

CONTACT

package com.example.baitaptuan4;  
  
public class Contact {  
 private int id;  
 private String name;  
 private String phoneNumber;  
  
 public Contact() {}  
  
 public Contact(int id, String name, String phoneNumber) {  
 this.id = id;  
 this.name = name;  
 this.phoneNumber = phoneNumber;  
 }  
  
 public Contact(String name, String phoneNumber) {  
 this.name = name;  
 this.phoneNumber = phoneNumber;  
 }  
  
 public int getId() { return id; }  
 public void setId(int id) { this.id = id; }  
  
 public String getName() { return name; }  
 public void setName(String name) { this.name = name; }  
  
 public String getPhoneNumber() { return phoneNumber; }  
 public void setPhoneNumber(String phoneNumber) { this.phoneNumber = phoneNumber; }  
}

DATABASEHANDLER

package com.example.baitaptuan4;  
  
import android.content.ContentValues;  
import android.content.Context;  
import android.database.Cursor;  
import android.database.sqlite.SQLiteDatabase;  
import android.database.sqlite.SQLiteOpenHelper;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class DatabaseHandler extends SQLiteOpenHelper {  
  
 // Phiên bản CSDL  
 private static final int *DATABASE\_VERSION* = 1;  
  
 // Tên CSDL  
 private static final String *DATABASE\_NAME* = "contactsManager";  
  
 // Tên bảng  
 private static final String *TABLE\_CONTACTS* = "contacts";  
  
 // Các cột  
 private static final String *KEY\_ID* = "id";  
 private static final String *KEY\_NAME* = "name";  
 private static final String *KEY\_PH\_NO* = "phone\_number";  
  
 public DatabaseHandler(Context context) {  
 super(context, *DATABASE\_NAME*, null, *DATABASE\_VERSION*);  
 }  
  
 @Override  
 public void onCreate(SQLiteDatabase db) {  
 String CREATE\_CONTACTS\_TABLE = "CREATE TABLE " + *TABLE\_CONTACTS* + "("  
 + *KEY\_ID* + " INTEGER PRIMARY KEY AUTOINCREMENT,"  
 + *KEY\_NAME* + " TEXT,"  
 + *KEY\_PH\_NO* + " TEXT" + ")";  
 db.execSQL(CREATE\_CONTACTS\_TABLE);  
 }  
  
 @Override  
 public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  
 db.execSQL("DROP TABLE IF EXISTS " + *TABLE\_CONTACTS*);  
 onCreate(db);  
 }  
  
 // Thêm contact  
 public void addContact(Contact contact) {  
 SQLiteDatabase db = this.getWritableDatabase();  
 try {  
 ContentValues values = new ContentValues();  
 values.put(*KEY\_NAME*, contact.getName());  
 values.put(*KEY\_PH\_NO*, contact.getPhoneNumber());  
 db.insert(*TABLE\_CONTACTS*, null, values);  
 } finally {  
 db.close();  
 }  
 }  
  
 // Lấy 1 contact theo id  
 public Contact getContact(int id) {  
 SQLiteDatabase db = this.getReadableDatabase();  
 Cursor cursor = db.query(*TABLE\_CONTACTS*,  
 new String[]{*KEY\_ID*, *KEY\_NAME*, *KEY\_PH\_NO*},  
 *KEY\_ID* + "=?",  
 new String[]{String.*valueOf*(id)},  
 null, null, null);  
  
 if (cursor != null && cursor.moveToFirst()) {  
 Contact contact = new Contact(  
 cursor.getInt(cursor.getColumnIndexOrThrow(*KEY\_ID*)),  
 cursor.getString(cursor.getColumnIndexOrThrow(*KEY\_NAME*)),  
 cursor.getString(cursor.getColumnIndexOrThrow(*KEY\_PH\_NO*))  
 );  
 cursor.close();  
 return contact;  
 }  
 return null;  
 }  
  
 // Lấy tất cả contact  
 public List<Contact> getAllContacts() {  
 List<Contact> contactList = new ArrayList<>();  
 String selectQuery = "SELECT \* FROM " + *TABLE\_CONTACTS*;  
 SQLiteDatabase db = this.getReadableDatabase();  
 Cursor cursor = db.rawQuery(selectQuery, null);  
  
 if (cursor.moveToFirst()) {  
 do {  
 Contact contact = new Contact();  
 contact.setId(cursor.getInt(cursor.getColumnIndexOrThrow(*KEY\_ID*)));  
 contact.setName(cursor.getString(cursor.getColumnIndexOrThrow(*KEY\_NAME*)));  
 contact.setPhoneNumber(cursor.getString(cursor.getColumnIndexOrThrow(*KEY\_PH\_NO*)));  
 contactList.add(contact);  
 } while (cursor.moveToNext());  
 }  
 cursor.close();  
 return contactList;  
 }  
  
 // Update contact  
 public int updateContact(Contact contact) {  
 SQLiteDatabase db = this.getWritableDatabase();  
 ContentValues values = new ContentValues();  
 values.put(*KEY\_NAME*, contact.getName());  
 values.put(*KEY\_PH\_NO*, contact.getPhoneNumber());  
 return db.update(*TABLE\_CONTACTS*, values,  
 *KEY\_ID* + " = ?", new String[]{String.*valueOf*(contact.getId())});  
 }  
  
 // Xóa contact  
 public void deleteContact(Contact contact) {  
 SQLiteDatabase db = this.getWritableDatabase();  
 db.delete(*TABLE\_CONTACTS*, *KEY\_ID* + " = ?",  
 new String[]{String.*valueOf*(contact.getId())});  
 db.close();  
 }  
}

DATABASEHELPER

package com.example.baitaptuan4;  
import android.content.Context;  
import android.database.sqlite.SQLiteDatabase;  
import android.database.sqlite.SQLiteOpenHelper;  
public class DatabaseHelper extends SQLiteOpenHelper {  
 private static final String *DATABASE\_CREATE* = "CREATE TABLE users (" +  
 "id INTEGER PRIMARY KEY AUTOINCREMENT, " +  
 "name TEXT NOT NULL" +  
 ");";  
 public DatabaseHelper(Context context, String name,  
 SQLiteDatabase.CursorFactory factory, int version) {  
 super(context, name, factory, version);  
// *TODO Auto-generated constructor stub* }  
 @Override  
 public void onCreate(SQLiteDatabase db) {  
// *TODO Auto-generated method stub* db.execSQL(*DATABASE\_CREATE*);  
 }  
 @Override  
 public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)  
 {  
// *TODO Auto-generated method stub* db.execSQL("DROP TABLE IF EXISTS users");  
 onCreate(db);  
 }  
}

MAINACTIVITY

package com.example.baitaptuan4;  
  
import android.os.Bundle;  
import android.util.Log;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import java.util.List;  
  
public class MainActivity extends AppCompatActivity {  
  
 DatabaseHandler db;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 db = new DatabaseHandler(this);  
  
 // Xóa hết dữ liệu cũ để dễ test  
 clearAllContacts();  
  
 // ====== 1. INSERT ======  
 Log.*d*("CRUD", "Inserting contacts...");  
 db.addContact(new Contact("Ravi", "9100000000"));  
 db.addContact(new Contact("Srinivas", "9199999999"));  
 db.addContact(new Contact("Tommy", "9522222222"));  
  
 // ====== 2. READ ALL ======  
 Log.*d*("CRUD", "Reading all contacts...");  
 List<Contact> contacts = db.getAllContacts();  
 for (Contact cn : contacts) {  
 String log = "Id: " + cn.getId() + " ,Name: " + cn.getName() + " ,Phone: " + cn.getPhoneNumber();  
 Log.*d*("CRUD", log);  
 }  
  
 // ====== 3. GET ONE (id = 1) ======  
 Contact firstContact = db.getContact(1);  
 if (firstContact != null) {  
 Log.*d*("CRUD", "Get Contact id=1 -> " + firstContact.getName() + " / " + firstContact.getPhoneNumber());  
 }  
  
 // ====== 4. UPDATE (id = 2) ======  
 Contact updateContact = db.getContact(2);  
 if (updateContact != null) {  
 updateContact.setName("Updated Name");  
 updateContact.setPhoneNumber("1234567890");  
 db.updateContact(updateContact);  
 Log.*d*("CRUD", "Updated Contact id=2");  
 }  
  
 // ====== 5. DELETE (id = 3) ======  
 Contact deleteContact = db.getContact(3);  
 if (deleteContact != null) {  
 db.deleteContact(deleteContact);  
 Log.*d*("CRUD", "Deleted Contact id=3");  
 }  
  
 // ====== 6. READ ALL AGAIN ======  
 Log.*d*("CRUD", "Reading contacts after update & delete...");  
 List<Contact> updatedContacts = db.getAllContacts();  
 for (Contact cn : updatedContacts) {  
 String log = "Id: " + cn.getId() + " ,Name: " + cn.getName() + " ,Phone: " + cn.getPhoneNumber();  
 Log.*d*("CRUD", log);  
 }  
 }  
  
 private void clearAllContacts() {  
 List<Contact> contacts = db.getAllContacts();  
 for (Contact c : contacts) {  
 db.deleteContact(c);  
 }  
 }  
}