Học và tên Nguyễn Văn Bằng

Lớp DCCNTT13.10.21

Mã sinh viên 20223965

MVC (Model-View-Controller) là một mẫu kiến trúc phần mềm được sử dụng rộng rãi trong việc phát triển ứng dụng để tạo ra các giao diện người dùng trực quan và có khả năng tương tác cao. MVC chia một ứng dụng thành 3 phần chính và mỗi phần có một vai trò riêng biệt:

1. Model đại diện cho dữ liệu và quy tắc nghiệp vụ của ứng dụng.
2. View chịu trách nhiệm hiển thị dữ liệu cho người dùng một cách trực quan và tương tác.
3. Controller đóng vai trò là cầu nối giữa Model và View, xử lý các yêu cầu từ người dùng và cập nhật giao diện tương ứng.



**View**

Về cơ bản, View đại diện cho cách dữ liệu được trình bày trong ứng dụng (UI). Các view được tạo ra dựa trên dữ liệu thu thập từ model. Bằng cách yêu cầu thông tin từ model, sau đó sẽ trả kết quả tới người dùng. Ngoài việc hiển thị dữ liệu từ các biểu đồ, sơ đồ và bảng, view còn hiển thị dữ liệu từ các nguồn khác. Tất cả các thành phần giao diện người dùng, chẳng hạn như hộp văn bản, menu thả xuống, v.v., sẽ xuất hiện trong bất kỳ view nào của khách hàng.

**Controller**

Controller là thành phần xử lý tương tác của người dùng. Dữ liệu đầu vào của người dùng được controller phân tích và xử lí, khi người dùng thao tác bất kì với hệ thống controller sẽ gửi thông tin đến model để xử lí và sau đó trả về kết quả view

Người dùng -tương tác-> Controller => Model => View (output) -trả kết quả-> Người dùng

Bằng cách giao tiếp với view liên quan của controller, người dùng có thể thay đổi giao diện của view (ví dụ: cuộn qua một tài liệu) và cập nhật trạng thái của model liên quan (ví dụ: lưu một tài liệu).

**Model**

Model là nơi lưu trữ dữ liệu và logic. Ví dụ, khi Controller truy xuất thông tin khách hàng từ cơ sở dữ liệu, dữ liệu được chuyển đổi giữa các thành phần controller hoặc giữa các yếu tố logic nghiệp vụ. Nó thao tác dữ liệu và gửi lại cơ sở dữ liệu, hoặc được sử dụng để hiển thị thông tin tương tự.

Ưu điểm của Kiến trúc MVC

* Mô hình MVC ra đời giúp các lập trình viên giải quyết nhiều vấn đề lúc bấy giờ, cùng điểm qua các ưu điểm của MVC:
* Mã nguồn dễ bảo trì và mở rộng: Kiến trúc MVC giúp mã nguồn dễ bảo trì, có thể mở rộng và phát triển một cách dễ dàng.
* Các thành phần có thể kiểm tra độc lập: Các thành phần của nó có thể được kiểm tra riêng biệt từ các thành phần người dùng.
* Hỗ trợ các loại khách hàng mới dễ dàng hơn: Kiến trúc này hỗ trợ việc thêm các loại khách hàng mới một cách dễ dàng.
* Phát triển song song các thành phần: Có thể phân chia phát triển các thành phần khác nhau một cách song song.
* Giảm bớt độ phức tạp: Bằng cách chia ứng dụng thành ba phần, bạn có thể tránh được sự phức tạp.
* Sử dụng mô hình Front Controller: Chỉ sử dụng mô hình Front Controller, trong đó mỗi yêu cầu được xử lý bởi một bộ điều khiển duy nhất.
* Hỗ trợ phát triển theo hướng kiểm thử tối đa: Hỗ trợ phát triển theo hướng kiểm thử (TDD) một cách tối đa.
* Dễ dàng cho các nhóm phát triển lớn: Một nhóm lớn các nhà thiết kế và phát triển web có thể tạo và duy trì các ứng dụng web với nó.
* Kiểm tra từng lớp và đối tượng riêng biệt: Bạn có thể kiểm tra tất cả các lớp và đối tượng riêng lẻ vì chúng đều độc lập với nhau.
* Hành động của bộ điều khiển có thể nhóm logic: Các hành động của bộ điều khiển có thể được nhóm lại một cách logic bằng cách sử dụng các mẫu thiết kế MVC.

Nhược điểm của Kiến trúc MVC

* Khó đọc, thay đổi và kiểm thử đơn vị: Khó đọc, thay đổi và kiểm thử đơn vị mô hình này vì không có thành phần riêng biệt để xử lý giao diện người dùng (UI). Tất cả phải được thực hiện trên lớp view. Vì vậy, cần phải phụ thuộc vào một framework khác để làm điều đó.
* Không hỗ trợ xác thực chính thức: Kiến trúc MVC không cung cấp hỗ trợ xác thực chính thức. Vì vậy, việc xác thực cần phải được thực hiện rõ ràng.
* Xử lý dữ liệu có thể không hiệu quả: Có thể dẫn đến quá trình xử lý dữ liệu không hiệu quả vì nó làm cho logic thực thi trở nên phức tạp.
* Khó sử dụng với giao diện người dùng hiện nay: Việc sử dụng MVC khó khăn với giao diện người dùng hiện đại. Hầu hết các framework UI phổ biến hiện nay đều có kiến trúc triển khai riêng của chúng và không thể nhúng vào MVC.
* Cần duy trì nhiều mã trong bộ điều khiển: Cần duy trì nhiều mã trong các bộ điều khiển, gần như cho mỗi hành động khác nhau trên trang, chúng ta cần khai báo các phương thức Action riêng biệt.