Lưu ý quan trọng: * Trong **Flutter** (Mobile App), chúng ta thường dùng thư viện `sqflite`. * Trong **Dart Console/Backend**, chúng ta dùng thư viện `sqlite3` (vì `sqflite` phụ thuộc vào hệ điều hành mobile). Hướng dẫn sử dụng thư viện **`sqlite3`** (gốc) cho ứng dụng Console. --- ### BƯỚC 1: Cài đặt thư viện Mở Terminal tại thư mục dự án và chạy lệnh: ```bash dart pub add sqlite3 ``` *Lưu ý cho người dùng Windows:* Để thư viện này chạy được, máy bạn cần có file `sqlite3.dll` trong hệ thống. Nếu chạy bị lỗi, bạn chỉ cần tải `sqlite3.dll` từ trang chủ SQLite và để vào cùng thư mục với file `dart.exe` hoặc thư mục `System32` (Nhưng thường Windows 10/11 hiện nay đã hỗ trợ tốt). --- ### BƯỚC 2: Cập nhật Model (Serialization) Database không hiểu Class `Student`, nó chỉ hiểu các dòng (Rows) và cột (Columns). Ta cần viết hàm chuyển đổi qua lại. Mở file `lib/student.dart` và cập nhật: ```dart // lib/student.dart // Import thư viện sqlite3 để dùng kiểu dữ liệu Row import 'package:sqlite3/sqlite3.dart'; class Student { String id; String name; double mathScore; double engScore; Student({ required this.id, required this.name, required this.mathScore, required this.engScore, }); double get averageScore => (mathScore + engScore) / 2; String get rank { if (averageScore >= 8.0) return 'Giỏi'; if (averageScore >= 6.5) return 'Khá'; if (averageScore >= 5.0) return 'Trung Bình'; return 'Yếu'; } // 1. Chuyển từ Object -> Map (Để lưu vào DB) // Tên key trong Map phải trùng tên cột trong Table Map toMap() { return { 'id': id, 'name': name, 'math_score': mathScore, 'eng_score': engScore, }; } // 2. Chuyển từ Database Row -> Object (Để hiển thị lên App) factory Student.fromRow(Row row) { return Student(id: row['id'] as String, name: row['name'] as String, mathScore: row['math_score'] as double, engScore: row['eng_score'] as double,); } @override String toString() { return 'ID: $id | Tên: $name | ĐTB: ${averageScore.toStringAsFixed(1)}'; } } ``` --- ### BƯỚC 3: Tạo lớp quản lý Database Tạo file mới `lib/database_helper.dart`. Đây là nơi chứa các câu lệnh SQL. ```dart // lib/database_helper.dart import 'package:sqlite3/sqlite3.dart'; import 'student.dart'; class DatabaseHelper { late final Database db; // Constructor: Mở kết nối và tạo bảng nếu chưa có DatabaseHelper() { print('🔄 Đang kết nối Database...'); // Mở file students.db (sẽ tự tạo nếu chưa có) db = sqlite3.open('students.db'); _createTable(); } void _createTable() { // SQL tạo bảng db.execute(''' CREATE TABLE IF NOT EXISTS students (id TEXT PRIMARY KEY, name TEXT NOT NULL, math_score REAL, eng_score REAL); '''); } // --- CÁC HÀM CRUD --- // 1. Lấy tất cả sinh viên List getAllStudents() { final ResultSet resultSet = db.select('SELECT * FROM students'); // Convert từng dòng (Row) thành Student object // map() ở đây trả về Iterable, cần .toList() return resultSet.map((row) => Student.fromRow(row)).toList(); } // 2. Thêm sinh viên (Dùng Prepare Statement để bảo mật & an toàn) void insertStudent(Student sv) { final stmt = db.prepare('INSERT INTO students (id, name, math_score, eng_score) VALUES (?, ?, ?, ?)'); try { stmt.execute([sv.id, sv.name, sv.mathScore, sv.engScore]); print('✅ Đã lưu vào Database!'); } catch (e) { print('❌ Lỗi: Có thể ID đã tồn tại.'); } stmt.dispose(); // Giải phóng bộ nhớ } // 3. Xóa sinh viên bool deleteStudent(String id) { final stmt = db.prepare('DELETE FROM students WHERE id = ?'); stmt.execute([id]); // db.getUpdatedRows() trả về số dòng bị ảnh hưởng bool isDeleted = db.getUpdatedRows() > 0; stmt.dispose(); return isDeleted; } // 4. Tìm kiếm theo tên List searchByName(String keyword) { final stmt = db.prepare('SELECT * FROM students WHERE name LIKE ?'); // %keyword% là cú pháp tìm kiếm tương đối trong SQL final ResultSet results = stmt.select(['%$keyword%']); List list = results.map((row) => Student.fromRow(row)).toList(); stmt.dispose(); return list; } // Đóng kết nối khi tắt app (Tuy nhiên console app ít khi cần gọi hàm này thủ công) void close() { db.dispose(); } } ``` --- ### BƯỚC 4: Cập nhật file `main.dart` Thay vì dùng `List students = []`, giờ chúng ta sẽ gọi `DatabaseHelper`. ```dart // bin/main.dart import 'dart:io'; import 'package:my_student_app/student.dart'; import 'package:my_student_app/database_helper.dart'; void main() { // Khởi tạo Database Helper final dbHelper = DatabaseHelper(); while (true) { print('\n=== QUẢN LÝ SINH VIÊN (SQLITE) ==='); print('1. Xem danh sách'); print('2. Thêm sinh viên'); print('3. Xóa sinh viên'); print('4. Tìm kiếm'); print('0. Thoát'); stdout.write('Chọn: '); String? choice = stdin.readLineSync(); switch (choice) { case '1': // Gọi dữ liệu từ DB thay vì List var list = dbHelper.getAllStudents(); showList(list); break; case '2': addStudentUI(dbHelper); break; case '3': deleteStudentUI(dbHelper); break; case '4': searchStudentUI(dbHelper); break; case '0': dbHelper.close(); // Đóng kết nối print('Bye!'); exit(0); default: print('Sai lựa chọn!'); } } } // --- CÁC HÀM UI --- void showList(List list) { print('\n--- KẾT QUẢ ---'); if (list.isEmpty) print('Trống!'); for (var sv in list) { print(sv.toString()); } } void addStudentUI(DatabaseHelper db) { stdout.write('ID: '); String id =

stdin.readLineSync() ?? ''; stdout.write('Tên: '); String name = stdin.readLineSync() ?? ''; stdout.write('Điểm Toán: '); double math = double.tryParse(stdin.readLineSync()!) ?? 0.0; stdout.write('Điểm Anh: '); double eng = double.tryParse(stdin.readLineSync()!) ?? 0.0; Student sv = Student(id: id, name: name, mathScore: math, engScore: eng); // Gọi hàm Insert của Database db.insertStudent(sv); } void deleteStudentUI(DatabaseHelper db) { stdout.write('Nhập ID cần xóa: '); String id = stdin.readLineSync() ?? ''; bool success = db.deleteStudent(id); if (success) { print('✅ Đã xóa thành công.'); } else { print('❌ Không tìm thấy ID này.'); } } void searchStudentUI(DatabaseHelper db) { stdout.write('Nhập tên cần tìm: '); String name = stdin.readLineSync() ?? ''; var list = db.searchByName(name); showList(list); } ``` ### Chạy chương trình 1. Gõ `dart run bin/main.dart`. 2. Thêm vài sinh viên. 3. **Tắt chương trình đi và bật lại.** 4. Chọn "Xem danh sách" -> Dữ liệu vẫn còn nguyên! (Kiểm tra trong thư mục dự án sẽ thấy file `students.db` xuất hiện). ### Giải thích các khái niệm SQL trong bài: 1. **Prepare Statement (`db.prepare`)**: Đây là kỹ thuật bảo mật. Thay vì nối chuỗi SQL (dễ bị hack SQL Injection), ta dùng dấu `?` làm chỗ trống, sau đó truyền dữ liệu vào. 2. **`sqlite3.open('students.db')`**: Nếu file chưa có, nó tự tạo. Nếu có rồi, nó mở ra đọc. 3. **`ResultSet` & `Row`**: Khi SELECT, kết quả trả về là một bảng (`ResultSet`) gồm nhiều dòng (`Row`). Ta dùng hàm `map` để biến đổi từng dòng Row thành object `Student` để code Dart dễ xử lý.