LENGTH_SHORT).show();
        } catch (Exception e) {
            Toast.makeText(this, "Internal Storage Error: " + e.getMessage(),
Toast.LENGTH_SHORT).show();
        }
    }

    private void saveExternalStorage(String fileName, String data) {
        if (isExternalStorageWritable()) {
            File file = new File(getExternalFilesDir(null), fileName);
            try (FileOutputStream fos = new FileOutputStream(file)) {
                fos.write(data.getBytes(StandardCharsets.UTF_8));
                Toast.makeText(this, "Saved to External Storage", Toast.LENGTH_SHORT).show();
            } catch (Exception e) {
                Toast.makeText(this, "External Storage Error: " + e.getMessage(),
Toast.LENGTH_SHORT).show();
            }
        } else {
            Toast.makeText(this, "External Storage Not Writable", Toast.LENGTH_SHORT).show();
        }
    }

    private String readInternalStorage(String fileName) throws Exception {
        try (FileInputStream fis = openFileInput(fileName);
             InputStreamReader isr = new InputStreamReader(fis, StandardCharsets.UTF_8);
             BufferedReader bufferedReader = new BufferedReader(isr)) {
            StringBuilder sb = new StringBuilder();
            String line;
            while ((line = bufferedReader.readLine()) != null) {
                sb.append(line).append("\n");
            }
            Toast.makeText(this, "Loaded from Internal Storage", Toast.LENGTH_SHORT).show();
            return sb.toString().trim();
        }
    }

    private String readExternalStorage(String fileName) throws Exception {
        if (isExternalStorageReadable()) {
            File file = new File(getExternalFilesDir(null), fileName);
            try (FileInputStream fis = new FileInputStream(file);
                 InputStreamReader isr = new InputStreamReader(fis, StandardCharsets.UTF_8);
                 BufferedReader bufferedReader = new BufferedReader(isr)) {

                StringBuilder sb = new StringBuilder();
                String line;
                while ((line = bufferedReader.readLine()) != null) {
                    sb.append(line).append("\n");
                }
                Toast.makeText(this, "Loaded from External Storage", Toast.LENGTH_SHORT).show();
                return sb.toString().trim();
            }
        } else {
            throw new Exception("External Storage Not Readable");
        }
    }
```

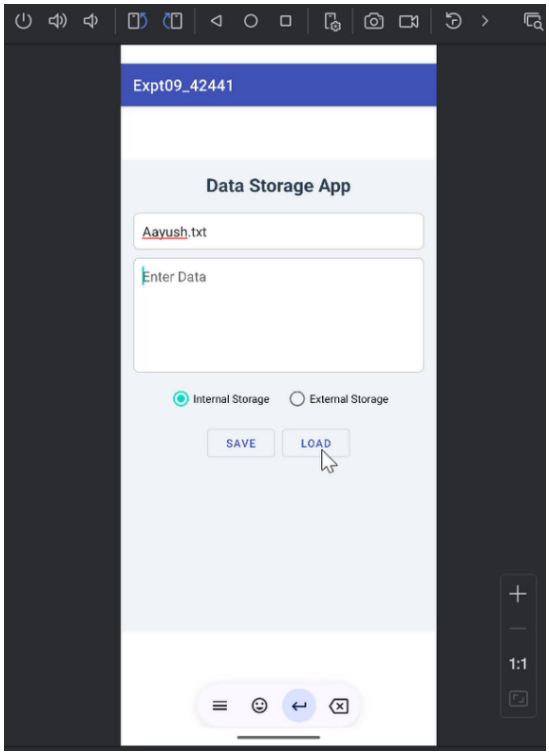
```
    private boolean isExternalStorageWritable() {
        String state = Environment.getExternalStorageState();
        return Environment.MEDIA_MOUNTED.equals(state);
    }

    private boolean isExternalStorageReadable() {
        String state = Environment.getExternalStorageState();
        return Environment.MEDIA_MOUNTED.equals(state) ||
                Environment.MEDIA_MOUNTED_READ_ONLY.equals(state);
    }
}
```

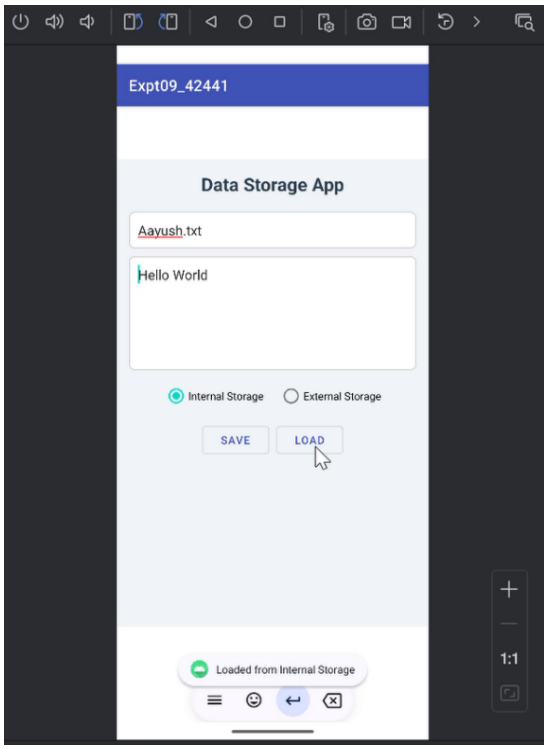
Output:

1) Saving to internal storage:

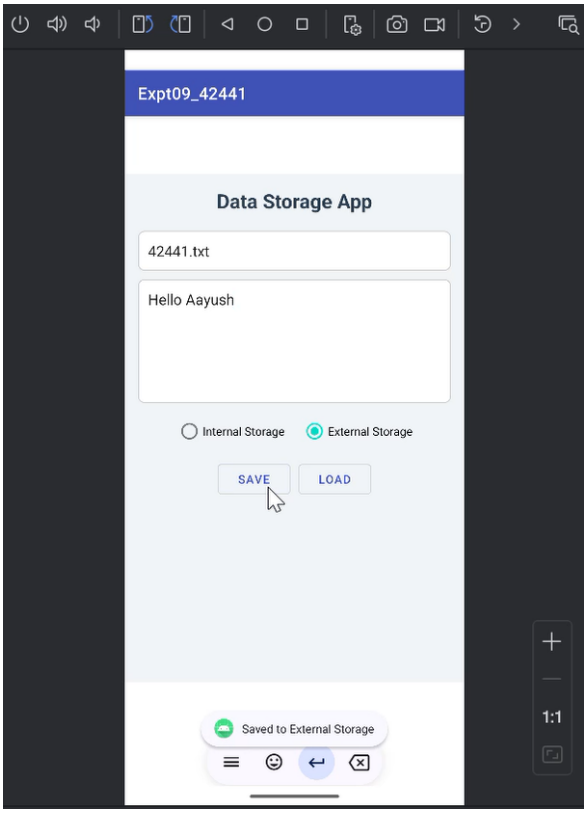


2) Loading from internal storage:

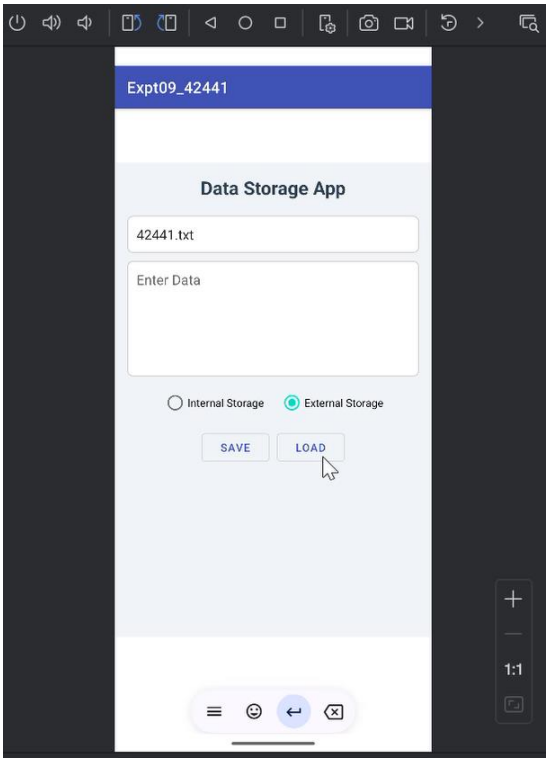


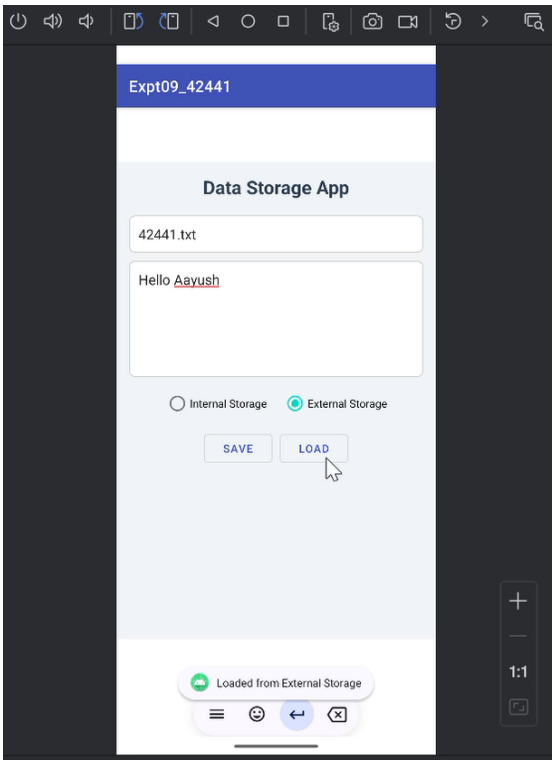


3) Saving to external storage:



4) Loading from external storage:





Conclusion:

……

……

……