Name of Student : Neha Yadav		
Roll Number : 62		**LAB Assignment Number :** 04
Title of LAB Assignment : Android program to perform CRUD operation using SQLite DB(create table students with fields RollNo, name , email-Id, course and contact no. perform add , update and delete record operations).		
DOP : 6th September 2024		**DOS :** 14th September 2024
CO Mapped : CO2, CO3	**PO Mapped :** PO2, PO3, PO5, PSO1, PSO2	**Signature :**

AIM: Android program to perform CRUD operation using SQLite DB(create table students with fields RollNo, name , email-Id, course and contact no. perform add , update and delete record operations).

THEORY:

1. Database Creation and Management

- **SQLiteOpenHelper**:
  - **Purpose**: Manages database creation and version management.
  - **Key Methods**:
    - `onCreate(SQLiteDatabase db)`: Creates tables and initializes the database.
    - `onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)`: Handles schema changes and updates the database structure.

2. Table Definition

- **Table Name**: `students`
- **Fields**:
  - `RollNo` (INTEGER, PRIMARY KEY): Unique student identifier.
  - `name` (TEXT): Student's name.
  - `emailId` (TEXT): Student's email address.
  - `course` (TEXT): Course the student is enrolled in.
  - `contactNo` (TEXT): Student's contact number.

3. CRUD Operations

1. Create (Insert)

- **Objective**: Add a new record to the database.
- **Action**: Use `insert()` method to add a record into the `students` table.

2. Read (Query)

- **Objective**: Retrieve and display records from the database.
- **Action**: Use `query()` or `rawQuery()` methods to fetch data from the `students` table.

3. Update

- **Objective**: Modify an existing record in the database.
- **Action**: Use `update()` method to change data for a specific record in the `students` table.

4. Delete

- **Objective**: Remove a record from the database.
- **Action**: Use `delete()` method to remove a record from the `students` table.

Each operation involves interacting with an instance of `SQLiteDatabase` obtained through the `SQLiteOpenHelper` class, ensuring proper database management and CRUD functionality.

CODE:

DatabaseHelper.java

```
package com.example.a62b_practical04;

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

public class DatabaseHelper extends SQLiteOpenHelper {

  private static final String DATABASE_NAME = "students.db";
  private static final int DATABASE_VERSION = 1;

  // Table name and columns
  private static final String TABLE_STUDENTS = "students";
  private static final String COLUMN_ROLLNO = "rollno";
  private static final String COLUMN_NAME = "name";
  private static final String COLUMN_EMAIL = "email";
  private static final String COLUMN_COURSE = "course";
  private static final String COLUMN_CONTACT = "contact";

  // Create table SQL query
  private static final String TABLE_CREATE =
       "CREATE TABLE " + TABLE_STUDENTS + " (" +
            COLUMN_ROLLNO + " INTEGER PRIMARY KEY, " +
            COLUMN_NAME + " TEXT, " +
            COLUMN_EMAIL + " TEXT, " +
            COLUMN_COURSE + " TEXT, " +
            COLUMN_CONTACT + " TEXT);";

  public DatabaseHelper(Context context) {
      super(context, DATABASE_NAME, null, DATABASE_VERSION);
  }

  @Override
  public void onCreate(SQLiteDatabase db) {
      db.execSQL(TABLE_CREATE);
  }

  @Override
  public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
      db.execSQL("DROP TABLE IF EXISTS " + TABLE_STUDENTS);
      onCreate(db);
  }

  // Add student
  public long addStudent(int rollno, String name, String email, String course, String contact) {
      SQLiteDatabase db = this.getWritableDatabase();
      ContentValues values = new ContentValues();
      values.put(COLUMN_ROLLNO, rollno);
      values.put(COLUMN_NAME, name);
      values.put(COLUMN_EMAIL, email);
      values.put(COLUMN_COURSE, course);
      values.put(COLUMN_CONTACT, contact);
      return db.insert(TABLE_STUDENTS, null, values);
  }

  // Update student
  public int updateStudent(int rollno, String name, String email, String course, String contact) {
      SQLiteDatabase db = this.getWritableDatabase();
      ContentValues values = new ContentValues();
      values.put(COLUMN_NAME, name);
      values.put(COLUMN_EMAIL, email);
      values.put(COLUMN_COURSE, course);
      values.put(COLUMN_CONTACT, contact);
      return db.update(TABLE_STUDENTS, values, COLUMN_ROLLNO + "=?", new
String[]{String.valueOf(rollno)});
  }

  // Delete student
  public void deleteStudent(int rollno) {
      SQLiteDatabase db = this.getWritableDatabase();
      db.delete(TABLE_STUDENTS, COLUMN_ROLLNO + "=?", new
String[]{String.valueOf(rollno)});
  }
```

```
  // Get all students
  public Cursor getAllStudents() {
    SQLiteDatabase db = this.getReadableDatabase();
    return db.query(TABLE_STUDENTS, null, null, null, null, null, null);
  }
}
```

MainActivity.java

```
package com.example.a62b_practical04;

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 androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

  private DatabaseHelper dbHelper;
  private EditText etRollno, etName, etEmail, etCourse, etContact;
  private Button btnAdd, btnUpdate, btnDelete, btnView;
  private TextView tvResults;

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

    dbHelper = new DatabaseHelper(this);

    etRollno = findViewById(R.id.etRollno);
    etName = findViewById(R.id.etName);
    etEmail = findViewById(R.id.etEmail);
    etCourse = findViewById(R.id.etCourse);
    etContact = findViewById(R.id.etContact);
    btnAdd = findViewById(R.id.btnAdd);
    btnUpdate = findViewById(R.id.btnUpdate);
    btnDelete = findViewById(R.id.btnDelete);
    btnView = findViewById(R.id.btnView);
```

```
tvResults = findViewById(R.id.tvResults);

btnAdd.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        int rollno = Integer.parseInt(etRollno.getText().toString());
        String name = etName.getText().toString();
        String email = etEmail.getText().toString();
        String course = etCourse.getText().toString();
        String contact = etContact.getText().toString();
        dbHelper.addStudent(rollno, name, email, course, contact);
        clearFields();
    }
});

btnUpdate.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        int rollno = Integer.parseInt(etRollno.getText().toString());
        String name = etName.getText().toString();
        String email = etEmail.getText().toString();
        String course = etCourse.getText().toString();
        String contact = etContact.getText().toString();
        dbHelper.updateStudent(rollno, name, email, course, contact);
        clearFields();
    }
});

btnDelete.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        int rollno = Integer.parseInt(etRollno.getText().toString());
        dbHelper.deleteStudent(rollno);
        clearFields();
    }
});

btnView.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Cursor cursor = dbHelper.getAllStudents();
        if (cursor.getCount() == 0) {
            tvResults.setText("No records found.");
            return;
            }

            StringBuilder sb = new StringBuilder();
            while (cursor.moveToNext()) {
                int rollno = cursor.getInt(cursor.getColumnIndex("rollno"));
                String name = cursor.getString(cursor.getColumnIndex("name"));
                String email = cursor.getString(cursor.getColumnIndex("email"));
                String course = cursor.getString(cursor.getColumnIndex("course"));
                String contact = cursor.getString(cursor.getColumnIndex("contact"));

                sb.append("Roll No: ").append(rollno).append("\n");
                sb.append("Name: ").append(name).append("\n");
                sb.append("Email: ").append(email).append("\n");
                sb.append("Course: ").append(course).append("\n");
                sb.append("Contact: ").append(contact).append("\n\n");
            }
            tvResults.setText(sb.toString());
            cursor.close();
        }
    });
  }

  private void clearFields() {
      etRollno.setText("");
      etName.setText("");
      etEmail.setText("");
      etCourse.setText("");
      etContact.setText("");
  }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:padding="16dp">

  <EditText
      android:id="@+id/etRollno"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="Roll No"
      android:inputType="number" />
```

```
<EditText
    android:id="@+id/etName"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@id/etRollno"
    android:hint="Name" />

<EditText
    android:id="@+id/etEmail"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@id/etName"
    android:hint="Email"
    android:inputType="textEmailAddress" />

<EditText
    android:id="@+id/etCourse"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@id/etEmail"
    android:hint="Course" />

<EditText
    android:id="@+id/etContact"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_below="@id/etCourse"
    android:hint="Contact"
    android:inputType="phone" />

<Button
    android:id="@+id/btnAdd"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@id/etContact"
    android:layout_marginTop="16dp"
    android:text="Add" />

<Button
    android:id="@+id/btnUpdate"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_toRightOf="@id/btnAdd"
```

```
        android:layout_marginStart="16dp"
        android:layout_below="@id/etContact"
        android:layout_marginTop="16dp"
        android:text="Update" />

    <Button
        android:id="@+id/btnDelete"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_toRightOf="@id/btnUpdate"
        android:layout_marginStart="16dp"
        android:layout_below="@id/etContact"
        android:layout_marginTop="16dp"
        android:text="Delete" />

    <Button
        android:id="@+id/btnView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_toRightOf="@id/btnDelete"
        android:layout_marginStart="16dp"
        android:layout_below="@id/etContact"
        android:layout_marginTop="16dp"
        android:text="View All" />

    <TextView
        android:id="@+id/tvResults"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/btnAdd"
        android:layout_marginTop="16dp"
        android:text="Results will be shown here"
        android:textSize="16sp" />

</RelativeLayout>
```

AndroidManifest.xml

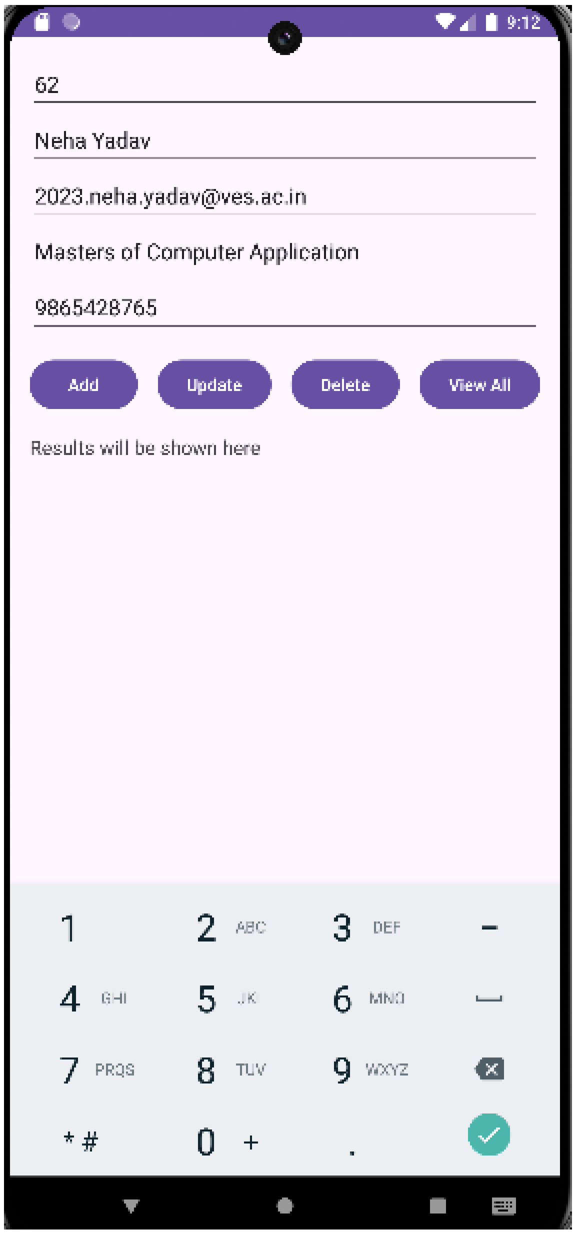
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <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._40B_practical04"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

ADD



62

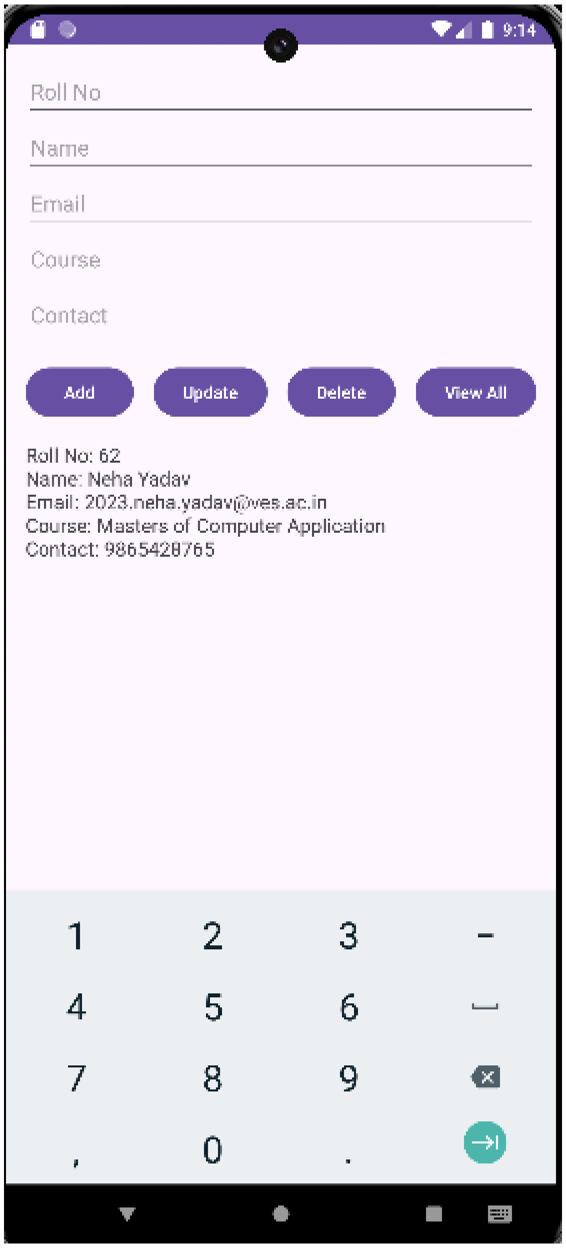
Neha Yadav

2023.neha.yadav@ves.ac.in

Masters of Computer Application

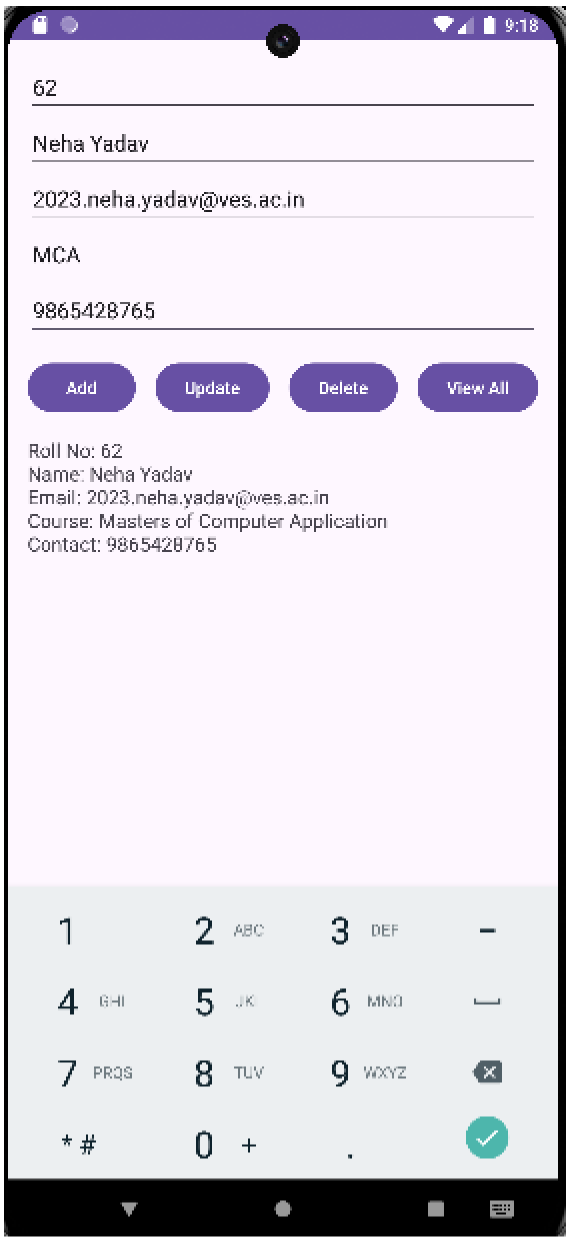
9865428765

Results will be shown here



Roll No: 62
Name: Neha Yadav
Email: 2023.neha.yadav@ves.ac.in
Course: Masters of Computer Application
Contact: 9865428765

UPDATE


62

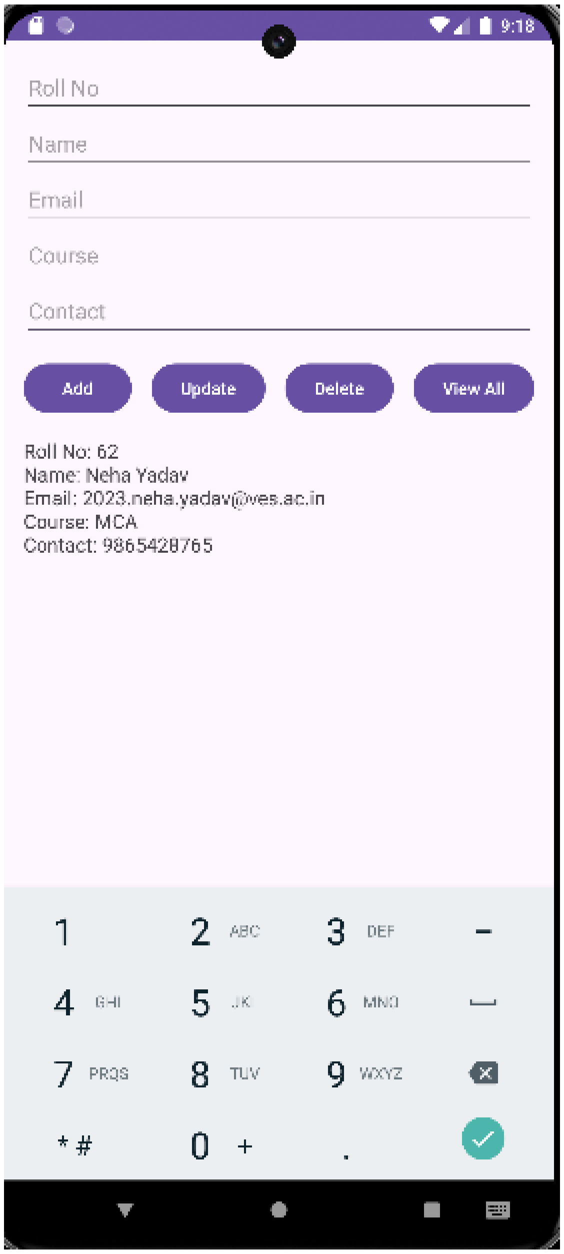
Neha Yadav

2023.neha.yadav@ves.ac.in

MCA

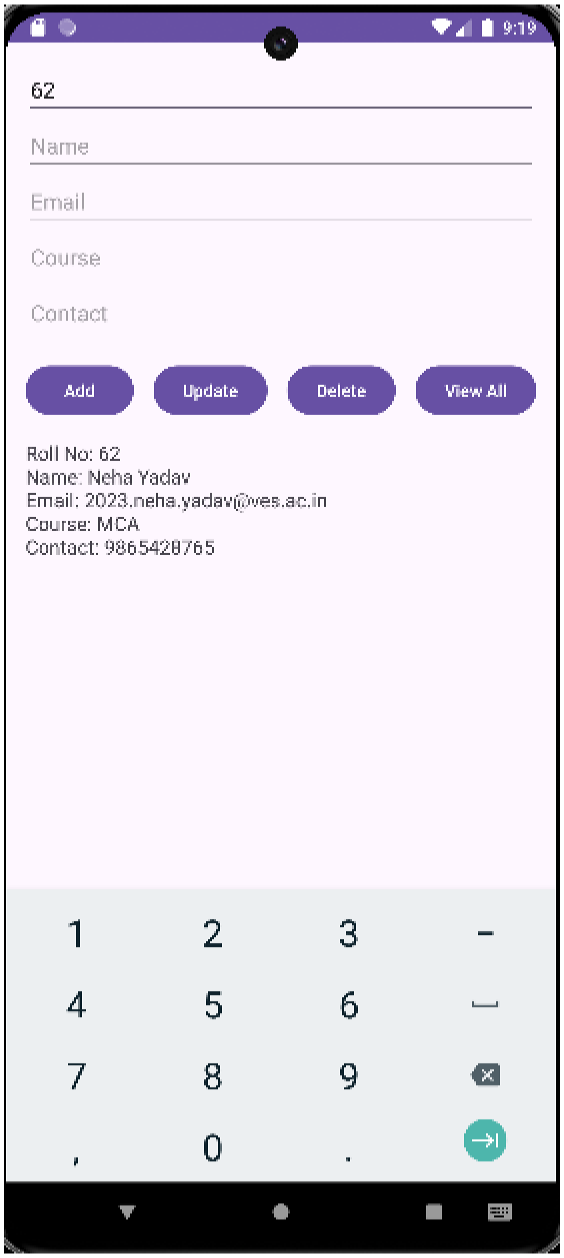
9865428765

Roll No: 62
Name: Neha Yadav
Email: 2023.neha.yadav@ves.ac.in
Course: Masters of Computer Application
Contact: 9865428765

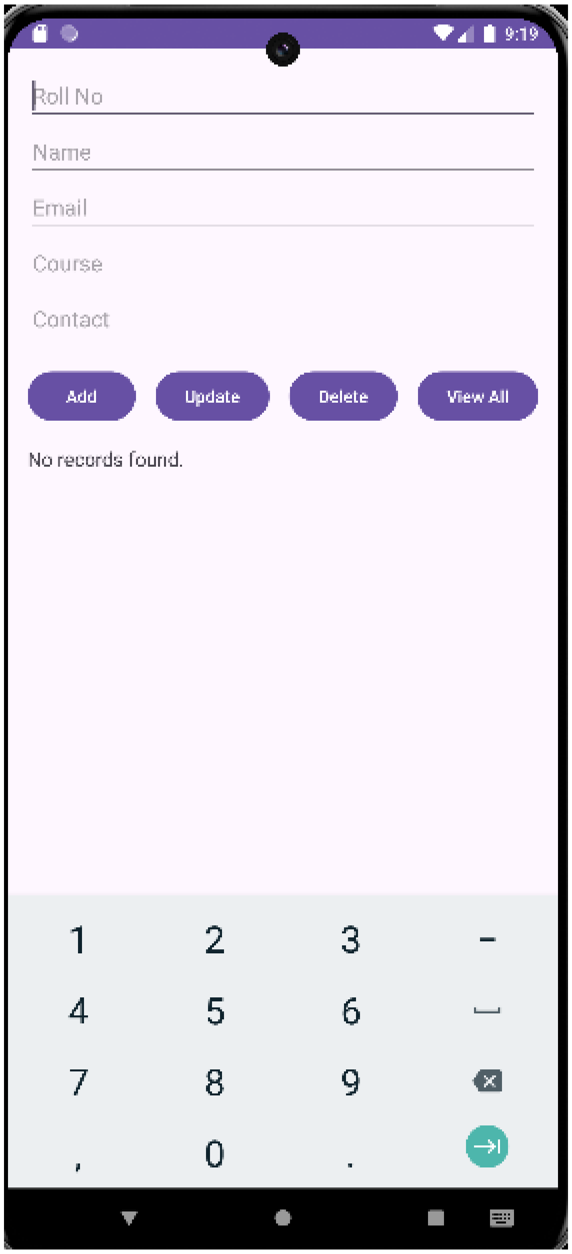


Roll No: 62
Name: Neha Yadav
Email: 2023.neha.yadav@ves.ac.in
Course: MCA
Contact: 9865428765

DELETE



Roll No: 62
Name: Neha Yadav
Email: 2023.neha.yadav@ves.ac.in
Course: MCA
Contact: 9865428765



No records found.

CONCLUSION:

In conclusion, implementing CRUD operations with SQLite in an Android application allows for efficient management of student records through creating, reading, updating, and deleting data in a local database. By defining a `students` table with fields for RollNo, name, email, course, and contact number, developers can ensure structured and persistent data storage. This approach supports dynamic data handling and enhances the app's functionality by allowing users to manage student information seamlessly.