# **Tìm hiểu mô hình MVC**

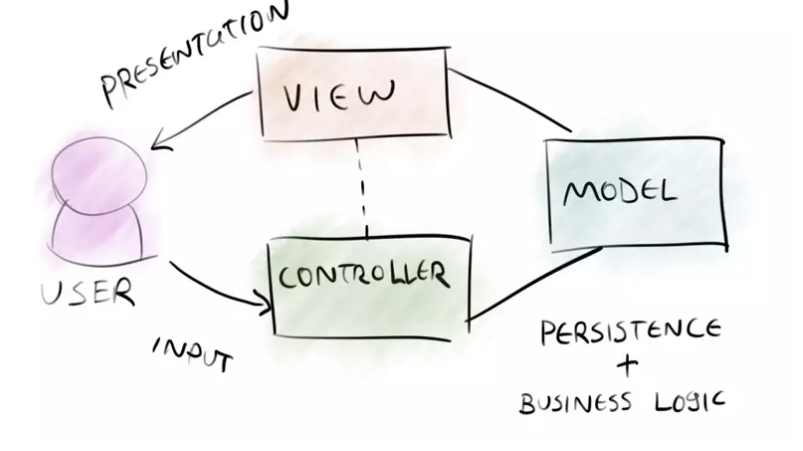
## **Mô hình MVC là gì?**

Mô hình Model-View-Controller (MVC) là một mẫu kiến ​​trúc phân tách một ứng dụng thành ba thành phần logic chính Model, View và Controller. Do đó viết tắt MVC. Mỗi thành phần kiến ​​trúc được xây dựng để xử lý khía cạnh phát triển cụ thể của một ứng dụng. MVC tách lớp logic nghiệp vụ và lớp hiển thị ra riêng biệt. Ngày nay, kiến ​​trúc MVC đã trở nên phổ biến để thiết kế các ứng dụng web cũng như ứng dụng di động.

## **MVC ra đời khi nào?**

* Kiến trúc MVC được thảo luận lần đầu vào năm 1979 bởi Trygve Reenskaug.
* Mô hình MVC được giới thiệu lần đầu tiên vào năm 1987 bằng ngôn ngữ lập trình Smalltalk.
* MVC lần đầu tiên được chấp nhận như một khái niệm chung, trong một bài báo năm 1988.
* Trong thời gian gần đây, MVC pattern được sử dụng rộng rãi trong các ứng dụng web hiện đại.

## **Kiến trúc MVC!**



****MVC quan trọng bao gồm:****

* Model: Nó bao gồm tất cả dữ liệu và logic liên quan của nó.
* View: Trình bày dữ liệu cho người dùng hoặc xử lý tương tác của người dùng.
* Controller: Là phần quan trọng nhất trong mô hình, nó liên kết phần Model và View.

### **View**

View là một phần của ứng dụng đại diện cho việc trình bày dữ liệu.

View được tạo bởi các dữ liệu mà chúng ta lấy từ dữ liệu trong model. Một view yêu cầu model cung cấp đầy đủ dữ liệu để nó hiển thị đầu ra cho người dùng.

View chính là nới chứa những giao diện như một nút bấm, khung nhập, menu, hình ảnh… nó đảm nhiệm nhiệm vụ hiển thị dữ liệu và giúp người dùng tương tác với hệ thống.

### **Controller**

Controller là một phần của ứng dụng xử lý tương tác của người dùng. Bộ điều khiển diễn giải đầu vào chuột và bàn phím từ người dùng, thông báo cho model và view để thay đổi khi thích hợp.

Controller là nới tiếp nhận những yêu cầu xử lý được gửi từ người dùng, nó sẽ gồm những class/ function xử lý nhiều nghiệp vụ logic giúp lấy đúng dữ liệu thông tin cần thiết nhờ các nghiệp vụ lớp Model cung cấp và hiển thị dữ liệu đó ra cho người dùng nhờ lớp View.

Controller gửi các lệnh đến model để làm thay đổi trạng thái của nó (Ví dụ: ta thêm mới 1 user hoặc cập nhật tên 1 user). Controller cũng gửi các lệnh đến view liên quan của nó để thay đổi cách hiển thị của view (Ví dụ: xem thông tin 1 user).

### **Model**

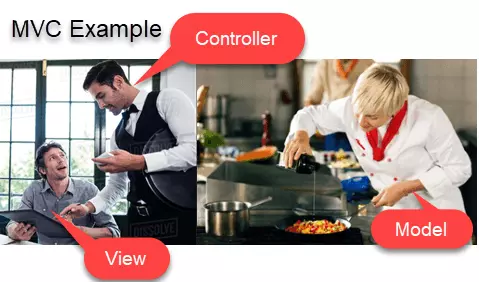
* Thành phần model lưu trữ dữ liệu và logic liên quan của nó. Bao gồm các class function xử lý các tác vụ như truy vấn, thêm, sửa hoặc xóa dữ liệu. Ví dụ, một đối tượng Controller sẽ lấy thông tin khách hàng từ cơ sở dữ liệu. Nó thao tác dữ liệu và gửi trở lại cơ sở dữ liệu hoặc sử dụng nó để hiển thị dữ liệu.

## **Sự tương tác giữa các thành phần**

* Controller tương tác với qua lại với View.
* Controller tương tác qua lại với Model.
* Model và View không có sự tương tác với nhau trực tiếp mà nó tương tác với nhau thông qua Controller.

****Ví dụ cho sự tương tác:**** Khi người dùng ấn đăng nhập từ view thì request sẽ được gửi từ trình duyệt đến controller, controller sẽ gọi đến model xử lý logic và trả lại kết quả đó cho user thông qua view .

## **Ví dụ về MVC**

****Hãy xem ví dụ về Model View Controller từ cuộc sống hàng ngày của chúng ta:****

* Giả sử bạn đi đến một nhà hàng. Bạn sẽ không phải vào bếp và chuẩn bị thức ăn mà bạn chắc chắn có thể làm ở nhà của mình. Thay vào đó, bạn chỉ cần đến đó và đợi người phục vụ đến.
* Bây giờ người phục vụ đến với bạn, và bạn chỉ việc gọi món ăn. Người phục vụ không biết bạn là ai và bạn muốn gì, anh ta chỉ ghi chi tiết đơn hàng thức ăn của bạn.
* Sau đó, người phục vụ di chuyển đến nhà bếp. Trong nhà bếp, người phục vụ không chuẩn bị thức ăn cho bạn.
* Đầu bếp chuẩn bị thức ăn cho bạn. Người phục vụ sẽ đưa món của bạn cho anh ta cùng với số bàn của bạn.
* Nấu thức ăn sau đó chuẩn bị cho bạn. Anh ấy sử dụng các nguyên liệu để nấu món ăn. Hãy giả sử rằng bạn đặt một bánh sandwich rau. Sau đó, anh ta cần bánh mì, cà chua, khoai tây, ớt chuông, hành tây, một chút, pho mát, v.v. mà anh ta lấy từ tủ lạnh.
* Nấu xong cuối cùng giao đồ ăn cho người phục vụ. Bây giờ công việc của người phục vụ là chuyển thực phẩm này ra ngoài nhà bếp.
* Bây giờ người phục vụ biết bạn đã đặt món ăn nào và chúng được phục vụ như thế nào.

****Sau khi xem ví dụ này ta thấy được :****

* View: Chính là bạn.
* Controller: Là người phục vụ.
* Model: Là đầu bếp.
* Database: là tủ lạnh.

## **Ưu điểm của MVC:**

* Bảo trì code dễ dàng, dễ dàng mở rộng và phát triển.
* Hỗ trợ dễ dàng hơn cho khách hàng mới.
* Việc phát triển các thành phần khác nhau có thể được thực hiện song song.
* Nó giúp bạn tránh sự phức tạp bằng cách chia ứng dụng thành ba đơn vị Model, View và Controller.
* Cung cấp hỗ trợ tốt nhất cho phát triển theo hướng thử nghiệm.
* Nó hoạt động tốt cho các ứng dụng Web được hỗ trợ bởi các nhóm lớn các nhà thiết kế và phát triển web.
* Cung cấp khả năng phân tách rõ ràng các mối quan tâm.
* Thân thiện với Công cụ Tìm kiếm (SEO).
* Tất cả các đối tượng được phân loại và đối tượng độc lập với nhau để bạn có thể kiểm tra chúng một cách riêng biệt.

## **Nhược điểm của việc sử dụng MVC**

* Khó đọc, thay đổi, kiểm tra và sử dụng lại mô hình này.
* Không có hỗ trợ xác thực chính thức.
* Tăng độ phức tạp và tính kém hiệu quả của dữ liệu.
* Khó khăn khi sử dụng MVC với giao diện người dùng hiện đại.
* Cần có nhiều người lập trình để tiến hành lập trình song song.
* Cần có kiến ​​thức về nhiều công nghệ.
* Bảo trì nhiều code trong Controller.

## **Kết luận**

* MVC là một mẫu kiến ​​trúc phân tách ứng dụng thành 1) Model, 2) View và 3) Controller.
* Model: Nó bao gồm tất cả dữ liệu và logic liên quan của nó.
* View: Trình bày dữ liệu cho người dùng hoặc xử lý tương tác của người dùng.
* Controller: Là nơi kết nối giữa các thành phần Model và View.
* Kiến trúc MVC được thảo luận lần đầu vào năm 1979 bởi Trygve Reenskaug.
* MVC là một mô hình dễ dàng mở rộng và bảo trì code.
* Một số Framework MVC phổ biến là Rails, Zend Framework, CodeIgniter, Laravel, Fuel PHP, v.v.