**Lớp: DCCNTT13.10.11**

**Họ tên: Phạm Thị Thảo**

**Mã sinh viên: 20222082**

**Tăng tính tái sử dụng và dễ bảo trì**

Các thành phần được tách biệt giúp dễ dàng sửa đổi, nâng cấp từng phần mà không ảnh hưởng đến các phần khác.

View và Controller có thể thay đổi mà không cần sửa Model nếu không thay đổi dữ liệu cốt lõi.

**Hỗ trợ phát triển song song**

Nhóm lập trình có thể chia nhỏ công việc: một nhóm làm giao diện (View), một nhóm làm xử lý (Controller), nhóm khác làm việc với dữ liệu (Model).

**Dễ mở rộng và kiểm thử**

Với cấu trúc rõ ràng, các thành phần có thể dễ dàng được mở rộng.

Logic nghiệp vụ trong Model có thể được kiểm thử độc lập mà không cần liên quan đến giao diện.

**Tăng tính linh hoạt trong phát triển giao diện người dùng**

Có thể sử dụng nhiều View khác nhau cho cùng một Model (ví dụ: giao diện web, giao diện di động).

**Giảm sự phụ thuộc giữa giao diện và logic xử lý**

Thay vì để giao diện trực tiếp xử lý dữ liệu, Controller đóng vai trò trung gian, từ đó làm giảm sự rối rắm trong code.

**Thích hợp với các ứng dụng có tính tương tác cao**

MVC thường được sử dụng trong các framework như ASP.NET MVC, Spring MVC, Laravel (PHP), Django (Python), v.v.

**Model**: Chịu trách nhiệm quản lý dữ liệu, logic nghiệp vụ và quy tắc xử lý. Đây là nơi tương tác với cơ sở dữ liệu hoặc các nguồn dữ liệu khác.

**View**: Chịu trách nhiệm hiển thị dữ liệu cho người dùng. View không xử lý logic nghiệp vụ mà chỉ hiển thị những gì Model cung cấp.

**Controller**: Nhận các yêu cầu từ người dùng (qua giao diện), xử lý và điều phối tương tác giữa Model và View.

**Sơ đồ mô hình MVC**

