PUNE INSTITUTE OF COMPUTER TECHNOLOGY, PUNE – 411043

Department of Electronics & Telecommunication Engineering

BATCH AND ROLL NO: P5-42114
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"?>
<androidx.constraintlayout.widget.ConstraintLayout
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"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/titleId"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="164dp"
        android:fontFamily="sans-serif-condensed-medium"
        android:text="Internal Storage"
        android:textSize="30sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/inputText"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="95dp"
        android:hint="TYPE YOUR TEXT HERE TO STORE"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/titleId" />

    <Button
        android:id="@+id/saveButtonId"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="32dp"
        android:onClick="saveData"
        android:text="SAVE"
        app:layout_constraintStart_toStartOf="@+id/loadButtonId"
        app:layout_constraintTop_toBottomOf="@+id/loadButtonId" />

    <Button
        android:id="@+id/loadButtonId"
        android:layout_width="wrap_content"
```

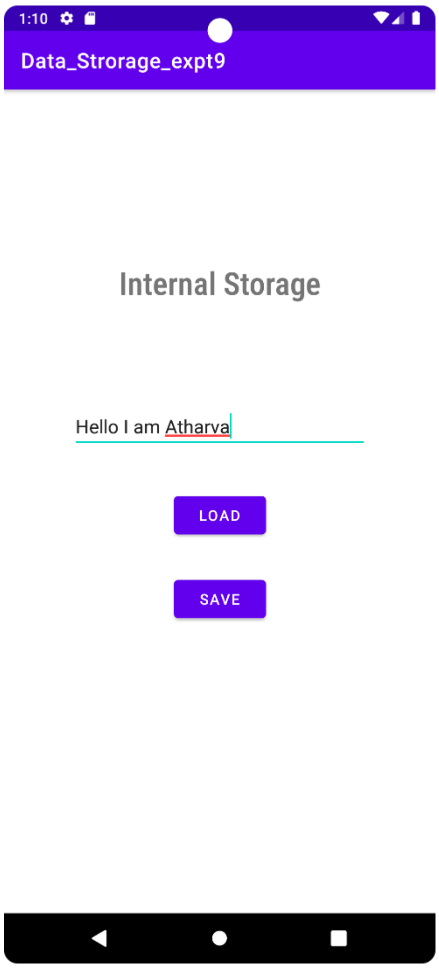
```
android:layout_height="wrap_content"
        android:layout_marginTop="38dp"
        android:onClick="loadData"
        android:text="load"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/inputText" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

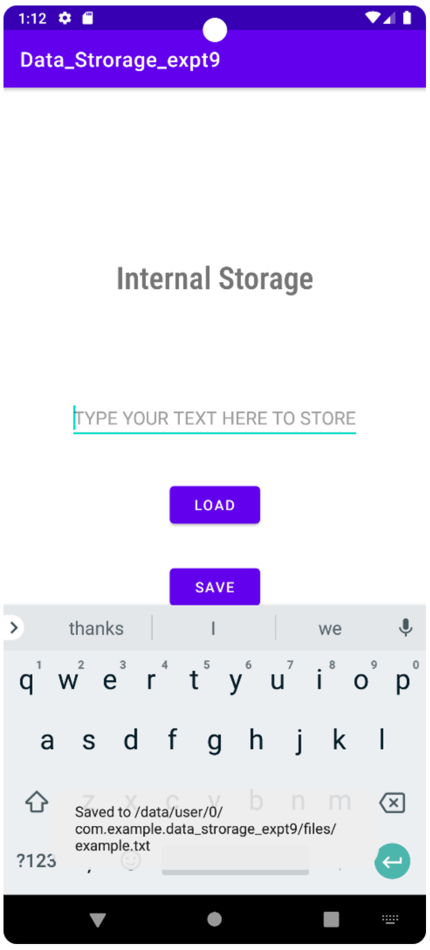
Java Code:

```
package com.example.data_strorage_expt9;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;
import java.io.BufferedReader;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;

public class MainActivity extends AppCompatActivity {

    private static final String FILE_NAME = "example.txt";
    EditText inputText;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        inputText = findViewById(R.id.inputText);
    }

    public void saveData(View view) throws Exception    {
        String text=inputText.getText().toString();
        FileOutputStream fileOutputStream=null;
        try{
            fileOutputStream=openFileOutput(FILE_NAME,MODE_PRIVATE);
            fileOutputStream.write(text.getBytes());
            inputText.getText().clear();
            Toast.makeText(this,"Saved to
"+getFilesDir()+"/"+FILE_NAME,Toast.LENGTH_LONG).show();
        }
        catch(Exception e){
            throw new RuntimeException(e);
        }
```

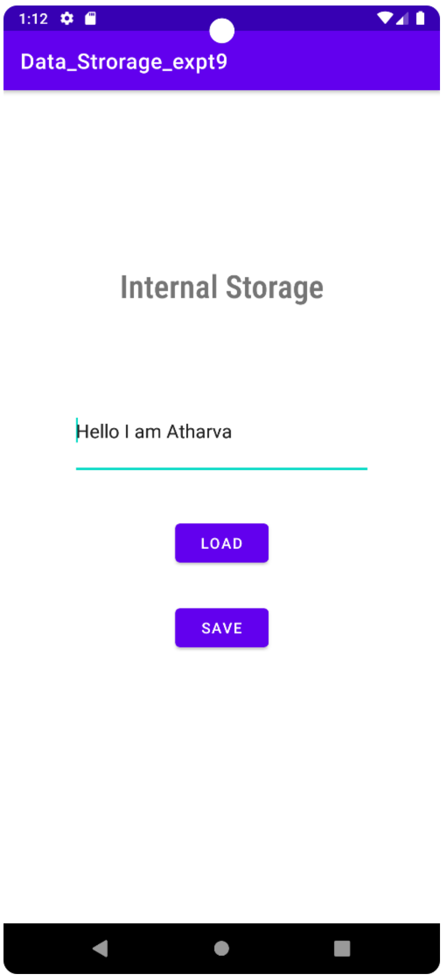
```
            finally{
        if(fileOutputStream!=null){
            try{
                fileOutputStream.close();
            }
            catch(IOException e){
                throw new RuntimeException(e);
            }
        }
    }
}
public void loadData(View view){
    FileInputStream fileInputStream=null;

    try{
        fileInputStream=openFileInput(FILE_NAME);
        InputStreamReader inputStreamReader=new InputStreamReader(fileInputStream);
        BufferedReader bufferedReader=new BufferedReader(inputStreamReader);
        StringBuilder stringBuilder=new StringBuilder();
        String text;
        while((text=bufferedReader.readLine())!=null){
            stringBuilder.append(text).append("\n");
        }
        inputText.setText(stringBuilder.toString());
    }
    catch(Exception e){
        throw new RuntimeException(e);
    }
    finally{
        if(fileInputStream!=null){
            try{
                fileInputStream.close();
            }
            catch(Exception e){
                throw new RuntimeException(e);
            }
        }
    }
}
}
```

Output:





Saved to /data/user/0/
com.example.data_strorage_expt9/files/
example.txt



Conclusion:

....

....

....