**CÁC MẪU THIẾT KẾ ĐƯỢC SỬ DỤNG**

**Singleton pattern**

Singleton Pattern đảm bảo rằng chỉ có một instance duy nhất của lớp RevenueManager trong toàn bộ ứng dụng. Điều này đảm bảo rằng chỉ có một bộ quản lý doanh thu và thông tin liên quan duy nhất được sử dụng, tránh việc tạo ra nhiều bộ quản lý không cần thiết và đảm bảo tính nhất quán giữa các thành phần của hệ thống, được sử dụng để quản lý số lượng sản phẩm, số lượng người dùng và doanh thu của website.

**Strategy pattern**

Sử dụng Strategy Pattern giúp tách rời logic giảm giá khỏi lớp sản phẩm chính. Điều này đảm bảo rằng lớp sản phẩm chỉ chịu trách nhiệm về các nhiệm vụ cơ bản liên quan đến sản phẩm, trong khi logic giảm giá được đặc tả và quản lý bởi các lớp chiến lược riêng biệt. Việc tách rời này giúp mã nguồn trở nên dễ đọc, dễ bảo trì và dễ mở rộng.

**Simple Factory pattern**

Sử dụng Simple Factory Pattern giúp tách rời quyết định tạo đối tượng khỏi mã nguồn chính. Với Simple Factory Pattern, việc tạo đối tượng được ẩn đi và chỉ cần gọi phương thức tạo đối tượng từ Factory Class. Điều này giảm sự phụ thuộc vào các lớp cụ thể và dễ dàng thay đổi hoặc thêm mới các lớp giảm giá mà không ảnh hưởng đến mã nguồn chính.

**Observer pattern**

Sử dụng Observer Pattern giúp tách rời quyết định gửi thông báo khỏi quản trị viên. Quản trị viên chỉ cần thông báo cho một đối tượng Observer, và các Observer sẽ tự động nhận thông báo và xử lý nó theo cách riêng. Điều này giảm sự phụ thuộc giữa các thành phần trong hệ thống và giúp mã nguồn trở nên rõ ràng và dễ bảo trì.

**Façade pattern**

Facade Pattern giúp đơn giản hóa giao tiếp với các thành phần phức tạp trong hệ thống. Thay vì người dùng phải tương tác trực tiếp với nhiều lớp kiểm tra khác nhau để đăng ký tài khoản, hệ thống cung cấp một lớp kiểm tra đơn giản (facade) để xử lý tất cả các yêu cầu liên quan đến đăng ký. Điều này giúp giảm sự phức tạp của mã nguồn và làm cho giao tiếp với hệ thống trở nên dễ dàng và rõ ràng hơn.

**Builder pattern**

Builder Pattern cho phép xây dựng đối tượng người dùng một cách linh hoạt và tiện lợi. Thay vì phải khởi tạo đối tượng người dùng bằng cách truyền đầy đủ thông tin vào hàm khởi tạo, hệ thống sử dụng một lớp Builder để tạo đối tượng theo từng bước. Điều này cho phép người dùng chỉ cần cung cấp các thông tin cần thiết và bỏ qua những thông tin không cần thiết cho đến khi chúng được sử dụng

**State pattern**

Sử dụng State Pattern giúp thay đổi hành vi của sản phẩm dựa trên trạng thái hiện tại. Khi số lượng sản phẩm hết, có thể thực hiện các hành vi như thông báo cho quản lý, chuyển đổi trạng thái sang "Hết hàng" và ngăn ngừa các hoạt động không hợp lệ như đặt hàng khi không còn hàng.