SIDDHI S. KOTRE
ROLL NO: 50

Assignment - 3 Database Connectivity and REST API Integration

1. **Aim:** Write a program to read and write content in internal storage.

Objective :
The objective of this program is to demonstrate how to perform file operations in internal storage on an Android device. The program will:

1. Allow the user to write text content to a file stored in internal storage.
2. Read and display the contents of the file from internal storage.
3. Highlight safe file handling and basic file I/O operations within Android.

Theory:
In Android, internal storage is a secure location in the device's file system where an app can store data. Files saved here are private to the app, meaning no other apps can access or modify them. Internal storage is typically used to save files like user preferences, sensitive information, or data that should remain confined to the app itself.
When a file is created in internal storage, it's placed in a directory associated with the app. Android's internal storage persists until the app is uninstalled, at which point any stored files are also deleted. This provides a secure and persistent storage solution without requiring external permissions from the user.

Code:
MainActivity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
   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"
   android:orientation="vertical"
   android:layout_gravity="center"
   android:gravity="center"
   tools:context=".MainActivity_internalstrg">

   <EditText
      android:id="@+id/text1"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:gravity="center"
      android:textSize="20dp"
      android:hint="Enter Data" />
```

```
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/btn1"
        android:layout_gravity="center"
        android:layout_marginTop="50dp"
        android:textSize="20dp"
        android:background="#6691A5"
        android:text="Save"
        android:onClick="safe"/>

    <Button
        android:id="@+id/btn2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="50dp"
        android:background="#6691A5"
        android:text="Load"
        android:onClick="loadMethod"
        android:textSize="20dp" />
</LinearLayout>
```

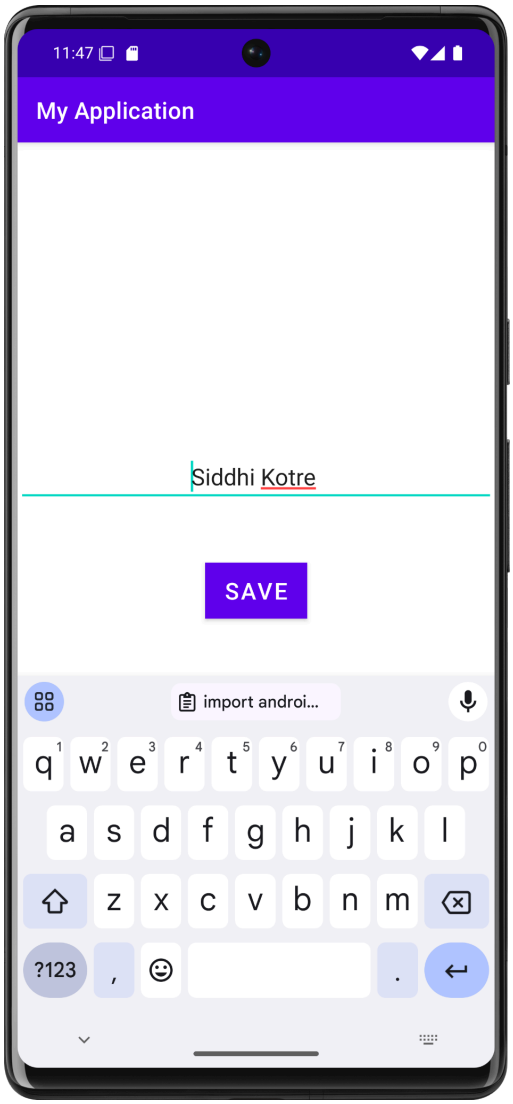
MainActivity.java

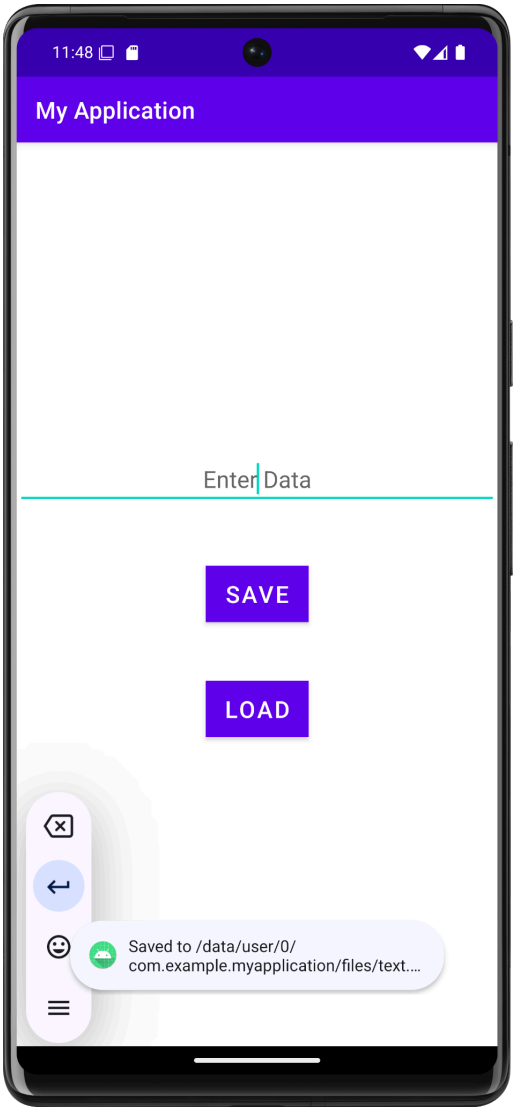
```
package com.example.assignment_3;

import androidx.appcompat.app.AppCompatActivity;
import android.annotation.SuppressLint;
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_internalstrg extends AppCompatActivity {
    EditText mEditText;
    private static final String FILE_NAME = "text.txt";

    @SuppressLint("MissingInflatedId")
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main_internalstrg);
        mEditText = findViewById(R.id.text1);
    }

    public void safe(View view) throws IOException {
        String text = mEditText.getText().toString();
        FileOutputStream fos = null;
        try {
            fos = openFileOutput(FILE_NAME, MODE_PRIVATE);
            fos.write(text.getBytes());
            mEditText.getText().clear();
            Toast.makeText(this, "Saved to " + getFilesDir() + "/" + FILE_NAME,
Toast.LENGTH_LONG).show();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            if (fos != null) {
                fos.close();
            }
        }
    }

    public void loadMethod(View view) {
        FileInputStream fis = null;
        try {
            fis = openFileInput(FILE_NAME);
            InputStreamReader isr = new InputStreamReader(fis);
            BufferedReader br = new BufferedReader(isr);
            StringBuilder sb = new StringBuilder();
            String text;
            while ((text = br.readLine()) != null) {
                sb.append(text).append("\n");
            }
            mEditText.setText(sb.toString());
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
```

```
            if (fis != null) {
                try {
                    fis.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }
    }
}
```

Output:





2. **Aim:** Write a program to read and write content in external storage.

Objective :

The objective of this program is to demonstrate how to perform file operations on an Android device's external storage. The program will:

1. Allow the user to write text content to a file stored in external storage.
2. Read and display the contents of the file from external storage.
3. Understand permissions and best practices for handling external storage in Android.

Theory:

In Android, external storage refers to storage outside the app's internal data space, typically the device's shared storage (e.g., SD card or a dedicated storage partition). External storage is ideal for storing files that can be accessed by other apps or that the user may need to move across devices, like photos, documents, or large datasets. However, unlike internal storage, external storage is more vulnerable to deletion, corruption, or unauthorized access.

Code:

AndroidManifest.xml

```
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

MainActivity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
   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"
   android:orientation="vertical"
   android:layout_gravity="center"
   android:gravity="center"
   tools:context=".MainActivity_externalstrg">

   <EditText
      android:id="@+id/text1"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:gravity="center"
      android:textSize="20dp"
      android:hint="Enter Data" />

   <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
```

```
        android:id="@+id/btn1"
        android:layout_gravity="center"
        android:layout_marginTop="50dp"
        android:textSize="20dp"
        android:background="#6691A5"
        android:text="Save"
        android:onClick="safe"/>

    <Button
        android:id="@+id/btn2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="50dp"
        android:background="#6691A5"
        android:text="Load"
        android:onClick="load"
        android:textSize="20dp" />

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/textview"
        android:textSize="20dp"
        />
</LinearLayout>
```

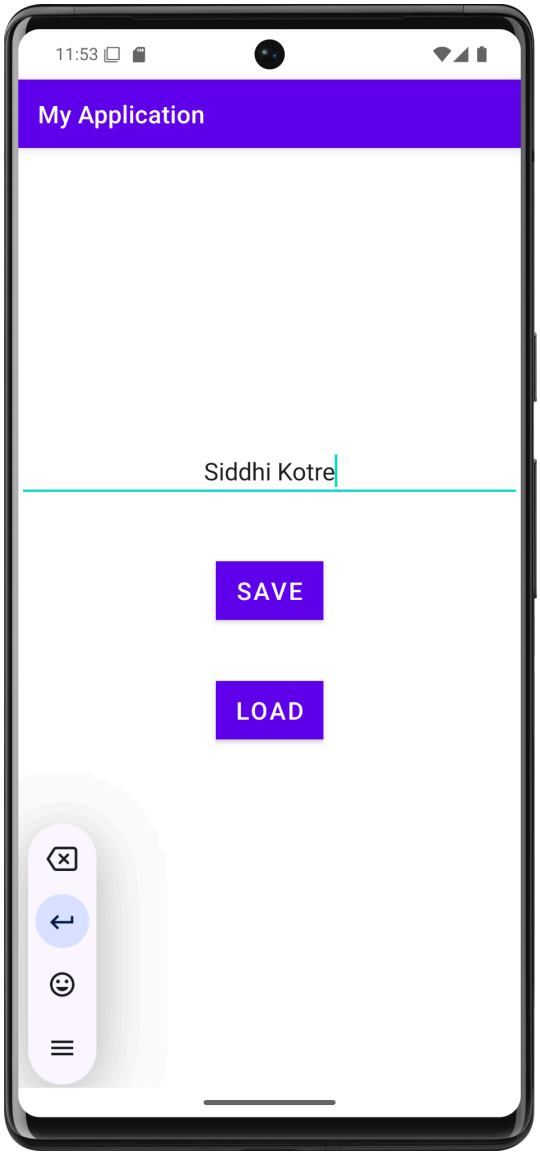
MainActivity.java

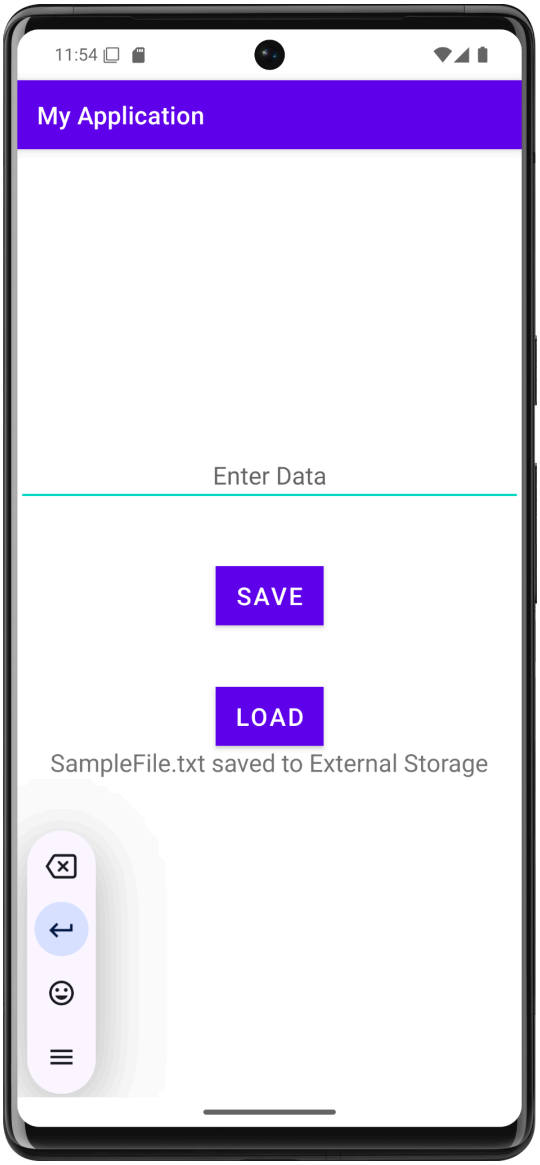
```
package com.example.assignment_3;

import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.os.Environment;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
```

```
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;

public class MainActivity_externalstrg extends AppCompatActivity {

    EditText mEditText;
    TextView t1;
    Button saveButton, readButton;
    private String filename = "Samplefile.txt";
    private String filepath = "MyFileStorage";
    File myExternalFile;
    String myData;

    @SuppressLint("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_main_externalstrg);

        mEditText = findViewById(R.id.text1);
        t1 = findViewById(R.id.textview);
        saveButton = findViewById(R.id.btn1);
        readButton = findViewById(R.id.btn2);

        if (!isExternalStorageAvailable() || isExternalStorageReadOnly()) {
            saveButton.setEnabled(false);
        } else {
            myExternalFile = new File(getExternalFilesDir(filepath), filename);
        }
    }

    private boolean isExternalStorageReadOnly() {
        String extStorageState = Environment.getExternalStorageState();
        return Environment.MEDIA_MOUNTED_READ_ONLY.equals(extStorageState);
    }

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

```
    public void safe(View view) {
        try {
            FileOutputStream fos = new FileOutputStream(myExternalFile);
            fos.write(mEditText.getText().toString().getBytes());
            fos.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
        mEditText.setText("");
        t1.setText("SampleFile.txt saved to External Storage");
    }

    public void load(View view) {
        try {
            FileInputStream fis = new FileInputStream(myExternalFile);
            InputStreamReader i = new InputStreamReader(fis);
            BufferedReader br = new BufferedReader(i);
            StringBuilder sb = new StringBuilder();
            String strLine;
            while ((strLine = br.readLine()) != null) {
                sb.append(strLine).append("\n");
            }
            myData = sb.toString();
            i.close();
        } catch (FileNotFoundException e) {
            throw new RuntimeException(e);
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
        mEditText.setText(myData);
        t1.setText("SampleFile.txt data retrieved from external Storage");
    }
}
```

Output:


11:53
My Application
Siddhi Kotre
SAVE
LOAD


11:54
My Application
Enter Data
SAVE
LOAD
SampleFile.txt saved to External Storage

3. **Aim:** Write a program to demonstrate shared preference.

Objective :

The objective of this program is to showcase how to use Shared Preferences in Android to store, retrieve, and modify small amounts of data persistently. Shared Preferences are ideal for storing simple key-value pairs, such as user settings or basic configurations.

Theory:

Shared Preferences in Android provide a way to store and retrieve small amounts of primitive data, such as String, int, boolean, float, and long, in the form of key-value pairs. It's a lightweight storage option ideal for saving user preferences, settings, or simple app configurations that need to persist across app sessions.

Code:

Activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
  android:orientation="vertical"
  tools:context=".MainActivity">

  <EditText
    android:id="@+id/edit_Text"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="16dp"
    android:hint="Enter username here"
    android:padding="16dp"
    android:minHeight="48dp" />

  <Button
    android:id="@+id/save"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="16dp"
    android:padding="16dp"
    android:text="Save"
    android:minHeight="48dp" />

  <Button
    android:id="@+id/load"
    android:layout_width="match_parent"
```

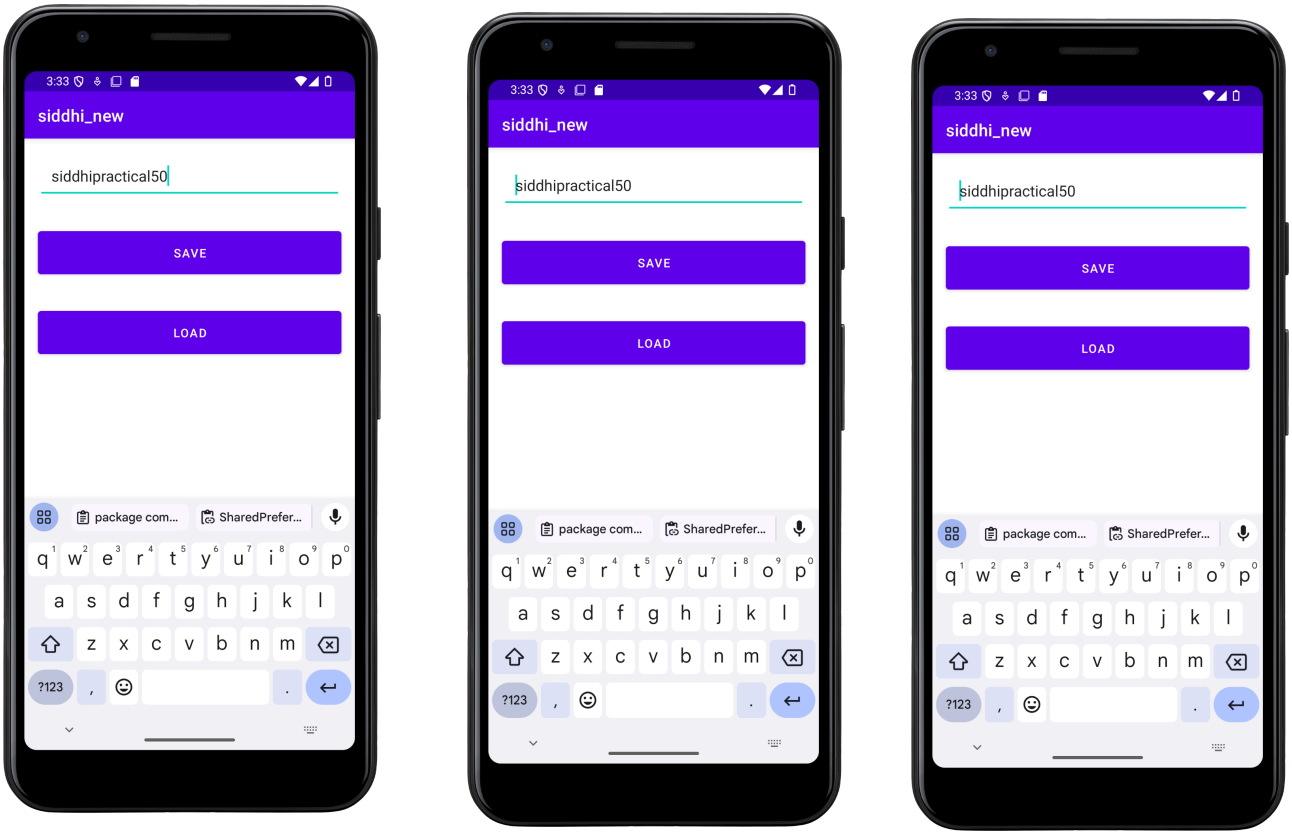
```
    android:layout_height="wrap_content"
    android:layout_margin="16dp"
    android:padding="16dp"
    android:text="Load"
    android:minHeight="48dp" />

</LinearLayout>
```

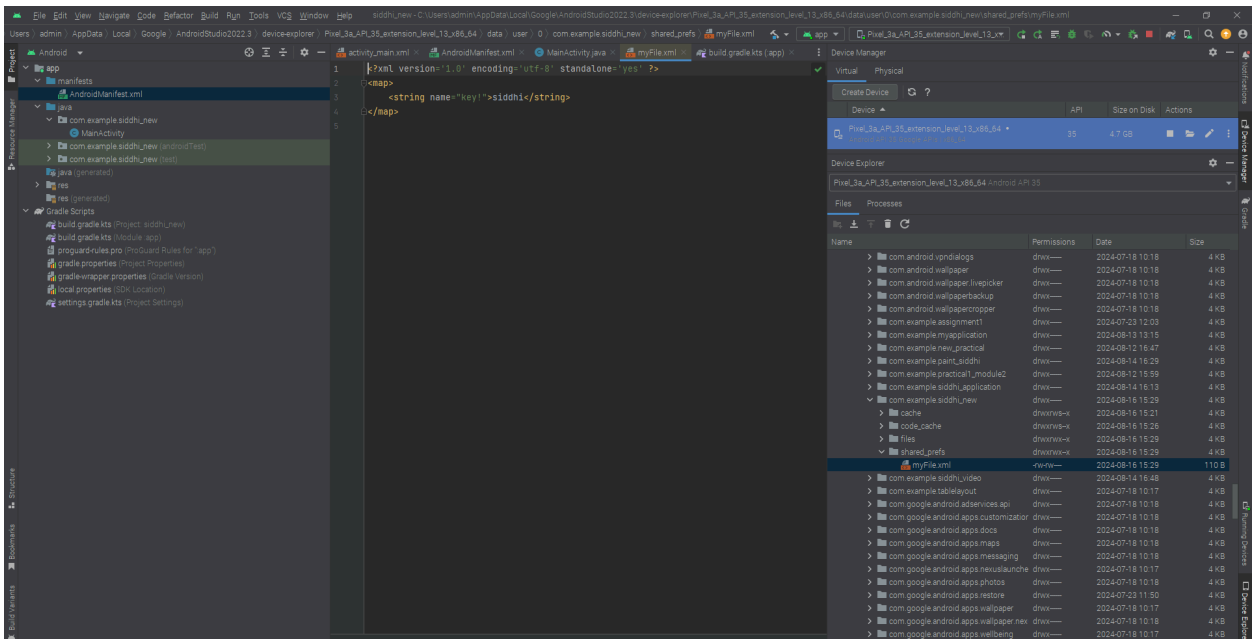
MainActivity.java

```
package com.example.siddhi_new;

import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

public class MainActivity extends AppCompatActivity {

  private EditText tv;
  private Button saveButton, loadButton;

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

     tv = findViewById(R.id.edit_Text);
     saveButton = findViewById(R.id.save);
     loadButton = findViewById(R.id.load);


     saveButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
           save();
        }
     });
```

```
        loadButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                load();
            }
        });
    }

    public void save() {
        String msg = tv.getText().toString();
        SharedPreferences s = getSharedPreferences("myFile", MODE_PRIVATE);
        SharedPreferences.Editor e = s.edit();
        e.putString("key!", msg);
        e.commit();
        tv.setText("");
    }

    public void load() {
        SharedPreferences sp = getSharedPreferences("myFile", MODE_PRIVATE);
        String message1 = sp.getString("key!", "key doesn't match");
        tv.setText(message1);
    }
}
```

Output:


siddhi_new
siddhipractical50
SAVE
LOAD


<?xml version="1.0" encoding="utf-8" standalone="yes" ?>
<map>
<string name="key1">siddhi</string>
</map>

4. **Aim:** Write a program to create a login screen and save user credentials in shared preference and display user name in second activity.

Objective :

The objective of this program is to demonstrate how to use Shared Preferences to save user login credentials and retrieve them in another activity. This program will:

1. Allow users to log in by entering their username and password.
2. Save the credentials in Shared Preferences after a successful login.
3. Display the username on a second screen, simulating a welcome message.

Theory:

In Android, Shared Preferences is a simple storage option used to store small pieces of persistent data as key-value pairs. It is a good option for saving settings, configurations, or user preferences.

When creating a login screen, you can use Shared Preferences to save user credentials, such as the username, and retrieve it later in another activity. This approach is useful for scenarios where basic information (like usernames) needs to be stored securely and retrieved across different sessions or screens.

Code:

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
  android:orientation="vertical"
  tools:context=".MainActivity">

  <EditText
    android:id="@+id/username"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="16dp"
    android:hint="Enter username here"
    android:padding="16dp"
    android:minHeight="48dp" />
  <EditText
    android:id="@+id/password"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="16dp"
```

```
    android:hint="Enter password"
    android:padding="16dp"
    android:minHeight="48dp" />
  <Button
    android:id="@+id/login"
    android:onClick="login"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="16dp"
    android:padding="16dp"
    android:text="Login"
    android:minHeight="48dp" />

</LinearLayout>
```

MainActivity.java

```
package com.example.login_app;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
TextView username,password;
SharedPreferences s;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    username = findViewById(R.id.username);
    password = findViewById(R.id.password);
     s = getSharedPreferences("login_user", MODE_PRIVATE);

  }
  public void login(View view) {
    String user = username.getText().toString();
    String pass = password.getText().toString();
    SharedPreferences.Editor e = s.edit();
```

```
        if (user.equals("Siddhi") && pass.equals("siddhi")) {
            e.putString("username", user);
            e.putString("password", pass);
            e.commit();

            Intent i = new Intent(this, Dashboard.class);
            startActivity(i);
        } else {
            Toast.makeText(this, "Invalid credentials", Toast.LENGTH_LONG).show();


        }
    }
}
```

Activity_dashboard.xml

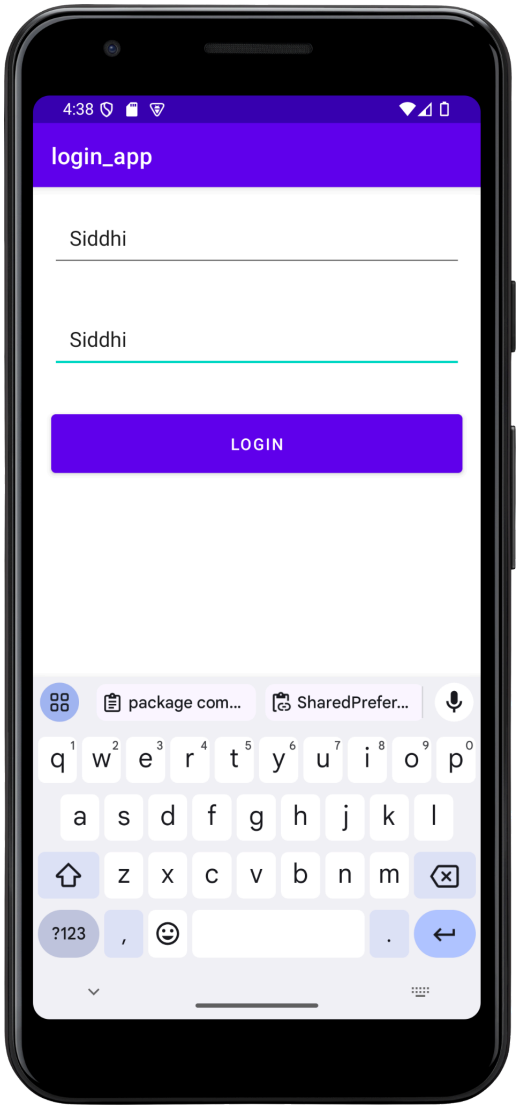
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    tools:context=".MainActivity">


    <TextView
        android:id="@+id/welcome_text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="TextView" />

    <Button
        android:id="@+id/logout"
        android:onClick="logout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="16dp"
        android:padding="16dp"
        android:text="Logout"
        android:minHeight="48dp" />
```

```
</LinearLayout>
```

Dashboard.java

```
package com.example.login_app;

import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;
import android.widget.TextView;

public class Dashboard extends AppCompatActivity {
TextView welcome_msg;
  SharedPreferences s;
  @SuppressLint("MissingInflatedId")
  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_dashboard);
     welcome_msg = findViewById(R.id.welcome_text);

     s = getSharedPreferences("login_user",MODE_PRIVATE);
     String username = s.getString("username", null);
     welcome_msg.setText("Welcome "+username);

  }

  public void logout(View view){
     SharedPreferences.Editor e = s.edit();
     e.clear();
     Intent i = new Intent(this,MainActivity.class);
     startActivity(i);
  }
}
```

Output:


4:38
login_app
Siddhi
Siddhi
LOGIN
package com...
SharedPrefer...


4:39
login_app
Welcome Siddhi
LOGOUT

5. **Aim:** Write a program to turn on and off bluetooth service.

Objective :

The objective of this program is to demonstrate how to programmatically control the Bluetooth functionality on an Android device. The program will allow the user to toggle Bluetooth on or off with a button click.

Theory:

Bluetooth in Android enables wireless communication between devices, making it useful for tasks like data exchange, audio streaming, and device connectivity. The Android BluetoothAdapter class provides the means to interact with Bluetooth features, including turning Bluetooth on and off, discovering devices, and managing connections.

Code:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
  android:orientation="vertical"
  android:gravity="center"
  tools:context=".MainActivity">

  <Button
     android:onClick="Enable"
     android:id="@+id/enablebt"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:text="Enable Button" />

  <Button
     android:onClick="Disable"
     android:id="@+id/disablebt"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:text="Disable Button" />
</LinearLayout>
```

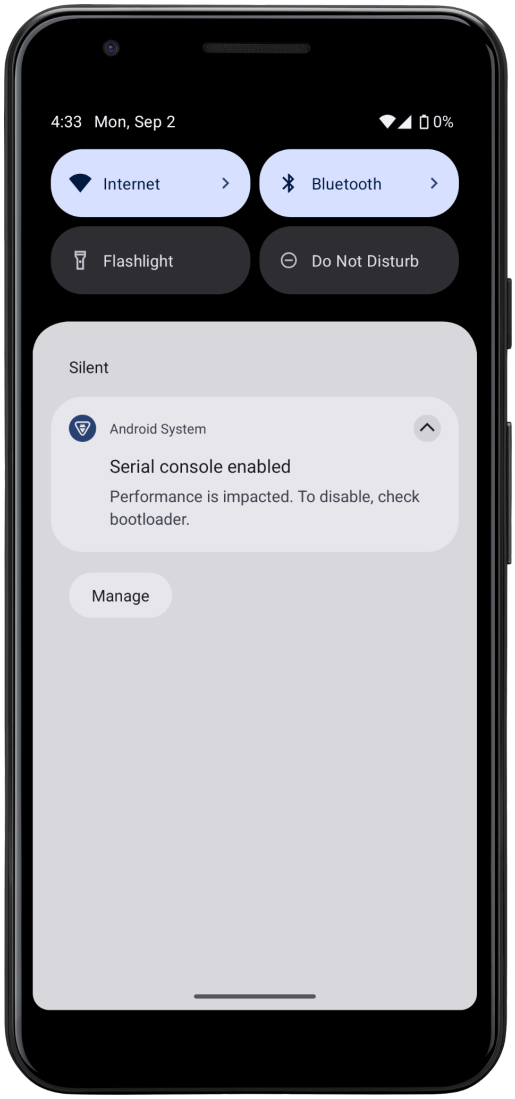
MAINACTIVITY.java

```
package com.example.newprac;

import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
```

```
import android.bluetooth.BluetoothAdapter;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
  BluetoothAdapter bAdapter = null;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);
     bAdapter = BluetoothAdapter.getDefaultAdapter();
  }

  public void Enable(View view) {
     if(bAdapter == null) {
        Toast.makeText(this, "Bluetooth is not supported", Toast.LENGTH_SHORT).show();
     } else {
        if (!bAdapter.isEnabled()) {
           if (ActivityCompat.checkSelfPermission(this,
android.Manifest.permission.BLUETOOTH_CONNECT) !=
PackageManager.PERMISSION_GRANTED) {

              return;
           }
           Log.d("ON","ON");
           Intent intentBtEnabled = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
           int REQUEST_ENABLE_BT = 2;
           startActivityForResult(intentBtEnabled,REQUEST_ENABLE_BT);

        }
     }
  }

  public void Disable(View view) {

           if (ActivityCompat.checkSelfPermission(this,
android.Manifest.permission.BLUETOOTH_CONNECT) !=
PackageManager.PERMISSION_GRANTED) {
```

```
            bAdapter.disable();
             Toast.makeText(this, "Bluetooth is off", Toast.LENGTH_SHORT).show();
             return;
          }
          Intent intentBtEnabled = new Intent("android.bluetooth.action.REQUEST_DISABLE");
          startActivityForResult(intentBtEnabled,2);

    }
}
```

Output:



6. **Aim:** Write a program to turn on and off wifi service.

Objective :

The objective of this program is to demonstrate how to programmatically control Wi-Fi functionality on an Android device. This program will allow the user to toggle Wi-Fi on or off with a button click.

Theory:

Wi-Fi in Android is a network technology that allows devices to communicate without internet cables. Managing Wi-Fi on Android devices is handled using the WifiManager class, which provides methods to enable, disable, and check the Wi-Fi status.

Code:

activity_main.xml

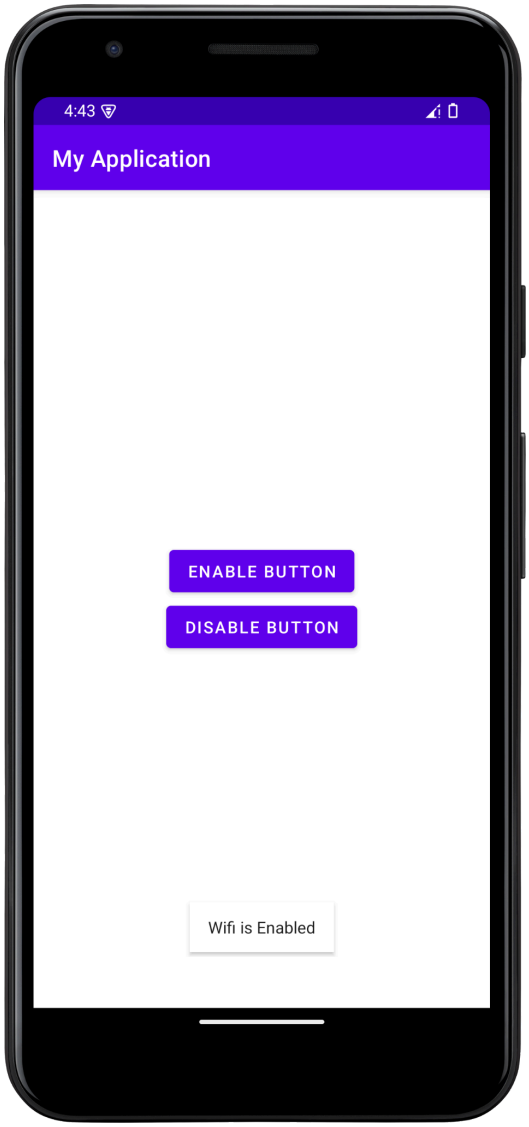
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"
  android:gravity="center"
  tools:context=".MainActivity">

  <Button
    android:onClick="Enable"
    android:id="@+id/enablewifi"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Enable Button" />

  <Button
    android:onClick="Disable"
    android:id="@+id/disablewifi"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Disable Button" />
</LinearLayout>
```

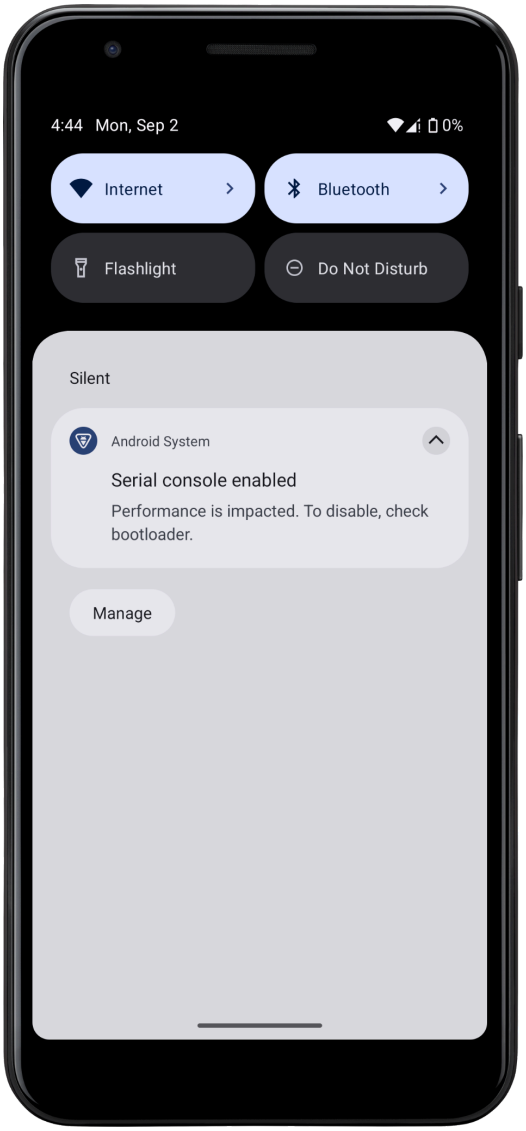
MainActivity.java

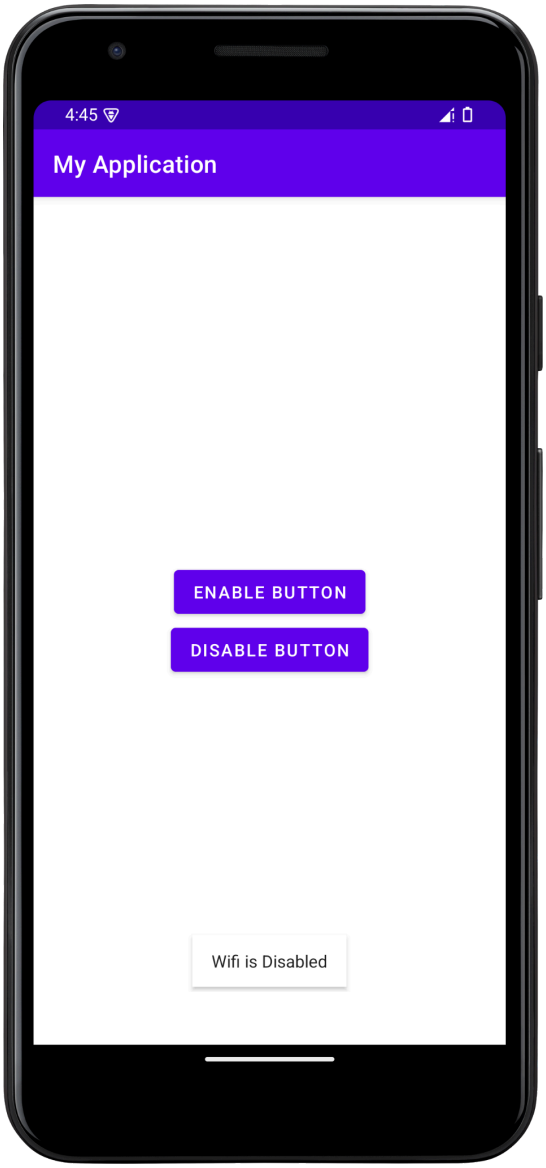
```
package com.example.myapplication;

import androidx.appcompat.app.AppCompatActivity;
```

```
import android.net.wifi.WifiManager;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
WifiManager wifi;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);
     wifi = (WifiManager) getApplicationContext().getSystemService(WIFI_SERVICE);

  }

  public void Enable(View view){
     wifi.setWifiEnabled(true);
     Toast.makeText(this, "Wifi is Enabled", Toast.LENGTH_SHORT).show();
  }

  public void Disable(View view){
     wifi.setWifiEnabled(false);
     Toast.makeText(this, "Wifi is Disabled", Toast.LENGTH_SHORT).show();
  }
}
```

Output:


4:43
My Application
ENABLE BUTTON
DISABLE BUTTON
Wifi is Enabled


4:44 Mon, Sep 2
0%
Internet
Bluetooth
Flashlight
Do Not Disturb
Silent
Android System
Serial console enabled
Performance is impacted. To disable, check bootloader.
Manage


4:45
My Application
ENABLE BUTTON
DISABLE BUTTON
Wifi is Disabled

7. **Aim:** Write a program to create user registration form after registration data will be inserted in sqlite database and also design activity which displays that information.

Objective :
Write a program to create a user registration form in an Android application. After registration, the user's data will be inserted into an SQLite database, and another activity will display the stored information.

Theory:
SQLite in Android: SQLite is a lightweight, relational database engine that is embedded within Android devices. It allows for local data storage and retrieval without the need for an external server. The database is stored in the device's internal storage, making it accessible only to the app that created it. In this example, SQLite is used to store user registration information.

Code:

```
<?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">

  <EditText
    android:id="@+id/editTextName"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Name:"
    android:padding="12dp" />

  <EditText
    android:id="@+id/editTextSurname"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Surname:"
    android:padding="12dp" />

  <EditText
    android:id="@+id/editTextMarks"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Marks:"
```

```
        android:inputType="number"
        android:padding="12dp" />

    <Button
        android:id="@+id/buttonAddData"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Add Data"
        android:padding="12dp"
        android:layout_marginTop="8dp" />

    <Button
        android:id="@+id/buttonViewAll"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="View All"
        android:padding="12dp"
        android:layout_marginTop="8dp" />

    <TextView
        android:id="@+id/textViewData"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:text="Data:"
        android:textSize="16sp" />

</LinearLayout>
```

Helper.java

```
package com.example.siddhi_prac;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

import androidx.annotation.Nullable;

public class Helper extends SQLiteOpenHelper {
```

```
  public static  final String DATABASE_NAME = "Mystudent_db";
  public static final String TABLE_NAME = "student_table";

  public static final String col_1 = "ID";
  public static final String col_2 = "NAME";
  public static final String col_3 = "SURNAME";
  public static final String col_4 = "MARKS";



  public Helper(@Nullable Context context) {
     super(context, DATABASE_NAME, null, 1);
  }

  @Override
  public void onCreate(SQLiteDatabase db) {
     db.execSQL("create table " + " " + TABLE_NAME+"(ID INTEGER PRIMARY KEY
AUTOINCREMENT, NAME TEXT, SURNAME TEXT,MARKS INTEGER)");
  }

  @Override
  public void onUpgrade(SQLiteDatabase db, int i, int i1) {

  }
  public boolean insert_data(String Name, String Surname, int Marks){

     SQLiteDatabase db= this.getWritableDatabase();

     ContentValues contentValues = new ContentValues();
     contentValues.put(col_2,Name);
     contentValues.put(col_3,Surname);
     contentValues.put(col_4,Marks);

    long result=  db.insert(TABLE_NAME,null, contentValues);
        if(result == -1){
           return false;
        }
        else{
           return true;
        }
  }
  public Cursor getAllData(){
     SQLiteDatabase db = this.getReadableDatabase();
```

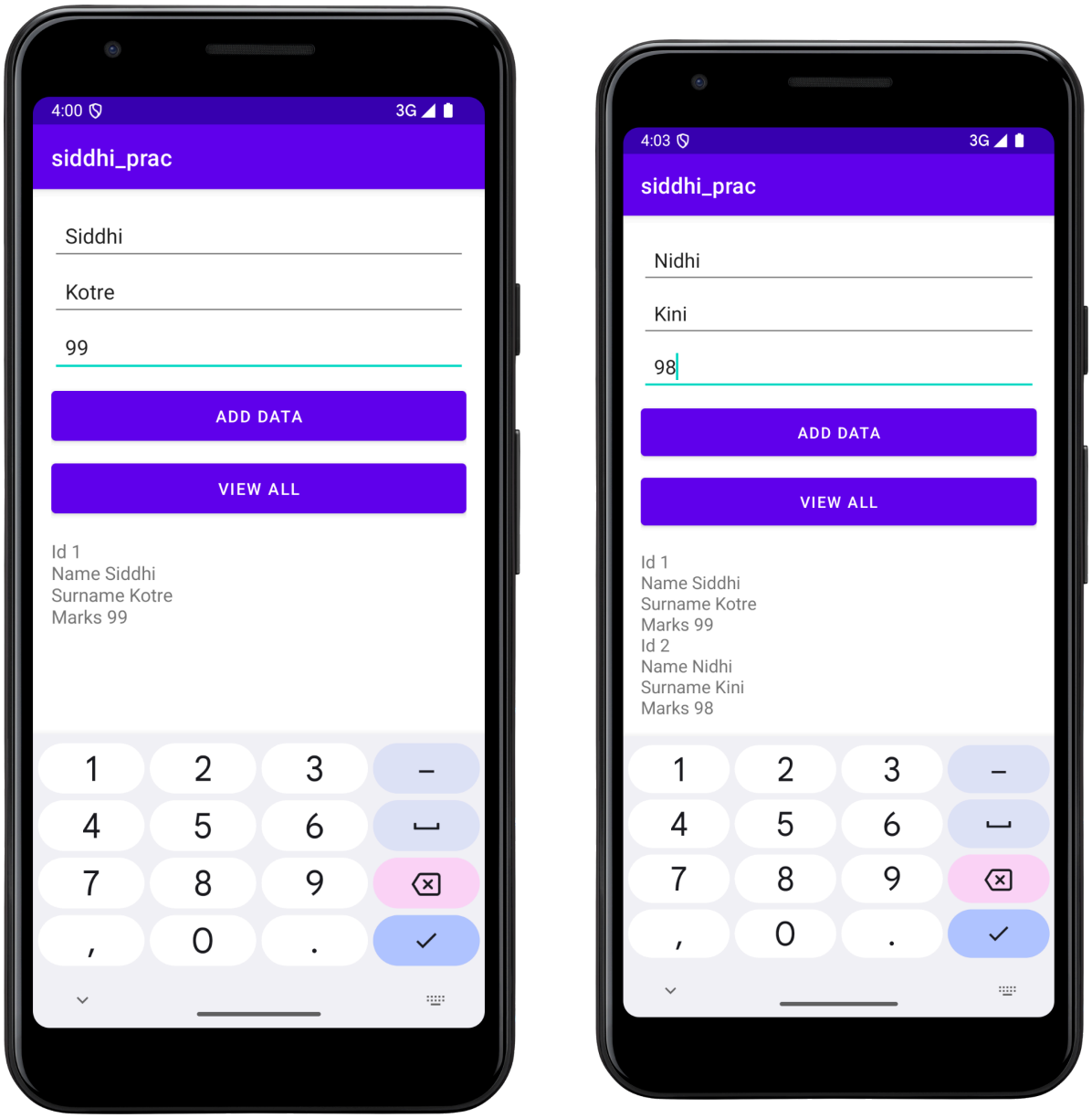
```
        Cursor res = db.rawQuery("select * from "+TABLE_NAME,null);
        return  res;
    }
}
```

MainActivity.java

```
package com.example.siddhi_prac;

import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;
import android.database.Cursor;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
EditText Name, Surname, Marks;
TextView data;

Helper helper;

    @SuppressLint("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Name = findViewById(R.id.Name);
        Surname  = findViewById(R.id.Surname);
        Marks  = findViewById(R.id.Marks);
        data = findViewById(R.id.data);

        helper = new Helper(this);

    }
    public void add_data(View view){
       boolean isInserted=  helper.insert_data(Name.getText().toString(),
Surname.getText().toString(), Integer.parseInt(Marks.getText().toString()));
        if(isInserted==true){
            Toast.makeText(this, "Data is inserted", Toast.LENGTH_SHORT).show();
```

```
        }else {
            Toast.makeText(this, "Data id not inserted", Toast.LENGTH_SHORT).show();
        }
    }

    public void view_data(View view){
        Cursor res = helper.getAllData();
        if(res.getCount()==0){
            Toast.makeText(this, "No data found", Toast.LENGTH_SHORT).show();
            return;
        }
             StringBuffer buffer = new StringBuffer();
        while(res.moveToNext()){
            buffer.append("Id "+res.getString(0)+"\n");
            buffer.append("Name "+res.getString(1)+"\n");
            buffer.append("Surname "+res.getString(2)+"\n");
            buffer.append("Marks "+res.getString(3)+"\n");



        }
        data.setText(buffer.toString());
    }
}
```

Output:


4:00
3G
siddhi_prac
Siddhi
Kotre
99
ADD DATA
VIEW ALL
Id 1
Name Siddhi
Surname Kotre
Marks 99
4:03
3G
siddhi_prac
Nidhi
Kini
98
ADD DATA
VIEW ALL
Id 1
Name Siddhi
Surname Kotre
Marks 99
Id 2
Name Nidhi
Surname Kini
Marks 98

8. **Aim:** Write a program to perform all CRUD operations using SQLite database.

Objective :

The objective of this program is to demonstrate the implementation of basic CRUD operations in an Android application. This program allows users to add, view, update, and delete records in an SQLite database.

Theory:

SQLite Database in Android: SQLite is a self-contained, serverless, and zero-configuration database engine embedded in Android devices. It is used for local data storage, allowing applications to persist data between sessions.

Code:

Activity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
  android:orientation="vertical"
  android:padding="16dp"
  tools:context=".MainActivity">

  <EditText
    android:id="@+id/id"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="ID"
    android:inputType="numberDecimal"
    tools:ignore="TouchTargetSizeCheck" />


  <EditText
    android:id="@+id/name"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Name"
    android:inputType="textPersonName"
    tools:ignore="TouchTargetSizeCheck" />


  <EditText
```

```
    android:id="@+id/salary"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Salary"
    android:inputType="numberDecimal"
    tools:ignore="TouchTargetSizeCheck" />


  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:weightSum="2">

    <Button
      android:id="@+id/add"
      android:onClick="addData"
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:layout_weight="1"
      android:text="Add" />

    <Button
      android:id="@+id/view"
      android:onClick="viewData"
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:layout_weight="1"
      android:text="View" />
  </LinearLayout>


  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:weightSum="2">

    <Button
      android:id="@+id/update"
      android:onClick="update"
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:layout_weight="1"
```

```
        android:text="Update" />

    <Button
        android:id="@+id/delete"
        android:onClick="delete_Data"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Delete" />
</LinearLayout>


<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:weightSum="2">

    <Button
        android:id="@+id/search"
        android:onClick="search"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Search" />

    <Button
        android:id="@+id/clear"
        android:onClick="clear"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Clear" />
</LinearLayout>


<TextView
    android:id="@+id/data"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="text"
    android:textSize="16sp"
    android:paddingTop="20dp" />
```

```
</LinearLayout>
```

Helper.java

```
package com.example.newprac;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

import androidx.annotation.Nullable;

public class Helper extends SQLiteOpenHelper {
   public static final String DATABASE_NAME = "employee.db";
   public static final String TABLE_NAME = "employee_table";
   public static final String COL_1 = "ID";
   public static final String COL_2 = "NAME";
   public static final String COL_3 = "SALARY";


   public Helper(@Nullable Context context) {
      super(context, DATABASE_NAME, null, 1);
   }

   @Override
   public void onCreate(SQLiteDatabase sqLiteDatabase) {
      sqLiteDatabase.execSQL("create table " + TABLE_NAME + "(ID INTEGER PRIMARY KEY
AUTOINCREMENT," + "NAME TEXT, SALARY INTEGER)");
   }

   @Override
   public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {

   }

   public boolean insert_data(String name, int salary) {
      SQLiteDatabase db = this.getWritableDatabase();
      ContentValues contentValues = new ContentValues();
      contentValues.put(COL_2, name);
      contentValues.put(COL_3, salary);

      long result = db.insert(TABLE_NAME, null, contentValues);
      if (result == -1) {
```

```
            return false;
        } else {
            return true;
        }

    }

    public Cursor getAllData() {
        SQLiteDatabase db = this.getReadableDatabase();
        Cursor res = db.rawQuery("select * from " + TABLE_NAME, null);
        return res;
    }


    public Cursor Search_Data(String name){
        SQLiteDatabase db = this.getWritableDatabase();
        Cursor res = db.rawQuery("select * from " + TABLE_NAME+"where"+COL_2+"LIKE
'%"+name+"%'", null);
        return res;
    }

    public boolean update_data(String id, String name, int salary) {
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues contentValues = new ContentValues();
        contentValues.put(COL_1, id);
        contentValues.put(COL_2, name);
        contentValues.put(COL_3, salary);
        db.update(TABLE_NAME,contentValues , "id=?",new String[]{id});
        return true;
    }

    public Integer delete_data(String id){
        SQLiteDatabase db = this.getWritableDatabase();
Integer res = db. delete(TABLE_NAME,"ID=?",new String[]{id});
return res;
    }

    }
```

MainActivity.java

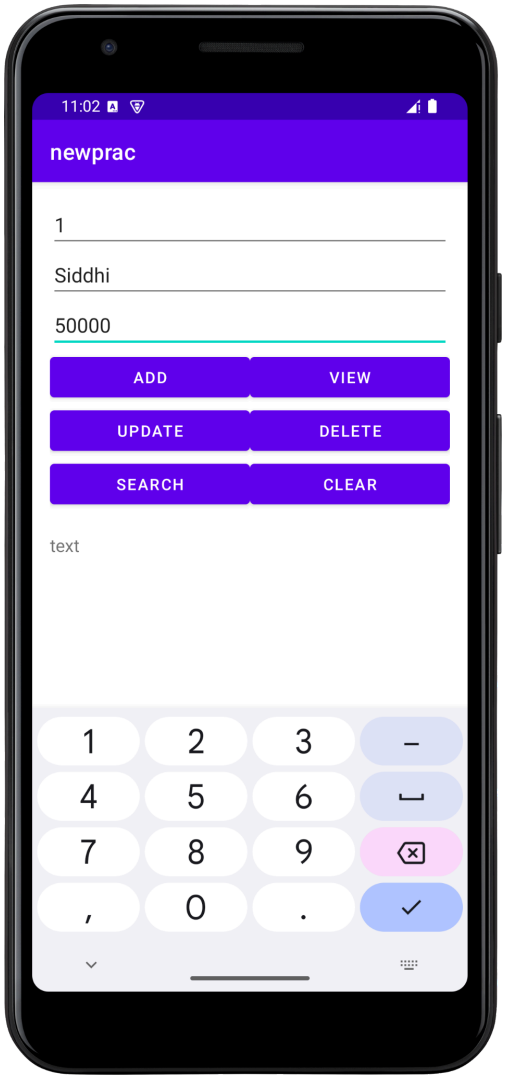
```
package com.example.newprac;

import androidx.appcompat.app.AppCompatActivity;
```

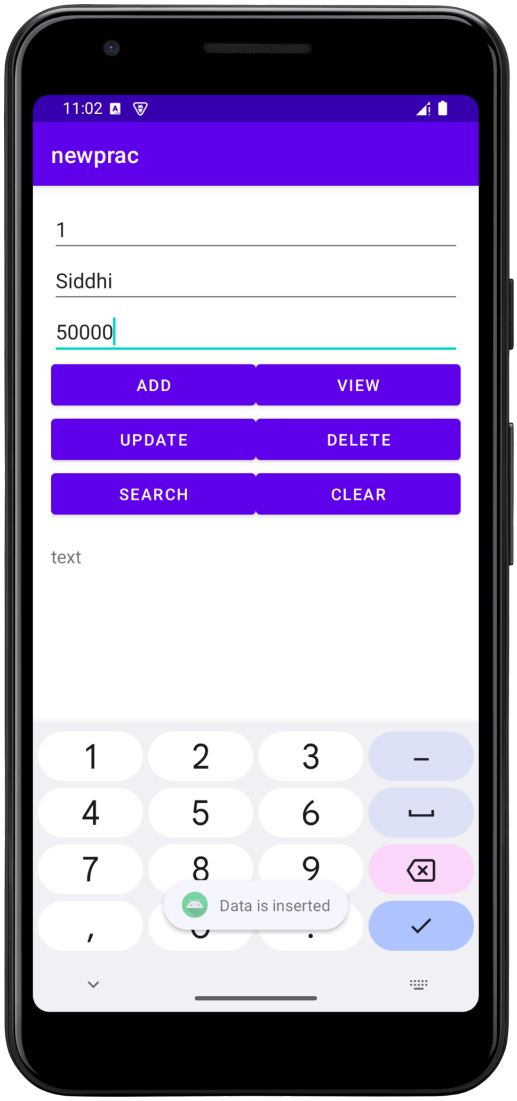
```
import android.database.Cursor;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
  Helper eDb;
  EditText editID,editName, editSalary;
  TextView tv;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);
     eDb = new Helper(this);
     editID=findViewById(R.id.id);
     editName=findViewById(R.id.name);
     editSalary=findViewById(R.id.salary);
     tv = findViewById(R.id.data);
  }

  public void addData(View view){
     boolean isInserted = eDb.insert_data(editName.getText().toString(),
Integer.parseInt(editSalary.getText().toString()));
     if(isInserted==true){
        Toast.makeText(this, "Data is inserted", Toast.LENGTH_SHORT).show();
     }else {
        Toast.makeText(this, "Data id not inserted", Toast.LENGTH_SHORT).show();
     }

  }

  public void viewData(View view){
     Cursor res = eDb.getAllData();

     if(res.getCount()==0){
        Toast.makeText(this, "No data found", Toast.LENGTH_SHORT).show();
        return;
     }
     StringBuffer buffer = new StringBuffer();
     while(res.moveToNext()) {
        buffer.append("Id " + res.getString(0) + "\n");
```

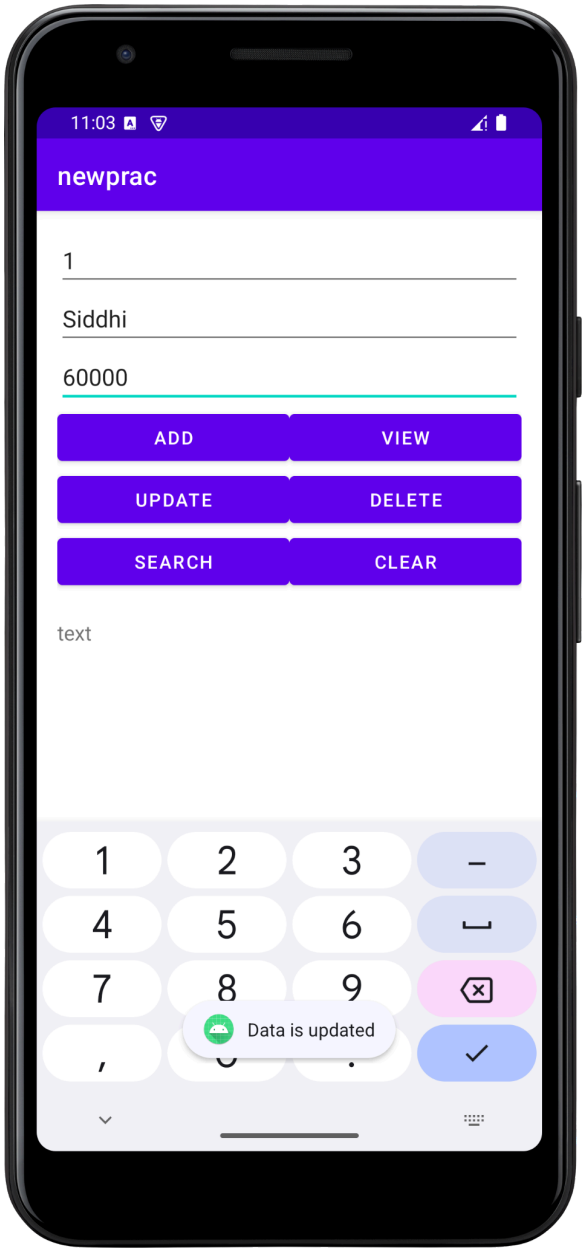
```
        buffer.append("Name " + res.getString(1) + "\n");
        buffer.append("Salary " + res.getString(2) + "\n");
    }
    }

    public void update(View view){
     boolean isupdated =   eDb.update_data(editID.getText().toString(),
            editName.getText().toString(), Integer.parseInt(editSalary.getText().toString()));
        if(isupdated){
            Toast.makeText(this, "Data is updated", Toast.LENGTH_SHORT).show();
        }else {
            Toast.makeText(this, "Data id not updated", Toast.LENGTH_SHORT).show();

        }
    }

    public void delete_Data(View view){
    Integer deletedRows= eDb.delete_data(editID.getText().toString());
    if(deletedRows>0){
        Toast.makeText(this, "Data Deleted", Toast.LENGTH_SHORT).show();
    }
    else{
        Toast.makeText(this, "Data is not deleted", Toast.LENGTH_SHORT).show();
    }
    }


    public  void search(View view){
        Cursor res = eDb.Search_Data(editName.getText().toString());

        if(res.getCount()==0){
            Toast.makeText(this, "No data found", Toast.LENGTH_SHORT).show();
        }
        else{
            while(res.moveToNext()){
                editID.setText(res.getString(0));
                editName.setText(res.getString(1));
                editSalary.setText(res.getString(2));
            }

        }

    }
```

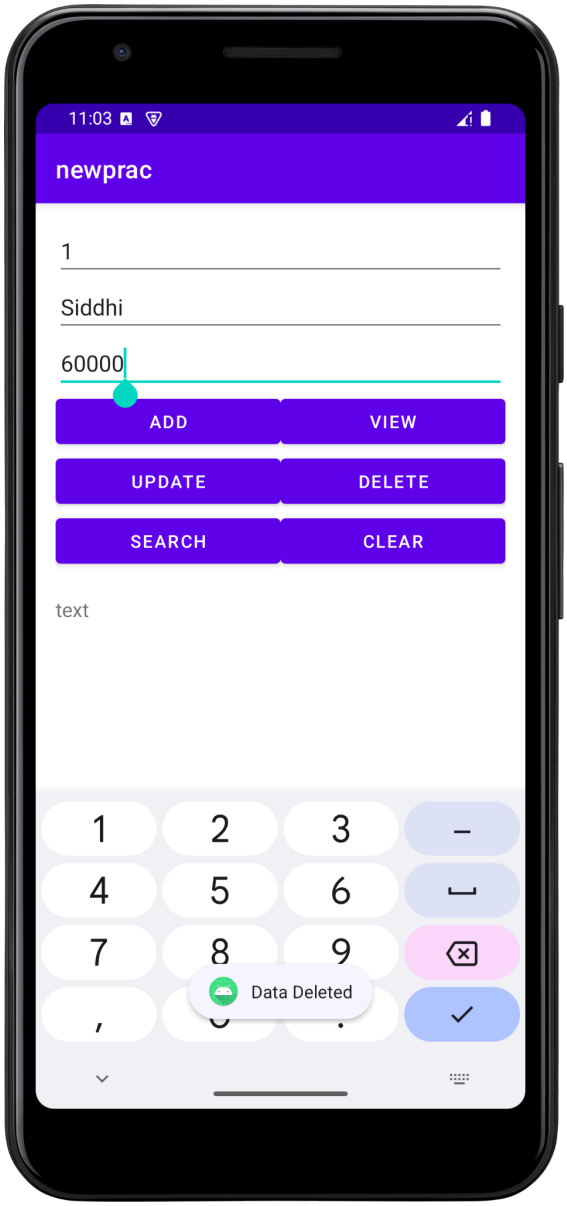
```
    public void clear(View view){
        editID.setText("");
        editName.setText("");
        editSalary.setText("");
        tv.setText("");
    }
}
```

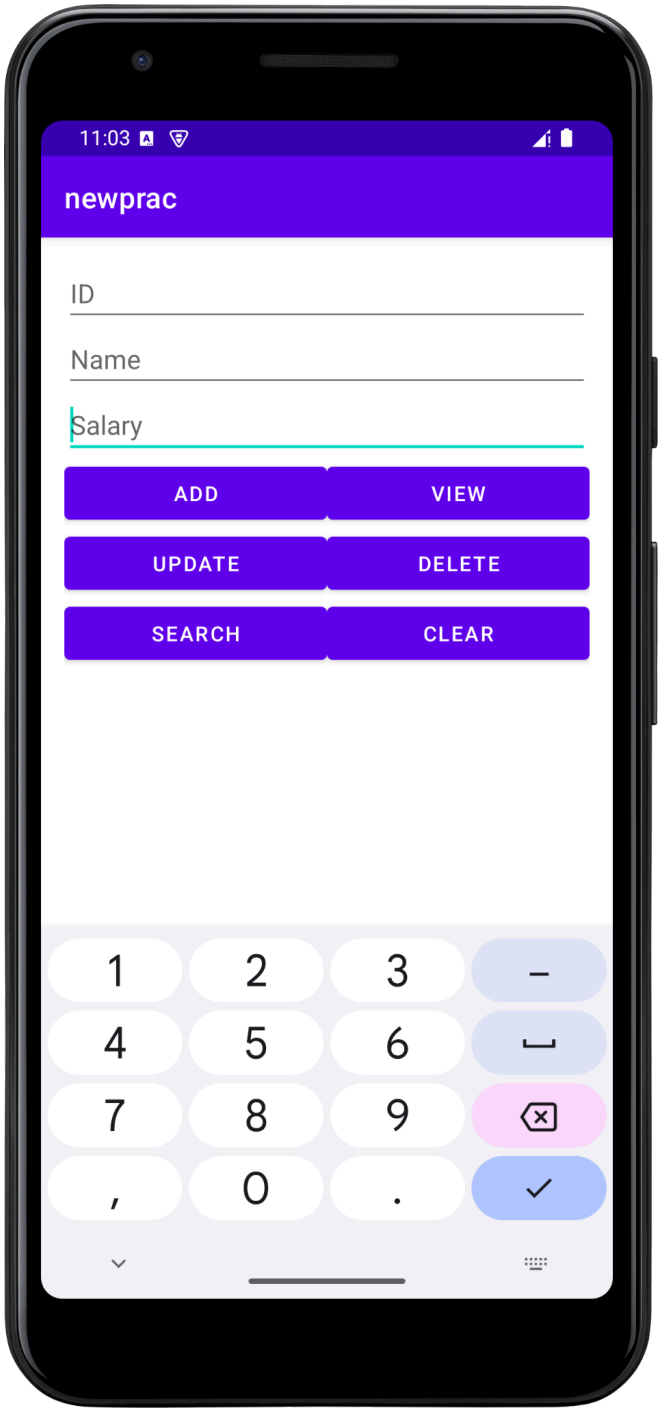
Output:






11:03
newprac
1
Siddhi
60000
ADD
VIEW
UPDATE
DELETE
SEARCH
CLEAR
text
Data is updated


11:03
newprac
1
Siddhi
60000
ADD
VIEW
UPDATE
DELETE
SEARCH
CLEAR
text
Data Deleted


11:03
newprac
ID
Name
Salary
ADD
VIEW
UPDATE
DELETE
SEARCH
CLEAR

9. **Aim:** Write a program to read all contacts using a content provider.

Objective :

The objective of this program is to access and retrieve all contact information stored on an Android device by using a content provider. This program demonstrates how to interact with the Contacts Content Provider and display the retrieved contact names and phone numbers within the app.

Theory:

Content Providers in Android: Content providers are a component of Android that manage access to a structured set of data, enabling data sharing across applications. They allow apps to access data from other applications and to provide data to others in a controlled, secure manner. The most common content providers are those built into the Android platform, such as the Contacts, Calendar, and Media content providers.

Contacts Content Provider: The Contacts Content Provider in Android provides access to the user's contact data, allowing applications to read, modify, and delete contact information with the appropriate permissions. The data is accessed through ContentResolver, which acts as a bridge to interact with the content provider. Each contact has multiple data types (like name, phone number, email) organized in URIs, such as ContactsContract.Contacts.CONTENT_URI. When accessing contacts, it's important to request the READ_CONTACTS permission in the manifest, as this data is considered sensitive.

Code:

activity.xml

```
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="vertical"
  android:gravity="center">

  <Button
     android:onClick="setContact"
     android:id="@+id/button"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:text="Contacts" />

  <TextView
     android:id="@+id/textview"
     android:layout_width="299dp"
     android:layout_height="514dp"
     android:layout_marginTop="16dp"
     android:text="" />
```

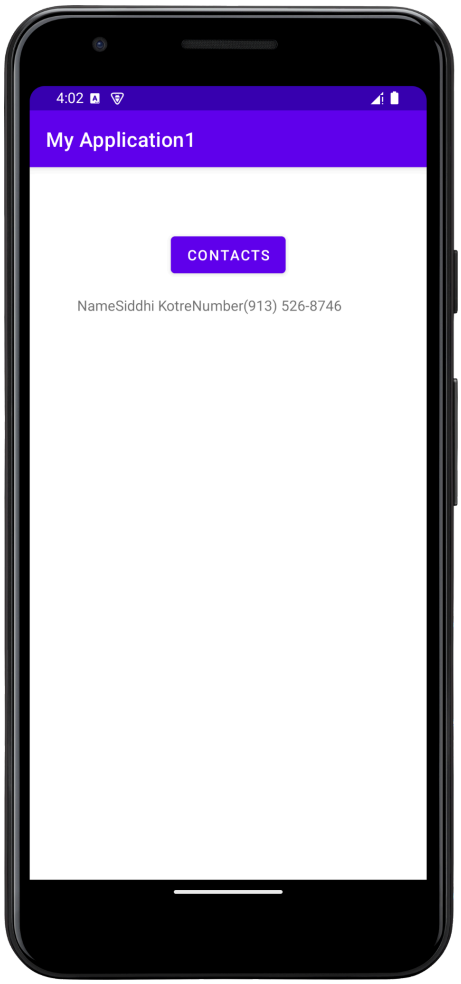
```
</LinearLayout>
```

MainActivity.java

```
package com.example.myapplication1;

import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;

import android.Manifest;
import android.annotation.SuppressLint;
import android.content.ContentResolver;
import android.content.pm.PackageManager;
import android.database.Cursor;
import android.net.Uri;
import android.os.Bundle;
import android.provider.ContactsContract;
import android.text.method.ScrollingMovementMethod;
import android.view.View;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {
TextView tv;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);
     tv = findViewById(R.id.textview);

  }
  @SuppressLint("Range")
  public void setContact(View view){
     if (ContextCompat.checkSelfPermission(this, Manifest.permission.READ_CONTACTS) !=
PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.READ_CONTACTS}, 10);
     }
     ContentResolver contentResolver = getContentResolver();
     Uri uri = ContactsContract.CommonDataKinds.Phone.CONTENT_URI;
     Cursor cursor= contentResolver.query(uri,null,null,null,null);
     if(cursor.getCount()>0){
        while(cursor.moveToNext()){
```

```
            String contactName =
cursor.getString(cursor.getColumnIndex(ContactsContract.CommonDataKinds.Phone.DISPLAY
_NAME));
            String contactNumber =
cursor.getString(cursor.getColumnIndex(ContactsContract.CommonDataKinds.Phone.NUMBER
));
            tv.setText("Name"+contactName+"Number" + contactNumber);
        }
        tv.setMovementMethod(new ScrollingMovementMethod());
    }
  }
}
```

Output:



10. **Aim:** Write a program to demonstrate JSON data parsing using HTTPUrlConnection (you can use https://api.github.com/users json data).

Objective :
The objective of this program is to fetch JSON data from the GitHub API, parse it, and display user information such as username and user ID on the screen. This example illustrates how to make HTTP requests, handle JSON data, and update the UI accordingly.

Theory:
JSON (JavaScript Object Notation): JSON is a lightweight data interchange format that is easy for humans to read and write, and easy for machines to parse and generate. It is often used to transmit data between a server and a web application as an alternative to XML.
HttpURLConnection: HttpURLConnection is a class in Java used to make HTTP requests. It provides methods for sending and receiving data over the web, supporting various HTTP methods like GET, POST, PUT, and DELETE. In this example, we will primarily use the GET method to retrieve data from the GitHub API.
Parsing JSON Data: To parse JSON data in Android, we can use the org.json library, which provides classes like JSONObject and JSONArray to work with JSON objects and arrays. In our program, we will parse the JSON data retrieved from the API to extract specific fields (like username and ID).

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.BLUETOOTH_CONNECT"/>
    <uses-permission android:name="android.permission.BLUETOOTH"/>
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>
    <uses-permission android:name="android.permission.INTERNET"/>

    <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.Assignment_3"
        tools:targetApi="31">
```

```
        <activity
            android:name=".MainActivity_JSONhttpurl"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

MainActivity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    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"
    android:orientation="vertical"
    tools:context=".MainActivity_JSONhttpurl">

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Siddhi Kotre"
        android:textSize="20dp"
        android:padding="10dp"
        android:layout_marginTop="15dp"
        android:layout_gravity="center"
        />

    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="FETCH"
        android:layout_marginTop="90dp"
        android:padding="10dp"
        android:onClick="FetchData"
        android:layout_gravity="center"
        />
```

```
    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="TextView"
        android:layout_gravity="center"
        android:layout_marginTop="90dp"
        android:padding="10dp"
        />

</LinearLayout>
```

MainActivity.java

```
package com.example.assignment_3;

import androidx.appcompat.app.AppCompatActivity;
import android.os.AsyncTask;
import android.os.Bundle;
import android.text.method.ScrollingMovementMethod;
import android.util.Log;
import android.view.View;
import android.widget.TextView;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;

public class MainActivity_JSONhttpurl extends AppCompatActivity {
    TextView tv;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main_jsonhttpurl);
        tv = findViewById(R.id.textView2);
    }

    public void FetchData(View view) {
        RequestData req = new RequestData();
        req.execute();
    }
```

```
    class RequestData extends AsyncTask {
        @Override
        protected Object doInBackground(Object[] objects) {
            HttpURLConnection connection = null;
            BufferedReader reader = null;

            try {
                URL url = new URL("https://api.github.com/users");
                connection = (HttpURLConnection) url.openConnection();
                connection.connect();
                Log.d("request!", "starting");
                InputStream stream = connection.getInputStream();
                reader = new BufferedReader(new InputStreamReader(stream));

                StringBuffer buffer = new StringBuffer();
                String line = "";

                while ((line = reader.readLine()) != null) {
                    buffer.append(line);
                }
                return buffer.toString();
            } catch (MalformedURLException e) {
                throw new RuntimeException(e);
            } catch (IOException ee) {
                throw new RuntimeException(ee);
            }
        }

        @Override
        public void onPostExecute(Object o) {
            super.onPostExecute(o);
            tv.setText(o.toString());
            tv.setMovementMethod(new ScrollingMovementMethod());
        }
    }
}
```

Output:


FETCH
[{"login":"mojombo","id":1,"node_id":"MDQ6VXNlcjE=","avatar
_url":"https://avatars.githubusercontent.com/u/1?v=4"
,"gravatar_id":"","url":"https://api.github.com/users/mojombo"
,"html_url":"https://github.com/mojombo","followers_url":"https:
//api.github.com/users/mojombo/followers","following_url":
"https://api.github.com/users/mojombo/following{/other
_user}","gists_url":"https://api.github.com/users/mojombo
/gists{/gist_id}","starred_url":"https://api.github.com/users
/mojombo/starred{/owner}{/repo}","subscriptions_url":
"https://api.github.com/users/mojombo/subscriptions"
,"organizations_url":"https://api.github.com/users/mojombo
/orgs","repos_url":"https://api.github.com/users/mojombo
/repos","events_url":"https://api.github.com/users/mojombo
/events{/privacy}","received_events_url":"https://api.github
.com/users/mojombo/received_events","type":"User","site
_admin":false},{"login":"defunkt","id":2,"node_id":"MDQ6VXNlcjI
=","avatar_url":"https://avatars.githubusercontent.com/u/2?v=
4","gravatar_id":"","url":"https://api.github.com/users/defunkt"
,"html_url":"https://github.com/defunkt","followers_url":"https:
//api.github.com/users/defunkt/followers","following_url":
"https://api.github.com/users/defunkt/following{/other
_user}","gists_url":"https://api.github.com/users/defunkt
/gists{/gist_id}","starred_url":"https://api.github.com/users
/defunkt/starred{/owner}{/repo}","subscriptions_url":"https://
api.github.com/users/defunkt/subscriptions","organizations
_url":"https://api.github.com/users/defunkt/orgs","repos_url":
"https://api.github.com/users/defunkt/repos","events_url":

11. **Aim:** Write a program to demonstrate JSON data parsing using OkHttp (you can use https://api.github.com/users json data).

Objective :

The objective of this program is to fetch JSON data from the GitHub API using the OkHttp library, parse it, and display user information such as usernames and user IDs. This example illustrates how to make HTTP requests with OkHttp, handle JSON data, and update the user interface accordingly.

Theory:

JSON (JavaScript Object Notation): JSON is a lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate. It is commonly used to transmit data between a server and a web application, making it a preferred format for APIs.
OkHttp: OkHttp is an open-source HTTP client for Android and Java applications. It simplifies making network requests and handling responses. It provides features such as connection pooling, transparent GZIP compression, and response caching, making it a popular choice for developers.
Parsing JSON Data: To parse JSON data in Android, we can use the org.json library, which provides classes like JSONObject and JSONArray to work with JSON objects and arrays. In this program, we will parse the JSON data retrieved from the GitHub API to extract specific fields (like username and ID).

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.INTERNET"/>

    <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.Assignment_3"
        tools:targetApi="31">

        <activity
            android:name=".MainActivity_JSONokhttp"
            android:exported="true">
```

```
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
  </application>
</manifest>
```

MainActivity.xml

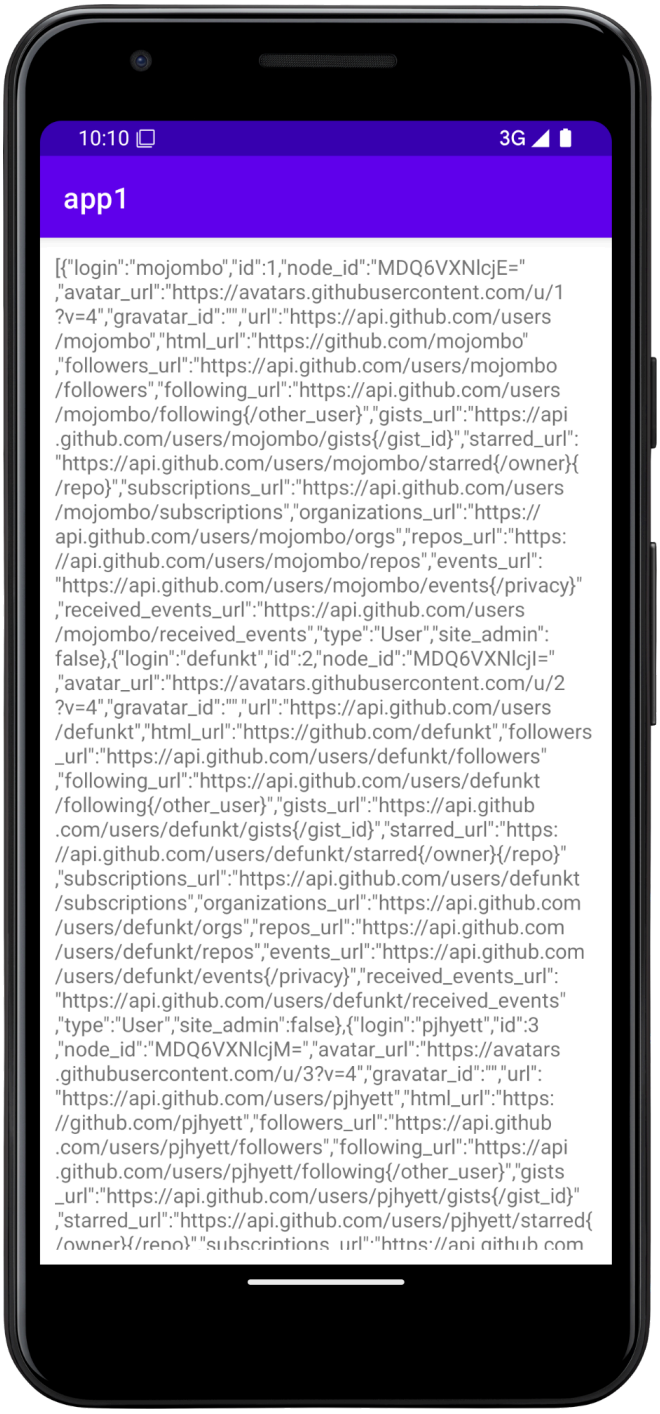
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:app="http://schemas.android.com/apk/res-auto"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:padding="10dp"
  android:orientation="vertical"
  tools:context=".MainActivity_JSONokhttp">

  <TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="10dp"
    android:layout_gravity="center"
    android:text="Siddhi Kotre" />

  <TextView
    android:id="@+id/textView2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="100dp"
    android:textSize="25dp"
    android:layout_gravity="center"
    android:text="TextView" />

</LinearLayout>
```

MainActivity.java

```
package com.example.assignment_3;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
```

```
import androidx.annotation.NonNull;
import android.widget.TextView;
import java.io.IOException;
import okhttp3.Call;
import okhttp3.Callback;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.Response;

public class MainActivity_JSONokhttp extends AppCompatActivity {
    TextView tv;

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

        tv = findViewById(R.id.textView2);

        OkHttpClient client = new OkHttpClient();
        String url = "https://api.github.com/users";
        Request request = new Request.Builder().url(url).build();

        client.newCall(request).enqueue(new Callback() {
            @Override
            public void onFailure(@NonNull Call call, @NonNull IOException e) {
                e.printStackTrace();
            }

            @Override
            public void onResponse(@NonNull Call call, @NonNull Response response) throws
IOException {
                final String myResponse = response.body().string();
                MainActivity_JSONokhttp.this.runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        tv.setText(myResponse);
                    }
                });
            }
        });
    }
}
```

Output:


10:10
3G
app1
[{"login":"mojombo","id":1,"node_id":"MDQ6VXNlcjE="
,"avatar_url":"https://avatars.githubusercontent.com/u/1
?v=4","gravatar_id":"","url":"https://api.github.com/users
/mojombo","html_url":"https://github.com/mojombo"
,"followers_url":"https://api.github.com/users/mojombo
/followers","following_url":"https://api.github.com/users
/mojombo/following{/other_user}","gists_url":"https://api
.github.com/users/mojombo/gists{/gist_id}","starred_url":
"https://api.github.com/users/mojombo/starred{/owner}{
/repo}","subscriptions_url":"https://api.github.com/users
/mojombo/subscriptions","organizations_url":"https://
api.github.com/users/mojombo/orgs","repos_url":"https:
//api.github.com/users/mojombo/repos","events_url":
"https://api.github.com/users/mojombo/events{/privacy}"
,"received_events_url":"https://api.github.com/users
/mojombo/received_events","type":"User","site_admin":
false},{"login":"defunkt","id":2,"node_id":"MDQ6VXNlcjI="
,"avatar_url":"https://avatars.githubusercontent.com/u/2
?v=4","gravatar_id":"","url":"https://api.github.com/users
/defunkt","html_url":"https://github.com/defunkt","followers
_url":"https://api.github.com/users/defunkt/followers"
,"following_url":"https://api.github.com/users/defunkt
/following{/other_user}","gists_url":"https://api.github
.com/users/defunkt/gists{/gist_id}","starred_url":"https:
//api.github.com/users/defunkt/starred{/owner}{/repo}"
,"subscriptions_url":"https://api.github.com/users/defunkt
/subscriptions","organizations_url":"https://api.github.com
/users/defunkt/orgs","repos_url":"https://api.github.com
/users/defunkt/repos","events_url":"https://api.github.com
/users/defunkt/events{/privacy}","received_events_url":
"https://api.github.com/users/defunkt/received_events"
,"type":"User","site_admin":false},{"login":"pjhyett","id":3
,"node_id":"MDQ6VXNlcjM=","avatar_url":"https://avatars
.githubusercontent.com/u/3?v=4","gravatar_id":"","url":
"https://api.github.com/users/pjhyett","html_url":"https:
//github.com/pjhyett","followers_url":"https://api.github
.com/users/pjhyett/followers","following_url":"https://api
.github.com/users/pjhyett/following{/other_user}","gists
_url":"https://api.github.com/users/pjhyett/gists{/gist_id}"
,"starred_url":"https://api.github.com/users/pjhyett/starred{
/owner}{/repo}","subscriptions_url":"https://api.github.com

12. **Aim:** Write a program to demonstrate JSON data parsing using Volley (you can use https://api.github.com/users/mojombo json data).

Objective :

The objective of this program is to fetch JSON data from the GitHub API using the Volley library, parse it, and display user information such as username and user ID. This example illustrates how to make network requests with Volley, handle JSON data, and update the user interface accordingly.

Theory:

JSON (JavaScript Object Notation): JSON is a lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate. It is commonly used to transmit data between a server and a web application, making it a preferred format for APIs.
Volley: Volley is a library developed by Google for Android that simplifies networking tasks. It handles asynchronous HTTP requests and provides built-in features like request prioritization, caching, and automatic scheduling of network requests, making it easier for developers to manage complex networking operations.

Code:

MainActivity.java

```
package com.example.app2;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;

import com.android.volley.Request;
import com.android.volley.RequestQueue;
import com.android.volley.Response;
import com.android.volley.VolleyError;
import com.android.volley.toolbox.JsonObjectRequest;
import com.android.volley.toolbox.Volley;
import com.squareup.picasso.Picasso;

import org.json.JSONException;
import org.json.JSONObject;

public class MainActivity extends AppCompatActivity {
TextView tv;
ImageView iv;
```

```
Button btn_load;

private RequestQueue mQueue;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);
     tv = findViewById(R.id.textView);
     btn_load = findViewById(R.id.loaddata);
     iv = findViewById(R.id.imageView);

     mQueue = Volley.newRequestQueue(this);

  }
  public void ClickButton(View view){
     JsonParse();
  }

  private void JsonParse() {
     String url = "https://api.github.com/users/mojombo";
     JsonObjectRequest jReq = new JsonObjectRequest(Request.Method.GET, url,
           null, new Response.Listener<JSONObject>() {
        @Override
        public void onResponse(JSONObject response) {
           try {
              String login = response.getString("login");
              String id = response.getString("id");
              String node_id = response.getString("node_id");
              String img_url = response.getString("avatar_url");
              tv.setText("login : " + login + "id" + id + "node_id" + node_id + "img_url" + img_url);
              Picasso.get().load(img_url).into(iv);
           } catch (JSONException e) {
              e.printStackTrace();
           }
        }
     }, new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
           error.printStackTrace();
        }
     });
```

```
        mQueue.add(jReq);
    }
}
```

activity_main.xml

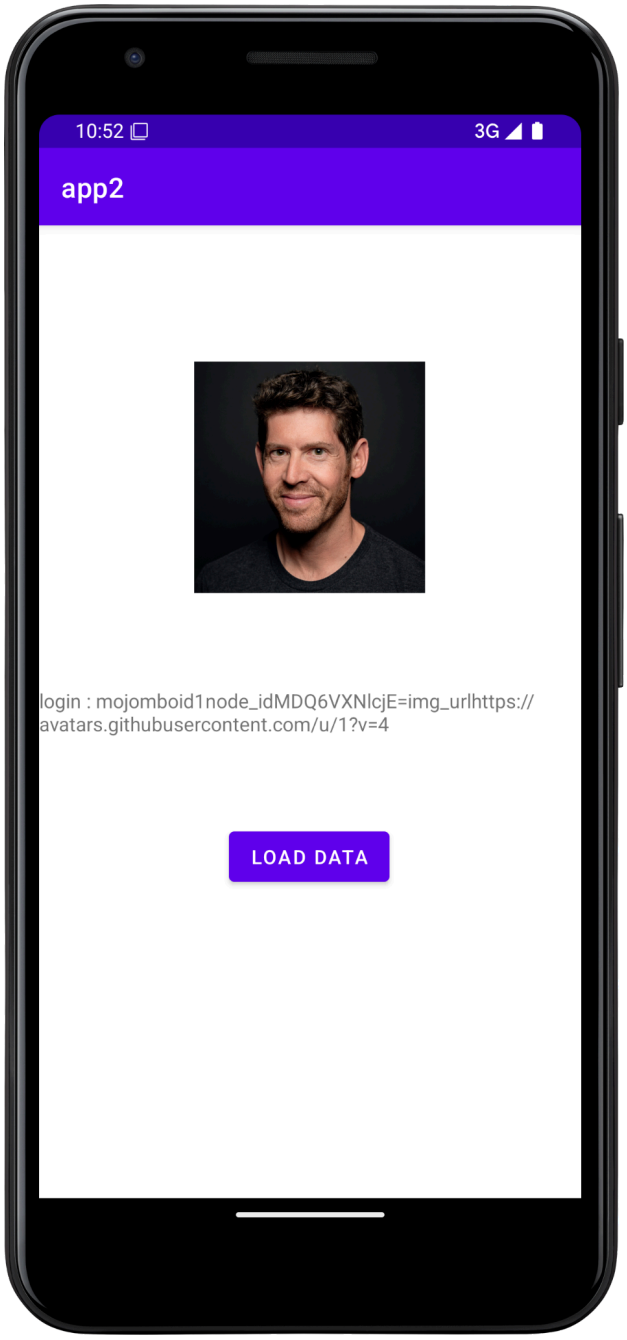
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.184"
        tools:srcCompat="@tools:sample/avatars" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Loading data..."
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/loaddata"
        android:onClick="ClickButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Load Data"
        app:layout_constraintBottom_toBottomOf="parent"
```

```
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.498"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.66" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

build.gradle.kts

```
dependencies {

  implementation("androidx.appcompat:appcompat:1.7.0")
  implementation("com.google.android.material:material:1.12.0")
  implementation("androidx.constraintlayout:constraintlayout:2.1.4")
  testImplementation("junit:junit:4.13.2")
  androidTestImplementation("androidx.test.ext:junit:1.2.1")
  androidTestImplementation("androidx.test.espresso:espresso-core:3.6.1")
  implementation("com.android.volley:volley:1.2.1")
  implementation("com.squareup.picasso:picasso:2.71828")
}
```

Output:


10:52
3G
app2
login : mojomboid1node_idMDQ6VXNlcjE=img_urlhttps://avatars.githubusercontent.com/u/1?v=4
LOAD DATA

13. **Aim:** Write a program to demonstrate JSON data parsing using Retrofit (you can use https://api.github.com/users json data).

Objective :

The objective of this program is to fetch JSON data from the GitHub API using the Retrofit library, parse it, and display user information such as usernames and user IDs. This example illustrates how to define a REST API interface with Retrofit, make network requests, handle JSON data, and update the user interface accordingly.

Theory:

JSON (JavaScript Object Notation): JSON is a lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate. It is widely used for transmitting data between a server and a web application, particularly in API responses.
Retrofit: Retrofit is a type-safe HTTP client for Android and Java developed by Square.

It simplifies the process of making network requests and handling responses. Retrofit allows developers to define a simple interface for API calls, automatically converts JSON responses to Java objects using converters (like Gson), and handles threading for network operations.
Parsing JSON Data: In Retrofit, JSON parsing is managed by converters like Gson, which automatically convert JSON objects into Java objects based on the structure defined in the model classes.

Code:

MainActivity.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    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"
    android:orientation="vertical"
    android:layout_marginTop="20dp"
    tools:context=".MainActivity_JSONretrofit">


    <TextView
        android:id="@+id/retroView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

API.java

```
package com.example.assignment_3;
```

```
import java.util.List;
import retrofit2.Call;
import retrofit2.http.GET;

public interface API {
   String BASE_URL = "https://api.github.com/";

   @GET("users")
   Call<List<MyClass>> getRecords();
}
```

AndroidManifest.xml

```
<uses-permission android:name="android.permission.INTERNET" />
```

MainActivity.java

```
package com.example.assignment_3;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.TextView;
import android.widget.Toast;
import java.util.List;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;

public class MainActivity_JSONretrofit extends AppCompatActivity {
   TextView tv;

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

      tv = findViewById(R.id.retroView);
      getRecords();
   }

   private void getRecords() {
      Retrofit retrofit = new Retrofit.Builder()
         .baseUrl(API.BASE_URL)
```

```
            .addConverterFactory(GsonConverterFactory.create())
            .build();

        API api = retrofit.create(API.class);
        Call<List<MyClass>> call = api.getRecords();
        call.enqueue(new Callback<List<MyClass>>() {
            @Override
            public void onResponse(Call<List<MyClass>> call, Response<List<MyClass>>
response) {
                List<MyClass> list = response.body();
                for (MyClass user : list) {
                    tv.append(user.getLogin() + ", " + user.getNode_id() + "\n");
                }
            }

            @Override
            public void onFailure(Call<List<MyClass>> call, Throwable t) {
                Toast.makeText(MainActivity_JSONretrofit.this, "No input",
Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

MyClass.java

```
package com.example.assignment_3;

public class MyClass {
    String login;
    String node_id;

    public MyClass(String login, String node_id) {
        this.login = login;
        this.node_id = node_id;
    }

    public String getLogin() {
        return login;
    }

    public void setLogin(String login) {
        this.login = login;
    }
```

```
    public String getNode_id() {
        return node_id;
    }

    public void setNode_id(String node_id) {
        this.node_id = node_id;
    }
}
```

Output: