Báo Cáo Đánh Giá & Cải Tiến Thiết Kế Cho Unit Test

1. Giới Thiệu Chung

Ứng dụng hiện tại là một hệ thống quản lý sinh viên, bao gồm các chức năng:

- Thêm, cập nhật, xóa, tìm kiếm sinh viên
- Quản lý dữ liệu sinh viên trong file
- Kiểm tra tính hợp lệ của email, số điện thoại, v.v.
- Cho phép người dùng export, import dữ liệu ra csv, txt, …

Trong quá trình viết unit test, tôi nhận thấy một số vấn đề có thể cải tiến để giúp tăng tính kiểm thử được (“testability”) của hệ thống.

---

2. Vấn Đề Gặp Phải Khi Viết Unit Test

2.1. Phụ thuộc quá nhiều vào file/database

- **Mô tả vấn đề:** Hiện tại, dữ liệu sinh viên được quản lý trong file và chương trình đọc/ghi file trực tiếp. Điều này gây khó khăn cho unit test vì file I/O thường chậm và không đồng nhất giữa các test case.
- **Giải pháp:** Dùng **Mocking** (Google Mock) để giả lập dữ liệu, thay vì làm việc với file thật.

2.2. Luồng code logic và giao diện (Console I/O)

- **Mô tả vấn đề:** Các chức năng như menu() hiện tại kết hợp hiển thị giao diện console và xử lý logic. Điều này khiến việc test trực tiếp trở nên khó khăn vì chương trình luôn yêu cầu người dùng nhập dữ liệu.
- **Giải pháp:** Tách giao diện Console (đọc/ghi dữ liệu) và business logic (xử lý sinh viên) thành hai phần riêng biệt. Điều này cho phép test đơn vị business logic mà không phụ thuộc Console I/O.

2.3. Hàm quá lớn, nhiều trách nhiệm

- **Mô tả vấn đề:** Một số hàm như docDanhSachSinhVien() làm quá nhiều việc:
    - Đọc file
    - Kiểm tra dữ liệu
    - Lưu danh sách sinh viên vào danh sách

Điều này làm cho việc test trở nên phức tạp và không linh hoạt.

- **Giải pháp:** Áp dụng nguyên tắc **Single Responsibility Principle (SRP)**, tách nhỏ thành nhiều hàm:
    - readFile() để đọc dữ liệu
    - validateData() để kiểm tra dữ liệu
    - storeStudents() để lưu dữ liệu

---

3. Cải Tiến Thiết Kế

3.1. Refactor lại chương trình theo mô hình Model-View-Controller (MVC)

- **Model**: Chứa class Student và xử lý dữ liệu sinh viên.
- **View**: Chỉ hiển thị thông tin và nhận input từ người dùng.
- **Controller**: Gọi các phương thức trong Model để xử lý nghiệp vụ, sau đó gửi kết quả cho View.

3.2. Thêm Mock Object để kiểm thử dễ dàng hơn

Sử dụng Google Mock để giả lập database hoặc file, giúp kiểm tra riêng lẻ từng module mà không cần dữ liệu thật.

3.3. Viết thêm unit test bao phủ nhiều trường hợp hơn

- Kiểm thử Student với dữ liệu hợp lệ và không hợp lệ.
- Test riêng validateEmail(), validatePhoneNumber().
- Kiểm thử khi danh sách sinh viên trống, khi có dữ liệu lớn.

---

4. Kết Luận

Bằng cách áp dụng các cải tiến trên, chương trình sẽ: Dễ bảo trì hơn, Unit test hiệu quả hơn, Tách biệt rõ ràng giữa logic xử lý và giao diện

Cần ưu tiên refactor để đảm bảo ứng dụng vừa chạy tốt vừa dễ kiểm thử.