**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

**KHOA CÔNG NGHỆ THÔNG TIN**

-------------o0o------------



**BÁO CÁO BÀI TẬP LỚN**

**MÔN HỌC KIẾN TRÚC VÀ THIẾT KẾ PHẦN MỀM**

***Đề tài:***

**TÌM HIỂU VỀ KIẾN TRÚC MVC**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | **ThS. Nguyễn Thị Dung** |
| **Sinh viên thực hiện:** | **Nguyễn Quang Huy**  **Lô Văn Đại**  **Ngô Hồng Thái**  **Phạm Thị Thuỳ Dương**  **Trần Thị Loan** |
| **Lớp:** | **KTPM K18B** |

***Thái Nguyên, tháng 10 năm 2022***

*Thái Nguyên, ngày 24 tháng 1 năm 2014*



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Họ và tên | Công việc | Thời gian hoàn thành | Ghi chú |
| 1 | Lô Văn Đại (Nhóm trưởng) | Tiếp cận vấn đề,mô tả bài toán và xây dựng kế hoạch quản lý yêu cầu. | 03/03/2022 | Tổng hợp tài liệu đến khi hoàn tất bản báo cáo |
| Xác định các stakeholder | 07/3/2022 |
| Phân tích và rút ra các yêu cầu | 10/03/2022 |
| Phát triển tài liệu trực quan | 14/03/2022 |
| Phân tích thiết kế hệ thống (Góp ý chung) |  |
| 2 | Nguyễn Quang Huy | Xác định các stakeholder | 07/3/2022 |  |
| Phân tích và rút ra các yêu cầu | 10/03/2022 |
| Phân tích thiết kế hệ thống ( Thiết kế giao diện) | 27/03/2022 |
| 3 | Phạm Thị Thùy Dương | Xác định các stakeholder | 07/3/2022 |  |
| Phân tích và rút ra các yêu cầu | 10/03/2022 |
| Phân tích thiết kế hệ thống ( Đặc tả chi tiết các UC) | 23/03/2022 |
| 4 | Ngô Hồng Thái | Xác định các stakeholder | 07/3/2022 |  |
| Phân tích và rút ra các yêu cầu | 10/03/2022 |
| Phân tích thiết kế hệ thống (Xây dựng biểu đồ hoạt động) | 25/03/2022 |
| 5 | Trần Thị Loan | Xác định các stakeholder | 07/3/2022 |  |
| Phân tích và rút ra các yêu cầu | 10/03/2022 |
| Phân tích thiết kế hệ thống ( bản vẽ UC và biểu đồ lớp) | 17/03/2022 |

# Chương 1: Tổng quan về mô hình MVC

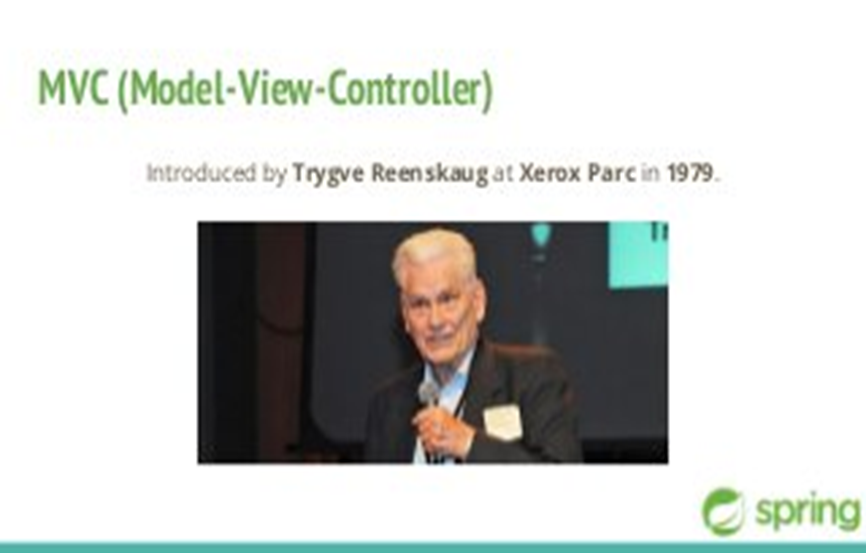
## 1.1 Mô hình MVC

### 1.1.1 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.

### 1.1.2 Lịch sử ra đời của mô hình MVC

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.



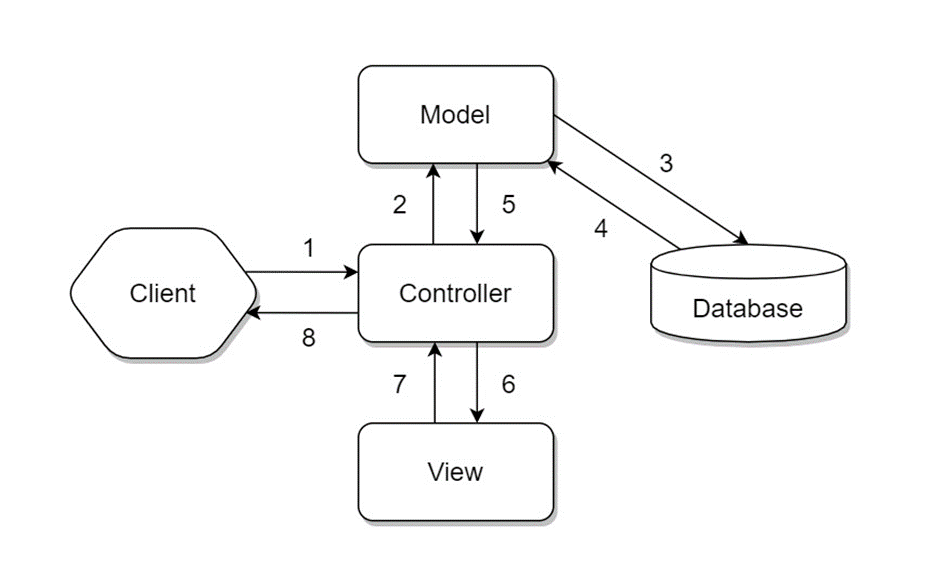
Hình 1.1 Tác giả

### 1.1.3 Các thành phần chính

* **Model:** Quản lý xử lý các dữ liệu.
* 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.
* **View:** Nơi hiển thị dữ liệu cho người dùng.
* 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.
* **Controller:** Điều khiển sự tương tác của hai thành phần Model và View.
* 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).

### 1.1.4 Sơ đồ giao tiếp

* MVC đại diện cho kiến trúc mà các nhà phát triển ứng dụng áp dụng vào dự án, chúng ta xem xét cách luồng dữ liệu ứng dụng hoạt động như thế nào.
* Có rất nhiều kịch bản cho luồng xử lý MVC trên ứng dụng web. Dưới đây là luồng xử lý căn bản và phổ biến nhất.



Hình 1.2 Sơ đồ giao tiếp

1. Client gửi yêu cầu đến server thông qua Controller, Controller sẽ tiếp nhận yêu cầu.

2. Controller sẽ xử lý dữ liệu đầu vào, và quyết đinh luồng đi tiếp theo của yêu cầu. Trả về kết quả hay tương tác với database để lấy dữ liệu. Nếu cần tương tác với Model để lấy dữ liệu, Controller sẽ gọi tới Model để lấy dữ liệu đầu ra. Nếu không, Controller sẽ trả về kết quả theo mũi tên số (8).

3. Model tương tác với Database để truy xuất dữ liệu phù hợp với yêu cầu.

4. Database trả về cho Model dữ liệu theo yêu cầu của Model.

5. Model trả về dữ liệu cho Controller xử lý.

6. Controller sẽ gọi đến View phù hợp với yêu cầu và kèm theo dữ liệu cho View. View chịu trách nhiệm hiển thị dữ liệu phù hợp với yêu cầu.

7. Sau khi xử lý hiển thị dữ liệu, View trả về cho Controller kết quả (HTML, XML hoặc JSON…).

8. Sau khi hoàn tất, Controller sẽ trả về kết quả cho Client.

## 1.2 Đánh giá kiến trúc của mô hình MVC

### 1.2.1 Ưu điểm của mô hình MVC

* Dễ dàng thao tác chỉnh sửa bộ phận Model hoạt động tách biệt với View đồng nghĩa với việc bạn có thể đưa ra các thay đổi, chỉnh sửa hoặc cập nhật dễ dàng ở từng bộ phận.
* Có thể chia nhiều developer làm việc cùng một lúc. Công việc của các developer sẽ không ảnh hưởng đến nhau
* Bảo trì code dễ dàng, dễ dàng mở rộng và phát triển.
* Nó giúp bạn tránh sự phức tạp bằng cách chia ứng dụng thành ba đơn vị Model, View và Contronller.
* 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.
* Thân thiện với công cụ tìm kiếm(SEO).

### 1.2.2 Nhược điểm của mô hình MVC

* Nhiều khung hoạt động đồng thời việc phân tách một tính năng thành ba bộ phận khác nhau dễ dẫn đến hiện tượng phân tán. Do đó, đòi hỏi các nhà phát triển phải duy trì tính nhất quán của nhiều bộ phận cùng một lúc.
* Tăng độ phức tạp và tính kém hiệu quả của dữ liệu.
* Đối với các mô hình, ứng dụng nhỏ thì việc triển khai sử dụng MVC có vẻ quá cồng kềnh.

### 1.2.3 Lợi ích khi sử dụng MVC

* Quy trình phát triển nhanh hơn.
* Khả năng cung cấp nhiều chế độ View.
* Trong mô hình MVC, bạn có thể tạo nhiều View cho chỉ một mô hình. Ngày nay, nhu cầu có thêm nhiều cách mới để truy cập ứng dụng và đang ngày càng tăng. Do đó, việc sử dụng MVC để phát triển chắc chắn là một giải pháp tuyệt vời.
* Hơn nữa, với phương pháp này, việc nhân bản code rất hạn chế. Vì nó tách biệt dữ liệu và logic nghiệp vụ khỏi màn hình.
* Các sửa đổi không ảnh hưởng đến toàn mô hình.
* Đối với bất kỳ ứng dụng web nào, người dùng có xu hướng thay đổi thường xuyên. Bạn có thể quan sát thông qua những thay đổi thường xuyên về màu sắc, font chữ, bố cục màn hình. Hay là thêm hỗ trợ thiết bị mới cho điện thoại hay máy tính bảng.
* Sửa đổi giao diện, font, hình ảnh, ... không ảnh hưởng đến toàn bộ mô hình MVC.
* Việc thêm một kiểu view mới trong MVC rất đơn giản. Vì phần Model không phụ thuộc vào phần View. Do đó, bất kỳ thay đổi nào trong Model sẽ không ảnh hưởng đến toàn bộ kiến trúc.
* MVC Model trả về dữ liệu mà không cần định dạng.
* MVC có thể trả về dữ liệu mà không cần áp dụng bất kỳ định dạng nào. Do đó, các thành phần giống nhau có thể được sử dụng với bất kỳ giao diện nào.
* Nền tảng MVC thân thiện với SEO.
* Nền tảng MVC hỗ trợ phát triển các trang web thân thiện với SEO. Bằng nền tảng này, bạn có thể dễ dàng phát triển các URL thân thiện với SEO để tạo ra nhiều lượt truy cập hơn.
* Những ngôn ngữ như JavaScript hay jQuery có thể được tích hợp với MVC. Từ đó phát triển nhiều ứng dụng web giàu tính năng, đặc biệt là với mô hình MVC trong Java

### 1.2.4 Tính ứng dụng của mô hình MVC

* Mô hình MVC được ứng dụng trong nhiều ngôn ngữ lập trình khác nhau, nhưng phổ biến nhất là ứng dụng ASP.NET MVC hay PHP MVC.
* MVC đang là mô hình được ứng dụng rất nhiều trong lập trình.
* Hệ thống MVC phát triển tốt sẽ cho phép phát triển front – end, back – end cùng trên hệ thống mà không có sự can thiệp, chia sẻ, chỉnh sửa các tập tin trong khi một hoặc hai bên vẫn đang làm việc.
* Việc sử dụng mô hình tương đối đơn giản. Chỉ cần hiểu rõ quy trình vận hành, nắm được các chức năng của từng bộ phận thì việc triển khai mô hình MVC tương đối dễ dàng.
* Trên đây là một số kiến thức cơ bản về mô hình MVC. Hi vọng bài viết trên đây của Mona Media sẽ giúp bạn hiểu rõ hơn về MVC và ứng dụng một cách hiệu quả.

## 1.3 So sánh mô hình MPV và MVC

* **Điểm khác biệt cơ bản của 2 mô hình MVC và MVP là sự khác nhau về vai trò của Presenter (MVP) và Controller (MVC), nó cũng dẫn đến sự khác nhau về vai trò của View. Trong MVP, sự hiện diện của Controller đã bị loại bỏ, thay vào đó việc sử lý các dữ liệu input được view đảm nhận và chuyển cho Presents khi có yêu cầu tương tác đến Model.**
* **Các mô hình MVP được dựa trên mô hình MVC. Các Presenter (MVP) và Controller (MVC) có vai trò gần tương tự nhau. Chúng có trách nhiệm giao tiếp giữa Model và View. Điều đó nói rằng, Controller (MVC) không quản lý Model và View như Presenter (MVP).**
* **Trong mô hình MVC, tầng View có phần thông minh và có thể lấy dữ liệu trực tiếp từ Model. Trong các mô hình MVP, View là hoàn toàn thụ động và lấy dữ liệu luôn được phân phối đến các View theo Presenter. Controller trong MVC cũng có thể được chia sẻ giữa nhiều View. Ngược lại trong MVP, View và Presenter có mỗi quan hệ một một, do đó một Presenter được gắn với 1 View.**
* **Những khác biệt về cách hoạt động làm mô hình MVP đảm bảo tính tách biệt giữa tầng dữ liệu và tầng hiển thị, và nó cũng làm tăng đáng kể khả năng kiểm thử của ứng dụng từ việc tách biệt đó.**

**Thu thập và Phân tích yêu cầu**

# Chương 2: Tài liệu kiến trúc

## 2.1 Giới thiệu

### 2.1.1 Mục tiêu

Tài liệu này cung cấp tổng quan kiến trúc toàn diện của hệ thống, sử dụng một số quan điểm kiến trúc khác nhau để mô tả các khía cạnh khác nhau của hệ thống. Nó nhằm nắm bắt và truyền đạt các quyết định kiến trúc quan trọng đã được thực hiện trên hệ thống ***Chia sẻ sách trực tuyến.***

### 2.1.2 Phạm vi

Mô tả phạm vi của tài liệu kiến trúc như tài liệu này cung cấp kiến trúc tổng quan cho hệ thống ***Chia sẻ sách trực tuyến.***

Tài liệu này được tạo ra từ các mô hình từ phần mềm StarUML.

### 2.1.3 Định nghĩa, các từ viết tắt

### 2.1.4 Tài liệu tham khảo

* Giao-trinh-PTTKHT-2019 Bộ môn HTTT
* Bài giảng môn kiến trúc phần mềm ICTU.

### 2.1.5 Đại diện kiến trúc

* Tài liệu này trình bày kiến trúc dưới dạng một loạt các khung nhìn: khung nhìn ca sử dụng, khung nhìn logic, khung nhìn quy trình và khung nhìn triển khai. Đây là các quan điểm về mô hình Ngôn ngữ mô hình thống nhất (UML) cơ bản được phát triển bằng StarUML

## 2.2 Các mục tiêu và ràng buộc kiến trúc

* Kỳ vọng về độ tin cậy:
* Tùy vào loại tài khoản người dùng khác nhau sẽ có những chức năng khác nhau, chặt chẽ về mặt quản lý
* Kỳ vọng về hiệu suất:
* Phần mềm chịu được cường độ cao, hoạt động liên tục 24h, cho phép ít nhất 4000 người truy cập cùng lúc.
* Các phản hồi của hệ thống không được quá 1 phút.
* Kỳ vọng về bảo trì:
* Bất cứ khi nào có sự cố xảy ra, thì phải được sửa chữa kịp thời
* Các yêu cầu về cài đặt và cầu hình:
* Phần mềm tương thích với bất các trình duyệt phổ biến tại Việt Nam hiện nay.
* Những yêu cầu khác:
* Bất kỳ yêu cầu hoặc tiêu chuẩn nào về quy định hoặc môi trường phải được hỗ trợ.

## 2.3 Tổng quan về kiến trúc

### 2.3.1 Khung nhìn kiến trúc

Mô hình hóa, triển khai và lập hồ sơ một hệ thống yêu cầu hệ thống phải được nhìn nhận từ các khía cạnh khác nhau. Do đó, kiến trúc của SNPKW sẽ được biểu diễn theo cách tiếp cận 5 dạng xem: Use Case View, Logical View, Process View, Physical View, và Deployment View. Dưới đây là mô tả ngắn gọn cho từng chế độ xem:

Logical View

Process View

Physical View

Deployment View

Use Case View: mục đích chính của dạng xem ca sử dụng là xác định các trình điều khiển của hệ thống, các yêu cầu hệ thống

Logical View: dạng xem này chứa bất kỳ định nghĩa hệ thống nào cũng như các biểu đồ lớp và đối tượng mô tả các dịch vụ mà hệ thống sẽ cũng cấp cho người dùng cuối của nó

Process View: sẽ hiển thị các quy trình hình thành cơ chế của hệ thống. Chúng sẽ được biểu diễn dưới dạng sơ đồ cộng tác, trình tự và hoạt động

Component View: bao gồm các thông số kỹ thuật của hệ thống và giao diện người dùng, ý nghĩa, các thành phần khác nhau tạo nên hệ thống

Deployment View: miêu tả các nút phần cứng của các hệ thống khác nhau hoạt động cùng nhau như cách mỗi nút phần cứng sẽ được cài đặt và triển khai.

## 2.4 Các khung nhìn kiến trúc

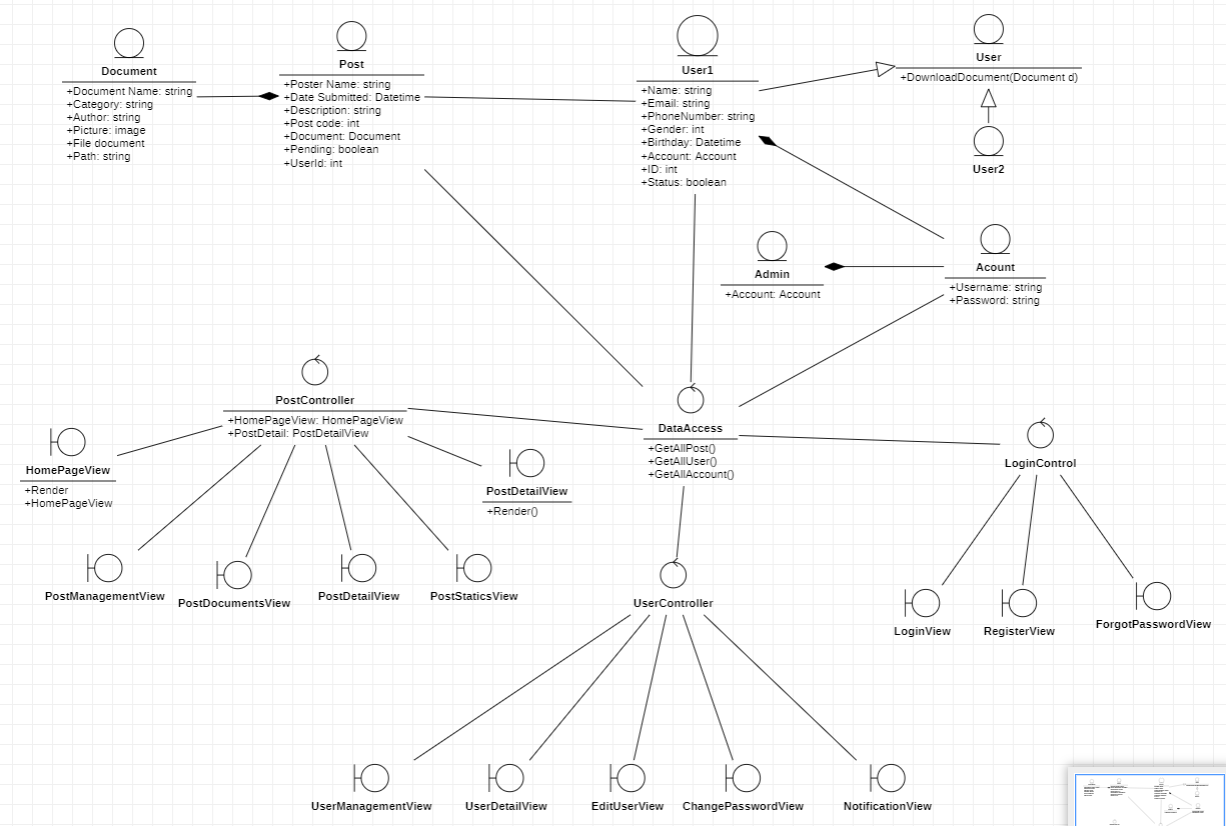
### 2.4.1 Khung nhìn Use Case

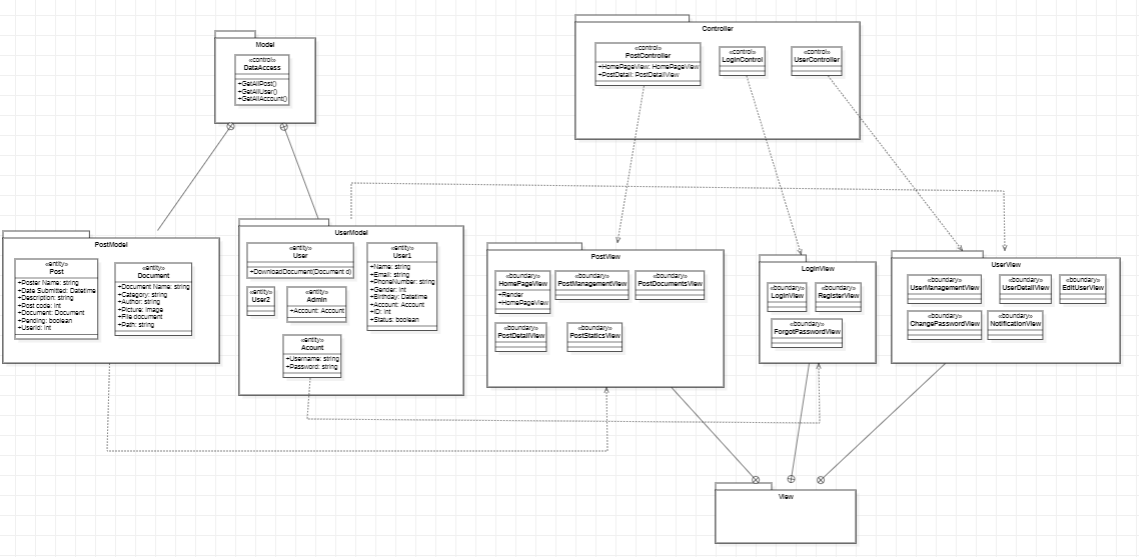
Biểu đồ use case



### 2.4.2 Khung nhìn logic

Biểu đồ lớp





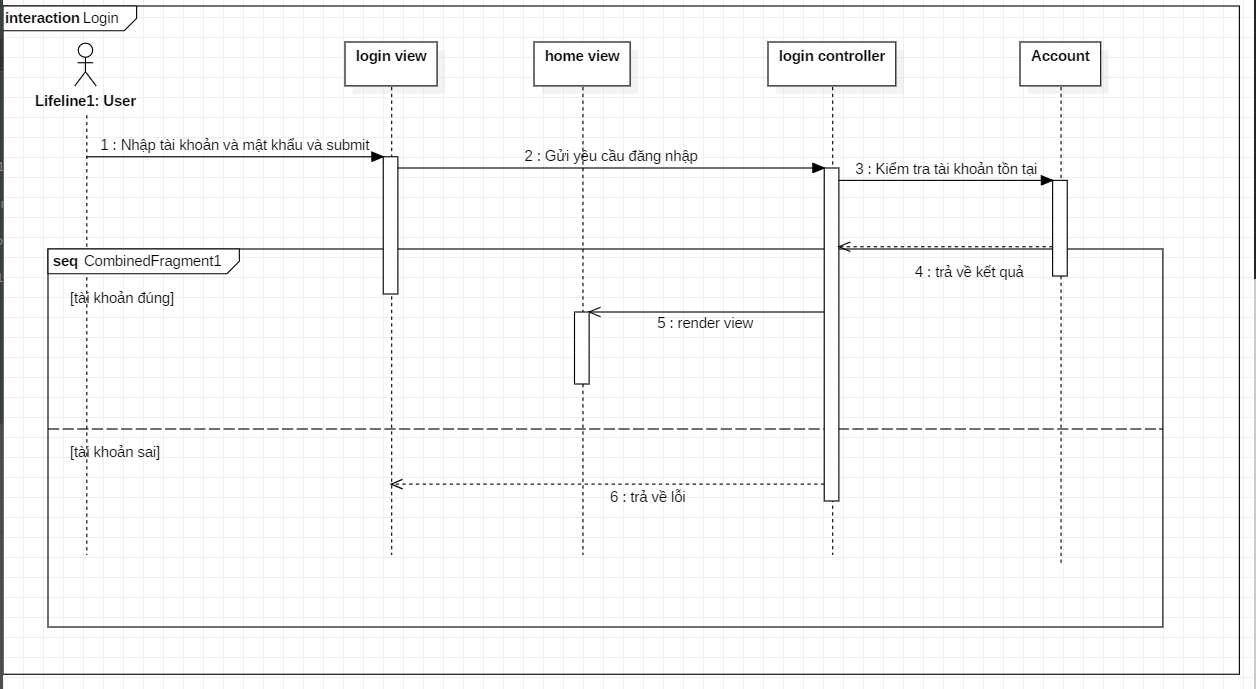
### 2.4.3 Khung nhìn quy trình

1. Đăng nhập

* Kịch bản

|  |  |
| --- | --- |
| Tên use case | Đăng nhập |
| Tác nhân chính | Admin |
| Mức |  |
| Tiền điều kiện | Admin ấn vào phần đăng nhập |
| Đảm bảo tối thiểu | Tài khoản phải chính xác |
| Đảm bảo thành công | Đăng nhập thành công |
| Kích hoạt | Admin chọn vào phần đăng nhập |
| Chuỗi sự kiện chính   1. Hệ thống hiện thị from đăng nhập 2. Admin sẽ nhập tên đăng nhập và pass 3. Hệ thống nhập thông tin vào SQL 4. Hệ thống hiển thị đăng nhập thành công | |
| Ngoại lệ  3.a Admin đăng nhập sai tên hoặc pass thì hệ thống sẽ hiển thị đăng nhập không thành công | |

* Biểu đồ trình tự

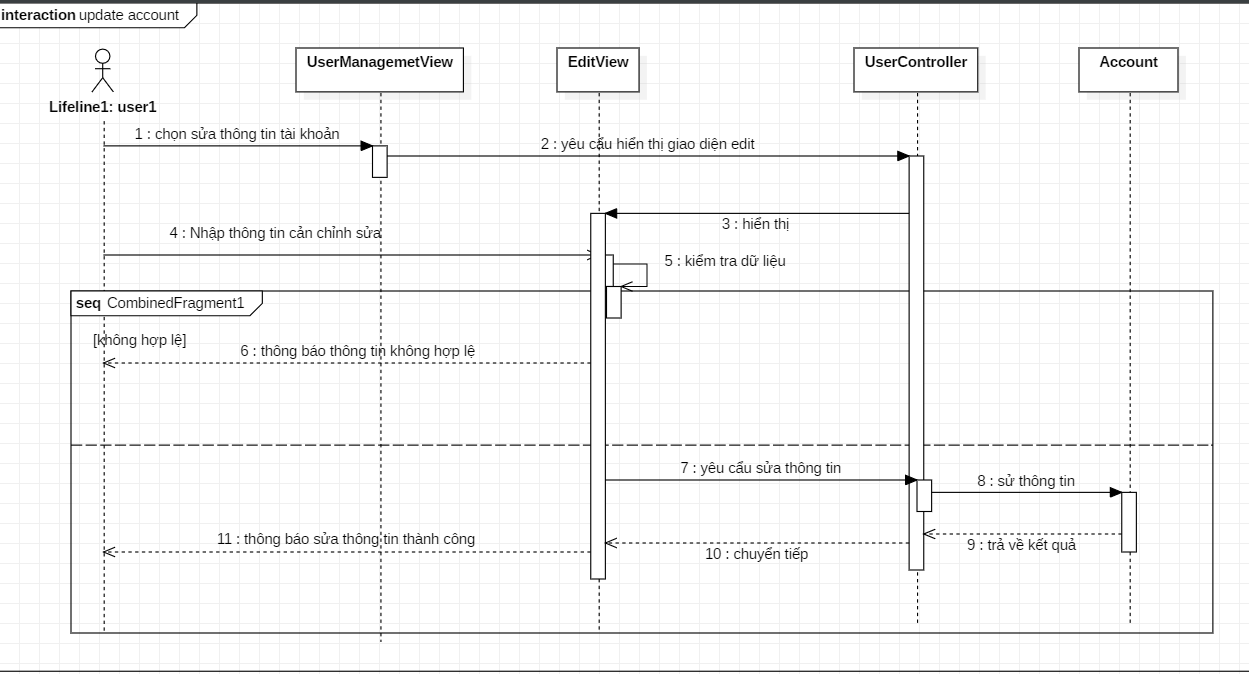


1. Sửa thông tin tài khoản

* Kịch bản

|  |  |
| --- | --- |
| Tên use case | Sửa thông tin tài khoản |
| Tác nhân chính | User 1 |
| Mức |  |
| Tiền điều kiện | User 1 đã đăng nhập vào hệ thống và ở trong trang quản lý tài khoản |
| Đảm bảo tối thiểu | Thông tin tài khoản được giữ nguyên |
| Đảm bảo thành công | Sửa thông tin tài khoản thành công |
| Kích hoạt | User 1 chọn chức năng sửa thông tin tài khoản |
| Chuỗi sự kiện chính   1. Hệ thống hiển thị form nhập liệu 2. User 1 nhập thông tin và ấn lưu 3. Hệ thống xác nhận thông tin tài khoản hợp lệ 4. Hệ thống cập nhập thông tin tài khoản trong csdl 5. Hệ thống thống báo sửa thông tin tài khoản thành công | |
| Ngoại lệ  3a. Hệ thống xác nhận thống tin tài khoản không hợp lệ  3a1. Hệ thống thông báo thông tin tài khoản không hợp lệ. | |

* Biểu đồ trình tự

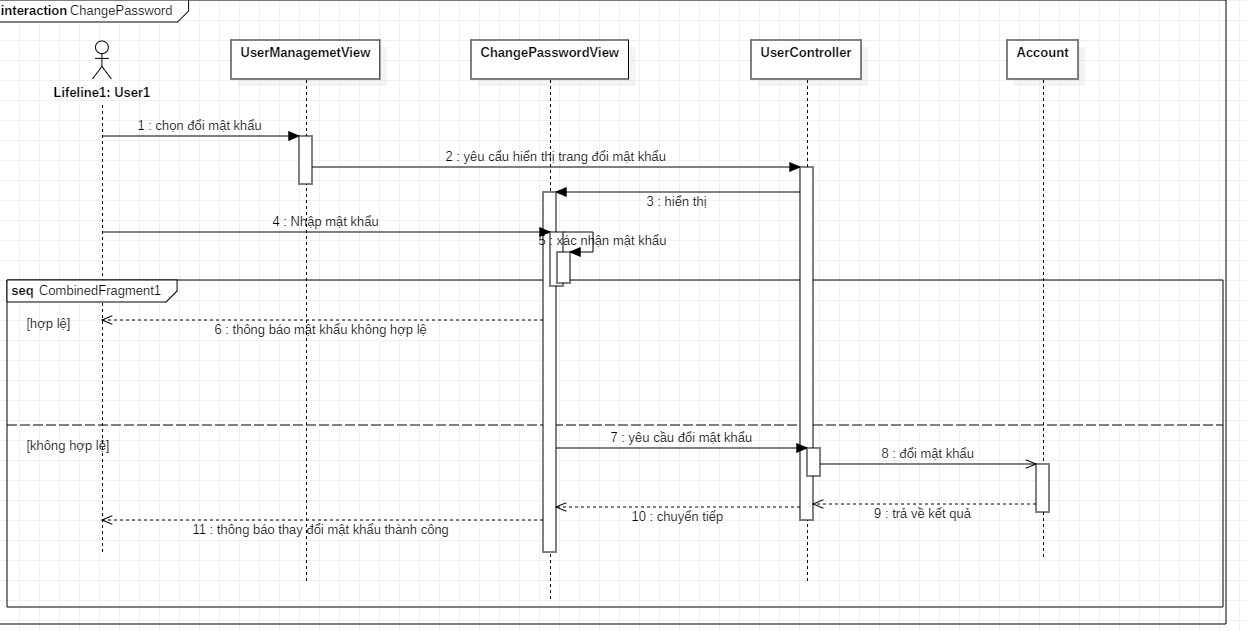


1. Đổi mật khẩu

* Kịch bản

|  |  |
| --- | --- |
| Tên use case | Đổi mật khẩu |
| Tác nhân chính | User 1 |
| Mức |  |
| Tiền điều kiện | User 1 đã đăng nhập vào hệ thống và ở trong trang quản lý tài khoản |
| Đảm bảo tối thiểu | Mật khẩu được giữ nguyên |
| Đảm bảo thành công | Đổi mật khẩu thành công |
| Kích hoạt | User 1 chọn chức năng đổi mật khẩu |
| Chuỗi sự kiện chính   1. Hệ thống hiển thị form nhập liệu 2. User 1 nhập thông tin và ấn lưu 3. Hệ thống xác nhận mật khẩu hợp lệ 4. Hệ thống cập nhập thông tin tài khoản trong csdl 5. Hệ thống thống báo sửa mật khẩu thành công | |
| Ngoại lệ  3a. Hệ thống xác nhận mật khẩu vừa nhập không hợp lệ.  3a1. Hệ thống thông báo mật khẩu không hợp lệ. | |

* Biểu đồ trình tự

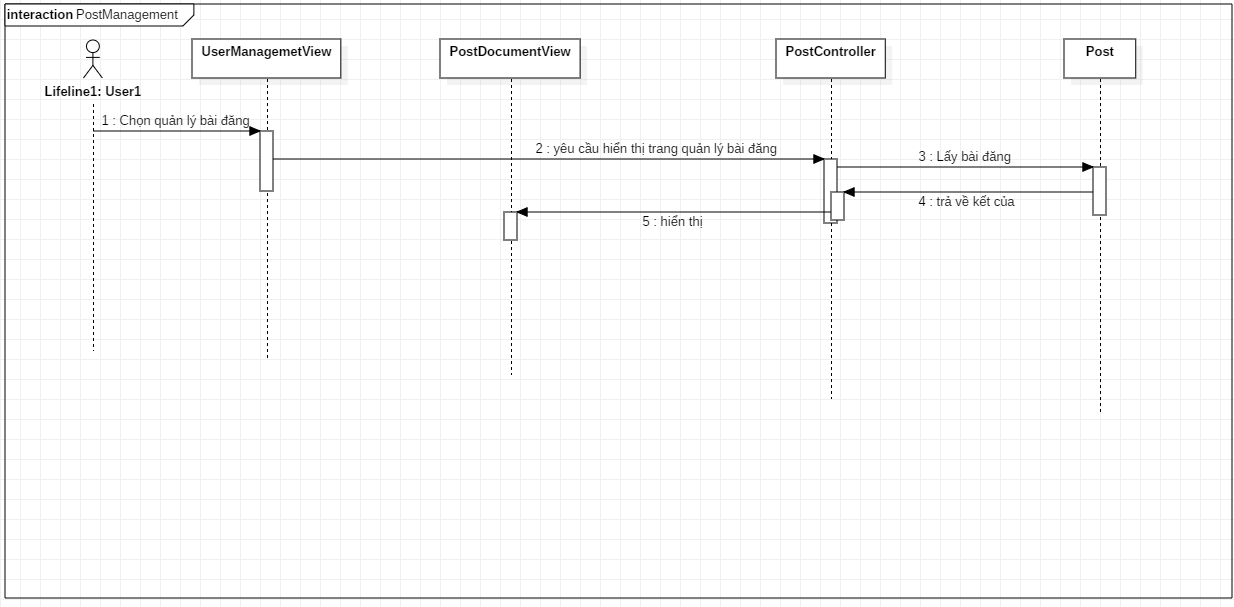


1. Quản lý bài đăng

* Kịch bản

|  |  |
| --- | --- |
| Tên use case | Quản lý bài đăng |
| Tác nhân chính | User 1 |
| Mức |  |
| Tiền điều kiện | User 1 đã đăng nhập vào hệ thống |
| Đảm bảo tối thiểu | Không có |
| Đảm bảo thành công | Hiển thị danh sách những bài đã đăng, đã hủy và đang chờ duyệt |
| Kích hoạt | User 1 chọn chức năng quản lý bài đăng |
| Chuỗi sự kiện chính   1. Hệ thống hiển thị danh sách những bài đã đăng, đã hủy và đang chờ duyệt | |
|  | |

* Biểu đồ trình tự

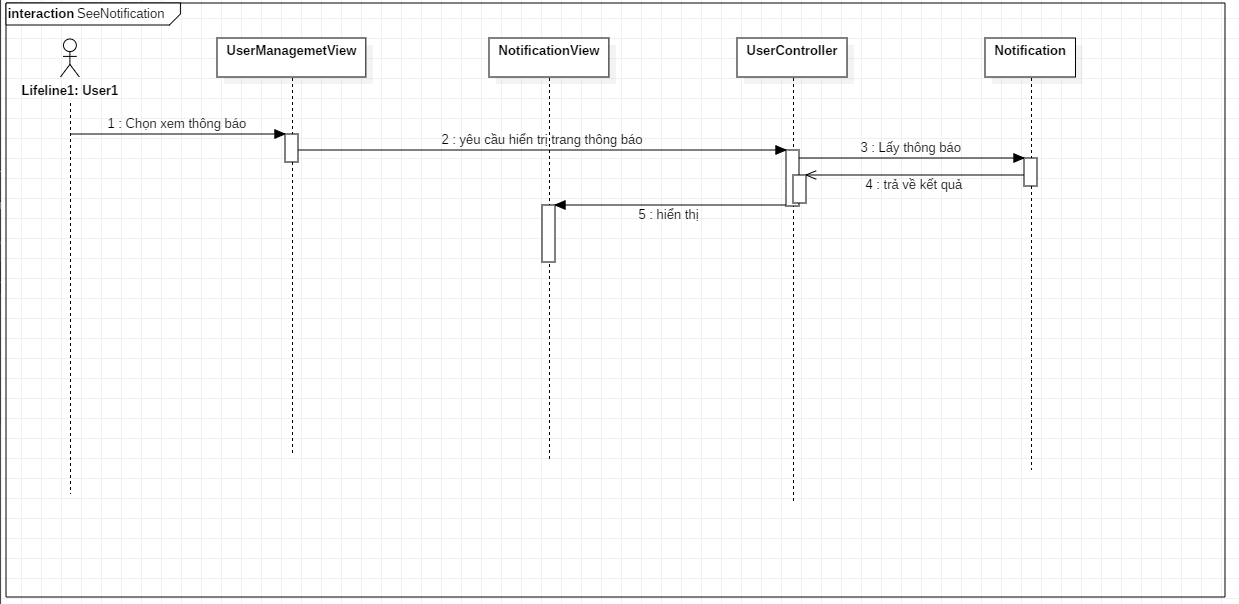


1. Xem thông báo

* Kịch bản

|  |  |
| --- | --- |
| Tên use case | Xem thông báo |
| Tác nhân chính | User 1 |
| Mức |  |
| Tiền điều kiện | User 1 đã đăng nhập vào hệ thống |
| Đảm bảo tối thiểu | Không có |
| Đảm bảo thành công | Hiển thị ra danh sách tất cả những thông báo |
| Kích hoạt | User 1 chọn chức năng xem thông báo |
| Chuỗi sự kiện chính   1. Hệ thống hiển thị ra danh sách tất cả những thông báo | |
|  | |

* Biểu đồ trình tự

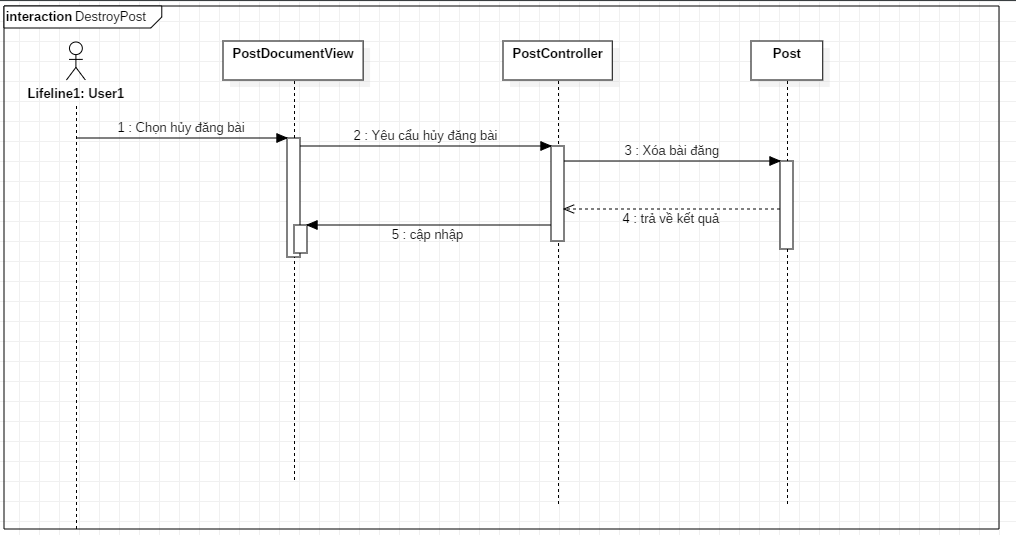


1. Hủy đăng bài

* Kịch bản

|  |  |
| --- | --- |
| Tên use case | Hủy đăng bài |
| Tác nhân chính | User 1 |
| Mức |  |
| Tiền điều kiện | User 1 đã đăng nhập vào hệ thống và ở trong trang quản lý bài đăng |
| Đảm bảo tối thiểu | bài đăng tiếp tục trong trạng thái chờ duyệt |
| Đảm bảo thành công | Hủy đăng bài thành công |
| Kích hoạt | User1 chọn chức năng xem những bài chờ duyệt |
| Chuỗi sự kiện chính   1. Hệ thống hiển thị ra danh sách những bài đang chờ duyệt 2. User 1 chọn hủy đăng bài. 3. Hệ thống hiển thị ra hộp thoại xác nhận hủy 4. User 1 chọn Ok 5. Hệ thống thông báo hủy đăng bài thành công. | |
|  | |

* Biểu đồ trình tự



### 2.4.4 Khung nhìn triển khai

Biểu đồ thành phần

## 2.5 Kích cỡ và hiệu suất

## 2.6 Chất lượng

# Chương 3: Demo chương trình