**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH**

**Ma Hoàng Hải Nguyên**

**Sử Dụng Spring MVC Framework**

**Xây Dựng Mạng Xã Hội Từ Thiện**

**Đồ Án Tốt Nghiệp**

**NGÀNH Khoa Học Máy Tính**

**TP. HỒ CHÍ MINH, Năm 2021BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH**

**Ma Hoàng Hải Nguyên**

**Sử Dụng Spring MVC Framework**

**Xây Dựng Mạng Xã Hội Từ Thiện**

**Mã số sinh viên: 1851010084**

**Đồ Án Tốt Nghiệp**

**NGÀNH HỌC: Khoa Học Máy Tính**

**Giảng viên hướng dẫn: Thạc Sĩ Dương Hữu Thành**

**TP. HỒ CHÍ MINH, Năm 2021**

TRƯỜNG ĐẠI HỌC MỞ **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**

Mẫu số: 5

THÀNH PHỐ HỒ CHÍ MINH **Độc lập – Tự do – Hạnh phúc**

**KHOA CÔNG NGHỆ THÔNG TIN**

**\_\_\_\_\_\_**

**GIẤY XÁC NHẬN**

Tôi tên là: Ma Hoàng Hải Nguyên

Ngày sinh: 16/10/2000 Nơi sinh: TP Hồ Chí Minh

Chuyên ngành: KHMT Mã sinh viên: 1851010084

Tôi đồng ý cung cấp toàn văn thông tin đồ án/ khóa luận tốt nghiệp hợp lệ về bản quyền cho Thư viện Trường Đại học Mở Thành phố Hồ Chí Minh. Thư viện Trường Đại học Mở Thành phố Hồ Chí Minh sẽ kết nối toàn văn thông tin đồ án/ khóa luận tốt nghiệp vào hệ thống thông tin khoa học của Sở Khoa học và Công nghệ Thành phố Hồ Chí Minh.

Ký tên

*(Ghi rõ họ và tên)*

…………………………

**Ý KIẾN CHO PHÉP BẢO VỆ ĐỒ ÁN/ KHÓA LUẬN TỐT NGHIỆP**

Mẫu số: 6

**CỦA GIẢNG VIÊN HƯỚNG DẪN**

**Giảng viên hướng dẫn:**

**Sinh viên thực hiện: Lớp:**

**Ngày sinh: Nơi sinh:**

**Tên đề tài:**

**Ý kiến của giảng viên hướng dẫn về việc cho phép sinh viên được bảo vệ đồ án/ khóa luận trước Hội đồng:**

*Thành phố Hồ Chí Minh, ngày … tháng … năm ……*

Người nhận xét

…………………………

LỜI CẢM ƠN

Trước hết, em xin gửi lời cảm ơn chân thành đến thầy Dương Hữu Thành. Người đã hướng dẫn chi tiết , giúp đỡ em chỉnh sửa và giải đáp các thắc mắc, khó khăn của em trong lúc thực hiện đồ án tốt nghiệp của mình.

Em cũng mong được bày tỏ tình cảm và lời cảm ơn của mình đến các thầy cô giáo Khoa Công Nghệ Thông Tin Trường Đại Học Mở Thành Phố Hồ Chí Minh đã chỉ dạy , giải đáp các thắc mắc của em không chỉ về mặt kiến thức mà còn về mặt đạo đức , đối nhân xử thế của một người sinh viên, một công nhân phục vụ tổ quốc . Cảm ơn các thầy cô đã giúp đỡ em củng cố kiến thức để thực hiện đồ án tốt nghiệp của mình.

Em cũng xin gửi lời cảm ơn đến thầy Tiến Sĩ Lê Xuân Trường – Trưởng Khoa Công Nghệ Thông Tin, ban giám hiệu nhà trường , các phòng ban, thủ thư đã tạo điều kiện cung cấp cho em môi trường học tập tốt nhất trong suốt khoảng thời gian học tại trường.

Trong quá trình học tập nghiên cứu phát triển đồ án tốt nghiệp em không thể tránh khỏi các thiếu sót các sai sót thiếu hiểu biết. Em rất mong được sự góp ý quý báu của thầy cô chấm đồ án tốt nghiệp cũng như tất cả quý thầy cô giáo để kết quả đồ án được hoàn thiện hơn.

Một lần nữa em xin chân thành cảm ơn.

Tp Hồ Chí Minh , Ngày 20 Tháng 8 Năm 2021

Sinh Viên

Ma Hoàng Hải Nguyên

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

TÓM TẮT KHÓA LUẬN

Mạng Xã Hội Từ Thiện được xây dựng với mục đích kết nối các người dùng trên toàn thế giới bất kể khoảng cách và vùng miền. Ai cũng có thể tương tác trên mạng xã hội để đăng các bài viết về sản phẩm mình có quyên góp cho người có hoàn cảnh khó khăn hay đăng các bài viết ủng hộ hoạt động từ thiện quyên góp các sản phẩm cho các người dùng có nhu cầu muốn có nhưng không có kinh phí tài chính để sở hữu.

Toàn bộ giao diện của mạng xã hội được sử dụng công nghệ Front-end FrameWork Bootstrap giúp cho việc xây dựng 1 giao diện đẹp thân thiện với người dùng dễ dàng.

Giúp cho Ứng dụng hoạt động tốt trên và hiển thị ổn định trên mọi hệ điều hành với khả năng Reponsive của Bootstrap. Ứng Dụng được xây dựng bằng ngôn ngữ Lập Trình Java sử dụng Spring FrameWork Mvc , một mô hình nổi tiếng dùng để phát triển các ứng dụng web.

Ứng dụng web sử dụng công nghệ Spring Security của Spring MVC giúp cho việc phân quyền giữa người quản trị ADMIN ( quản trị viên ) và USER ( người dùng ) được xác thực họ có thể vào trang nào trong ứng dụng web. Cũng như bảo vệ trang web khỏi các request ẩn danh, các request chưa đăng nhập không qua xác nhận người dùng có thể vào tương tác với ứng dụng web một cách không chính thống.

Ứng dụng Web mang lại cho người dùng các chức năng như xem bài viết của Mạng Xã Hội Từ Thiện, like và comment trên bài viết của người khác.

Người dùng có thể nhận các thông báo ( notification ) khi có người dùng khác like comment bài viết của mình.

Người dùng có thể Đấu Giá Các Sản Phẩm Của Các Bài Viết Trên Mạng Xã Hội Từ Thiện.

Người quản trị có thể xem thống kê dữ liệu người dùng và các thông tin bài viết , tìm kiếm bài viết.

Mạng Xã Hội Cũng có chức năng Report Người Dùng khi có các hành vi không chuẩn mực đối với cộng đồng.

TÓM TẮT KHÓA LUẬN

(PHIÊN BẢN TIẾNG ANH NẾU CÓ)

**MỤC LỤC**

[LỜI CẢM ƠN 3](#_Toc84844273)

[NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN 4](#_Toc84844274)

[TÓM TẮT KHÓA LUẬN 5](#_Toc84844275)

[TÓM TẮT KHÓA LUẬN 6](#_Toc84844276)

[(PHIÊN BẢN TIẾNG ANH NẾU CÓ) 6](#_Toc84844277)

[DANH MỤC TỪ VIẾT TẮT 10](#_Toc84844278)

[DANH MỤC HÌNH VẼ 12](#_Toc84844279)

[DANH MỤC BẢNG 14](#_Toc84844280)

[Chương 1. Tổng quan,đặt vấn đề,giới thiệu về đề tài 15](#_Toc84844281)

[1.1. Nội dung chương 15](#_Toc84844282)

[1.2. Giới Thiệu Đề Tài 15](#_Toc84844283)

[1.3. Chức Năng của hệ thống 16](#_Toc84844284)

[1.4. Lý Do Chọn Đề Tài 16](#_Toc84844285)

[1.5. Mục Tiêu Nghiên Cứu 16](#_Toc84844286)

[1.6. Bố Cục Đề Tài 17](#_Toc84844287)

[Chương 2. Giới Thiệu Tổng Quan 18](#_Toc84844288)

[2.1. Nội dung chương 18](#_Toc84844289)

[2.2. Giới Thiệu Ứng Dụng Web 18](#_Toc84844290)

[2.3. Giới thiệu Mô Hình Kiến Trúc MVC 18](#_Toc84844291)

[2.3.1. Ưu & nhược điểm của MVC 19](#_Toc84844292)

[2.4. Giới Thiệu Spring MVC FrameWork 20](#_Toc84844293)

[2.4.1. Các Đặc trưng nổi bật của Spring MVC 20](#_Toc84844294)

[2.5. Các Thành Phần tạo nên Ứng Dụng Web Spring MVC 26](#_Toc84844295)

[2.5.1. Giới Thiệu về TomCat Server 26](#_Toc84844296)

[2.5.2. Java Development Kit (JDK ) 27](#_Toc84844297)

[2.5.3. Giới thiệu về NETBEAN IDE 28](#_Toc84844298)

[2.5.4. Giới Thiệu về Hibernate theo giải pháp ORM 28](#_Toc84844299)

[2.5.5. Bootstrap 35](#_Toc84844300)

[2.5.6. MySQL Workbench 37](#_Toc84844301)

[2.5.7. Giới Thiệu JDBC 38](#_Toc84844302)

[2.6. Xây Dựng Chương Trình Đầu Tiên sử dụng Spring MVC 39](#_Toc84844303)

[2.6.1. Cài đặt bộ JDK 39](#_Toc84844304)

[2.6.2. Cài đặt biến môi trường (Environment Variables) trong Java : PATH và CLASSPATH 40](#_Toc84844305)

[2.6.3. Cài đặt Apache Tomcat Server 42](#_Toc84844306)

[2.6.4. Tạo Project Maven mới. 42](#_Toc84844307)

[Chương 3. Hệ thống mạng xã hội từ thiện 62](#_Toc84844308)

[3.1. Nội dung chương 62](#_Toc84844309)

[3.2. Giới thiệu hệ thống 63](#_Toc84844310)

[3.3. Phân tích thiết kế hệ thống 64](#_Toc84844311)

[3.3.1. Các yêu cầu chức năng của hệ thống. 64](#_Toc84844312)

[3.3.2. Các yêu cầu phi chức năng của hệ thống 64](#_Toc84844313)

[3.3.3. Các ký hiệu cơ bản trong UML 65](#_Toc84844314)

[3.3.4. Biểu đồ trường hợp sử dụng(Use Case diagram) 66](#_Toc84844315)

[3.3.5. Lược đồ tuần tự(SEQUENCE DIAGRAM) 83](#_Toc84844316)

[3.3.6. Sơ đồ lớp(Class Diagram) 93](#_Toc84844317)

[3.3.7. Mô hình khái niệm 95](#_Toc84844318)

[3.3.8. Mô Hình Hóa Tương tác đối tượng 99](#_Toc84844319)

[3.3.9. Biểu đồ trình tự 99](#_Toc84844320)

[3.4. Thiết kế cơ sở dữ liệu 110](#_Toc84844321)

[3.4.1. Cấu trúc và mối quan hệ của các bảng 110](#_Toc84844322)

[3.5. Hiện thực hệ thống 110](#_Toc84844323)

[3.6. Chủ đề cấp độ 2 111](#_Toc84844324)

[3.6.1. Chủ đề cấp độ 3 111](#_Toc84844325)

[Chương 4. TÊN CHƯƠNG 3 112](#_Toc84844326)

[4.1. Chủ đề cấp độ 2 112](#_Toc84844327)

[4.1.1. Chủ đề cấp độ 3 112](#_Toc84844328)

[4.2. Chủ đề cấp độ 2 112](#_Toc84844329)

[TÀI LIỆU THAM KHẢO 113](#_Toc84844330)

[PHỤ LỤC 114](#_Toc84844331)

DANH MỤC TỪ VIẾT TẮT

|  |  |
| --- | --- |
| MVC | MODEL – VIEW - CONTROLLER |
| IDE | Integrated Development Environment |
| ORM | Object-Relational Mapping |
| JDBC | Java Database Connectivity |
| JSF | JavaServer Faces |
| JSP | JavaServer Pages |
| IOC | Inversion of Control |
| DI | Dependency Injection |
| JSTL | Java Standard Tag Library |
| URL | Uniform Resource Locator |
| HTML | Hypertext Markup Language |
| XML | Extensible Markup Language |
| JEL | Java Expression Language |
| JVM | Java Virtual Machine |
| JRE | Java Runtime Environment |
| JDK | Java Development Kit |
| POJO | Plain Old Java Object |
| JPA | Java Persistence API |
| API | Application Programming Interface |
| HQL | Hibernate Query Language |
| JTA | Java Transaction API |
| IE | Internet Explorer |

DANH MỤC HÌNH VẼ

[Hình 2.1: Mô Hình Kiến Trúc MVC 18](#_Toc84878786)

[Hình 2.2: Mô Hình Front Controller Design Pattern 21](#_Toc84878787)

[Hình 2.3 Mô Hình Dispatcher Servlet 22](#_Toc84878788)

[Hình 2.4 Cấu trúc Đường Dẫn URL của một Ứng Dụng Web 23](#_Toc84878789)

[Hình 2.6 Apache Tomcat Server 26](#_Toc84878790)

[Hình 2.7 Java Development Kit 27](#_Toc84878791)

[Hình 2.8 Tương Tác Giữa Database và Persistence Layer Với Hibernate/ORM 29](#_Toc84878792)

[Hình 2.9 Kỹ Thuật ORM Mapping Các Lớp Hướng Đối Tượng vào các table 29](#_Toc84878793)

[Hình 2.10 Kiến trúc Hibernate 30](#_Toc84878794)

[Hình 2.11 Bootstrap 36](#_Toc84878795)

[Hình 2.12 Kiến trúc JDBC 38](#_Toc84878796)

[Hình 2.13 Các gói JDK 40](#_Toc84878797)

[Hình 2.14 Cài Đặt Project Maven 43](#_Toc84878798)

[Hình 2.15Thiết Lập TomCat Server 43](#_Toc84878799)

[Hình 2.16 Chọn Server 44](#_Toc84878800)

[Hình 2.17 Cấu Hình Server 44](#_Toc84878801)

[Hình 2.18 Cấu trúc project vừa mới tạo 45](#_Toc84878802)

[Hình 2.19 Package Controllers 48](#_Toc84878803)

[Hình 2.20 Chương Trình Đầu Tiên 49](#_Toc84878804)

[Hình 3.3.4.3.1 Biểu đồ trường hợp sử dụng của người dùng (Use Case Diagram) 68](#_Toc84878805)

[Hình 3.3.4.3.2Biểu đồ trường hợp sử dụng của quản trị viên(Use Case Diagram) 69](#_Toc84878806)

[Hình 3.3.5.1 Sequence Diagram Đăng ký tài khoản người dùng 83](#_Toc84878807)

[Hình 3.3.5.2 Sequence Diagram Đăng nhập 84](#_Toc84878808)

[Hình 3.3.5.3 Sequence Diagram Đăng Xuất 84](#_Toc84878809)

[Hình 3.3.5.4 Sequence Diagram Tìm kiếm bài viết 85](#_Toc84878810)

[Hình 3.3.5.5 Sequence Diagram Trang chủ hiển thị 85](#_Toc84878811)

[Hình 3.3.5.6 Sequence Diagram chức năng like bài viết 86](#_Toc84878812)

[Hình 3.3.5.7 Sequence Diagram Chức năng bình luận bài viết 87](#_Toc84878813)

[Hình 3.3.5.8 Sequence Diagram Trang thông tin cá nhân 87](#_Toc84878814)

[Hình 3.3.5.9 Sequence Diagram Trang đăng bài viết mới 88](#_Toc84878815)

[Hình 3.3.5.10 Sequence Diagram Trang cập nhật bài viết 88](#_Toc84878816)

[Hình 3.3.5.11 Sequence Diagram Xóa Bài Viết 89](#_Toc84878817)

[Hình 3.3.5.12 Báo cáo bài viết Sequence Diagram 89](#_Toc84878818)

[Hình 3.3.5.13 Báo cáo bình luận Sequence Diagram 90](#_Toc84878819)

[Hình 3.3.5.14 Người tổ chức đấu giá chọn người chiến thắng 90](#_Toc84878820)

[Hình 3.3.5.15 Người tham gia đấu giá 91](#_Toc84878821)

[Hình 3.3.5.16 Hệ thống thông báo Sequence Diagram 91](#_Toc84878822)

[Hình 3.3.5.17 Thống kê dữ liệu Sequence Diagram 92](#_Toc84878823)

[Hình 3.3.5.18 Sequence Diagram Báo Cáo bài viết,bình luận 92](#_Toc84878824)

[Hình 3.3.6.1 Class Diagram Đăng ký tài khoản người dùng 93](#_Toc84878825)

[Hình 3.3.6.2 Class Diagram Đăng nhập 93](#_Toc84878826)

[Hình 3.3.6.3 Class Diagram Đăng Xuất 94](#_Toc84878827)

[Hình 3.3.6.4 Class Diagram Tim Kiếm Bài Viết 94](#_Toc84878828)

[Hình 3.3.6.5 Class Diagram Xử lý hiển thị trang chủ 95](#_Toc84878829)

[Hình 3.3.6.6 Class Diagram Chức năng Like Bài Viết 95](#_Toc84878830)

[Hình 3.3.6.7 Class Diagram Chức năng bình luận bài viết 96](#_Toc84878831)

[Hình 3.3.6.8 Class Diagram Hiển thị thông tin cá nhân 96](#_Toc84878832)

[Hình 3.3.6.9 Class Diagram Đăng bài viết mới 97](#_Toc84878833)

[Hình 3.3.6.10 Class Diagram cập nhật bài viết 97](#_Toc84878834)

[Hình 3.3.6.11 Class Diagram Xóa Bài Viết 98](#_Toc84878835)

[Hình 3.3.6.12A Class Diagram Báo cáo bài viết 98](#_Toc84878836)

[Hình 3.3.6.12B Class Diagram Báo cáo bình luận 99](#_Toc84878837)

[Hình 3.3.6.13 Class Diagram Chức năng đấu giá 99](#_Toc84878838)

[Hình 3.3.6.14 Class Diagram hệ thống thông báo 100](#_Toc84878839)

[Hình 3.3.6.15 Class Diagram thống kê dữ liệu 100](#_Toc84878840)

[Hình 3.3.6.16 Class Diagram phê duyệt báo cáo bài viết bình luận 101](#_Toc84878841)

[Hình 3.3.6.17 Class Diagram Biểu đồ mối quan hệ giữa các lớp 102](#_Toc84878842)

DANH MỤC BẢNG

[Bảng 3.3.3.1 Các ký hiệu cơ bản trong UML 67](#_Toc84878301)

[Bảng 2.1: Tên bảng 1 103](#_Toc84878302)

[Bảng 2.2: Tên bảng 2 103](#_Toc84878303)

[Bảng 2.3: Tên bảng 3 103](#_Toc84878304)

# 

# Tổng quan,đặt vấn đề,giới thiệu về đề tài

## Nội dung chương

Chương một sẽ trình bày và giới thiệu tổng quan về đồ án, các chức năng mà ứng dụng sẽ làm được, em cũng sẽ giới thiệu một chút về nền phát triển của mạng xã hội và bối cảnh hiện nay đối với mạng xã hội có sức ảnh hưởng như thế nào đến đời sống con người trong thế giới hiện đại.

## Giới Thiệu Đề Tài

Với quá trình phát triển đi lên của ứng dụng Công Nghệ Thông Tin trong đời sống thường nhật của chúng ta, khi công nghệ thông tin, điện tử kỹ thuật số đóng vai trò quan trọng không chỉ trong mục đích thương mại, quảng bá, truyền thông các thông tin mà còn trong mục đích giải trí, kết nối mọi người đến với nhau trên khắp thế giới chỉ bằng chiếc điện thoại hay máy tính để bàn. Việc liên lạc và tìm kiếm thông tin không còn bị cản trở về mặt khoảng cách. Tất cả các nước trên thế giới đều đang chạy đua theo xu hướng hiện đại. Sử dụng Công Nghệ Thông Tin và Kỹ Thuật Số trong việc giao tiếp như mạng xã hội: Facebook, Twitter… Các Ứng dụng Bán Hàng có chức năng thanh toán xuyên lục địa áp dụng không chỉ trong nước và ngoài nước như Lazada,Shopee,Amazon. Các cổng thông tin thanh toán khắp quốc gia như Paypal,….

Mạng Xã Hội Từ Thiện cung cấp hệ thống các bài viết để người dùng có thể cung cấp các sản phẩm từ thiện cho những người dùng khác có hoàn cảnh khó khăn không đủ nhu yếu phẩm để phục vụ đời sống ăn uống của bản thân gia đình họ. Với việc phát triển hệ thống này giúp mang lại 1 nơi để mọi người có thể chia sẽ giúp đỡ nhau, nâng cao đời sống vật chất và tinh thần của họ. Tuy nhiên, với những mục đích tốt đẹp mang lại của Mạng Xã Hội, trong quá trình phát triển chúng em vẫn còn gặp nhiều sai sót, khó khăn, nhược điểm của hệ thống cần được khắc phục về mặt chuyên môn.

Với mong muốn phát triển hệ thống mạng xã hội chia sẽ từ thiện, cũng như là bước khởi đầu cho hành trình trong sự nghiệp, chúng em mong muốn mang lại một hệ thống phát triển hoạt động đúng như yêu cầu và tiêu chí của một mạng xã hội nên có.

## Chức Năng của hệ thống

Mạng xã hội từ thiện sẽ có đủ các chức năng cơ bản mà một hệ thống mạng xã hội nên có:

Đăng nhập,Đăng ký.

Trang chính hiển thị các bài viết mới.

Trang Thông tin người dùng , người dùng có thể chỉnh sửa thông tin cá nhân

Đăng bài viết mới , cập nhật xóa bài viết.

Người dùng có thể like và comment bài viết của người khác và bình luận của người khác .

Khi người tương tác các chức năng trên với bài viết của người khác thì họ sẽ nhận thông báo mới về các hoạt động của người dùng khác tương tác với bình luận và bài viết của họ.

Trang Đấu giá: là trang người dùng có thể đấu giá các sản phẩm quyên góp trên bài viết của người khác đăng và người đăng có thể quyết định người đấu giá nào là người chiến thắng.

Các chức năng phụ : Report người dùng, Thống kê nội dung,like,comment và số lượng bài viết ( chỉ dành cho admin).

## Lý Do Chọn Đề Tài

Lý do em chọn đề tài liên quan đến mạng xã hội vì nó là một thứ ứng dụng được sử dụng rất phổ biến trong xã hội ngày nay, cùng với tình hình dịch bệnh khó khăn,em muốn phát triển một ứng dụng web có thể giúp đỡ những người có hoàn cảnh khó khăn, và nơi để họ có thể tương tác và trao đổi giúp cải thiện đời sống và tạo điều kiện cho người làm từ thiện có thể giúp đỡ những người khác bất kể khoảng cách gần xa.

## Mục Tiêu Nghiên Cứu

Trong tương lai, ngoài các chức năng sẵn có sẽ phát triển trong đồ án, em sẽ cố gắng cải tiển để tích hợp chức năng nhắn tin box chat theo thời gian thực,tạo nên sự tiện nghi và tăng khả năng tương tác giữa các người dùng với nhau,cũng như cải thiện tính bảo mật của chương trình và tích hợp một số chức năng thanh toán tiền mặt trực tuyến cho người dùng,cải thiện giao diện ứng dụng đẹp hơn và bắt mắt hơn.

## Bố Cục Đề Tài

Chương một: Giới Thiệu Tổng Quan,đặt vấn đề,ý tưởng phát triển hệ thống.

Chương hai: Cơ Sở Lý Thuyết,các công nghệ sử dụng trong chương trình,chương trình demo đầu tiên.

Chương ba: Xây dựng hệ thống,giao diện phân tích thiết kế,thiết kế cơ sở dữ liệu,các chức năng hệ thống minh họa giao diện.

Chương bốn:Kết Luận Và Hướng Phát Triển của hệ thống.

Tài Liệu Tham Khảo trình bày các trích dẫn và các tài liệu được sử dụng phục vụ cho việc phát triển đồ án và Phụ lục minh họa các dòng code phát triển hệ thống.

# Giới Thiệu Tổng Quan

## Nội dung chương

Chương hai sẽ trình bay tổng quan về cơ sở lý thuyết, các công nghệ sẽ được sử dụng trong đồ án này, các đoạn code minh họa về hệ thống, cấu hình nên ứng dụng và chương trình demo sử dụng công nghệ này để phát triển.

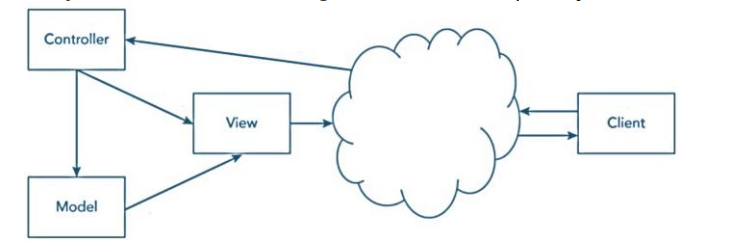
## Giới Thiệu Ứng Dụng Web

Với Nhu Cầu sử dụng mạng xã hội và tốt độ phát triển chóng mặt của Internet , theo đó các hoạt động từ thiện và tương tác với người có hoàn cảnh khó khăn trên mạng xã hội cũng vì thế mà phát triển theo từ quyên góp từ thiện truyền thống thành tương tác từ thiện qua mạng xã hội, vì thế mà ứng dụng web mang tên **“Mạng Xã Hội Từ Thiện“** được xây dựng.

Toàn bộ giao diện được sử dụng **Front END Framework Bootstrap** điều này giúp cho ứng dụng hoạt động và hiển thị ổn định trên mọi hệ điều hành và các đời máy, đó là tính năng **Responsive** của **Bootstrap** . Hệ thống được phát triển với Công Nghệ **Spring FrameWork**  theo kiến trúc **MVC** với các đặc trưng nổi bật như Front Controller Design Pattern, Spring Security,..

## Giới thiệu Mô Hình Kiến Trúc MVC

**MVC** là từ viết tắt của ba từ **Model – View – Controller.** Đây là mô hình thiết kế sử dụng trong kỹ thuật phần mềm. Mô hình source code được chia thành ba phần, mỗi phần tương ứng với chức năng riêng biệt phục vụ cho việc xử lý dữ liệu logic để hiển thị lên màn hình cho người dùng.



Hình 2.1: Mô Hình Kiến Trúc MVC

**Model** các lớp truy cập dữ liệu, xử lý dữ liệu lưu trữ trong cơ sở dữ liệu ( Entity FrameWork, Hibernate, JDBC ). Là cầu nối giữa hai thành phần View và Controller, Model được hiện thực trong cấu trúc project có thể là một file xml hay một persistent class có vai trò là thể hiện như một đối tượng, một table trong cơ sở dữ liệu.

**View** mẫu để tạo HTML động, trình bày nội dung trang web, cấu trúc được trình bày trong file html hoặc xhtml đối với FrameWork JSF và JSP đối với SPRING FrameWork. View cũng có chức năng ghi nhận hoạt động của người dùng để tương tác với Controller với mỗi request tương ứng phù hợp với các hoạt động của Controller. Controller sẽ thực hiện các xử lý và tương tác với Model ( nếu có ) và gọi một view tương ứng hiển thị cho người dùng.

**Controller** Bộ phận có nhiệm vụ xử lý các yêu cầu người dùng ( request ) đưa đến thông qua view, xử lý mối quan hệ giữa View và Model, phản hồi đầu vào của người dùng, tương tác với Model và trả kết quả dữ liệu phù hợp với View.

### Ưu & nhược điểm của MVC

**Ưu :** MVC không sử dụng viewstate nên khá tiết kiệm dữ liệu diện tích băng thông .

Khi tương tác, người dùng có thể sử dụng ứng dụng trên web cần tương tác gửi và nhận dữ liệu một cách liên tục. Vì vậy, việc giảm băng thông giúp cho website hoạt động tốt và ổn định hơn.

Với đặc điểm hoạt động độc lập của từng thành phần, lập trình viên có thể làm việc đồng thời trên từng bộ phận khác nhau của chương trình mà không cần phải lo về lỗi một chức năng nhỏ gây ảnh hưởng đến tính ổn định của cả một chương trình.

Hỗ trợ kỹ thuật không đồng bộ: MVC có thể hoạt động trên nền tảng JavaScript.

Dễ dàng thao tác chỉnh sửa: Bộ phận Model hoạt động tách biệt hoàn toàn với View và Controller giúp cho việc chỉnh sửa các components được dễ dàng mà không làm ảnh hưởng quá nhiều đến các components khác.

**Nhược:** Quá trình điều hướng code khó khăn, luồng dữ liệu phức tạp, mô hình gồm nhiều lớp tương tác với nhau đòi hỏi lập trình phải nắm chắc kiến thức cơ bản để phát triển tốt.

Không thích hợp phát triển các ứng dụng có quy mô nhỏ vì độ phức tạp và cấu trúc lưu trữ một lượng lớn các file cấu hình, file class Persistent, các lớp Controller và các tập tin View HTML tương ứng.

## Giới Thiệu Spring MVC FrameWork

Spring MVC là một framework mã nguồn mở được dùng để phát triển các ứng dụng Web theo mô hình MVC ( Model – View – Controller ), Spring MVC có nhiệm vụ cung cấp kiến trúc MVC và các component được sử dụng nhằm phát triển các ứng dụng web được khả chuyển và linh hoạt hơn.

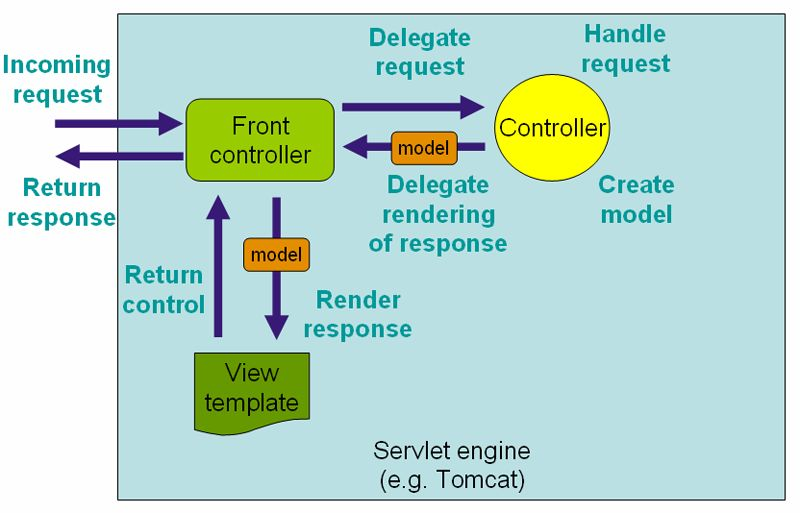
### Các Đặc trưng nổi bật của Spring MVC

Spring MVC hiện thực các đặc trưng của Spring Core như Inversion of Control ( IOC ), Dependency Injection ( DI ) sử dụng các đối tượng khởi tạo như Servlet listeners và Web Application Context. Trong đó các lớp cấu hình project Spring bao gồm Servlet module chứa phương thức kế thừa của Spring MVC dùng cho việc cấu hình để khởi chạy ứng dụng web. Hệ thống phải triển theo mô hình MVC có các package ( gói ) hiện thực cho các lớp tương ứng với chức năng cụ thể: Models sẽ có các đối tượng domain được xử lý bởi tầng service và được lưu trữ bởi các tầng persistence: chứa các lớp pojos hiện thực các đối tượng là các table trong cơ sở dữ liệu. View sử dụng JSP Template được viết bởi JSTL ( Java Standard Tag Library ). Tầng Repository có trách nhiệm kết nối để giao tiếp với Controller thông qua tầng Service để truy vấn các dữ liệu database của Server bằng các câu truy vấn theo kiểu Kỹ thuật Lập trình ORM phát triển bằng Hibernate trên Database MySQL và 1 số tầng phụ khác phục vụ cho việc chứng thực phân quyền người dùng bằng Spring Security một thành phần quan trọng không thể thiếu của Ứng Dụng Web Spring MVC.

#### Front Controller Design Pattern

Khác một chút so với mô hình MVC truyền thống , Spring MVC FrameWork có thêm 1 tầng xử lý các http Request của Clients gửi đến Server, Nó Cũng là nơi Khởi động vài thành phần quan trọng của FrameWork.

Front Controller nhận các request từ người dùng để thực hiện các chức năng chung, chuyển request đến các controller tương ứng dựa trên cấu hình của ứng dụng web và thông tin của HTTP Request.



Hình 2.2: Mô Hình Front Controller Design Pattern

Controller tương tác với tầng Repository để thực hiện các câu truy vấn với các lớp Pojos ( persistence Class ) thông qua tầng Service ( Service Layer ) để thực hiện các logic nghiệp vụ ( business logic ) và thực hiện các thao tác lưu trữ ( persistence logic ).

Sau đó cập nhật dữ liệu cho lớp model và view sẽ đóng gói dữ liệu và hiển thị và Gửi trả view về cho người dùng. Cuối cùng Front Controller phản hồi đến client với một view tương ứng với Request đó. Trong Spring MVC, DispatcherServlet làm việc như tầng xử lý Front Controller.

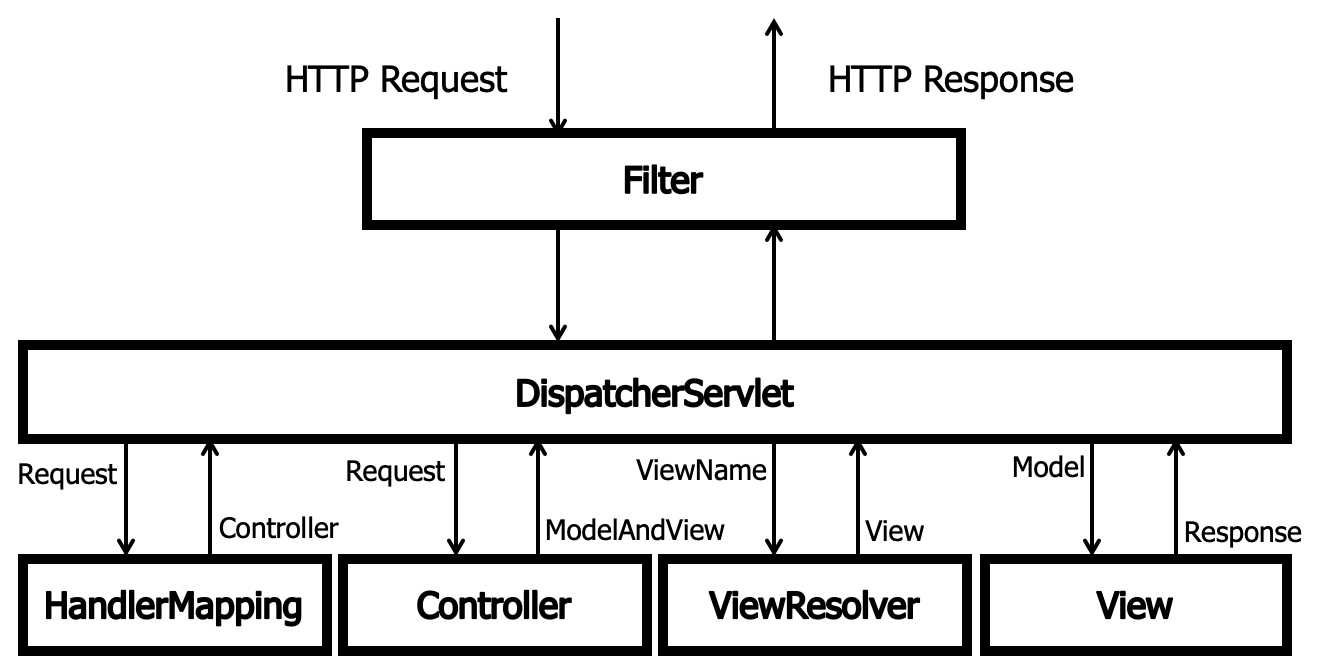
#### DispatcherServlet

Một Request được gửi lên từ phía client vào server thì nó sẽ đi qua

tầng Filter, tầng này cung cấp vai trò chứng thực người dùng trước khi đi vào tầng DispatcherServlet. Tầng DispatcherServlet là nơi nhận tất cả các request từ người dùng để thực hiện xử lý và trả ra Response phù hợp cho người dùng.

Trước tiên DispatcherServlet đi vào xác thực với Handler Mapping để kiểm tra URL người dùng gửi lên có map với View nào trả ra trong Controller không,

Sau đó Request được gửi qua Controller, Controller sẽ thực hiện tương tác với lớp Model để thực hiện các nghiệp vụ sau đó trả ra một dữ liệu là ModelAndView gửi về cho ViewResolver. ViewResolver được cấu hình để tìm tập tin View HTML hay JSP của Spring MVC tương ứng để hiển thị ( render ) dữ liệu cho người dùng. ViewResolver với chức năng phân giải tên View thành một file tập tin JSP tương ứng để trả ra View. Sau đó bên View sẽ đóng gói tất cả thành một Repsonse hoàn chỉnh và gửi trả về cho người dùng dưới dạng View.



Hình 2.3 Mô Hình Dispatcher Servlet

Trong Spring MVC, URL để gửi nhận các Request từ Clients về phía Server có cấu trúc được phân giải gồm 4 phần:

* Localhost:portname tên Domain để chạy server của Ứng Dụng Web Spring trên máy cài đặt Ứng dụng web , portname có ký pháp mặc định là 8080 nếu như trên máy không có các ứng dụng khác sử dụng port này để chạy tác vụ của nó.
* Application Name: Tên Ứng Dụng Web Spring MVC được tự động đặt theo tên của Project khi ta tạo ứng dụng web mới.
* ReQuestPath: Đường dẫn trõ tới các tài nguyên được xây dựng trong controller của ứng dụng. Các Controller có được xây dựng trong các lớp Controller đều có Các Annotation là @RequestMapping để chỉ định đường dẫn tương ứng trên địa chỉ URL. RequestPath phục vụ rất tốt cho lập trình viên trong việc định tuyến các đường dẫn để xây dựng một ứng dụng mang tính dễ tương tác và thân thiện với người dùng hơn.
* RequestParams: Là những tham số gửi lên trên Server Cú pháp khai báo những tham số (Params ) trên đường dẫn URL để tương tác với ứng dụng web bắt đầu bằng dấu ?paramsName=Food thêm & để chỉ định thêm nhiều tham số trên đường dẫn.

http://localhost:8080/SocialNetWork/user/profile/?userName=admin

Hình 2.4 Cấu trúc Đường Dẫn URL của một Ứng Dụng Web

Các lớp được cho là Controller có thể được các tập cấu hình nhận biết thông qua các @Annotation là @Controller và @RestController đặt trên đỉnh đầu của tên Lớp đó. Khi các Controller này nhận được request của Client với URL có Request Path giống với các đường dẫn mà phương thức Controller được cấu hình với Annotation @RequesMapping thì hệ thống sẽ tiến hành thực thi Controller đó với Request Path tương ứng . Mỗi phương thức như vậy đều trả về chuỗi String đại diện cho 1 view tương ứng sẽ được đóng gói và hiển thị lên giao diện cho người dùng.

#### Web Application Context

Dispatcher Servlet yêu cầu một đối tượng là Web Application Context, là một Extension của Application Context, được sử dụng như một Container dùng để tạo và chứa các đối tượng. Container có vai trò quản lý vòng đời của đối tượng, những đối tượng này trong Container được gọi là Spring Managed Beans.

Các Container sử dụng Dependency Injection (DI) để quản lý các đối tượng beans, một thể hiện Application Context dùng để tạo beans, kết hợp beans thông qua cấu hình và cung cấp các beans khi có request từ Clients. Beans: là các phương thức các hàm có gắn annotation là @Bean hoặc là các cấu hình Bean được cấu hình trong tập tin XML chứa trong một package ( gói ) các tập tin cấu hình nên ứng dụng web.

Ví Dụ sử dụng Bean để tạo đối tượng ViewResolver trong Container để chỉ định các view dùng để render để response cho một Request Từ Client.

@Bean

public InternalResourceViewResolver getInternalResourceViewResolver(){

InternalResourceViewResolver resolver = new InternalResourceViewResolver();

resolver.setViewClass(JstlView.class);

resolver.setPrefix("/WEB-INF/jsp/");

resolver.setSuffix(".jsp");

return resolver;

}

#### Dependency Injection ( DI ) và Inversion of Control ( Ioc)

Dependency Injection: Đây là một cách để hiện thực design pattern : Inversion of Control Pattern, các module phụ thuộc ( dependency ) sẽ được inject vào module cấp cao.

Inversion of Control: Đây là một Design Pattern được tạo ra để code tuân thủ các nguyên lý Dependency Inversion. Có nhiều cách để hiện thực các pattern này chẳng hạn như Service Locator, Event, Delegate,…. Dependency Injection là một trong những cách để hiện thực nó.

Khi nói tới Dependency Injection (DI) có một số các hệ thống DI Container như UNITY, Structure Map, Spring v…v, hỗ trợ trong việc cài đặt và áp dụng DI vào Code, nó còn có tên gọi khác là IoC Container, ý nghĩa của chúng là tương tự.

**Định Nghĩa của Dependency Injection**

Các module không trực tiếp giao tiếp với nhau mà thông qua các interface . Module cấp thấp ở đây là các Class sẽ kế thừa (implements) các Interface , và Các Module cấp cao sẽ gọi module cấp thấp thông qua interface.

Ví dụ để giao tiếp với Database , ta có interfact Idatabase và các module cấp thấp là XML Database, SQLDatabase. Module Cấp Cao là Customer Business Sẽ chỉ sử dụng interface Idatabase.

Việc khởi tạo các module cấp thấp sẽ do DI Container thực hiện trong module Customer Business ta không khởi tạo Idatabase db = new XMLDatabase(), việc này sẽ do DI Container thực hiện. Cụ thể ở Spring là Sử dụng cơ chế móc nối Autowired Để thực hiện tiêm các đối tượng XMLDatabase vào trong Module Cấp cao Customer Business. Module Cấp cao này sẽ không biết gì về các Module XML Database hay SQL Database.

Việc Module nào gắn với các interface nào sẽ được cấu hình thông qua tập tin XML hoặc các Annotaion trong Code, với việc cấu hình theo DI giúp giảm sự phụ thuộc giữa các Module, giúp cho việc bảo trì dễ thay đổi module và kiểm thử.

**Có 3 dạng Dependency Injection :**

**Constructor Injection:** các Dependency sẽ được các Container cụ thể ở đây là Web Application Context của Spring tiêm vào (inject) một class thông qua constructor của class đó, đây là cách thông dụng nhất.

**Interface Injection:** Class cần inject sẽ kế thừa một interface. Interface này sẽ chứa một hàm tên là Inject. Container sẽ tiêm các dependency vào một class thông qua việc gọi các hàm tiêm của Interface đó.

**Setter Injection:** các dependency trong container sẽ được vào class thông qua các phương thức getter.

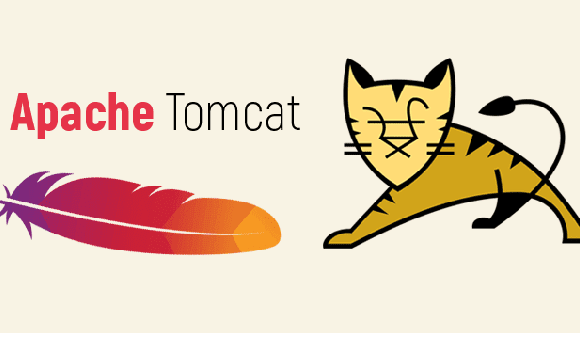
**Các Ưu điểm và nhược điểm DI và IOC**

**Ưu điểm** Giảm sự kết dính giữa các module, Code dễ bảo trì, dễ thay thế các module, rất dễ để viết các unit test. Dễ dàng thấy các relationship giữa các module.

**Nhược điểm** Khái niệm Dependency Injection vẫn còn rất mới và khó làm quen đối với những người bắt đầu học các Design Pattern mới cụ thể ở đây là **Dependency Inversion Principle.** Vì sử dụng các Interface chủ yếu nên việc debug để tìm lỗi trở nên khó khăn , làm cho cấu trúc chương trình trở nên phức tạp, không phù hợp với những dự án phát triển quy mô nhỏ.

## Các Thành Phần tạo nên Ứng Dụng Web Spring MVC

### Giới Thiệu về TomCat Server

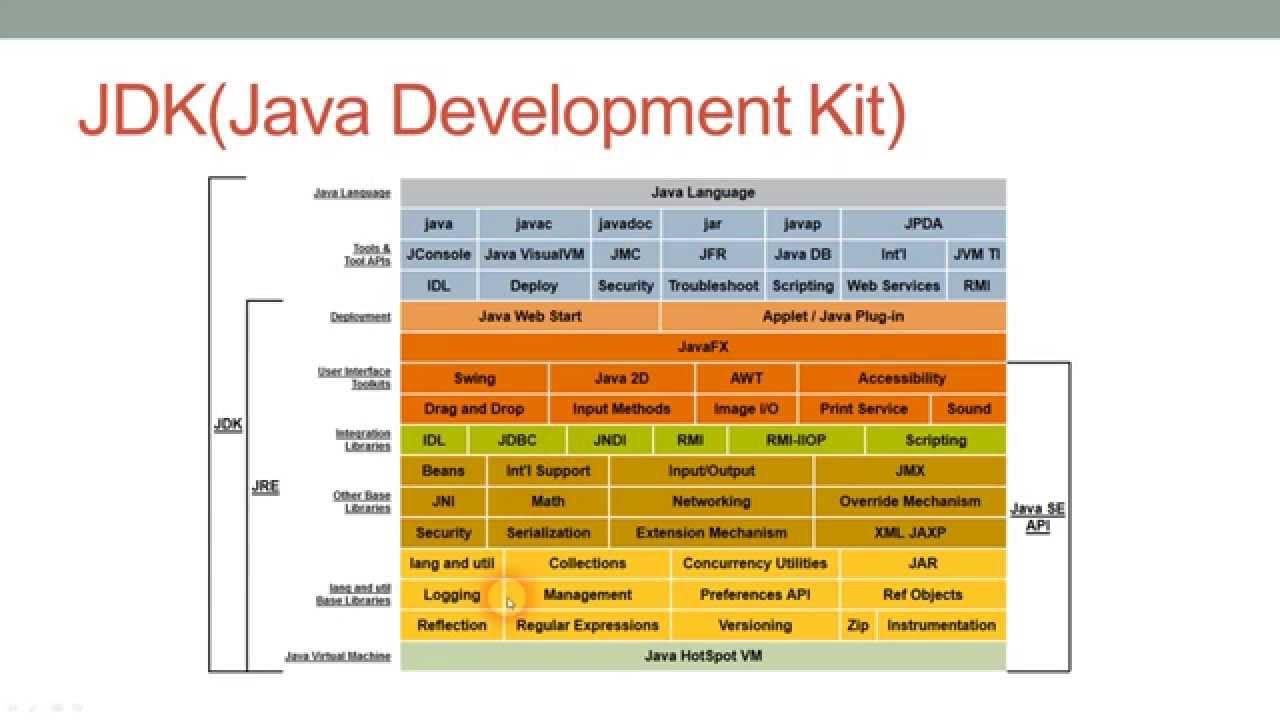


Hình 2.6 Apache Tomcat Server

Apache Tomcat Server là một web server HTTP được phát triển bởi Apache Software Foundation, được xây dựng để phát triển ứng dụng Java . Với khả năng chạy nhiều bản Java chuyên biệt như Java Servlet, Java Server Pages ( JSP ), Java EL và Web Socket.

Với TomCat Apache , lập trình viên có thể tương tác bằng nhiều ngôn ngữ lập trình khác nhau chẳng hạn như PHP, Python, Perl…. Giúp cho việc phát triển ứng dụng web được nhanh chóng an toàn hơn nhờ vào công nghệ mã nguồn mở và khả năng chứng thực bảo mật thông tin của Apache.

### Java Development Kit (JDK )



Hình 2.7 Java Development Kit

Là một trong ba gói công nghệ cốt lõi được sử dụng trong lập trình Java, Máy Ảo Java – Java Virtual Machine ( JVM ) và Môi trường lập trình Java RunTime – Java RunTime Enviroment (JRE). JDK được phát triển bởi Sun Microsystem và được tuyên bố cấp phép bởi GNU vào ngày 11 tháng 7 năm 2006. Giấy phép được trao bởi GNU là một dạng giấy phép về phần mềm tự do cho phép có thể sử dụng chia sẻ, sữa đổi và phát triển ứng dụng ngày càng thêm lớn mạnh.

JVM là thành phần Java nền tảng để chạy các chương trình

JRE là gói công cụ để chạy Java Code, phần tạo ra JVM

JDK cho phép các developer tạo ra các chương trình Java , trong đó các thành phần chương trình có thể tương tác với JVM và JRE để xử lý và chạy.

Để thiết lập và cài đặt một môi trường phát triển cũng rất đơn giản, khi tải xuống JDK bạn sẽ cần chọn phiên bản mà mình muốn sử dụng. Java 8 từng là phiên bản được sử dụng phổ biến nhất. Tuy nhiên để phát triển ứng dụng web trên Spring MVC được mượt mà và cập nhật tương thích cùng với framework này, ta nên sử dụng bộ JDK phát hành mới nhất của nhà phát hành cụ thể là bản 15 để có những trải nghiệm tốt nhất và cập nhật về mặt kiến thức nhanh nhất.

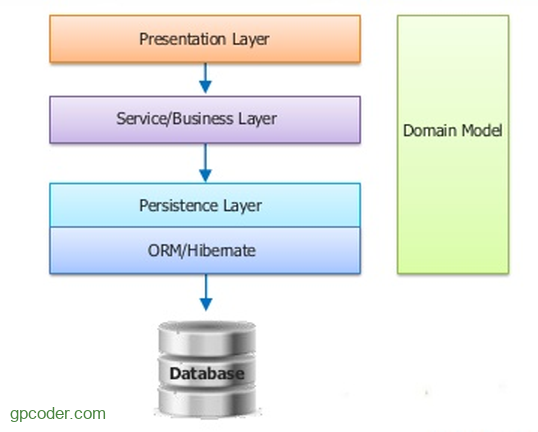
### Giới thiệu về NETBEAN IDE

Là một môi trường phát triển tích hợp (IDE ) cho phép người lập trình viên có thể viết và phát triển các ứng dụng, web được hỗ trợ các ngôn ngữ mà NetBean đã tích hợp trong môi trường cụ thể là Java. Netbean cho phép các ứng dụng phát triển từ một tập hợp các thành phần gọi là modules, trình duyệt này có thể chạy trên các hệ điều hành phổ biến hiện nay như Windows, macOS, Linux và Solaris. Ngoài việc phát triển Java , NetBean có các phân mở rộng cho các ngôn ngữ lập trình khác như PHP, C++, HTML5 và JavaScript, ngoài ra NetBean cũng hổ trợ việc phát triển các ứng dụng được phát triển ở bên trung gian – thứ ba.

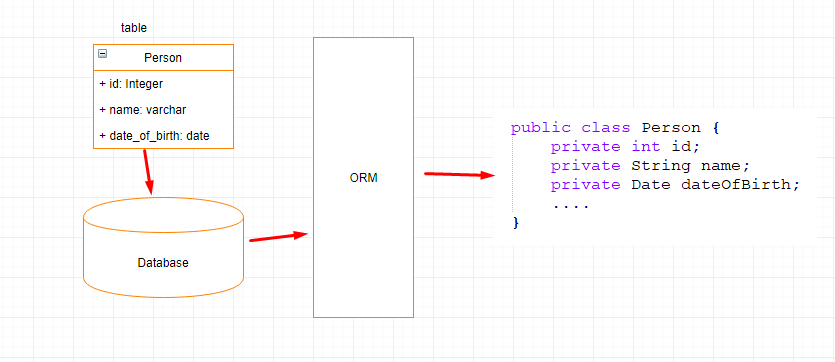
### Giới Thiệu về Hibernate theo giải pháp ORM

ORM (Object – Relational Mapping) là một kỹ thuật lập trình chuyển dữ liệu các lớp hướng đối tượng (persistence object) được xây dựng trong mã nguồn ( source code ) ánh xạ đến các bảng (Table) tương ứng trong Cơ Sở Dữ Liệu, ORM giúp cho việc thao tác với dữ liệu được đơn giản hóa. Lập trình viên có thể tương tác với các bảng trong cơ sở dữ liệu thông qua lớp đối tượng mà không cần quan tâm đến loại database đang sử dụng là SQL Server, MySQL, PostgreeSQL,….

Cơ Chế Hoạt Động của ORM khi tương tác với các lớp đối tượng trên mã nguồn hoàn toàn bằng code Java theo cấu trúc của API này cung cấp các câu truy vấn mang tính hướng đối tượng hoàn toàn mà không cần phải viết các câu truy vấn Select, Delete, Update theo kiểu truyền thống để tương tác với các bảng nữa. Khi thực hiện chạy chương trình tầng ORM sẽ có trách nhiệm là đối tác trung gian giữa Persistence Layer và Database để thực hiện chuyển các câu truy vấn mang tính hướng đối tượng thành câu truy vấn dạng bảng truyền thống của SQL Server để tương tác với cơ sở dữ liệu của chúng ta. ORM mang lại cho lập trình viên cảm giác sử dụng không còn quan tâm đến các câu truy vấn SQL mà tập trung vào xử lý logic nghiệp vụ, tương tác với các lớp hướng đối tượng nhiều hơn là các bảng dưới cơ sở dữ liệu, cung cấp các câu truy vấn API logic hướng đối tượng, không trực tiếp viết các câu truy vấn SQL.



Hình 2.8 Tương Tác Giữa Database và Persistence Layer Với Hibernate/ORM



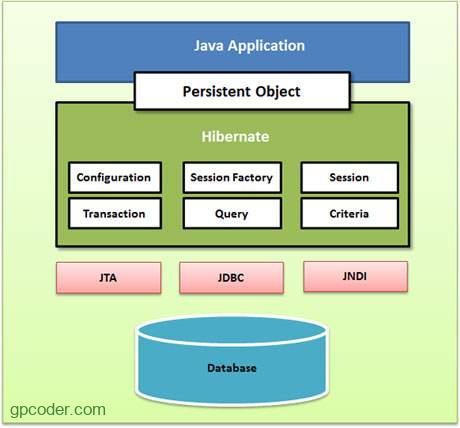
Hình 2.9 Kỹ Thuật ORM Mapping Các Lớp Hướng Đối Tượng vào các table

#### Hibernate

Hibernate là một framework mã nguồn nổi tiếng phát triển theo giải pháp ORM bởi Gavin King vào năm 2001 và còn cải tiến và phát triển đến tận bây giờ. Là một giải pháp ORM mạnh mẽ, hiệu năng cao cho Java, hỗ trợ hầu hết các cơ sở dữ liệu quan hệ như MySQL, PostgressSQL, MS SQL Server, DB2.

Hibernate giúp cho việc phát triển ứng dụng của bạn chỉ còn chú tâm vào các lớp hướng đối tượng mà không cần phải bận tâm nhiều về các table được mapping xuống dưới cơ sỡ dữ liệu nữa. Hibernate giúp hỗ trợ lưu trữ và truy vấn dữ liệu quan hệ được nhanh nhẹn cho kết quả nhanh chóng khi cho phép bạn truy vấn dữ liệu thông qua Java Persistence API (JPA) hoặc bằng ngôn ngữ SQL mở rộng của Hibernate (HQL) và SQL thuần (Native SQL).

**Kiến trúc Hibernate**



Hình 2.10 Kiến trúc Hibernate

**Cấu hình Hibernate**

Khi lập trình tương tác với những project sử dụng maven ta cần phải thêm vào các dependency gồm JDBC Connector tương ứng với hệ quản trị cơ sở dữ liệu sử dụng phía bên dưới và bộ hibernate core

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>5.4.10.Final</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.18</version>

</dependency>

**Persistence Object**

Là các lớp Pojo(Plain Old Java Object) object mapping với các table tương ứng của cơ sở dữ liệu quan hệ. Nó chính là các container lưu trữ các đối tượng của nó từ ứng dụng để lưu xuống Database, hay chứa dữ liệu tải lên từ database.

Việc cấu hình các đối tượng bằng Persistence có hai kiểu là cấu hình bằng tập tin XML và cấu hình các đối tượng này bằng các Java Class có gắn annotation đặc trưng của hibernate để mapping xuống các đối tượng dưới cơ sở dữ liệu. Phương thức cấu hình bằng annotation được sử dụng rất phổ biến trong cộng đồng lập trình java vì các ưu điểm của nó chẳng hạn như thấy các đối tượng và các ràng buộc về khóa ngoại khóa chính và các mối quan hệ manytoone,manytomany,vv được thể hiện trực tiếp trong đối tượng cấu hình lớp Java @Annotation,như vậy với cấu hình annotaiton hệ thống sẽ dễ dàng khả chuyển để bảo trì, có giao diện code thân thiện trực quan với các lập trình viên,nên việc sử dụng trong các dự án hiện đại ngày nay là chuyện dễ hiểu.

Cấu hình bằng tập tin xml

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate

Mapping DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping>

<class name="pojo.Author" table="author">

<meta attribute="class-description">Author Info</meta>

<id column="author\_code" name="id" type="string"/>

<property column="first\_name"

name="firstName" type="string"/>

<property column="last\_name"

name="lastName" type="string"/>

<property column="date\_of\_birth"

name="dateOfBirth" type="date"/>

<property column="gender"

name="gender" type="boolean"/>

</class>

</hibernate-mapping>

Cấu hình bằng code Java

@Entity

@Table(name ="user")

public class User implements Serializable{

public static final String ADMIN = "ROLE\_ADMIN";

public static final String USER = "ROLE\_USER";

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id ;

@Size(min = 1 , max = 50 , message = "{user.username.lenErr}")

private String username;

@NotEmpty(message = "user.password.notEmpty")

private String password;

@Pattern(regexp = "^[A-Za-z0-9+\_.-]+@(.+)$" ,

message = "{user.email.error.invalidMsg}")

private String email;

@Size(min = 3 , max = 50 , message = "{user.name.lenErr}")

private String name;

**Configuration**

Là đối tượng Hibernate đầu tiên được tạo trong ứng dụng sử dụng framework hibernate và chỉ cần được tạo một lần duy nhất trong suốt quá trình chạy ứng dụng . Nó đại diện cho một tập tin cấu hình hoặc một thuộc tính yêu cầu được viết bằng mã nguồn Java của hibernate. Các thông tin chi tiết của một tập tin cấu hình cụ thể gồm có các thông tin liên quan đến database và một số thông tin ánh xạ các lớp đối tượng java đến bảng dữ liệu.

Database Connection : Chứa các thông tin về server kết nối tên port, tên database, username, password được xử lý thông qua một hoặc nhiều tập tin cấu hình được Hibernate hỗ trợ, các tệp này thường có tên hibernate.properties và hibernate.cfg.xml.

Class Mapping Setup: thành phần tạo ra các ánh xạ kết nối giữa các lớp Pojo và các bảng dưới cơ sở dữ liệu.

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate

Configuration DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<property name="hibernate.dialect">

org.hibernate.dialect.MySQLDialect

</property>

<property name="hibernate.connection.driver\_class">

com.mysql.cj.jdbc.Driver

</property>

<property name="hibernate.connection.url">

jdbc:mysql://localhost:3308/socialnetwork

</property>

<property name="hibernate.connection.username">root</property>

<property name="hibernate.connection.password">17102000</property>

<mapping resource="SocialNetWork.hbm.xml"/>

</session-factory>

</hibernate-configuration>

Cấu hình hibernate bằng mã nguồn Java

Configuration conf = new Configuration();

Properties props = new Properties();

props.put(Environment.DIALECT,"org.hibernate.dialect.MySQLDialect");

props.put(Environment.DRIVER,"com.mysql.cj.jdbc.Driver");

props.put(Environment.URL,"jdbc:mysql://localhost:3308/socialnetwork");

props.put(Environment.USER, "root");

props.put(Environment.PASS, "17102000");

conf.setProperties(props);

**Session Factory**

Là một interface tạo ra các đối tượng session để tương tác thực hiện các câu truy vấn xuống dưới cơ sỡ dữ liệu bằng cách đọc cấu hình trong tập tin cấu hình Hibernate Configuration. Session Factory là một đối tượng nặng ( heavy weight object ) được tạo bởi Configuration và tồn tại xuyên suốt trong khi chương trình hoạt động cho đến khi chương trình kết thúc. Mỗi cơ sở dữ liệu ( có tập tin cấu hình riêng hay mã nguồn java riêng ) đều cần có một thể hiện Session Factory riêng biệt.

Code mở Session factory để thực hiện truy vấn ở tầng Repository

Session session = sessionFactory.getObject().getCurrentSession();

CriteriaBuilder builder = session.getCriteriaBuilder();

CriteriaQuery<User> query = builder.createQuery(User.class);

Root<User> uRoot = query.from(User.class);

Predicate p = builder.equal(uRoot.get("username").as(String.class),username.trim());

query.where(p);

query.select(uRoot);

Query q = session.createQuery(query);

return q.getResultList();

**Hibernate Session**

Session được sử dụng để tạo kết nối vật lý tới một cơ sở dữ liệu. Đối tượng session để thiết kế với kiểu dáng gọn, nhẹ và được thiết kế như một instance mỗi khi tương tác với cơ sở dữ liệu. Tác dụng của Session dùng để thực hiện thao tác tạo,đọc, cập nhật và xóa các thể hiện của lớp đối tượng được ánh xạ (persistence object).Các đối tượng Session không nên được mở trong một thời gian dài bởi vì chúng không phải là luồng an toàn (thread – unsafe) và bắt buộc phải được đóng lại xong khi được mở ra để tương tác các câu truy vấn.

**Hibernate Transaction ( Quản lý giao tác )**

Đại diện cho một đơn vị làm việc với cơ sở dữ liệu và hầu hết các chức năng giao tác transaction, các transaction trong hibernate được xử lý bởi trình quản lý transaction ( từ JDBC Java Database Connectivity đến JTA là Java Transaction API ).

Transaction đảm bảo tính toàn vẹn của phiên làm việc với cơ sở dữ liệu, khi mọi thao tác thêm xóa sửa đều được ghi lại và nếu như có một thao tác thất bại trong quá trình thực hiện session thì sẽ tiến hành rollback các thao tác thêm xóa sửa đó đảm báo tính toàn vẹn của dữ liệu và khả năng phục hồi lại sau sự cố đều do đối tượng transaction này đảm nhiệm.

**Query**

Các đối tượng tạo câu truy vấn có thể là chuỗi truy vấn SQL ( Native SQL ) hoặc là Hibernate Query Language ( HQL ) để lấy dữ liệu từ cơ sở dữ liệu và chỉ tương tác trực tiếp với các lớp đối tượng mặc dù vẫn còn bóng dáng của các câu lệnh truyền thống SQL như Select From Where ,vv.

Query q = session.createQuery("SELECT A.firstName "+ "FROM Author A");

List r = q.list();

**Criteria**

Các đối tượng Criteria được Hibernate cung cấp để thực hiện các thao tác với cơ sở dữ liệu dễ dàng, nhanh chóng mà không cần phải biết quá nhiều cú pháp truy vấn SQL.

CriteriaBuilder builder = session.getCriteriaBuilder();

CriteriaQuery<Product> query

= builder.createQuery(Product.class);

Root<Product> root = query.from(Product.class);

query.select(root);

Query<Product> q = session.createQuery(query);

List<Product> rs = q.getResultList();

rs.forEach(p -> System.out.println(p.getName()));

### Bootstrap

Bootstrap là một framework sử dụng các thành phần HTML,CSS,JavaScript cho phép người dùng dễ dàng thiết kế website theo một cái chuẩn đã được các lập trình viên phát triển và tích hợp vào mã nguồn của framework cho phép ứng dụng phát triển với giao diện khả chuyển và phù hợp với mọi nền tảng thiết bị phát triển chẳng hạn như ipad,tablet,phone,PC,Laptop.Lý do bootstrap phát triển mạnh là vì hệ thống được xây dựng với các thành phần có sẵn chẳng hạn như: typography,forms,buttons,tables,navigation,image carousels và những thứ khác,các components, javascript giúp cho việc thiết kế responsive của bạn được dễ dàng thuận tiện và nhanh chóng.

Những ưu điểm khi sử dụng có được kể đến như:

Rất dễ sử dụng,đơn giản vì nó được dựa trên HTML,CSS,JavaScript.

Bootstrap xây dựng các responsive css trên các thiết bị Iphones,Tablets và Deskop,tính năng này giúp cho người dùng tiết kiệm được rất nhiều thời gian trong việc tạo ra một website thân thiện với các thiết bị điện tử,thiết bị cầm tay.

Khả năng tương thích với các trình duyệt chẳng hạn như Chrome,FireFox,Internet Explorer,Safari và Opera.Tuy nhiên Bootstrap chỉ hỗ trợ từ IE9 trở len ( IE Browser ), điều này vô cùng dễ hiểu vì IE8 không hỗ trợ HTML5 và CSS3.



Hình 2.11 Bootstrap

#### Ba loại file chính của Boostrap

**Bootstrap.css**

Sử dụng và phát triển dựa trên CSS framework sẵn có, bootstrap CSS được xử lý với các tập tin file được lập trình viên xây dựng theo các class với mỗi class ấy là một struct có component là thành phần của CSS framework sẵn có tạo nên . Giúp cho việc phát triển front end của các ứng dụng web được nhanh chóng và khả chuyển trên mọi nền tảng mà người sử dụng không cần phải quan tâm quá chi tiết về các CSS tạo nên các thành phần class boostrap mà tập trung vào mục đích chính của việc thiết kế ứng dụng web là tạo nên một ứng dụng thân thiện và ưa nhìn.

**Boostrap.js**

Các thành phần tương tác website của bootstrap chứa các câu lệnh javascript nhằm tiết kiệm thời gian viết cú pháp JavaScript mà nhiều nhà phát triển đã sử dụng Jquery.Đây là thư viện JavaScript mã nguồn mở,đa nền tảng giúp bạn thêm nhiều chức năng vào trang web,chẳng hạn như:

Thực hiện các yêu cầu của Ajax,loại trừ dữ liệu từ một vị trí khác một cách linh hoạt,tạo các tiện ích bằng bộ sưu tập Plugin Javascript,cung cấp các tạo hình động tùy chỉnh bằng thuộc tính của CSS.

**Glyphicons**

Trong các giao diện web, những thành phần tạo nên sự bắt mắt của nó không thể thiếu đó là Icons.mục đích của chúng thường được kể đến như liên kết đến dữ liệu nhất định và các hành động trong giao diện người dùng.

### MySQL Workbench

MySQL là một cơ sở dữ liệu quan hệ, mã nguồn mở được sử dụng phổ biến trong việc phát triển các ứng dụng web từ lớn đến nhỏ, hiệu năng cao cho phép lưu trữ một lượng lớn dữ liệu trong các bảng và được các hệ thống lớn sử dụng hiện nay như Facebook,vv.MySQL hỗ trợ trên nhiều hệ điều hành và các ngôn ngữ lập trình khác nhau như là PHP,Java,Ruby,Python,v.v.Vì sử dụng hình thức chuẩn của ngôn ngữ truy vấn dữ liệu SQL phổ biến nên cú pháp truy vấn thông dụng của nó gần giống như các truy vấn trong hệ MS SQL Server.

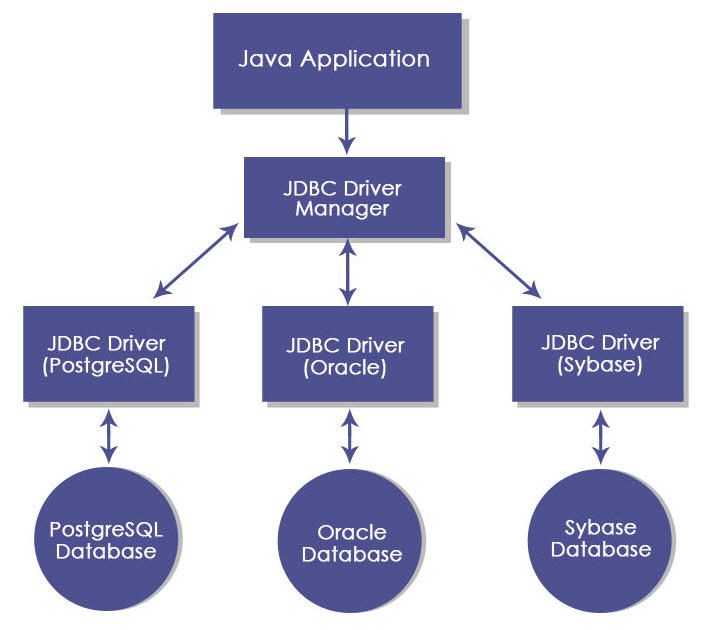
MySQL Workbench là một chương trình giúp cho lập trình viên có thể giao tiếp với hệ quản trị MySQL thông qua giao diện tương tác thân thiện với người dùng. Giúp cho việc phát triển tương tác với cơ sở dữ liệu nhanh chóng khả chuyển mà không cần phải tương tác trực tiếp qua các câu lệnh command-line phức tạp khó nhớ. Với hệ thống thiết kế đơn giản dễ sử dụng và có thể thích ứng với nhiều hệ điều hành khác như là Windows, Mac OS, Linus hay Ubuntu.

### Giới Thiệu JDBC

JDBC(Java Database Connectivity) cung cấp các bộ API để phát triển các ứng dụng tương tác với các cơ sở dữ liệu quan hệ bằng mã nguồn Java.JDBC hỗ trợ nhiều hệ quản trị cơ sở dữ liệu phổ biến hiện nay như SQL Server , MYSQL ,Postgres SQL ,

v.v.JDBC với các bộ API cung cấp các chức năng như tạo kết nối đến cơ sở dữ liệu,tạo câu lệnh SQL,thực thi câu lệnh SQL, xuất các dữ liệu thể hiện từ các câu truy vấn như là xem,thêm,xóa sửa tương tác với CSDL.

**Kiến trúc JDBC**



Hình 2.12 Kiến trúc JDBC

Để tương tác với từng cơ sở dữ liệu nhất định cần phải có một driver tương ứng của nó , các driver này được cung cấp bởi các nhà phát triển cơ sở dữ liệu đó với mục dich hiện thực hóa các JDBC API.

**Các đối tượng trong JDBC API**

Driver Manager quản lý các driver của cơ sở dữ liệu.

Driver giao diện xử lý các giao tiếp cơ sỡ dữ liệu, ta không tương tác trực tiếp với giao diện này mà thông qua Driver Manager.

Connection giao diện chứa các phương thức để tạo kết nối đến cơ sở dữ liệu.

Statement: cung cấp các phương thức để thực thi câu truy vấn , Stored procedure.

Result Set:đối tượng của giao diện này đại diện cho dữ liệu được trả về khi thực thi câu truy vấn.

SQL Exception:Xử lý và ném các lỗi ngoại lệ trong quá trình thực hiện tương tác với cơ sở dữ liệu.

**Transaction**

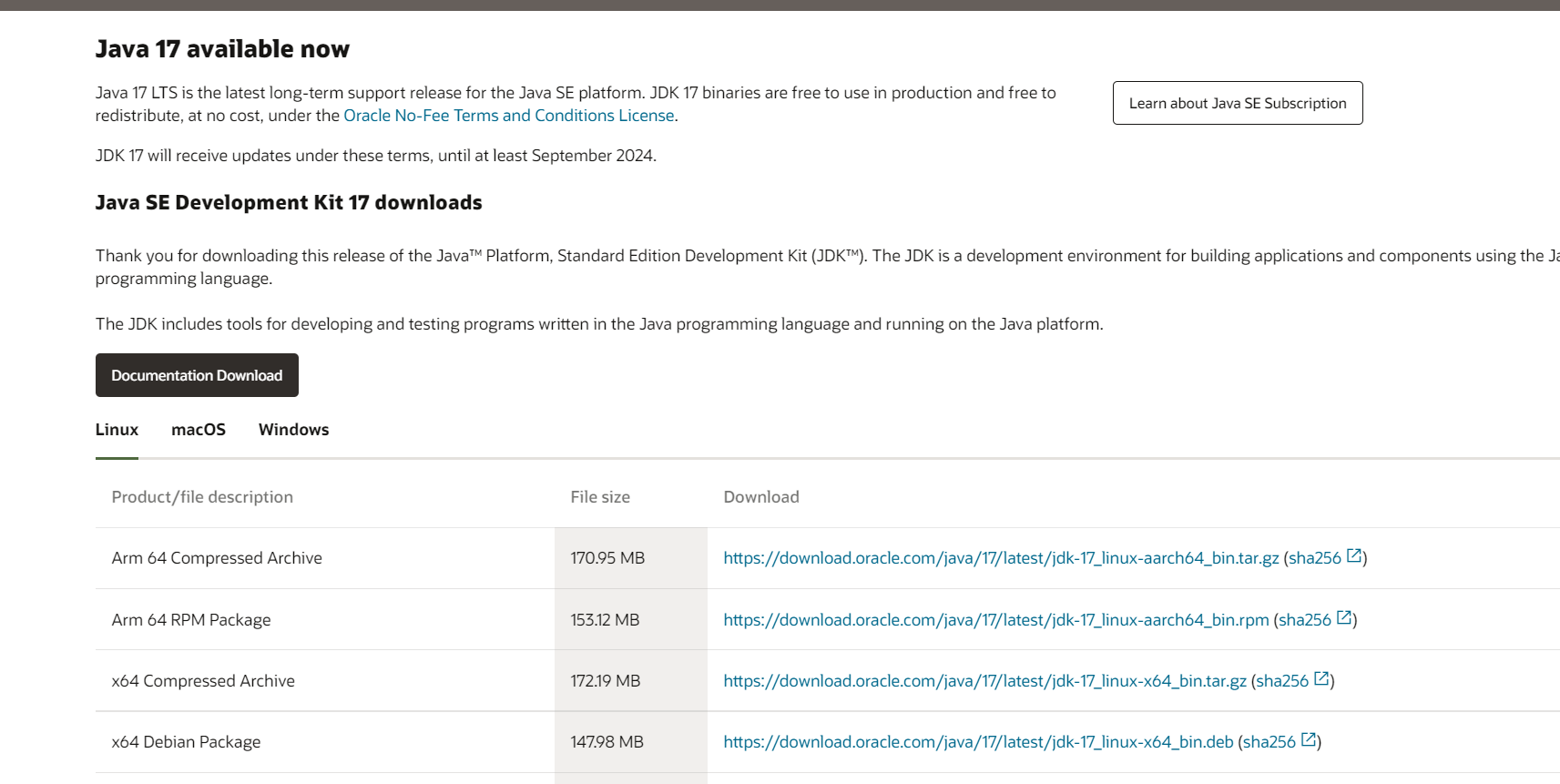
Đại diện cho một đơn vị xử lý, được hỗ trợ bởi hầu hết các hệ quản trị cơ sở dữ liệu.Interface Connection mặc định sử dụng chế độ autocommit để làm việc với giao tác, túc là mỗi câu lệnh truy vấn statement trong kết nối sẽ được thực thi và commit lưu trữ ngay xuống CSDL, nếu như có trường hợp lỗi xảy ra trong quá trình chèn về kiểu dữ liệu, hoặc dữ liệu chèn vào bảng dữ liệu không được nhất quán và đồng nhất thì quản lý giao tác sẽ thực hiện các thao tác rollback , trả CSDL về tình trạng nguyên vẹn trước khi được commit và ném ra các ngoại lệ SQL Exception để thông báo cho lập trình viên lỗi.

## Xây Dựng Chương Trình Đầu Tiên sử dụng Spring MVC

### Cài đặt bộ JDK

Truy cập vào đường link của ORACLE để tải phiên bản mới nhất phù hợp với Hệ Điều Hành tích hợp trên máy của mình.

<https://www.oracle.com/java/technologies/downloads/>



Hình 2.13 Các gói JDK

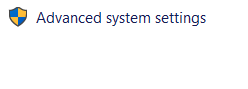
Sau khi đồng ý điều khoản và nhấp chọn dowload , mở tệp JDK.exe vừa dowload về để bắt đầu quá trình cài đặt. Nhấp chọn next và chọn đường dẫn thư mục phù hợp để lưu các file cài đặt và chờ đợi một khoảng thời gian cho đến khi chương trình cài đặt hoàn tất.

### Cài đặt biến môi trường (Environment Variables) trong Java : PATH và CLASSPATH

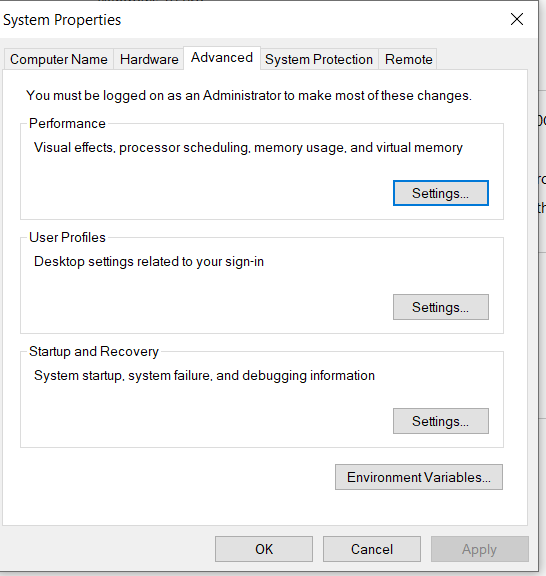
Đầu tiên ở màn hình deskop nhấp chuột phải vào my computer và chọn Properties.



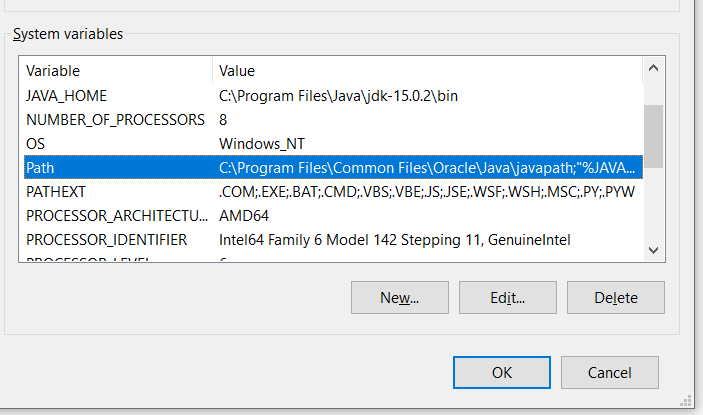
Click chọn Advanced system settings ở góc trái màn hình



Chọn Tab Advanced và chọn Environment Variables

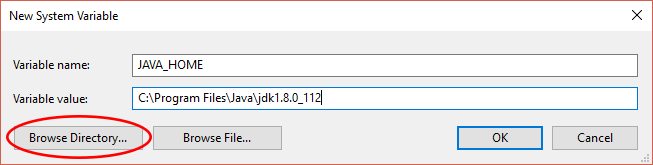


Ở mục System variables, nhấp chọn mục Path và bấm New



Thêm một biến môi trường mới vào

JAVA\_HOME Và Đường link tới vị trí cài đặt file jdk lúc nãy bạn đã chọn,nếu như không biết nó ở đâu bạn có thể chọn Browse Directory để tìm đường dẫn tuyệt đối của nó.



Sau đó chọn Oke và lưu biến môi trường mới vào Path.

Cuối cùng bạn mở cửa sổ command prompt – CMD của máy bạn và gõ java-version để kiểm tra nếu hiện như hình thì bạn đã cài đặt JDK thành công.

### Cài đặt Apache Tomcat Server

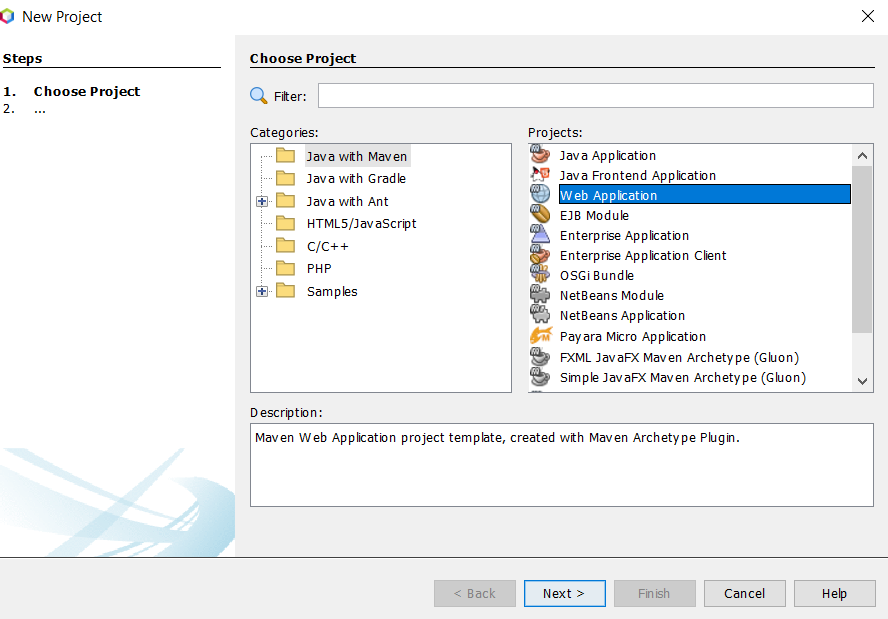
Truy cập vào trang chủ của Tomcat Server

<http://tomcat.apache.org> để tải bản Tomcat Server mới nhất, có thể cân nhắc dowload bản mới nhất bản 10 nhưng để sử dụng server một cách một ổn định, em đã sử dụng Tomcat bản 9.

Sau khi đã tải về phiên bản phù hợp và cài đặt bản exe, ta chọn lưu vào ổ thư mục phù hợp để khi tạo project maven ta sẽ chỉ định đường dẫn thư mục chứa tomcat server đã cài đăt.

### Tạo Project Maven mới.

Cấu hình Spring MVC với Project Maven



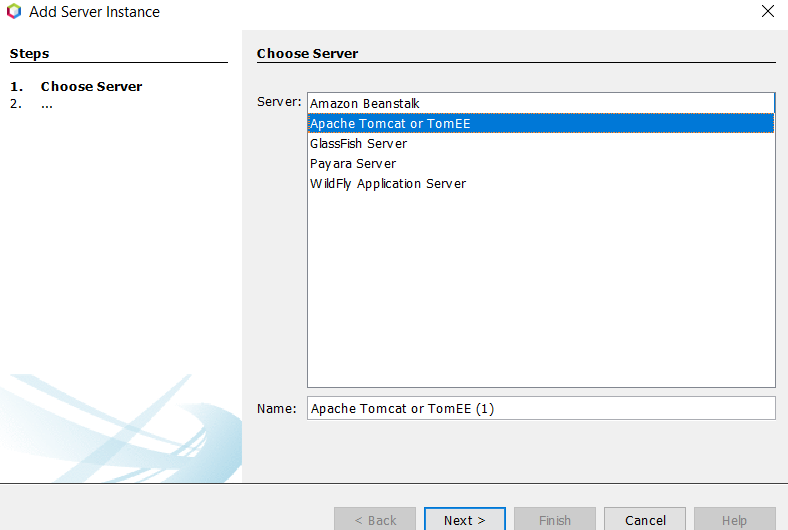
Hình 2.14 Cài Đặt Project Maven

Thiết Lập Tomcat Server mới vừa cài đặt.



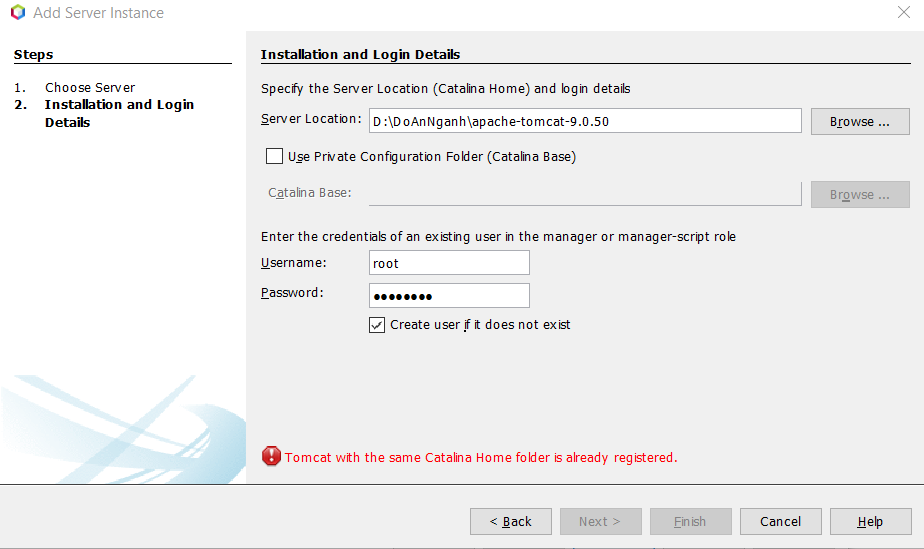
Hình 2.15Thiết Lập TomCat Server

Bấm vào Add nếu như chưa có Server và chọn Server Tomcat



Hình 2.16 Chọn Server

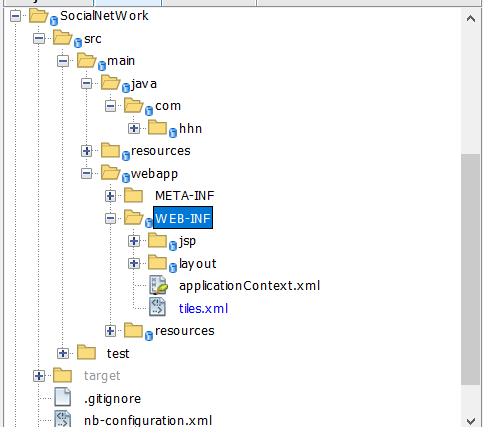
Chọn và chỉ định thư mục chứa Tomcat đã cài đặt lúc nãy và thiết lập Username và Password của nó.



Hình 2.17 Cấu Hình Server

Điền thông Tin Tên Project sau đó là tạo project

Cấu trúc Project sau khi được tạo



Hình 2.18 Cấu trúc project vừa mới tạo

Thêm các dependencies của Spring vào pom.xml

<dependency>

<groupId>javax</groupId>

<artifactId>javaee-web-api</artifactId>

<version>7.0</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>5.2.2.RELEASE</version>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

Trong thư mục webapp /WEB-INF/applicationContext.xml

Có gắn một mục context là cấu hình project này bằng annotation.

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xmlns:aop="http://www.springframework.org/schema/aop"

xmlns:c="http://www.springframework.org/schema/c"

xmlns:context="http://www.springframework.org/schema/context"

xmlns:flow="http://www.springframework.org/schema/webflow-config"

xmlns:jee="http://www.springframework.org/schema/jee"

xmlns:jms="http://www.springframework.org/schema/jms"

xmlns:lang="http://www.springframework.org/schema/lang"

xmlns:osgi="http://www.springframework.org/schema/osgi"

xmlns:p="http://www.springframework.org/schema/p"

xmlns:tx="http://www.springframework.org/schema/tx"

xmlns:util="http://www.springframework.org/schema/util"

xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.2.3.xsd

http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-5.2.xsd

http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-5.2.xsd

http://www.springframework.org/schema/webflow-config http://www.springframework.org/schema/webflow-config/spring-webflow-config-2.0.xsd

http://www.springframework.org/schema/jee http://www.springframework.org/schema/jee/spring-jee-5.2.xsd

http://www.springframework.org/schema/jms http://www.springframework.org/schema/jms/spring-jms-5.2.xsd

http://www.springframework.org/schema/lang http://www.springframework.org/schema/lang/spring-lang-5.2.xsd

http://www.springframework.org/schema/osgi http://www.springframework.org/schema/osgi/spring-osgi-1.2.xsd

http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-5.2.xsd

http://www.springframework.org/schema/util http://www.springframework.org/schema/util/spring-util-5.2.xsd">

<context:annotaion-config />

</beans>

Cấu hình thư mục WebApplicationContext để chỉ định tập tin view sẽ hiển thị ở phần view cụ thể ở đây là tập tin JSP được lưu trữ trong thư mục WEB-INF/JSP/ và có đuôi là .jsp.

File cấu hình được cấu hình bằng Annotation

@Configuration

@EnableWebMvc

@EnableTransactionManagement

@ComponentScan(basePackages = {

"com.hhn.controllers" ,

"com.hhn.repository" ,

"com.hhn.service" ,

"com.hhn.validator"

})

public class WebApplicationContextConfig implements WebMvcConfigurer {

@Override

public void configureDefaultServletHandling(DefaultServletHandlerConfigurer configurer) {

configurer.enable();

}

@Bean

public InternalResourceViewResolver getInternalResourceViewResolver(){

InternalResourceViewResolver resolver = new InternalResourceViewResolver();

resolver.setViewClass(JstlView.class);

resolver.setPrefix("/WEB-INF/jsp/");

resolver.setSuffix(".jsp");

return resolver;

}

Tạo một lớp DispatcherServletInitializer để chỉ định tập tin cấu hình WebApplication Config và chỉ định đường dẫn mặc định lúc khởi chạy chương trình.

public class DispatcherServletInitializer extends AbstractAnnotationConfigDispatcherServletInitializer{

@Override

protected Class<?>[] getRootConfigClasses() {

return new Class[]{

TilesConfig.class ,

HibernateConfig.class ,

SpringSecurityConfig.class

};

}

@Override

protected Class<?>[] getServletConfigClasses() {

return new Class[]{

WebApplicationContextConfig.class

};

}

@Override

protected String[] getServletMappings() {

return new String[]{"/"};

}

}

Trong thư mục WEB-INF/jsp trong thư mục webapp tạo một file jsp mới là welcome.jsp và ghi các dòng lệnh html

<%@page contentType="text/html" pageEncoding="UTF-8"%>

<!DOCTYPE html>

<html>

<head>

<title>Welcome Page</title>

</head>

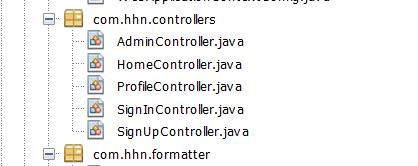
<body>

<h1>${message}</h1>

</body>

</html>  
</html>

Trong Thư mục Source Packages tạo một package riêng cho Controller



Hình 2.19 Package Controllers

Tạo một class Controller để chỉ định view sẽ thể hiện lên trên trang web.

@Controller

public class HomeController {

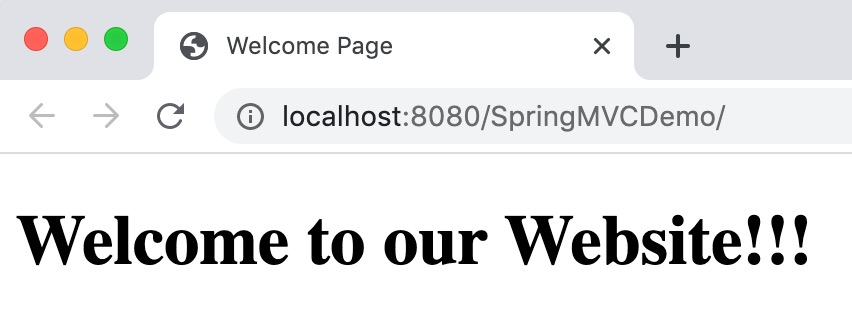
@RequestMapping(value = "/")

public String index(Model model) {

model.addAttribute("message",

"Welcome to our Website!!!");

return "welcome";  
}  
}



Hình 2.20 Chương Trình Đầu Tiên

#### Controller

Các controller là các Java Class thông thường được tạo với kết thúc là Controller để báo hiệu rằng mình là Controller, sau đó ta chỉ phương thức @Annotation Mapping Java là @Controller để các tập tin cấu hình trong mục config biết rằng tập tin đó là tập tin controller sẽ tương tác với các tập tin JSP và hiển thị view.

Các phương thức trong Controller thường được gắn Annotation @Request Mapping chỉ định đường dẫn URL ánh xạ vào phương thức controller đang viết.

@RequestMapping("/user/")

public String index(Model model , @RequestParam(value ="kw" , required = false , defaultValue = "") String kw ,

@RequestParam(required=false,defaultValue = "")Map<String,String> params , HttpServletRequest request ){

int page = Integer.parseInt(params.getOrDefault("page","1"));

model.addAttribute("postNewFeed" , this.postService.getPostNewFeed(kw,page));

model.addAttribute("counter" , this.postService.countPost());

model.addAttribute("likePost", new LikePost());

model.addAttribute("likeComment", new LikeComment());

model.addAttribute("notifications", new Notifications());

return "homePage";

}

Ta cũng có thể sử dụng @RequestMapiing cho lớp Controller, khi đó Spring sẽ xét các giá trị của @RequestMapping nằm trên lớp Controller trước khi ánh xạ phần còn lại của URL vào phương thức xử lý chỉ định.

#### Tag Libraries

JSP ( Java Server Page ) là công nghệ cho phép nhúng các mã nguồn Java vào trang HTML,các mã nguồn Java được bắt đầu bằng các thẻ của JSTL ( Java Standard Tag Library) hoặc cặp dấu <%%>.

JSTL là một bộ thư viện gồm các thẻ chuẩn được cung cấp bởi nhà phát triển của Java:Oracle,để sử dụng bộ thư viện ta có thể chỉ định nó thông qua tập thẻ taglib directives.Taglib được khai báo trong các trang jsp và chỉ định vị trí của thư viện bằng thuộc tính uri và thuộc tính prefix chỉ định biến bắt đầu để sử dụng các câu lệnh chứa trong đường dẫn uri.

<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

Ví dụ sử dụng model attribute lấy dữ liệu thông qua trang view JSP

@RequestMapping(value = "/user/profile")

public String profilePage(Model model , @RequestParam(value ="kw" , required = false , defaultValue = "") String kw ,

@RequestParam(value="userName" , required = false , defaultValue = "") String userName , HttpServletRequest request){

model.addAttribute("userProfile", this.userService.getUserProfile(userName));

model.addAttribute("postUserProfile" , this.postService.getPostsUserProfile(kw , userName ));

model.addAttribute("likePost", new LikePost());

model.addAttribute("likeComment", new LikeComment());

return "profilePage";

}

<form:form action="${addNewPost}" method="post" modelAttribute="post" enctype="multipart/form-data" >

<div class="form-group">

<label for="cate">Người Đăng Bài Viết </label>

<form:select id="user" path="user" class="form-control">

<c:forEach items="${userIdLoggedIn}" var="u">

<option value="${u.id}">${u.name}</option>

</c:forEach>

</form:select>

<form:errors path="categoryPost" cssClass="text-danger" element="div" />

</div>

<form:errors path="\*" cssClass="alert alert-danger" element="div" />

<div class="form-group">

<label for="content">Content</label>

<form:textarea rows="5" path="content" class="form-control" />

<form:errors path="content" cssClass="text-danger" element="div" />

</div>

#### WebDataBinder

Dùng để lấy dữ liệu từ đối tượng HttpServletRequest truyền lên từ phía clients để chuyển thành kiểu dữ liệu phù hợp của đối tượng Backing Bean và kiểm tra dữ liệu đầu vào.Để có thể điều chỉnh cách thức kết buộc dữ liệu (data binding), ta cần cấu hình đối tượng WebDatabinder trong Controller. Để các lớp Config quét thấy đối tượng này ta cần dùng @Annotation @Initbinder để chỉ định phương thức khởi động WebDatabinder.

@Autowired

private WebAppValidator userValidator;

@InitBinder

public void initBinder(WebDataBinder binder)

{

binder.setValidator(userValidator);

}

Trong các phương thức action khi cần gọi đối tượng này để Validate

@PostMapping("/signup")

private String addUser( Model model , @ModelAttribute(value="user")@Valid User user ,

BindingResult result){

if(userDetailsService.checkUserName(user.getUsername()) == true)

model.addAttribute("errUserNameDuplicate" , "UserName Đã Có Người Sử Dụng");

if(!result.hasErrors()){

if(this.userDetailsService.addUser(user) == true)

return "redirect:/signin";

else

model.addAttribute("errMsG" , "SomethingWrong !");

}

return "signUpPage";

}

#### Properties File

Các nhãn hiển thị trên trang web chẳng hạn như câu chữ tên ứng dụng “Social NetWork Charity” đều là những dòng câu lệnh hard-core trực tiếp từ tập tin .jsp.

Vì vậy Srping cung cấp một taglib tên là “Spring” với vai trò gọi các nhãn được lưu trữ tập tin message.properties được chỉ định trong file config webapplicationconfig.

<%@ taglib prefix="spring"

uri="http://www.springframework.org/tags" %>

<form:form method="post" modelAttribute="word">

<spring:message code="label.word" />

Để chỉ định tệp tin cấu properties ta cần cấu hình bean cho lớp ResourceBundleMessageSource để chỉ định giá trị tên của tập tin properties.

@Bean

public MessageSource messageSource(){

ResourceBundleMessageSource resource = new ResourceBundleMessageSource();

resource.setBasename("messages");

return resource;

}

#### View Resolver

Trong Interface InternalResourceViewResolver có chứa các kỹ thuật đặc biệt chẳng hạn như Redirect: một kỹ thuật chuyển người dùng đến một trang khác, đây là một thủ thuật được sử dụng rất phổ biến khi lập trình tương tác với các form khi người dùng submit một form nào đó để tránh việc người submit nhiều lần gây ra những lỗi ngoài ý muốn cho chương trình.Có 2 kỹ thuật Redirect View được View Resolver cung cấp đó là forward và redirect.

**Forward** sẽ chuyển một request hiện tại đến một phương thức request mapping khác. Dựa trên đường dẫn tiền tố forward và request đó vẫn là request ban đầu nên những giá trị đặt vào model khi bắt đầu request nó vẫn sẽ còn lưu giữ giá trị.

**Redirect** sẽ tạo nên một request mới và những giá trị đã đặt vào model khi bắt đầu request hiện tại sẽ mất đi.

**Multipart request** là một loại HTTP Request để gửi các tập tin và dữ liệu đến server.

Lớp **CommonsMultipartResolver** sẽquyết định một request có được chứa các file và chuyển các HTTP Request đó thành các thành các tham số và tập tin multipart hay không.

@Bean

public CommonsMultipartResolver multipartResolver(){

CommonsMultipartResolver resolver = new CommonsMultipartResolver();

resolver.setDefaultEncoding("UTF-8");

return resolver;

}

#### Upload các file ảnh lên Cloudinary

Đăng ký thông tin của bạn ở trên trang theo đường link [CloudinaryLink](https://cloudinary.com/users/register/free) để nhận các thông tin của cloud như là cloudname , api key , api secret, thêm các dependcy cần thiết để sử dụng API này.

<dependency>

<groupId>com.cloudinary</groupId>

<artifactId>cloudinary-http44</artifactId>

<version>1.29.0</version>

</dependency>

<dependency>

<groupId>com.cloudinary</groupId>

<artifactId>cloudinary-taglib</artifactId>

<version>1.29.0</version>

</dependency>

Tạo Bean Cloudinary và bắt đầu tạo phương thức upload từ form post có enctype=mulitpart.

@Bean

public Cloudinary cloudinary(){

Cloudinary c = new Cloudinary(ObjectUtils.asMap(

"cloud\_name", "decmhyieh",

"api\_key" , "413756871999116" ,

"api\_secret", "G34UwtLLRY2hg\_PCrmzs5pT9RTo",

"secure", true

));

return c;

}

Map r = this.cloudinary.uploader().upload(post.getFile().getBytes(),

ObjectUtils.asMap("resource\_type","auto"));

post.setImage((String) r.get("secure\_url"));

return this.postRepository.addNewPost(post);

} catch (IOException ex) {

System.out.println("Add Post" + ex.getMessage());

}

#### Tích hợp Hibernate vào Spring

Thêm các dependency cần thiết vào tập tin pom.xml

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>5.4.10.Final</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.18</version> </dependency>

Tạo thư mục resources trong thư mục src/main và tạo tập tin database.properties chứa các dữ liệu cấu hình của Hibernate.

hibernate.dialect=org.hibernate.dialect.MySQLDialect

hibernate.showSql=true

hibernate.connection.driverClass=com.mysql.cj.jdbc.Driver

hibernate.connection.url=jdbc:mysql://localhost:3308/socialnetwork

hibernate.connection.username=root

hibernate.connection.password=12345678

Tạo thêm Lớp Hibernate Config để cấu hình bằng mã nguồn Java.

@Configuration

@PropertySource("classpath:databases.properties")

public class HibernateConfig {

@Autowired

private Environment env;

@Bean

public LocalSessionFactoryBean getSessionFactory(){

LocalSessionFactoryBean sessionFactory = new LocalSessionFactoryBean();

sessionFactory.setPackagesToScan("com.hhn.pojos");

sessionFactory.setDataSource(dataSource());

sessionFactory.setHibernateProperties(hibernaProperties());

return sessionFactory;}

@Bean

public DataSource dataSource(){

DriverManagerDataSource dataSource = new DriverManagerDataSource();

dataSource.setDriverClassName(env.getProperty("hibernate.connection.driverClass"));

dataSource.setUrl(env.getProperty("hibernate.connection.url"));

dataSource.setUsername(env.getProperty("hibernate.connection.username"));

dataSource.setPassword(env.getProperty("hibernate.connection.password"));

return dataSource;

}

Kỹ thuật quản lý giao tác Transaction là một kỹ thuật lập trình sử dụng phổ biến hiện nay để phát triển các ứng dụng thương mại để đảm bảo tính nhất quán và toàn vẹn dữ liệu.Bằng cách sử dụng Hibernate Transaction Manager sẽ kết buộc session từ một Session Factory vào một luồng cho phép một Session cho mỗi Session Factory.

Thêm vào lớp Hibernate Config phương thức Transaction Manager

@Bean

public HibernateTransactionManager transactionManager(){

HibernateTransactionManager h = new HibernateTransactionManager();

h.setSessionFactory(getSessionFactory().getObject());

return h;

}

#### Template with Tiles

Apache là một framework mã nguồn mở giúp tái sử dụng lại các tập tin front-end khi xây dựng các front-end template,với khả năng định nghĩa các tiles con để lắp ráp thành một trang web hoàn chỉnh khi ứng dụng được chạy, những thành phần con này còn có thể kế thừa nhau để tạo nên các tiles mới có một phần tiles của tiles cha.

<dependency>

<groupId>org.apache.tiles</groupId>

<artifactId>tiles-extras</artifactId>

<version>3.0.8</version>

</dependency>

Tạo một file tiles.xml trong thư mục WEB-INF/ và định nghĩa các thẻ definition.

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE tiles-definitions PUBLIC "-//Apache

Software Foundation//DTD Tiles Configuration 3.0//EN"

"http://tiles.apache.org/dtds/tiles-config\_3\_0.dtd">

<tiles-definitions>

<definition name="homePage"

template="/WEB-INF/layout/base.jsp">

<put-attribute name="title" value="Home Page" />

<put-attribute name="header" value="/WEB-INF/layout/header.jsp" />

<put-attribute name="content" value="/WEB-INF/jsp/index.jsp"/>

<put-attribute name="footer" value="/WEB-INF/layout/footer.jsp" />

</definition>

<definition name="postByCategoryPost" extends ="homePage">

<put-attribute name="content" value="/WEB-INF/jsp/pagePostByCateId.jsp" />

</definition>

<definition name="commentPage" extends ="homePage">

<put-attribute name="content" value="/WEB-INF/jsp/comment.jsp" />

</definition>

<definition name="profilePage" extends="homePage">

<put-attribute name="title" value="Profile User Page" />

<put-attribute name="content" value="/WEB-INF/jsp/profile.jsp"/>

</definition>

Trong package Config tạo một Lớp TilesConfig để chỉ định tập tin xml chứa các thẻ definition định nghĩa các tiles.

@Configuration

public class TilesConfig {

@Bean

public UrlBasedViewResolver urlBasedViewResolver(){

UrlBasedViewResolver resolver = new UrlBasedViewResolver();

resolver.setViewClass(TilesView.class);

resolver.setOrder(-2);

return resolver;

}

@Bean

public TilesConfigurer tilesConfigurer(){

TilesConfigurer configurer = new TilesConfigurer();

configurer.setDefinitions("/WEB-INF/tiles.xml");

configurer.setCheckRefresh(true);

return configurer;

}

}

Trong Controller các phương thức trả về các view tương ứng với tên các tiles được tạo trong file xml.

@RequestMapping(value = "/user/profile")

public String profilePage(Model model , @RequestParam(value ="kw" , required = false , defaultValue = "") String kw ,

return "profilePage";

}

#### Bean Validation

Là một phương thức kiểm tra dữ liệu đầu vào quan trọng của Spring cho phép mô tả các ràng buộc trên các đối tượng thông qua cấu hình các @Annotation.

Để cài đặt Bean validator ta cũng cần thêm các dependency vào trong project.

<dependency>

<groupId>org.hibernate.validator</groupId>

<artifactId>hibernate-validator</artifactId>

<version>6.1.1.Final</version>

</dependency>

Tạo ra các ràng buộc dữ liệu cho một lớp bằng các Annotation như @Size,@Notnull,@Min,@Max,để hiển thị các lỗi đó ra các view ta có chỉ định các biến chứa nội dung lỗi được lưu trong tập messages.properties và cấu hình trong lớp WebApplication Config để kiểm tra dữ liệu.

@Bean(name = "validator")

public LocalValidatorFactoryBean validator(){

LocalValidatorFactoryBean bean = new LocalValidatorFactoryBean();

bean.setValidationMessageSource(messageSource());

return bean;

}

Entity

@Table(name ="user")

public class User implements Serializable{

public static final String ADMIN = "ROLE\_ADMIN";

public static final String USER = "ROLE\_USER";

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id ;

@Size(min = 1 , max = 50 , message = "{user.username.lenErr}")

private String username;

@NotEmpty(message = "user.password.notEmpty")

private String password;

@Pattern(regexp = "^[A-Za-z0-9+\_.-]+@(.+)$" ,

message = "{user.email.error.invalidMsg}")

private String email;

#### Spring Validation

Trong package Validator tạo ra một lớp cụ thể là UserValidator hiện thực lại Interface Validator chưa hai phương thức chính là supports() cho biết validator có được phép kiểm tra lớp chỉ định trong phương thức đó hay không và phương thức validate() là phương thức được để kiểm tra các trường dữ liệu trong lớp đó.

@Override

public boolean supports(Class<?> clazz) {

return User.class.isAssignableFrom(clazz);

}

@Override

public void validate(Object target, Errors errors) {

User u = (User) target;

if (!u.getPassword().trim().equals(u.getConFirmPassWord().trim()))

errors.rejectValue("password","user.password.error.notMatchMsg");

if(u.getFile().getSize() == 0 )

{

errors.rejectValue("file", "user.file.error.noFileChoose");

}

if(u.getUsername().contains(" "))

{

errors.rejectValue("username", "user.username.spaceError");

}

if(u.getBirthday() == null)

{

errors.rejectValue("birthday" , "user.birthday.DateError");

} }

Sau khi cấu hình lớp Validator thì trong controller tạo một phương thức initbinder chỉ định Lớp Validator vừa mới tạo trong Controller để hiển thị các lỗi ra view theo phương thức.

@InitBinder

public void initBinder(WebDataBinder binder)

{

binder.setValidator(userValidator);

}

@PostMapping("/signup")

private String addUser( Model model , @ModelAttribute(value="user")@Valid User user ,

BindingResult result){

if(!result.hasErrors()){

if(this.userDetailsService.addUser(user) == true)

return "redirect:/signin";

else

model.addAttribute("errMsG" , "SomethingWrong !");

}

return "signUpPage";

#### Spring Security

Là một lớp cấu hình quan trọng trong các ứng dụng web Spring khi nó quan tâm đến các đối tượng HTTPRequest và HttpResponse,các request thực hiện thông qua trình duyệt,Web Service,HTTP Client hoặc bằng Ajax.Với các Servlet Filter xây dựng sẵn và chỉ cần cấu hình các filter phù hợp với nhu cầu sử dụng để kiểm tra các HTTP Request trước khi được phép thực hiện một phương thức hay công việc cụ thể nào đó.

Cấu hình Spring Security bằng các dependency

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-web</artifactId>

<version>5.2.1.RELEASE</version>

</dependency>

<dependency>

<groupId>org.springframework.security</groupId>

<artifactId>spring-security-config</artifactId>

<version>5.2.1.RELEASE</version>

</dependency>

Tạo Lớp Spring Security Config để cấu hình đăng nhập và mã hóa password, cấu hình lớp UserDetailsService có phương thức getUserByUsername để gọi nó trong lớp Cấu hình Spring Security.

@Override

protected void configure(AuthenticationManagerBuilder auth) throws Exception {

auth.userDetailsService(userDetailsService)

.passwordEncoder(passwordEncoder());

}

@Override

protected void configure(HttpSecurity http) throws Exception {

http.formLogin().loginPage("/signin")

.usernameParameter("username")

.passwordParameter("password");

http.formLogin().defaultSuccessUrl("/user/").failureUrl("/signin?error");

http.logout().logoutSuccessUrl("/signin");

http.exceptionHandling().accessDeniedPage("/signin?accessDenied");

http.authorizeRequests().antMatchers("/").permitAll()

.antMatchers("/admin/\*\*").access("hasRole('ROLE\_ADMIN')")

.antMatchers("/user/\*\*").access("hasAnyRole('ROLE\_USER', 'ROLE\_ADMIN')");

// http.exceptionHandling().accessDeniedPage("/user/?accessAuthorize");

http.csrf().disable();

}

@Override

public void configure(WebSecurity web) throws Exception {

super.configure(web); //To change body of generated methods, choose Tools | Templates.

}

# Hệ thống mạng xã hội từ thiện

## Nội dung chương

Chương ba sẽ trình bày nội dung về giao diện trực quan của hệ thống khi chạy lên, các chức năng của hệ thống tương tác với người sử dụng. Nội dung phân tích thiết kế hệ thống và thiết kế cơ sở dữ liệu cho chương trình.

## Giới thiệu hệ thống

Hệ thống mạng xã hội từ thiện được xây dựng với mục đích sử dụng cho phép người dùng có thể đăng các bài viết chia sẻ về các món hàng từ thiện trên dòng trạng thái chung của cộng đồng.

Trên dòng trạng thái chính có danh sách các bài viết được phân trang theo số lượng các bài viết trên diễn đàn, người dùng có thể like và bình luận trên bài viết của người khác khi có các tác động này người đăng bài viết sẽ có 1 thông báo mới báo hiệu là có một người dùng khác đã tương tác trên bài viết của mình. Và họ cũng có thể trực tiếp giao tiếp với nhau thông qua bình luận dưới bài viết để hỏi thêm các thông tin chi tiết của món hàng cụ thể như: giá cả, địa điểm sản xuất và các thông tin phụ của món hàng, ngoài ra họ cũng có thể like bình luận người khác tất nhiên sẽ có thông báo bạn đã like bình luận của họ.

Đặc biệt họ cũng có thể tổ chức đấu giá cho các sản phẩm đăng trên diễn đàn này, người dùng khác khi muốn mua sản phẩm có thể chọn giá tiền mà mình muốn đấu giá và bắt đầu đăng ký thông tin đấu giá của mình vào món hàng. Người tổ chức đấu giá có thể thấy lịch sử đấu giá của những người đã đưa ra giá đấu giá của họ và người tổ chức đấu giá cũng có thể quyết định ai là người chiến thắng đấu giá cho bài viết này, sau khi quyết định người chiến thắng hệ thống sẽ có thông báo mới cho việc báo với người ra giá rằng họ là người chiến thắng, họ có thể xem thông tin chi tiết các món hàng mà mình đã chiến thắng đấu giá, sau khi cuộc đấu giá đã có người chiến thắng thì những người khác họ không thể đấu giá cho phiên đấu giá này nữa và người tổ chức cũng không thể chọn thêm người đấu giá cho phiên đấu giá này nữa.

Người dùng được hỗ trợ các chức năng tìm kiếm bài viết theo nội dung bài viết, họ cũng trang thông tin cá nhân của mình để xem các bài viết và tương tác với bài viết các chức năng cụ thể như cập nhật bài viết và xóa bài viết.

Người dùng cũng có thể báo cáo các bài viết và bình luận của người dùng khác kèm nội dung lý do báo cáo bài viết chẳng hạn như người dùng dùng từ ngữ không đúng đắn hoặc là có hàm ý khiêu khích.

Quản trị viên ngoài các chức năng đã có của người dùng họ cũng có thể duyệt các báo cáo đã được gửi lên người dùng, họ có thể xem xét là gỡ bài viết khỏi dòng trạng thái chung của cộng đồng hoặc là bỏ qua báo cáo, các báo cáo sau khi đã được duyệt sẽ không thể duyệt thêm được nữa. Sau khi duyệt báo cáo hệ thống sẽ có thông báo đến người gửi báo cáo hoặc người báo cáo hoặc bình luận rằng bài viết hay bình luận của họ đã được gỡ xuống dòng trạng thái chung của cộng đồng và người gửi báo cáo cũng có thông báo rằng quản trị viên đã duyệt báo cáo hay bỏ qua báo cáo của họ.

Hệ thống cũng có các yêu cầu phi chức năng như người dùng không thể báo cáo bài viết của chính họ hay là họ không thể vào các trang hệ thống tương tác của admin như khả năng báo cáo người dùng khác.

Hệ thống có hỗ trợ khả năng đăng nhập và đăng ký thông tin người dùng, đăng ký thông tin người dùng sẽ được mã hóa mật khẩu trong cơ sở dữ liệu và bắt buộc họ phải có avatar làm ảnh nền cho tài khoản cá nhân của họ.

## Phân tích thiết kế hệ thống

### Các yêu cầu chức năng của hệ thống.

Xây dựng hệ thống lưu trữ bài viết và thông tin các bình luận, các like của người dùng và thông tin cá nhân của người dùng.

Hệ thống có hỗ trợ tương tác giữa người dùng với người dùng và quản trị viên với người dùng.

Hệ thống cũng có có thể cập nhật nội dung bài viết và xóa các bài viết và tương tác các chức năng của bài viết như báo cáo bài viết like và bình luận bài viết.

### Các yêu cầu phi chức năng của hệ thống

Giao diện đẹp thân thiện với người dùng, phù hợp với các loại màn hình như máy tính để bàn, ipad, tablet và điện thoại.

Tốc độ truy cập nhanh chóng.

Bảo mật trang web khi người dùng không thể tương tác các chức năng mà chỉ có người quản trị viên mới có thể tương tác được.

Có hỗ trợ hệ thống notification center khi người dùng tương tác sẽ có thông báo cụ thể ai là người đã tương tác với bài viết hay đã đấu giá trên bài viết của minh.

### Các ký hiệu cơ bản trong UML

|  |  |
| --- | --- |
| **Tên ký hiệu** | **Ký hiệu** |
| Actor |  |
| Association |  |
| Directed Association |  |
| Dependency |  |
| Aggregation |  |
| Composition |  |
| UseCae |  |
| UseCase Subject |  |
| Include |  |
| Extend |  |
| Lifeline |  |
| Message |  |
| Reply Message |  |
| Self Message |  |
| Boundary |  |
| Entity |  |
| Control |  |

Bảng 3.3.3.1 Các ký hiệu cơ bản trong UML

### Biểu đồ trường hợp sử dụng(Use Case diagram)

#### Xác định các Actor

Quản trị viên(admin): Là người có chức vụ cao nhất có toàn bộ quyền hạn và có thể tương tác các chức năng mà chỉ có quản trị viên mới tương tác được.

Người dùng(user): Tài khoản người dùng được gắn quyền mặc định khi đăng ký tài khoản, họ sẽ có đầy đủ các chức năng có thể tương tác trên mạng xã hội và bị giới hạn các chức năng chi có admin mới tương tác được.

#### Các trường hợp sử dụng của actor

**Liệt kê các trường hợp sử dụng**

Đăng ký thông tin tài khoản.

Đăng nhập và Đăng xuất thông tin người dùng.

Tìm kiếm bài viết.

Trang thông tin trạng thái gồm các danh sách bài viết được sắp xếp mới nhất lên trên đầu tiên.

Trang thông tin cá nhân bao gồm thông tin chi tiết của người dùng và nơi họ có thể đăng, xóa, cập nhật bài viết của họ.

Chức năng like bài viết, bình luận bài viết, like bình luận của người dùng khác.

Chức năng báo cáo bài viết, báo cáo bình luận của người dùng khác.

Đăng bài viết mới, cập nhật bài viết, xóa bài viết.

Chức năng đấu giá

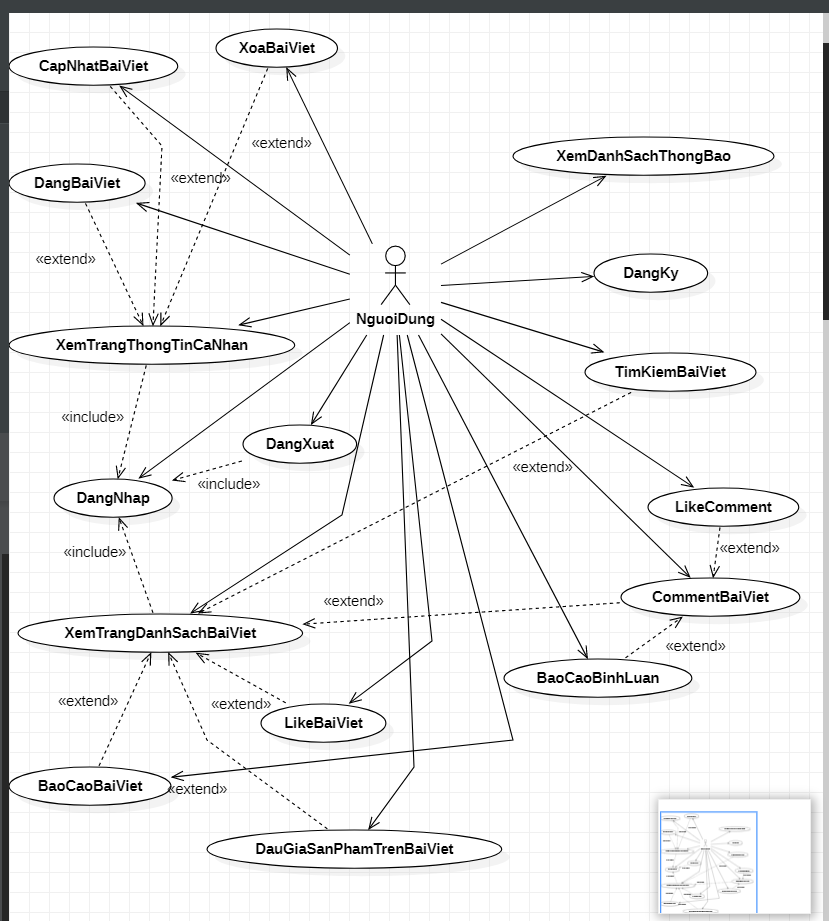
Hệ thống thông báo

Quản trị của admin: xem danh sách các báo cáo bài viết bình luận, thống kê số bình luận số like của bài viết của hệ thống.

#### Use Case Diagram

Use-Case diagram: Là các chuỗi sự kiện mô tả sự tương tác giữa user và hệ thống. Cung cấp cái nhìn tổng thể về những gì mà hệ thống phải làm và ai sẽ dùng nó.

**Use Case Diagram cho người dùng**



Hình 3.3.4.3.1 Biểu đồ trường hợp sử dụng của người dùng (Use Case Diagram)

**Use case cho quản trị viên**

Ngoài các chức năng được cấp quyền chỉ cho quản trị viên, họ cũng có thể sử dụng toàn bộ các chức năng của một người dùng thông thường.



Hình 3.3.4.3.2Biểu đồ trường hợp sử dụng của quản trị viên(Use Case Diagram)

**Đặc tả các Use Case**

**1. Đăng ký tài khoản người dùng**

**Tóm tắt**

Use-case cho phép actor đăng ký để trở thành thành viên sử dụng mạng xã hội này.

Người dùng chưa có tài khoản là actor của use-case này.

**Dòng sự kiện**

Use-Case bắt đầu khi actor chọn chức năng”Đăng ký thành viên”.

**Dòng sự kiện chính**

Actor chọn chức năng”đăng ký”.

Hệ thống hiển thị mục đăng ký thành viên.

Actor điền vào các thông tin trường dữ liệu đầy đủ.

Actor chọn nút “Đăng ký”.

Hệ thống lưu toàn bộ thông tin mà actor vừa xác nhận xuống database.

Hệ thống chuyển người dùng sang trang đăng nhập.

**Các dòng sự kiện phụ**

Nếu như người dùng không điền đầy đủ các trường dữ liệu mà đã bấm đăng ký thì hệ thống sẽ giữ người dùng ở trang đăng ký kèm các thông tin lỗi các trường dữ liệu còn thiếu dữ liệu hoặc nhấp không đúng theo yêu cầu.

**Các yêu cầu đặc biệt**

Người dùng phải có ảnh avatar.

**Điều kiện tiên quyết**

Các usernam để đăng nhập đều là đồng nhất không có usename được phép trùng lắp trong database.

**Điều kiện hoàn tất**

Điền đủ các trường dữ liệu hợp lệ sau đó đưa người dùng quay về trang Đăng nhập

**2. Đăng nhập**

**Tóm tắt**

Use-case này cho phép actor đăng nhập vào hệ thống với vai trò là người dùng hay quản trị viên.

Người dùng và quản trị viên là actor của use-case này.

**Dòng sự kiện**

Use-case bắt đầu khi actor chọn chức năng “đăng nhập” vào hệ thống.

**Dòng sự kiện chính**

Actor chọn chức năng đăng nhập.

Hệ thống hiển thị form hai khung nhập username và password.

Actor nhập tên truy cập và mật khẩu.

Actor chọn nút đăng nhập.

Hệ thống kiểm tra tên truy cập và mật khẩu, xác nhận là hợp lệ và cho phép actor đăng nhập vào hệ thống với chức vụ phù hợp được định sẵn có.

**Các dòng sự kiện khác**

Nếu actor nhập tên truy cập và mật khẩu không đúng thì hệ thống sẽ báo lỗi và giữ người dùng ở trang đăng nhập cho đến khi họ nhập đúng thì chuyển qua trang chủ.

**Các yêu cầu đặc biệt**

Không có

**Điều kiện tiên quyết**

Không có

**Điều kiện hoàn tất**

Hệ thống tạo session và lưu thông tin người dùng và chuyển người dùng qua trang chủ.

**Điểm mở rộng**

Không có

**3. Đăng xuất người dùng**

**Tóm tắt**

Use-case hoạt động khi người dùng muốn đăng xuất khỏi hệ thống.

**Dòng sự kiện**

Use-case này bắt đầu khi người dùng nhấp chọn nút “đăng xuất”.

**Dòng sự kiện chính**

Actor chọn chức năng đăng xuất

Hệ thống xử lý bằng Spring Security được cấu hình để khi người dùng chọn nút đăng xuất sẽ gỡ bỏ session và đưa người dùng trở về trang đăng nhập.

**Các dòng sự kiện khác**

Không có

**Điều kiện tiên quyết**

Người dùng đã đăng nhập từ trước.

**Điều kiện hoàn tất**

Người dùng được đưa trở về trang đăng xuất mà không có những lỗi xảy ra trong lúc thực hiện.

**Điểm mở rộng**

Không có.

**4. Tìm kiếm bài viết**

**Tóm tắt**

Người dùng muốn tìm bài viết theo từ khóa.

**Dòng sự kiện**

Use-case này hoạt động khi người dùng nhập vào thanh tìm kiếm từ khóa phù hợp và nhấn nút Enter.

**Dòng sự kiện chính**

Actor nhập vào từ khóa cụ thể và nhấn chọn enter, hệ thống sẽ xử lý khi có từ khóa truyền vào để sàng lọc ra các bài viết phù hợp và thể hiện tại trang chủ cho người dùng xem.

**Các dòng sự kiện khác**

Nếu như từ khóa truyền vào không khớp với bài viết nào thì danh sách bài viết hiển thị ra tại trang chủ sẽ không có bài viết nào và use-case kết thúc.

**Điều kiện tiên quyết**

Người dùng phải nhập từ khóa vào thanh tìm kiếm

Người dùng đã đăng nhập

**Điều kiện hoàn tất**

Hệ thống xử lý hiển thị bài viết đúng với từ khóa mà người dùng đã nhập vào.

**Điểm mở rộng**

Không có

**5. Trang thông tin trạng thái**

**Tóm tắt**

Trang hiển thị các bài viết mới nhất và được sắp xếp phân trang bài viết mới lên trước.

**Dòng sự kiện**

Sau khi người dùng đăng nhập chuyển qua trang chính hệ thống sẽ xử lý hiện danh sách các bài viết lên dòng trạng thái của người dùng.

**Dòng sự kiện chính**

Hệ thống xử lý hiển thị danh sách các bài viết mới nhất lên đầu tiên, cùng với các thông tin của đăng bài viết, số like, số bình luận, thông tin đấu giá, thông tin người đăng bài viết.

**Các dòng sự kiện khác**

Người dùng cũng có thể like và bình luận trên từng bài viết,người dùng cũng có thể đấu giá trên từng bài viết nếu như bài viết đó chưa có người chiến thắng.

**Điều kiện tiên quyết**

Người dùng phải đăng nhập

**Điều kiện hoàn tất**

Danh sách bài viết phải có bài viết để hiển thị.

**Điểm mở rộng**

Người dùng cũng có thể like và bình luận trên bài viết, người dùng cũng có thể báo cáo bài viết, và đấu giá trên sản phẩm đó.

**6. Chức năng like bài viết**

**Tóm tắt**

Là một chức năng cho phép người dùng có thể like bài viết trên trang Danh sách bài viết chính.

Người dùng và quản trị viên đều là actor của use-case này.

**Dòng sự kiện**

Use-case này được thực hiện khi người dùng chọn nút”Like”.

**Dòng sự kiện chính**

Người dùng chọn nút “like” trên từng bài viết khi thực hiện action hệ thống sẽ xử lý và lưu thông tin dữ liệu người like bài viết và post được like đó vào bảng trong cơ sở dữ liệu,đồng thời khi thực hiện thêm dữ liệu thì hệ thống cũng tự động load lại trang web và tăng số like tổng của bài viết lên một đơn vị.

Ngoài ra, khi người dùng nhấn lại nút like lần nữa hệ thống sẽ tự động xử lý xóa thông tin người vừa unlike bài viết đó và cập nhật lại tổng số like của bài viết đó trong bảng dữ liệu.

**Dòng sự kiện khác**

Khi người dùng like bài viết hệ thống cũng sẽ gửi một thông báo mới tới cho người đăng bài viết đó rằng bạn đã tương tác với bài viết của họ.

**Điều kiện tiên quyết**

Bài viết đó chưa bị report, người dùng đã đăng nhập.

**Điều kiện hoàn tất**

Bài viết xử lý toàn bộ các tác vụ liên quan trong việc like bài viết và load lại trang web để cập nhật.

**Điểm mở rộng**

Không có.

**7. Chức năng bình luận bài viết**

**Tóm tắt**

Là một chức năng cho phép người dùng có thể thêm một bình luận vào bài viết trên trang Danh sách bài viết chính.

Người dùng và quản trị viên đều là actor của use-case này.

**Dòng sự kiện**

Use-case này được thực hiện khi người dùng chọn nút”Bình luận”.

**Dòng sự kiện chính**

Người dùng sau khi nhấn nút”Bình luận” sẽ được chuyển đến trang bình luận có các bình luận trước đó của bài viết và dữ liệu số lượt bình luận trên bài viết kèm theo dữ khả năng report một bình luận của người khác.

Người sau khi nhập nội dung bình luận và chọn nút”bình luận” hệ thống sẽ xử lý thêm dữ liệu vào bảng Comments của cơ sở dữ liệu và load lại trang web để xem thay đổi mới.

**Dòng sự kiện khác**

Hệ thống sẽ báo lỗi nếu như dữ liệu bình luận quá ngắn do chưa nhập bình luận.

**Điều kiện tiên quyết**

Người dùng phải đăng nhập vào bài viết.

**Điều kiện hoàn tất**

Bài viết xử lý toàn bộ các tác vụ liên quan trong việc bình luận bài viết và load lại trang web để cập nhật.

**Điểm mở rộng**

Gửi thông báo khi có người mới tương tác khi thêm một bình luận mới vào bài viết đó,người đăng bài viết sẽ nhận thông báo rằng có người dùng khác đã tương tác với bài viết của họ.

Trên mỗi bình luận người dùng cũng có thể like bình luận của một người khác đã bình luận trong bài viết, cách thức hoạt động của nó cũng tương tự như khi người dùng like bài viết về mặt logic.

**8. Trang thông tin cá nhân**

**Tóm tắt**

Là trang người dùng có thể xem thông tin cá nhân của mình cũng như tương tác các chức năng sẵn có mà quyền hạn của người dùng đó được phép.

**Dòng sự kiện**

Khi người dùng chọn vào nút “My profile”

**Dòng sự kiện chính**

Sau khi được chuyển trang Trạng thái chính, người dùng có thể bấm chọn vào nút “My Profile” để vào trang cá nhân chính của mình. Ở trang này, hệ thống sẽ xử lý để lấy ra thông tin cá nhân của họ và danh sách các bài viết mà họ đã đăng lên,ngoài ra họ cũng có thể tương tác các chức năng như đăng bài viết mới,chỉnh sửa bài viết cũ và xóa những bài viết nếu như hệ thống cho phép họ xóa.Ngoài các chức năng sẵn có của người dùng, quản trị viên cũng có thêm các danh mục để thống kê và các chức năng liên quan chỉ có quản trị viên mới có thể tương tác.

**Dòng sự kiện khác**

Trang cá nhân là nơi để người dùng xem các thông tin cá nhân của mình, cũng như tương tác với các bài viết, quản trị viên có thể duyệt báo cáo và thống kê dữ liệu khi truy cập vào trang cá nhân chính của họ.

**Điều kiện tiên quyết**

Người dùng phải đăng nhập

**Điều kiện hoàn tất**

Người dùng được qua trang cá nhân của họ sau khi họ chọn nút “My Profile”

**Điểm mở rộng**

Đây là nơi để họ tương tác các chức năng mở rộng khác.

**9.Đăng,Xóa,Cập nhật bài viết.**

**Tóm tắt**

Khi người truy cập trang thông tin cá nhân họ có thể đăng bài viết mới của mình ở đây

Họ có thể cập nhật các bài viết đã đăng

Họ có thể xóa các bài viết nếu như bài viết đó có thể xóa

**Dòng sự kiện**

Khi người dùng tương tác với các sự kiện tương ứng.

**Dòng sự kiện chính**

Khi người dùng chọn nút “Đăng bài viết mới” hệ thống sẽ chuyển người dùng đến trang đăng bài viết, sau khi điền đủ các trường dữ liệu cần thiết, và ấn chọn nút Đăng bài, hệ thống sẽ upload ảnh lên cloudinary đồng thời thêm dữ liệu vào bảng cơ sở dữ liệu.

Khi chọn nút”Cập nhật bài viết” hệ thống sẽ chuyển đến trang cập nhật bài viết cho phép họ cập nhật lại nội dung bài viết hoặc ảnh bài viết mới,sau khi nhập lại nội dung bài viết mới, hoặc chọn giữ nguyên nội dung bài viết thì họ có thể chọn cập nhật và hệ thống sẽ tiến hành lưu các thay đổi sau đó đưa đưa người dùng quay trở lại trang chủ.

Khi người dùng chọn “Xóa bài viết” hệ thống sẽ tìm kiếm bài viết phù hợp theo id của bài viết đó trong danh sách, nếu như có thì sẽ bài viết đó và chuyển người dùng về trang chủ.

**Dòng sự kiện khác**

Không có

**Điều kiện tiên quyết**

Người dùng phải đăng nhập và truy cập vào trang cá nhân.

**Điều kiện hoàn tất**

Hệ thống xử lý xong request và chuyển người dùng quay về trang chủ để kết thúc một hành động để lưu dữ liệu vào các bảng trong cơ sở dữ liệu.

**Điểm mở rộng**

Không có

**10. Báo cáo bài viết, báo cáo bình luận**

**Tóm tắt**

Khi người truy cập trang Trạng thái chung,họ có thể chọn các bài viết hay các bình luận để báo cáo lên quản trị viên.

**Dòng sự kiện**

Tại trang trạng thái chung người dùng có thể chọn nút”Báo cáo bài viết” hay”Báo cáo bình luận” tại bài viết hay bình luận tương ứng.

**Dòng sự kiện chính**

Khi người dùng chọn các chức năng tương ứng như báo cáo bài viết hay báo cáo bình luận, hệ thống sẽ chuyển người dùng đến trang “Đăng báo cáo” ở đây họ có thể chọn các quản trị viên có sẵn đang trực thuộc ở trang web để gửi báo cáo, sau khi đã điền đầy đủ các trường thông tin dữ liệu,hệ thống sẽ chuyển người dùng đến trang xác nhận đã gửi báo cáo và đưa người dùng về trang chủ.

**Dòng sự kiện khác**

Dữ liệu đầu vào phù hợp thì hệ thống mới tiến hành thực hiện, nếu dữ liệu không phù hợp, hệ thống sẽ báo lỗi và giữ người tại trang báo cáo bài viết bình luận cho đến khi họ hoàn tất, hoặc bấm hủy bỏ để quay lại trang chủ,hủy bỏ việc gửi báo cáo.

**Điều kiện tiên quyết**

Người dùng phải đăng nhập.

**Điều kiện hoàn tất**

Hệ thống xử lý xong request và chuyển người dùng quay về trang chủ để kết thúc một hành động để lưu dữ liệu vào các bảng trong cơ sở dữ liệu.

**Điểm mở rộng**

Không có

**11. Chức năng đấu giá**

**Tóm tắt**

Trong các bài viết đều có thể là nơi để người dùng có thể đấu giá trên bài viết của người dùng khác đăng lên và người đăng bài viết họ có thể quyết định ai là người chiến thắng cuộc đấu giá đó.

**Dòng sự kiện**

Tại trang Trạng thái chung bấm chọn vào “đấu giá” để bắt đầu đấu giá sản phẩm.

**Dòng sự kiện chính**

Tại trang đấu giá Hệ thống sẽ xử lý và thể hiện danh sách thông tin của bài viết đấu giá và thông tin của người đăng bài đấu giá, ngoài ra còn có thông tin những người đấu giá khác đã đấu lên sản phẩm trước đó( chỉ có người đăng bán đấu giá mới nhìn thấy). Đối với người dùng vào đấu giá sẽ chỉ nhìn thấy giá khởi đầu của món hàng và giá cao nhất đang đấu giá hiện tại.

Người dùng khác khi họ muốn đấu giá sản phẩm có thể điền giá tiền mình muốn đấu giá vào mục ghi giá tiền,nếu như giá tiền hợp lệ hệ thống sẽ đưa ra thông báo “bạn đã đấu giá thành công”, nếu như giá tiền quá thấp so với giá khởi điểm hoặc giá hiện tại hệ thống sẽ báo lỗi.

Đối với người tổ chức đấu giá họ có thể quyết định ai là người chiến thắng đấu giá đối với sản phẩm của mình, sau khi chọn được người chiến thắng hệ thống sẽ thông báo cho người đã chiến thắng đấu giá thông tin họ đã chiến thắng đấu giá và họ có thể kiểm tra trong trang những sản phẩm đã chiến thắng đấu giá.

**Các dòng sự kiện khác**

Nếu như người dùng nhập giá tiền đấu giá sản phẩm quá nhỏ so với giá điểm hoặc giá tiền đang chiến thắng hiện giờ, hệ thống sẽ thông báo lỗi.

Ngoài ra đối với những bài viết đã có người chiến thắng đấu giá thì họ không thể chọn người chiến thắng thêm nữa và những người khác cũng không thể đưa ra giá đấu giá mới cho sản phẩm đó nữa.

**Các yêu cầu đặc biệt**

Không có

**Điều kiện tiên quyết**

Người dùng trước tiên phải đăng nhập hệ thống.

**Điều kiện hoàn tất**

Chọn được người chiến thắng cho cuộc đấu giá.

Người dùng thêm giá tiền mới cho buổi đấu giá.

Sau khi xử lý xong hệ thống sẽ có thông báo thành công và ngầm định thực hiện các hoạt động thêm dữ liệu vào các bảng liên quan và gửi thông báo cho người chiến thắng.

**Điểm mở rộng**

Không có.

**12 Hệ thống thông báo**

**Tóm tắt**

Use-case cho phép người dùng có thể xem các thông báo mới,khi các người dùng khác tương tác với các bài viết, bình luận và chương trình đấu giá của họ.

Người quản trị và người dùng là actor của use-case này.

**Dòng sự kiện**

Use-case bắt đầu khi người dùng khác tương tác với các chức năng có thể khiến cho hệ thống thông báo hoạt động.

**Dòng sự kiện chính**

Khi có các tương tác mới giữa người dùng và những người dùng khác, hệ thống sẽ ghi nhận các hoạt động đó trích xuất ra dữ liệu người dùng và đóng gói vào đối tượng thông báo thêm vào bảng Notifications.Khi dữ liệu được thêm vào người dùng cũng được xem như là nhận được thông báo mới từ hệ thống.

**Các dòng sự kiện khác**

Không có

**Các yêu cầu đặc biệt**

Không có

**Điều kiện hoàn tất**

Hệ thống thêm mới dữ liệu thông báo cho người dùng tương ứng khi có các tác động liên quan đến bài viết, bình luận và các thông tin đấu giá của họ.

**Điểm mở rộng**

Không có.

**13. Hệ thống quản trị của Admin Thống kê dữ liệu**

**Tóm tắt**

Use-case cho phép actor là quản trị viên có thể thực hiện các hoạt động thống kê các dữ liệu trên hệ thống.

**Dòng Sự kiện**

Use-case bắt đầu khi quản trị viên vào trang cá nhân để chọn”Thống kê dữ liệu”

**Dòng sự kiện chính**

Khi chọn chức năng “Thống kê dữ liệu”

Hệ thống sẽ chuyển quản trị viên đến trang “Danh Mục Thống Kê”

Ở đây người dùng có thể chọn thống kê cụ thể theo số like, số bình luận, thông tin đấu giá.

Hệ thống sẽ xử lý theo dữ liệu hiện có và thống kê dữ liệu sau đó vẽ biểu đồ hiển thị cho người dùng.

**Các dòng sự kiện khác**

Không có.

**Các yêu cầu đặc biệt**

Không có.

**Điều kiện tiên quyết**

Quản trị viên phải đăng nhập và vào trang cá nhân.

**Điều kiện hoàn tất**

Hệ thống vẽ được biểu đồ thống kê được đầy đủ các dữ liệu theo mục liên quan.

**Điểm mở rộng**

Không có.

**14. Chức năng xem danh sách báo cáo viết, bình luận và phê duyệt báo cáo bình luận**

**Tóm tắt**

Use-case cho phép quản trị viên có thêm danh sách các bài viết và bình luận bị báo cáo và có thể chọn gỡ bài viết khỏi cộng đồng hoặc bỏ qua báo cáo.

**Dòng sự kiện**

Quản trị viên ở trang cá nhân chọn nút “Xem Danh Sách Báo Cáo”

**Dòng sự kiện chính**

Quản trị viên ở trang cá nhân chọn nút”Xem Danh Sách Báo Cáo”, hệ thống sẽ chuyển đến trang Danh sách báo cáo bài viết hay bình luận, họ có thể chọn các chức năng hiện có sẵn hiển thị ở đó.

Khi chọn một trong hai chức năng hệ thống sẽ đưa đến trang hiển thị danh sách các bài viết,các bình luận tương ứng khi đó họ có thể chọn một trong hai phương án cho các bài viết hoặc bình luận bị báo cáo đó.

Chọn Phê duyệt báo cáo, hệ thống sẽ chuyển báo cáo đó thành đã duyệt và chuyển bài viết đó thành trạng thái đã bị report, sau đó hệ thống sẽ gửi thông báo đến người dùng gửi bài viết và người dùng đăng bài viết.

Chọn Bỏ qua báo cáo, hệ thống cũng sẽ chuyển báo cáo thành đã duyệt nhưng bài viết đó sẽ không bị ảnh hưởng và hệ thống chỉ gửi thông báo cho người gửi báo cáo rằng báo cáo của họ chưa đủ điều kiện phê duyệt.

Người dùng có thể gửi báo cáo nhiều lần và quản trị viên có thể kiểm duyệt nhiều các bài viết nếu như lý do báo cáo đúng sự thật với nội dung và ảnh của bài viết đó.

**Các dòng sự kiện khác**

Không có.

**Các yêu cầu đặc biệt**

Không có.

**Các điều kiện tiên quyết**

Quản trị viên cần đăng nhập vào hệ thống.

**Điều kiện hoàn tất**

Hệ thống hiển thị danh sách các bài viết các bình luận đang bị báo cáo và cho phép quản trị viên có thể duyệt các báo cáo và đảm bảo hệ thống đã xử lý đầy đủ các hoạt động liên quan sau khi đã phê duyệt hoặc bỏ qua báo cáo.

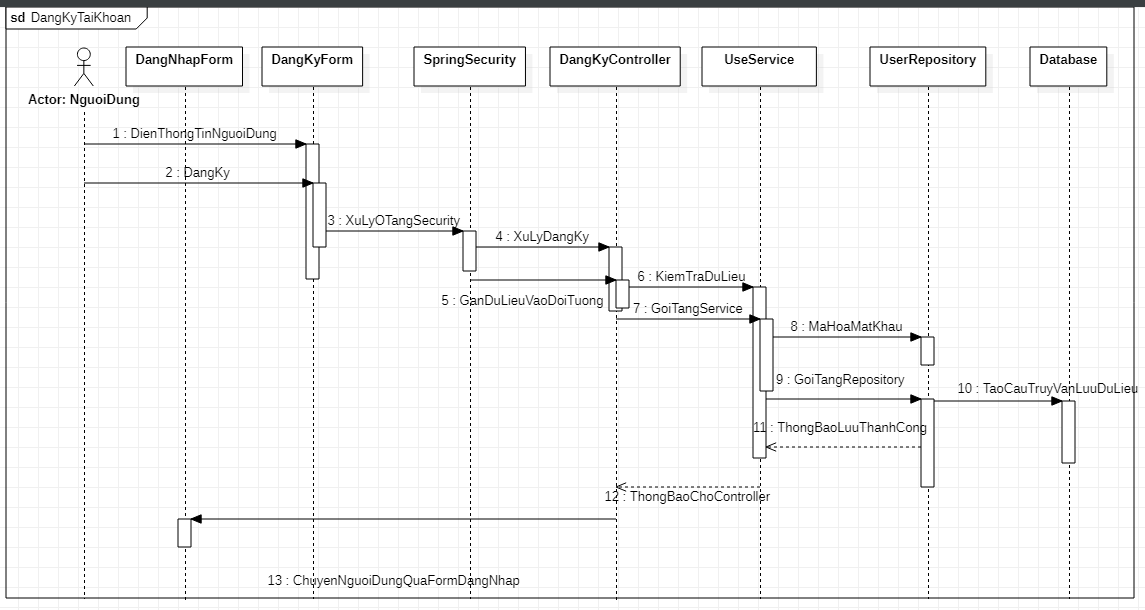
**Điều kiện mở rộng**

Không có.

### Lược đồ tuần tự(SEQUENCE DIAGRAM)

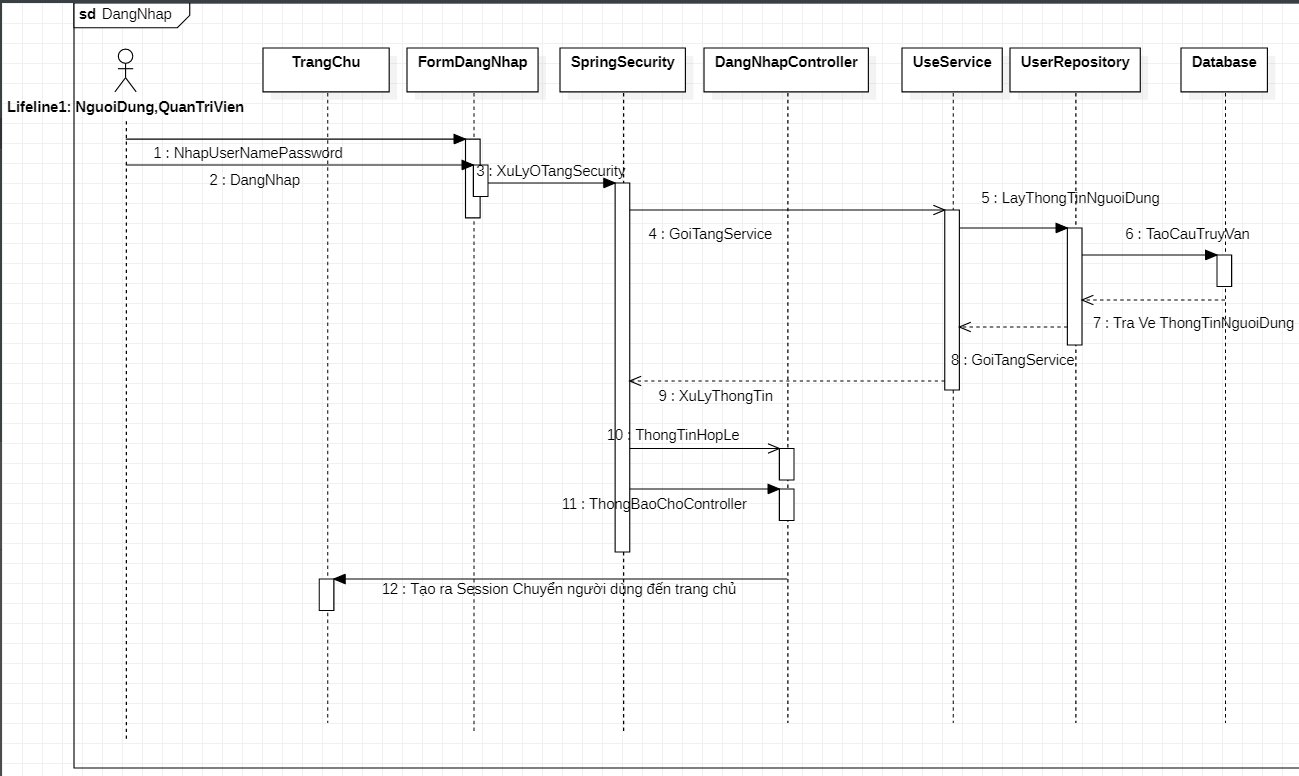
Biểu đồ tuần tự là phương tiện biểu diễn tương tác dưới dạng hình ảnh. Tập trung vào thứ tự các công việc thực hiện.

#### Đăng ký tài khoản người dùng



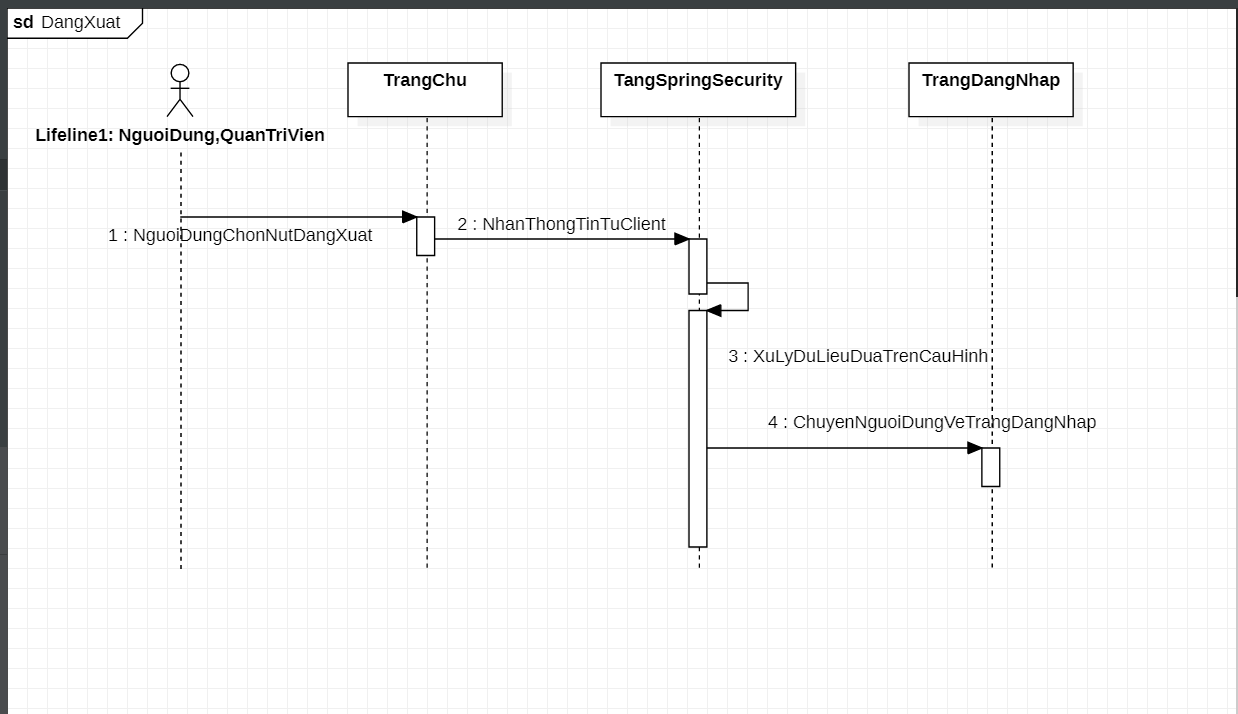
Hình 3.3.5.1 Sequence Diagram Đăng ký tài khoản người dùng

#### Đăng nhập vào hệ thống



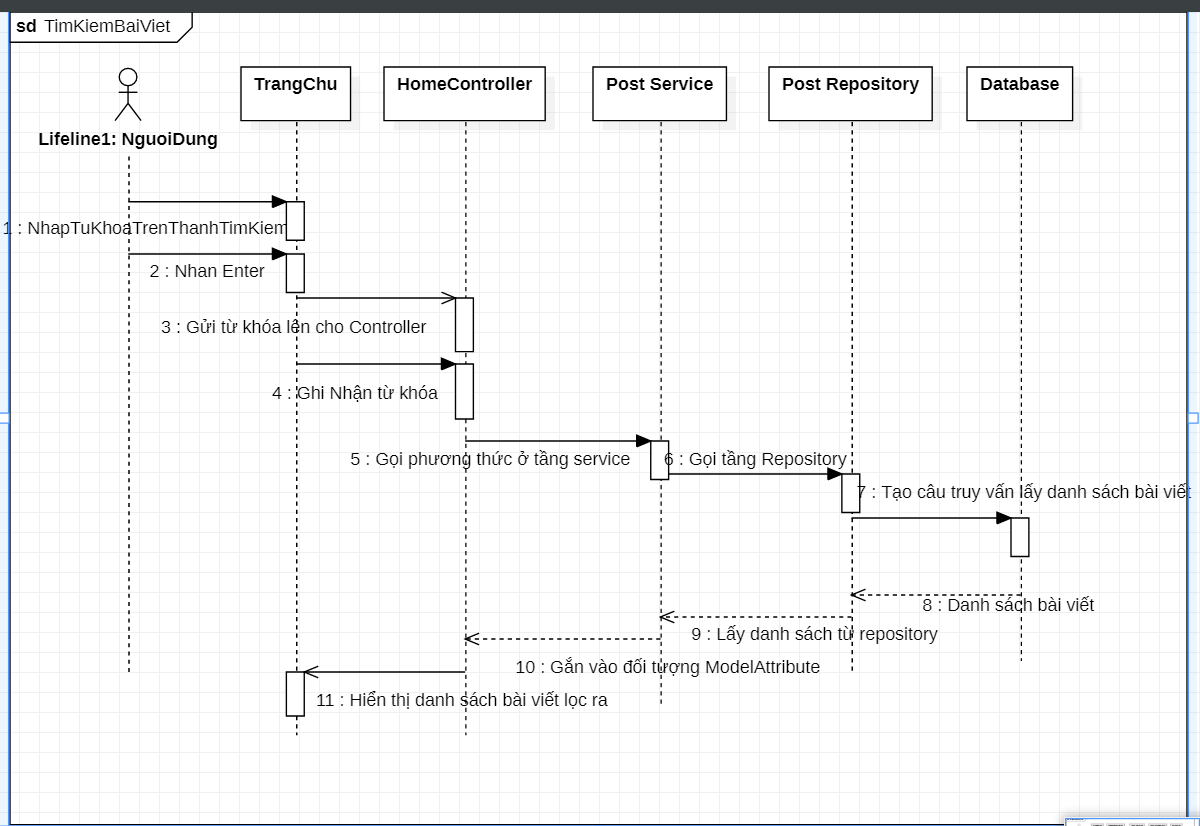
Hình 3.3.5.2 Sequence Diagram Đăng nhập

#### Đăng Xuất Người Dùng



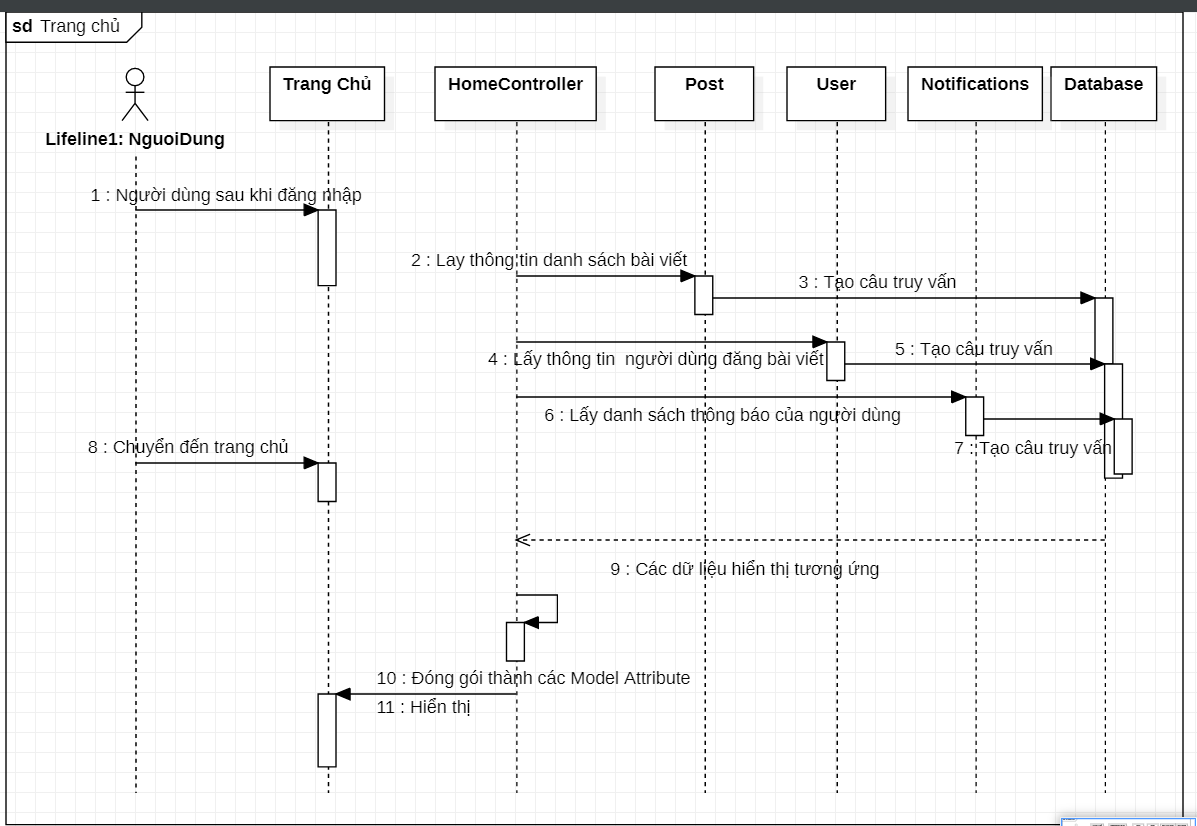
Hình 3.3.5.3 Sequence Diagram Đăng Xuất

#### Tìm kiếm bài viết



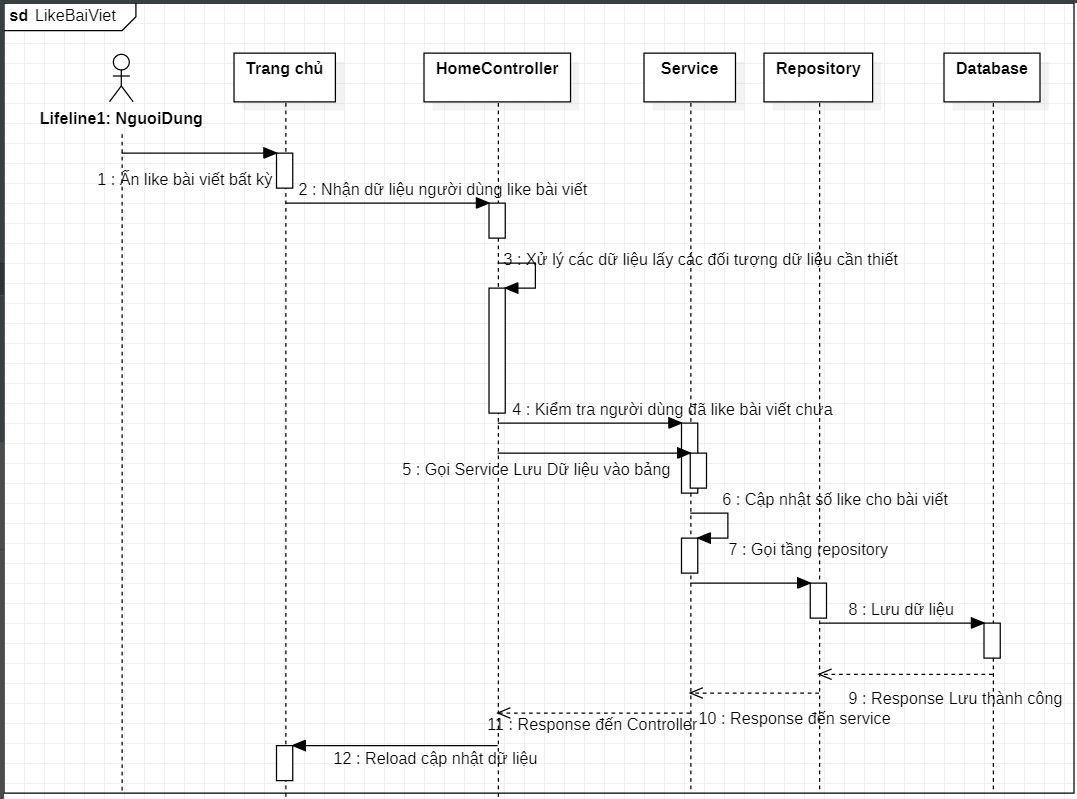
Hình 3.3.5.4 Sequence Diagram Tìm kiếm bài viết

#### Trang chủ hiển thị



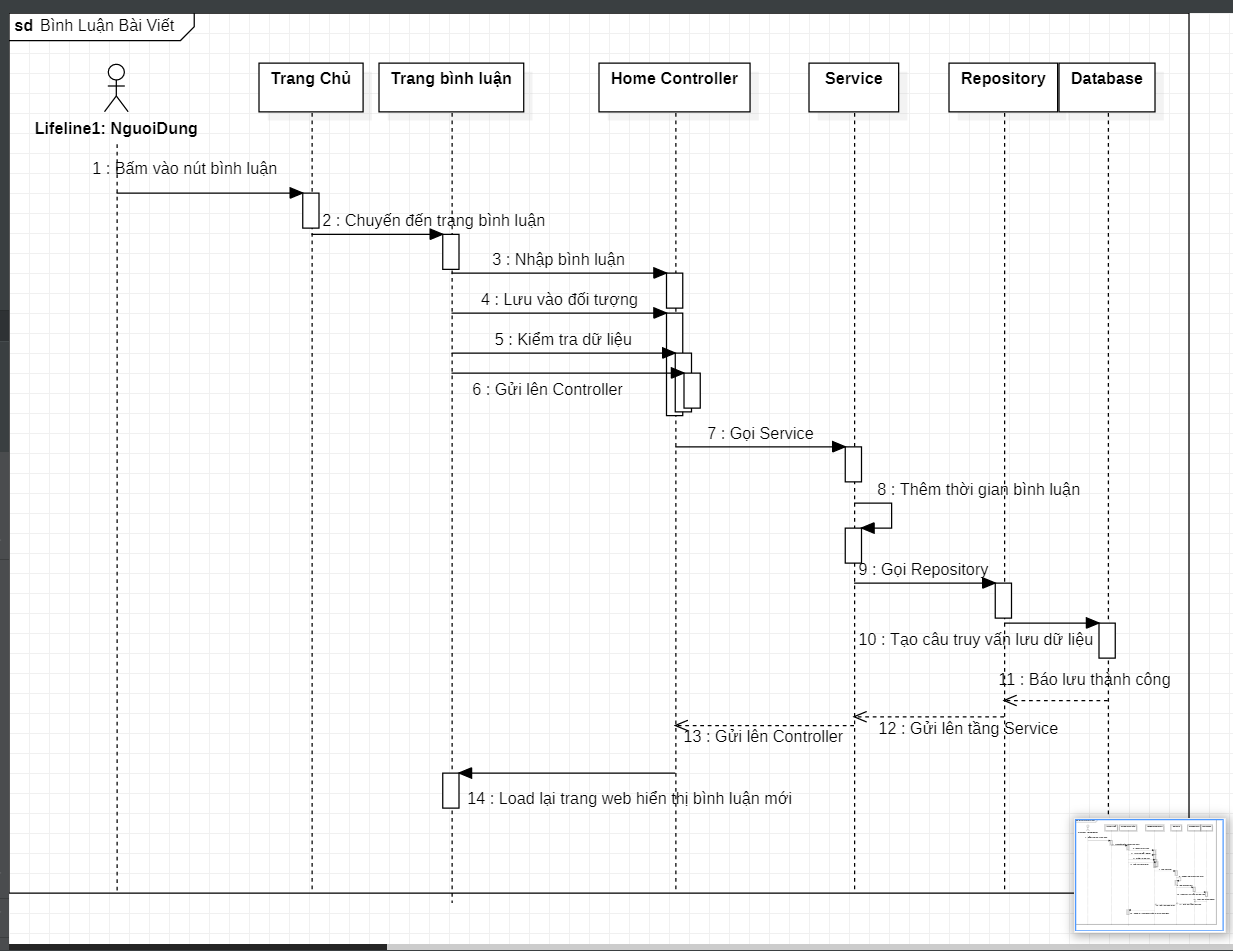
Hình 3.3.5.5 Sequence Diagram Trang chủ hiển thị

#### Chức năng like bài viết



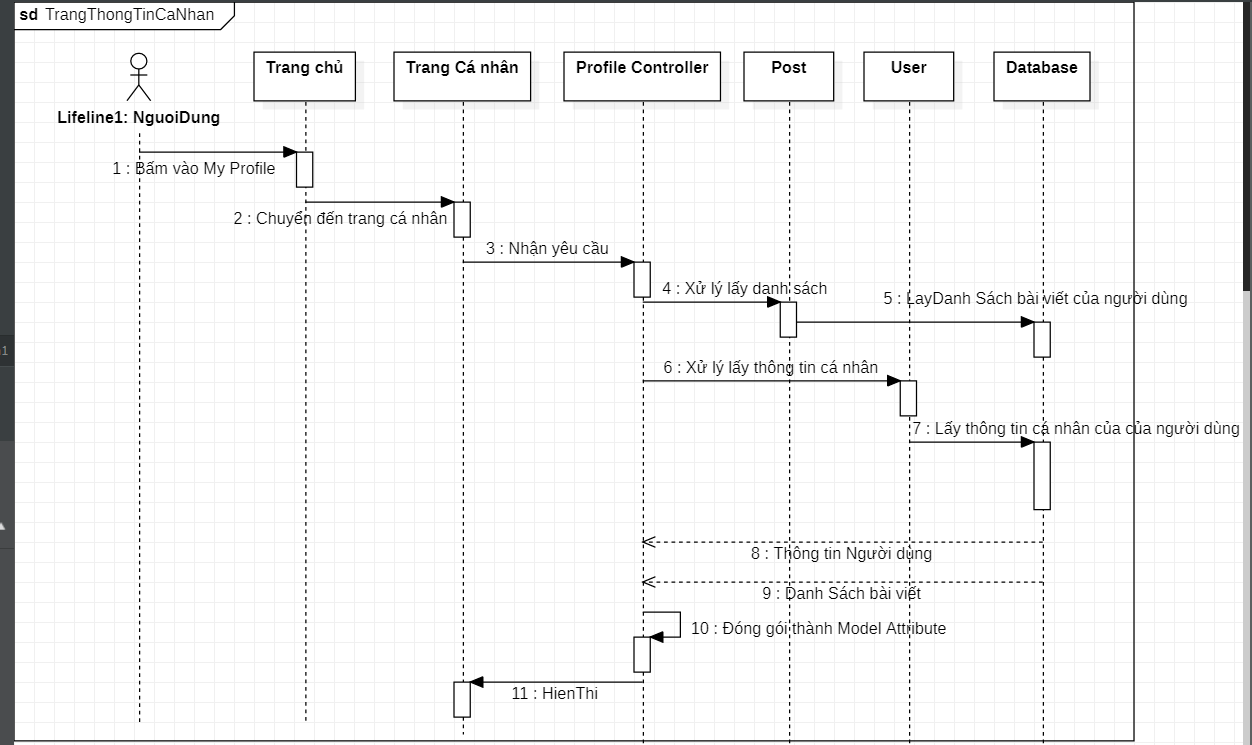
Hình 3.3.5.6 Sequence Diagram chức năng like bài viết

#### Chức năng bình luận bài viết



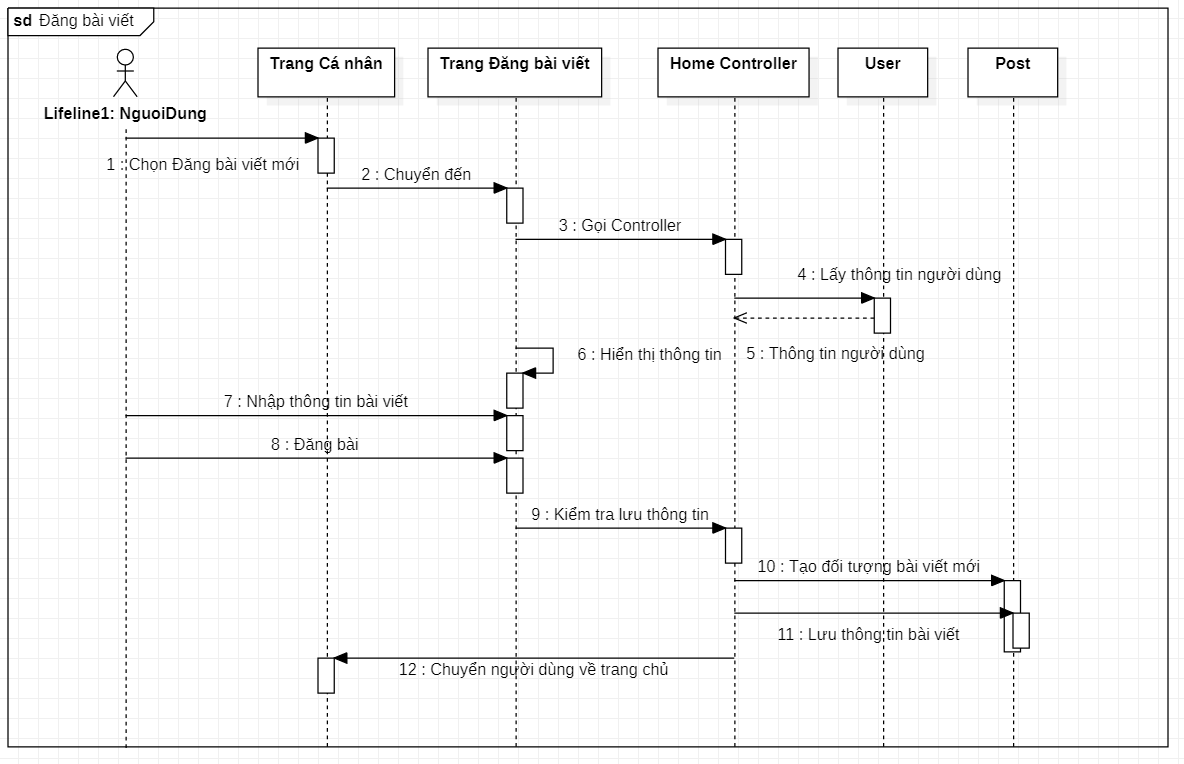
Hình 3.3.5.7 Sequence Diagram Chức năng bình luận bài viết

#### Trang thông tin cá nhân

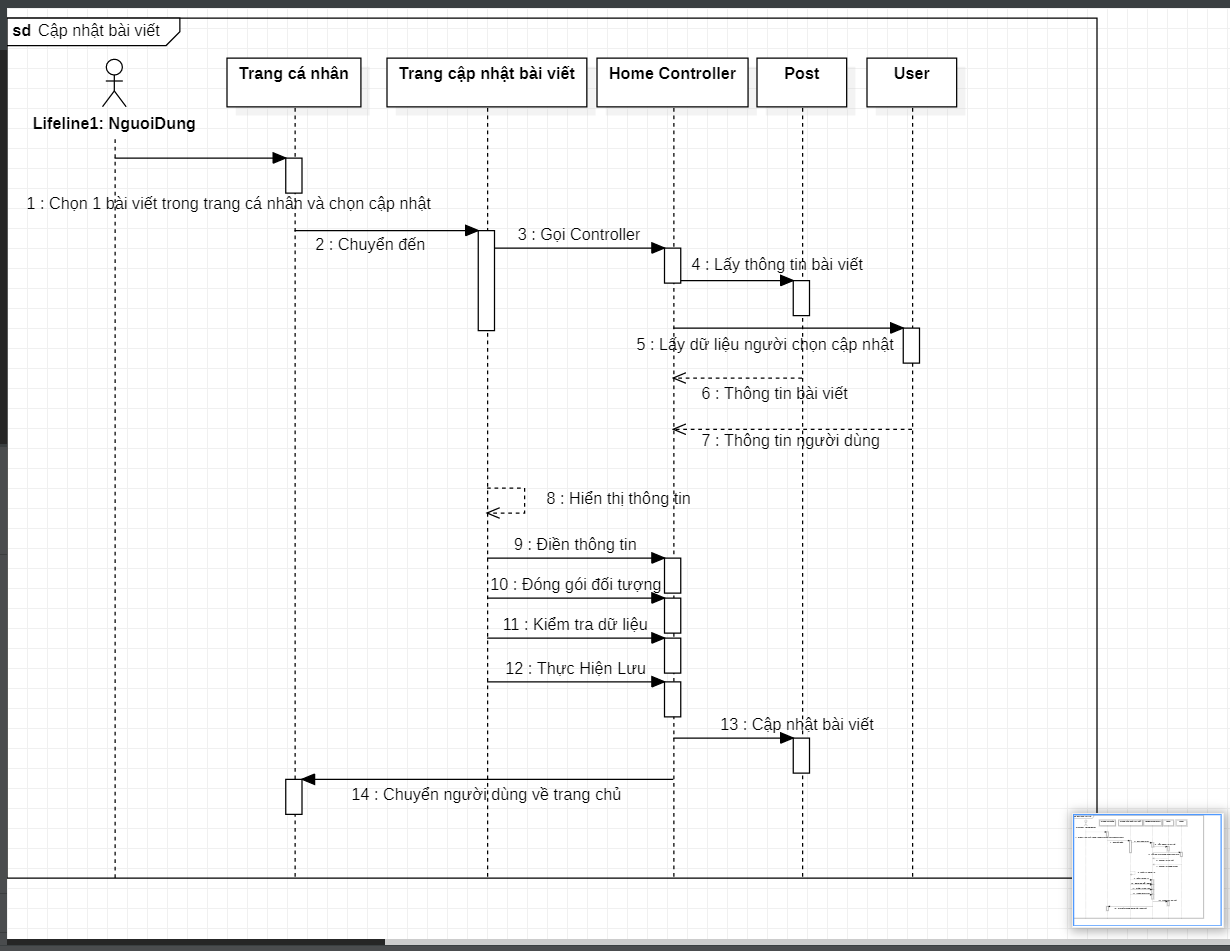


Hình 3.3.5.8 Sequence Diagram Trang thông tin cá nhân

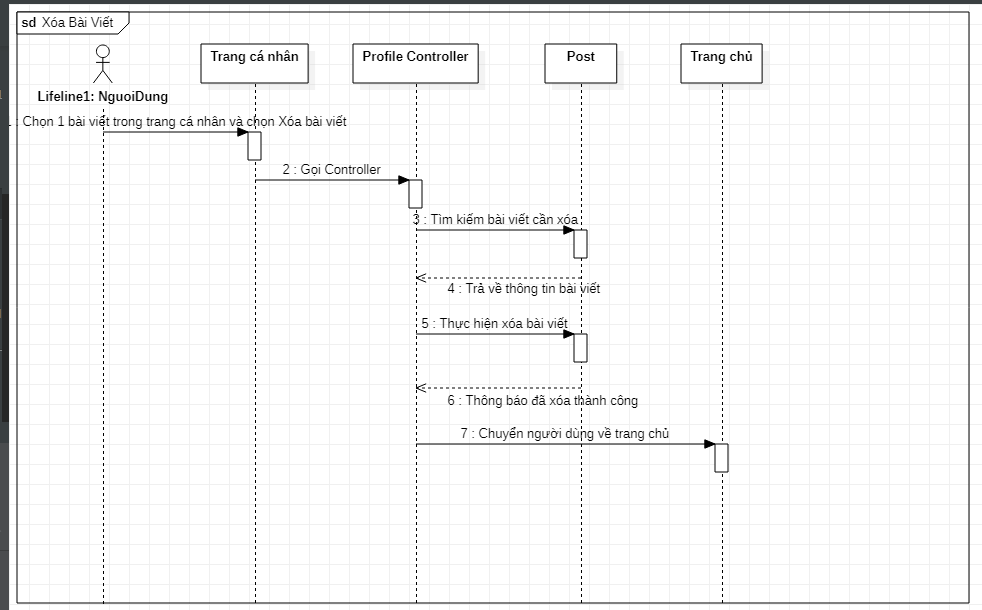
#### Đăng,Xóa,Cập nhật bài viết



Hình 3.3.5.9 Sequence Diagram Trang đăng bài viết mới

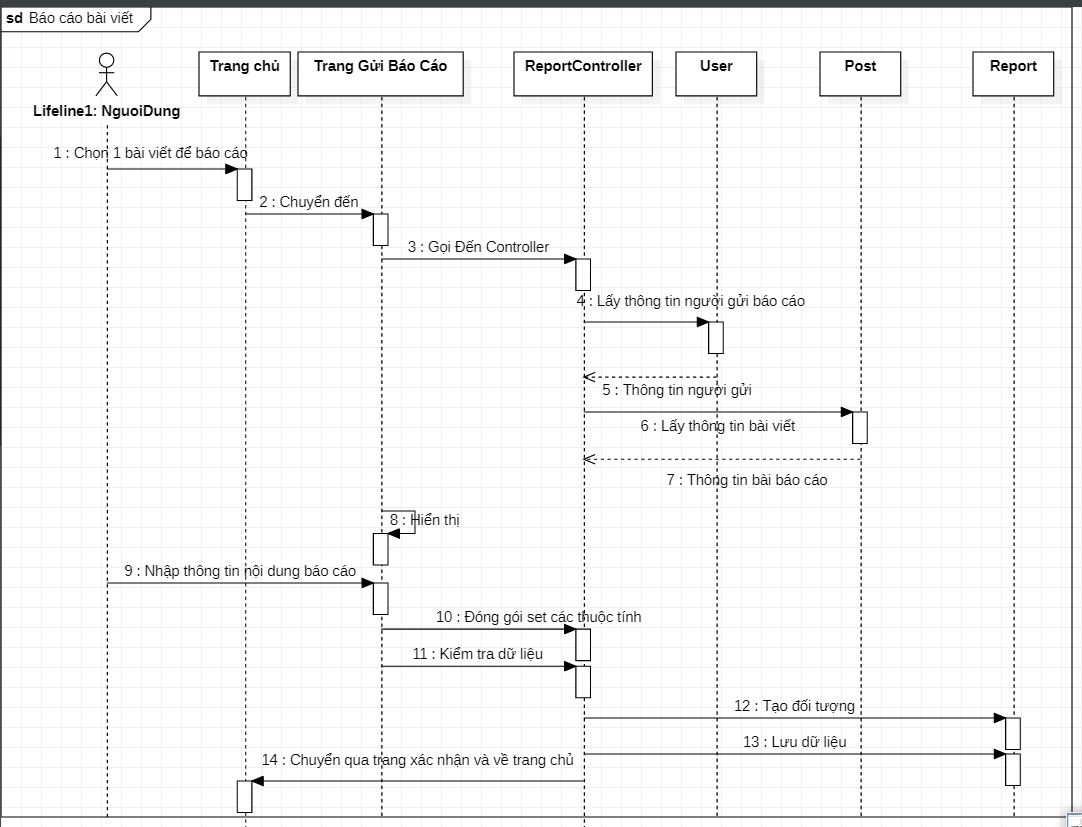


Hình 3.3.5.10 Sequence Diagram Trang cập nhật bài viết

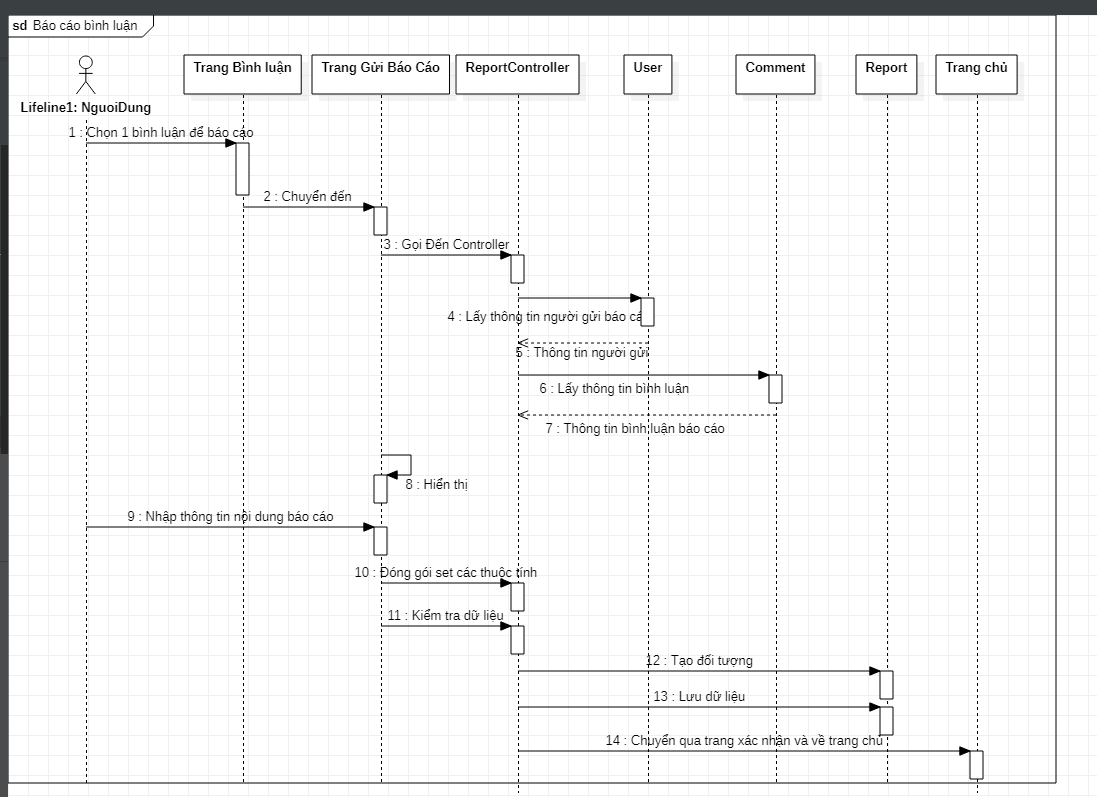


Hình 3.3.5.11 Sequence Diagram Xóa Bài Viết

#### Báo cáo bài viết



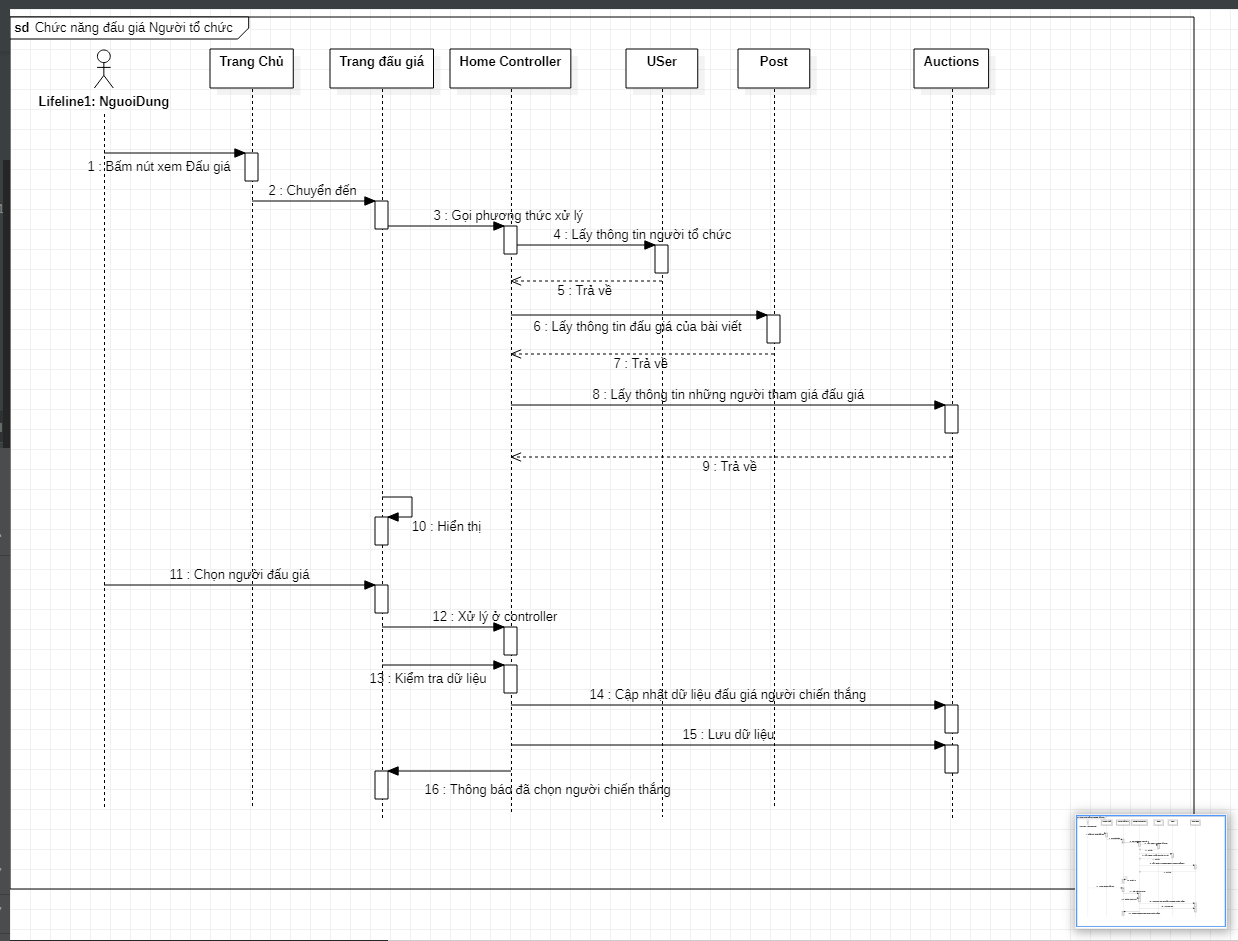
Hình 3.3.5.12 Báo cáo bài viết Sequence Diagram



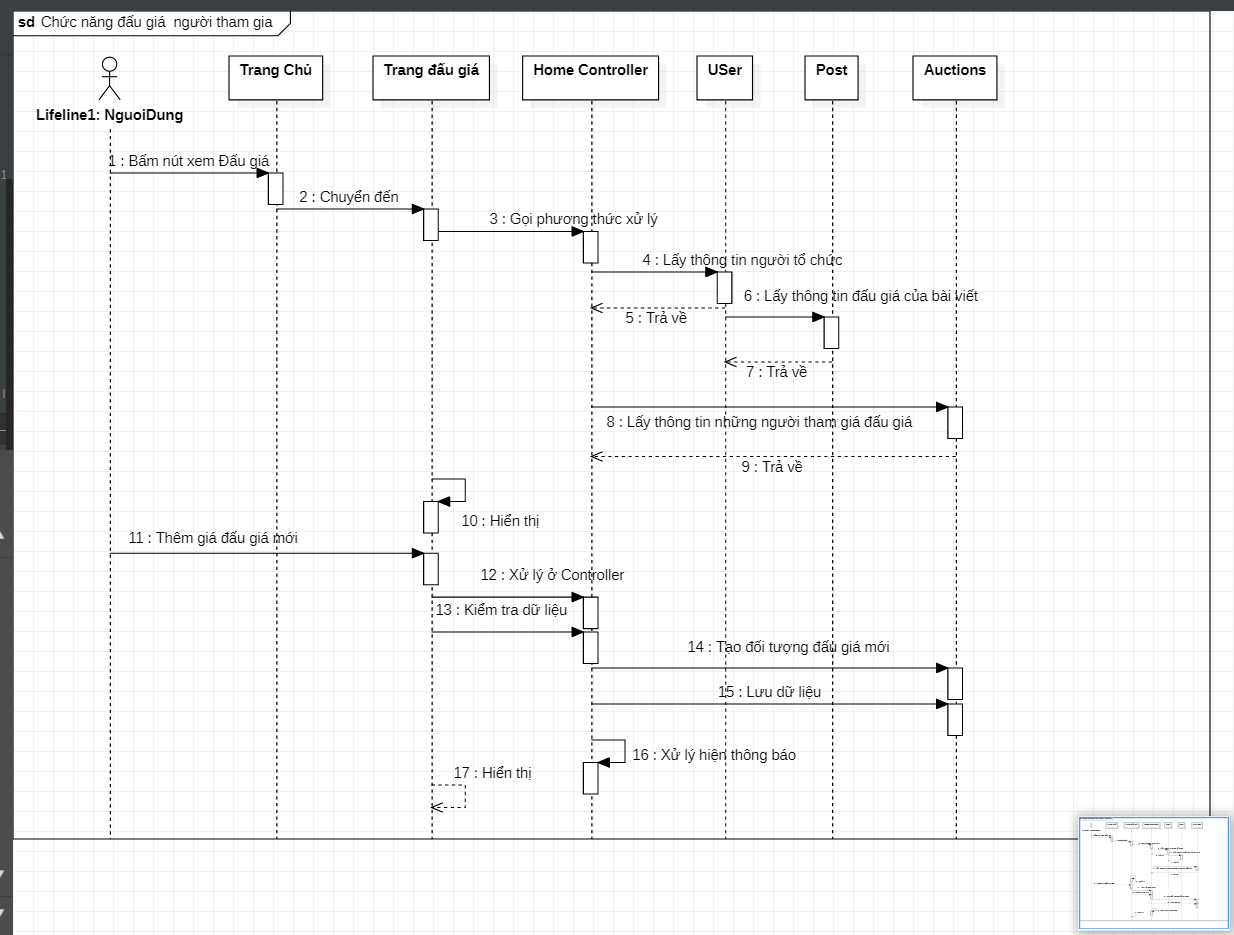
Hình 3.3.5.13 Báo cáo bình luận Sequence Diagram

#### Chức năng đấu giá

**Chức năng đấu giá của người tổ chức**

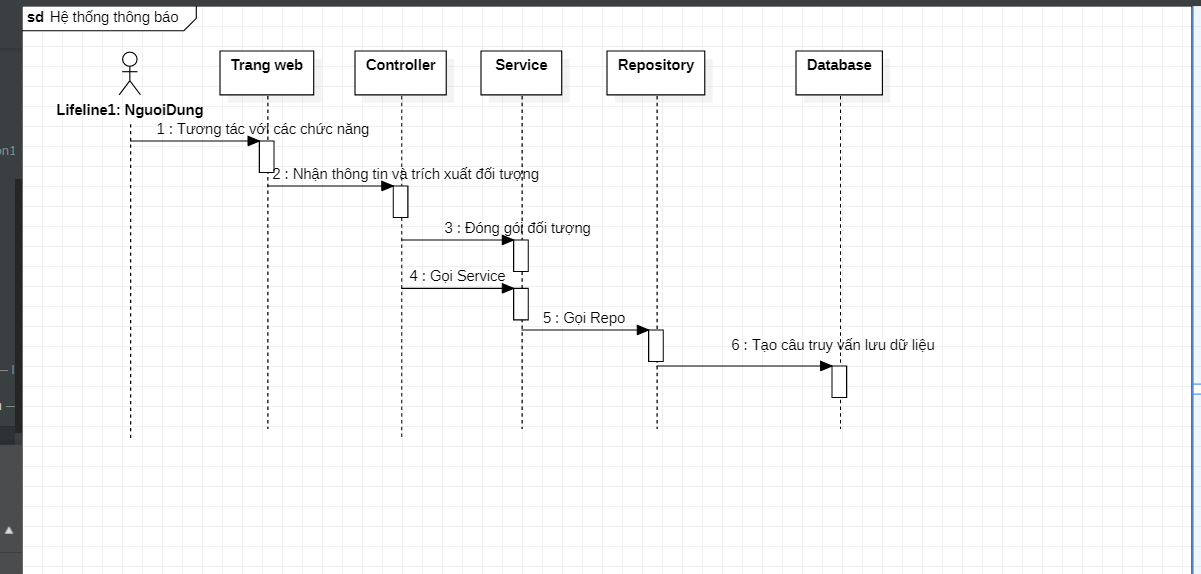
****

Hình 3.3.5.14 Người tổ chức đấu giá chọn người chiến thắng

****

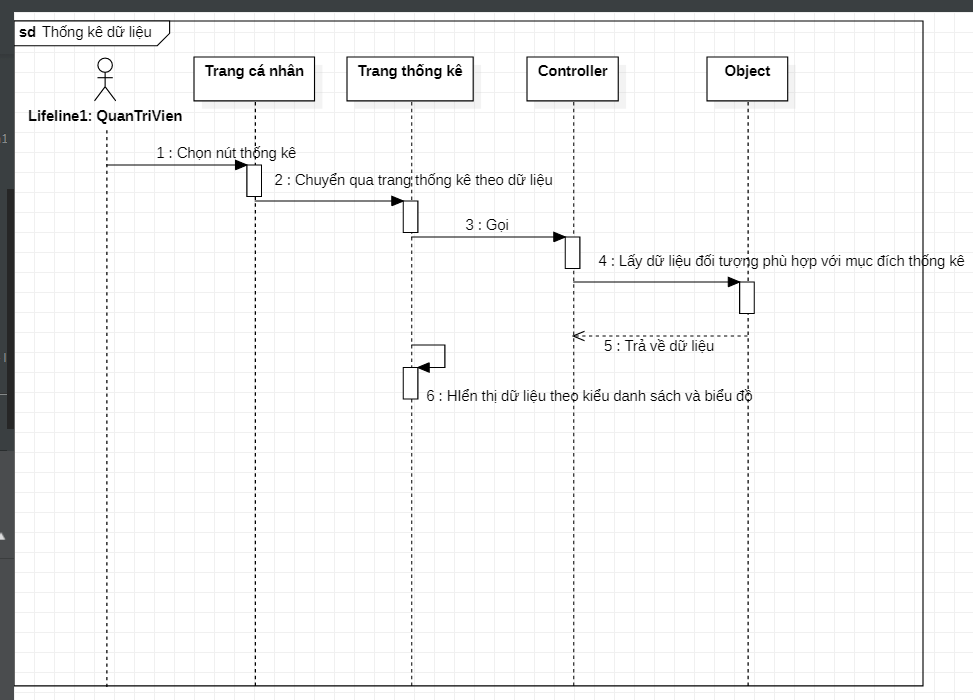
Hình 3.3.5.15 Người tham gia đấu giá

#### Hệ thống thông báo



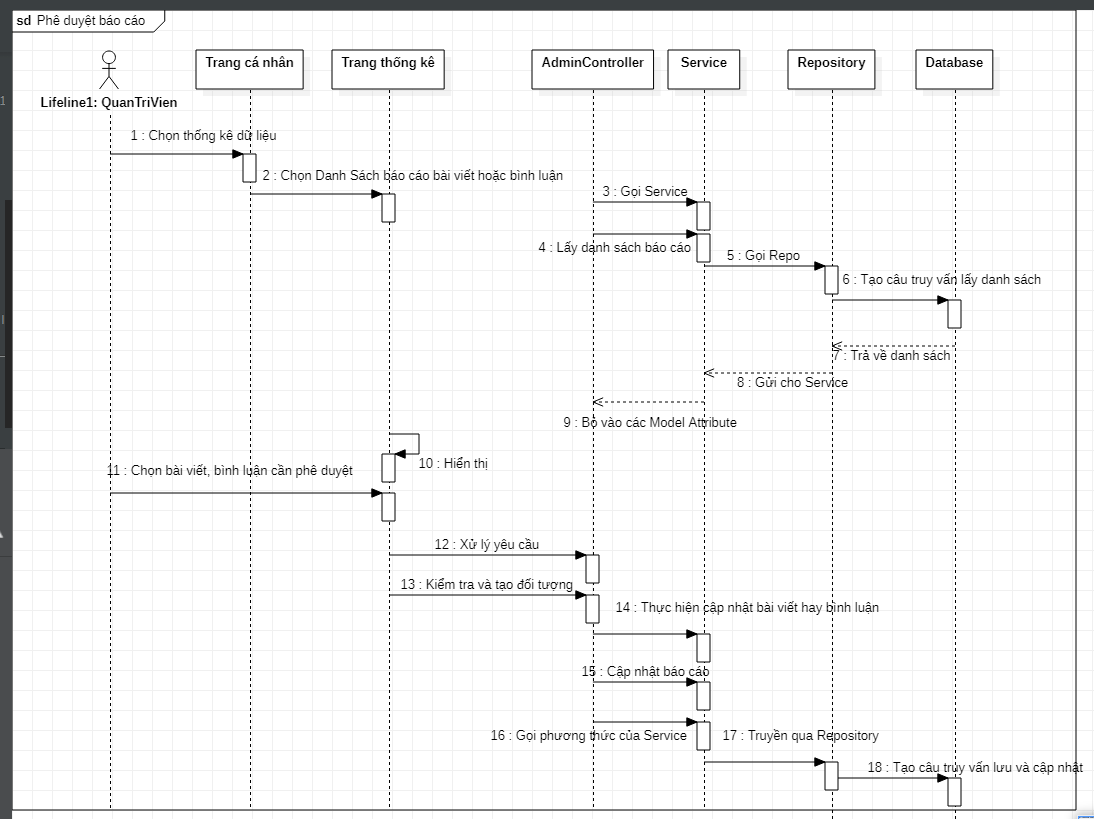
Hình 3.3.5.16 Hệ thống thông báo Sequence Diagram

#### Hệ thống quản trị admin thống kê dữ liệu



Hình 3.3.5.17 Thống kê dữ liệu Sequence Diagram

#### Chức năng phê duyệt báo cáo bài viết, bình luận của quản trị viên

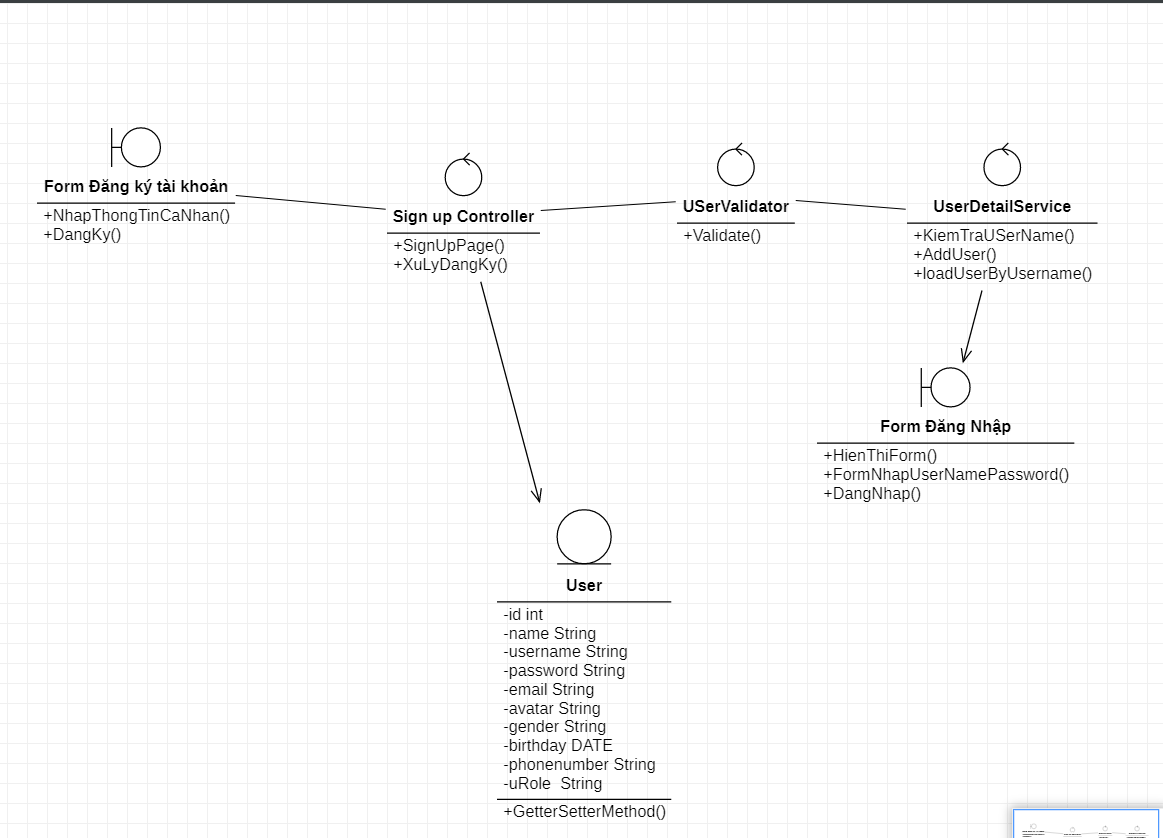


Hình 3.3.5.18 Sequence Diagram Báo Cáo bài viết,bình luận

### Sơ đồ lớp(Class Diagram)

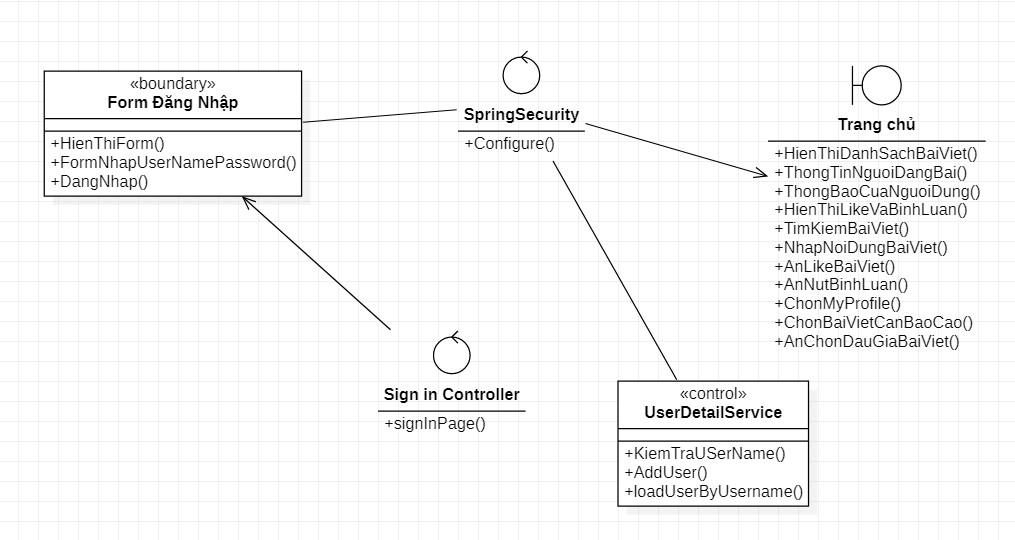
Biểu đồ mô tả các lớp, là viên gạch để xây dựng bất kỳ hệ thống theo cấu trúc hướng đối tượng nào.Khả năng tương tác giữa chúng, bằng cách truyền thông điệp, được thể hiện trong các mối quan hệ giữa chúng. Các thuộc tính và các thao tác của lớp, cũng như các quan hệ giữa các lớp.

#### Đăng ký Tài khoản người dùng



Hình 3.3.6.1 Class Diagram Đăng ký tài khoản người dùng

#### Đăng Nhập



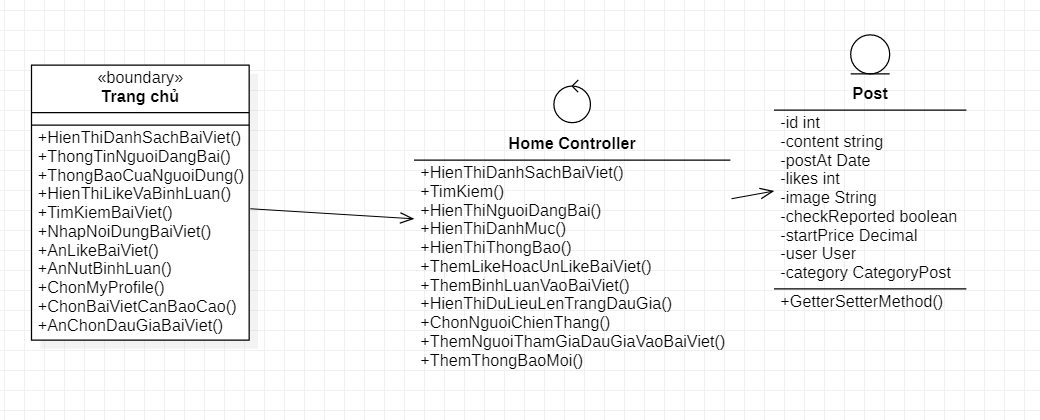
Hình 3.3.6.2 Class Diagram Đăng nhập

#### Đăng Xuất



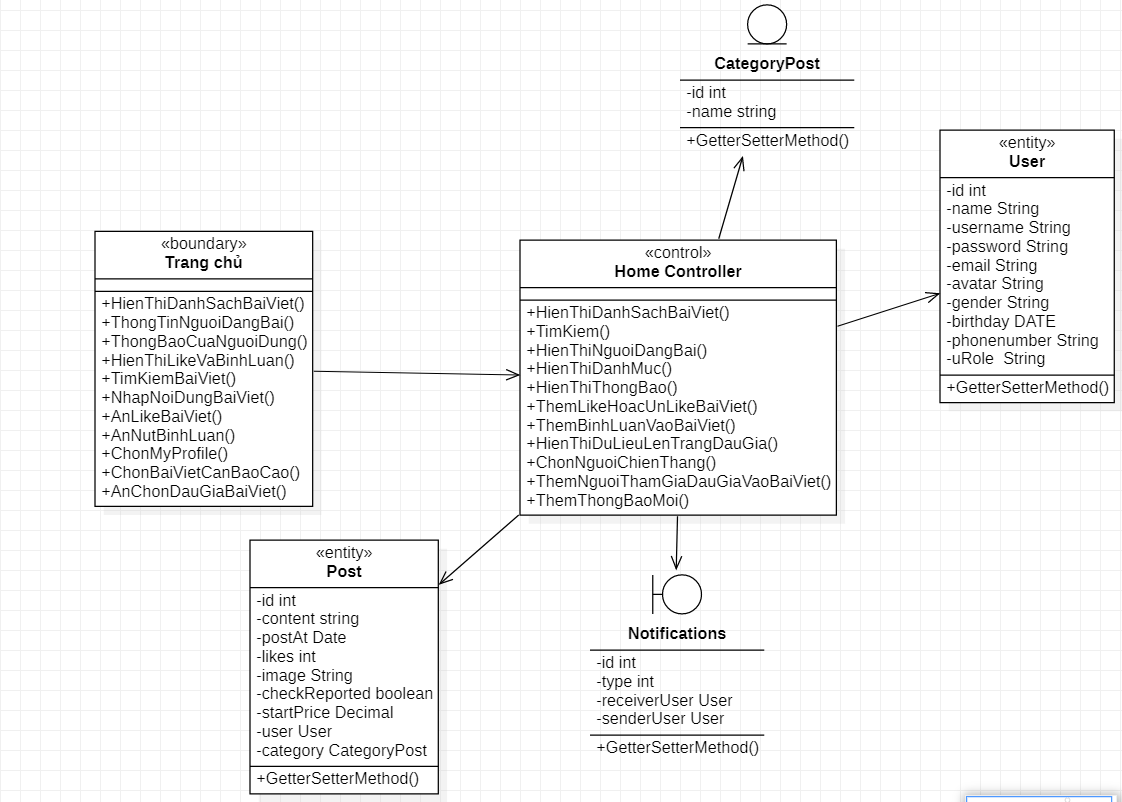
Hình 3.3.6.3 Class Diagram Đăng Xuất

#### Tìm kiếm bài viết



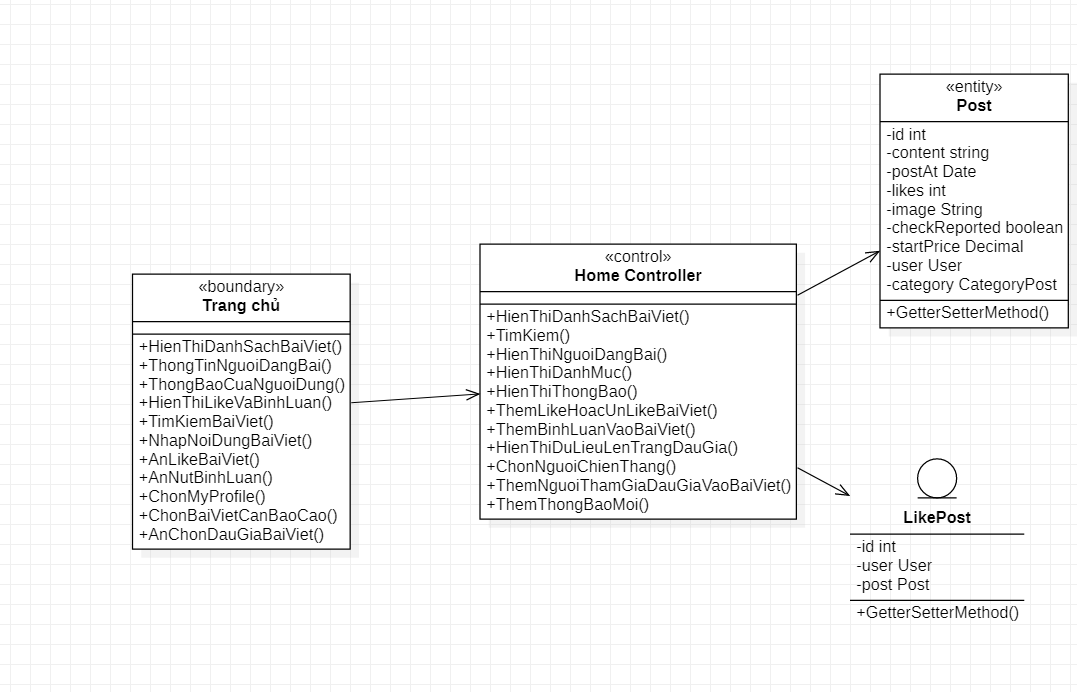
Hình 3.3.6.4 Class Diagram Tim Kiếm Bài Viết

#### Trang chủ



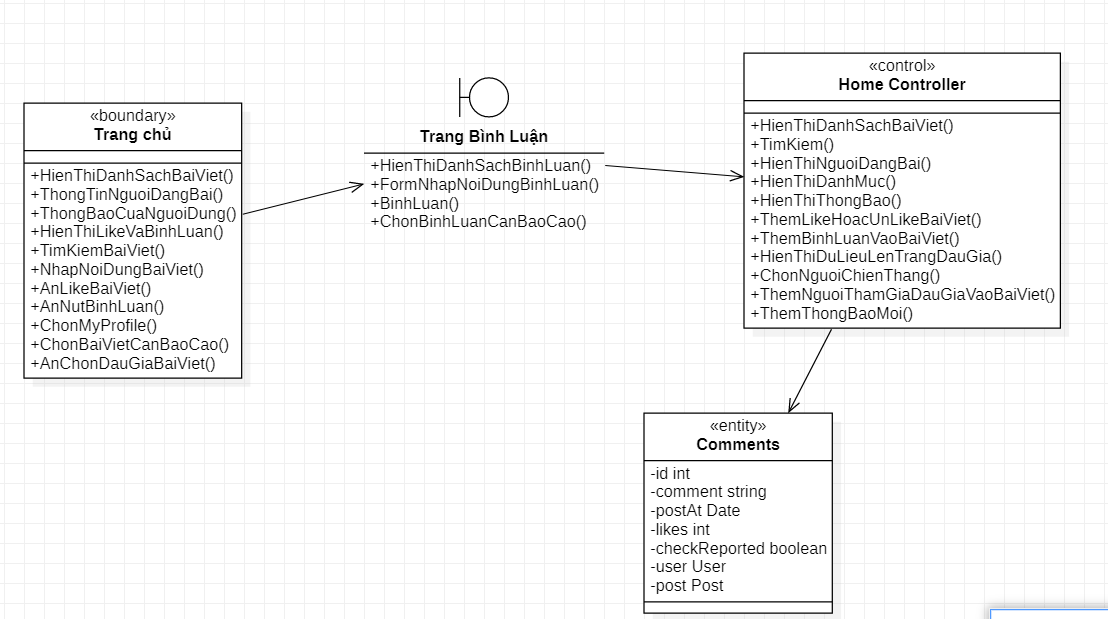
Hình 3.3.6.5 Class Diagram Xử lý hiển thị trang chủ

#### Chức năng like bài viết



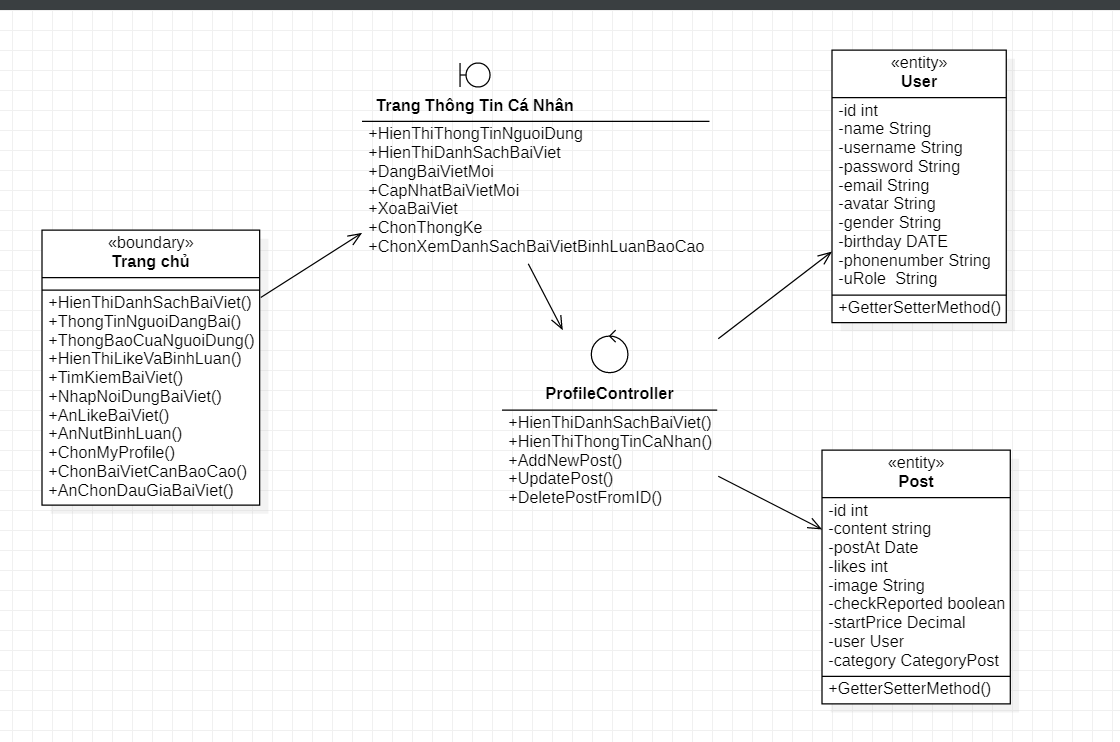
Hình 3.3.6.6 Class Diagram Chức năng Like Bài Viết

#### Chức năng bình luận bài viết



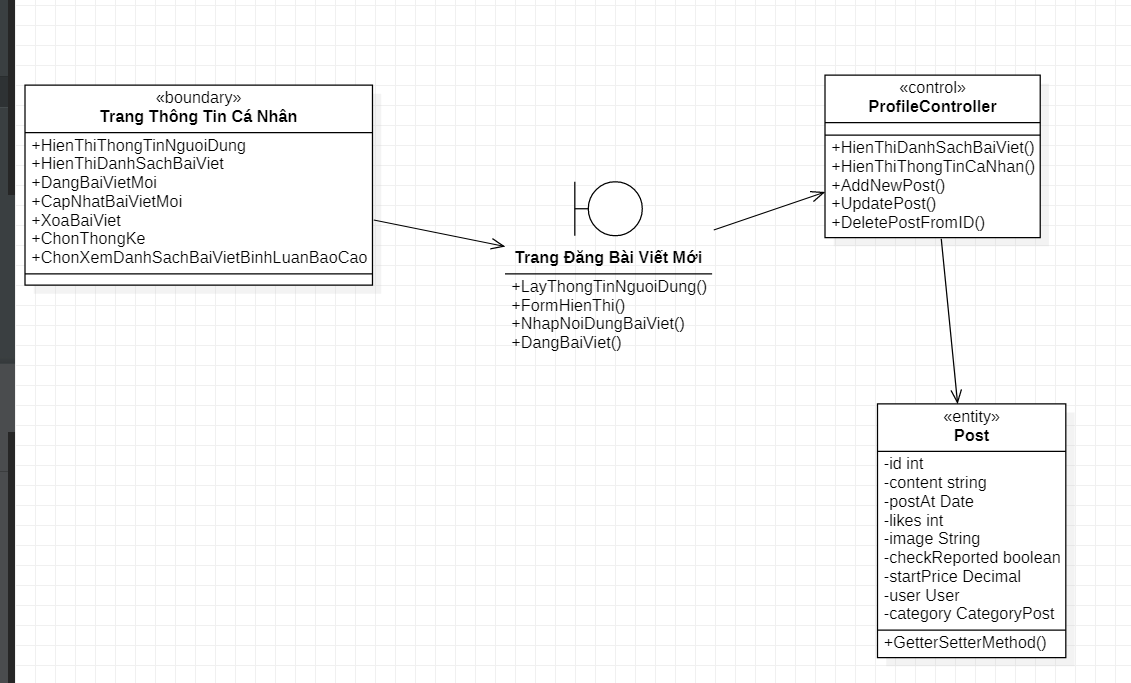
Hình 3.3.6.7 Class Diagram Chức năng bình luận bài viết

#### Trang thông tin cá nhân



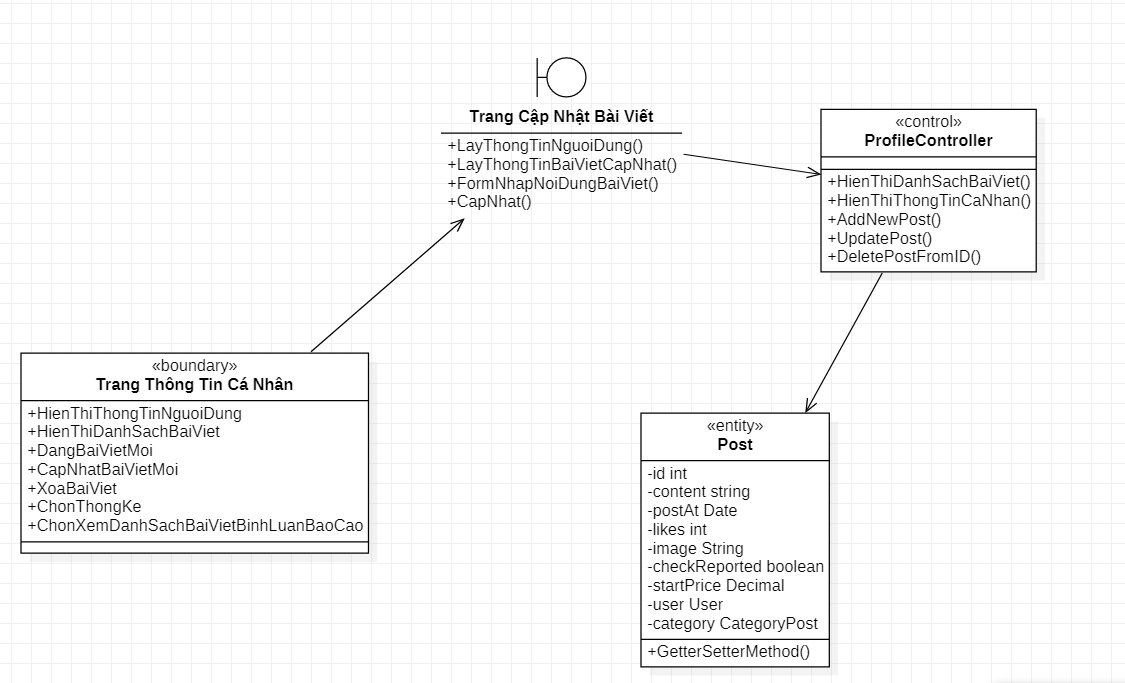
Hình 3.3.6.8 Class Diagram Hiển thị thông tin cá nhân

#### Đăng bài viết mới



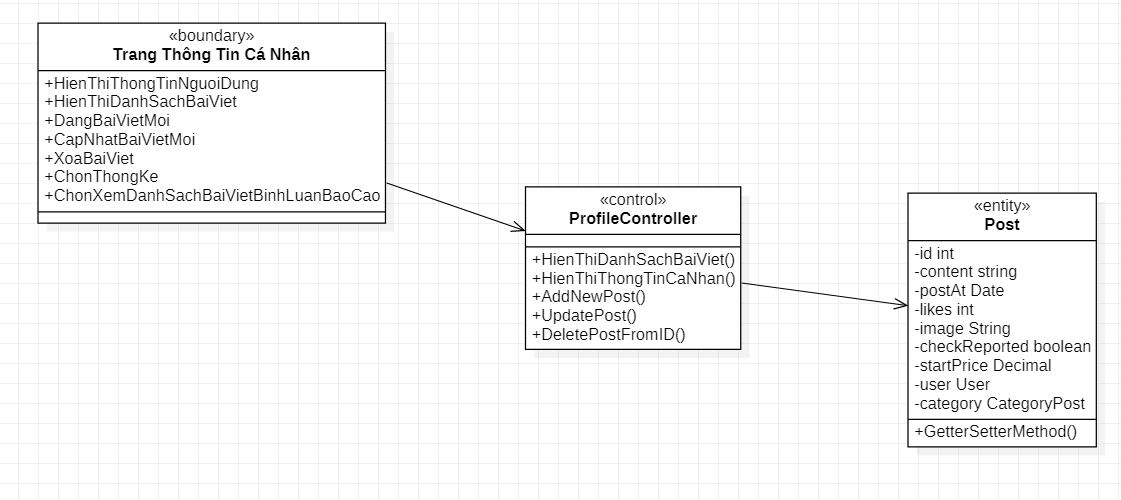
Hình 3.3.6.9 Class Diagram Đăng bài viết mới

#### Cập nhật bài viết



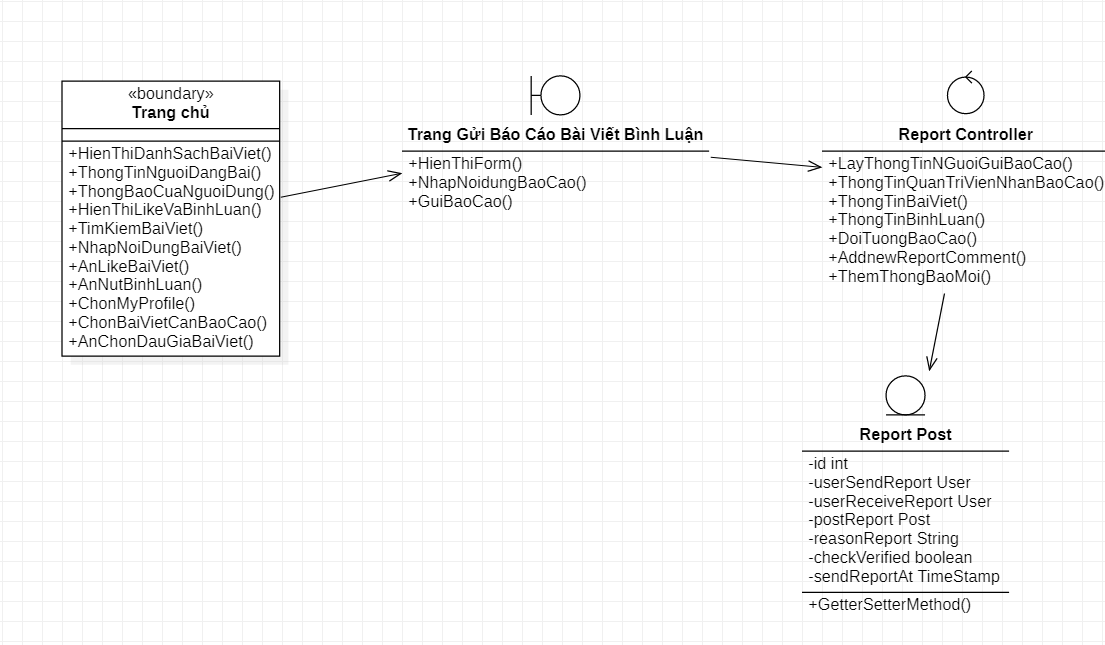
Hình 3.3.6.10 Class Diagram cập nhật bài viết

#### Xóa Bài Viết

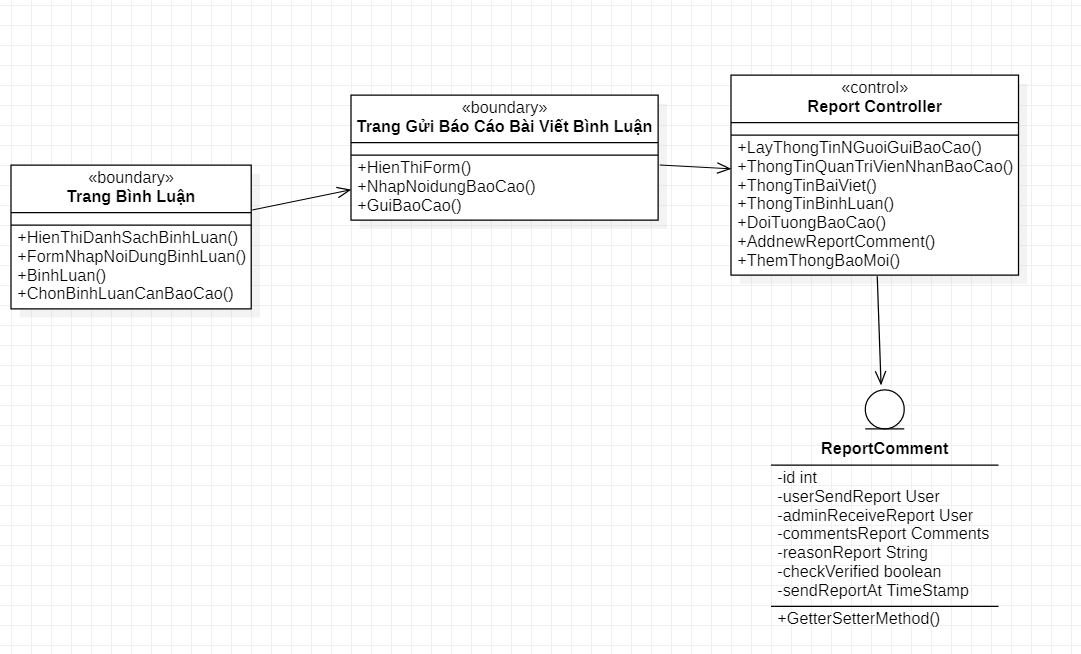


Hình 3.3.6.11 Class Diagram Xóa Bài Viết

#### Báo Cáo Bài Viết,Bình Luận



Hình 3.3.6.12A Class Diagram Báo cáo bài viết



Hình 3.3.6.12B Class Diagram Báo cáo bình luận

#### Chức năng Đấu Giá



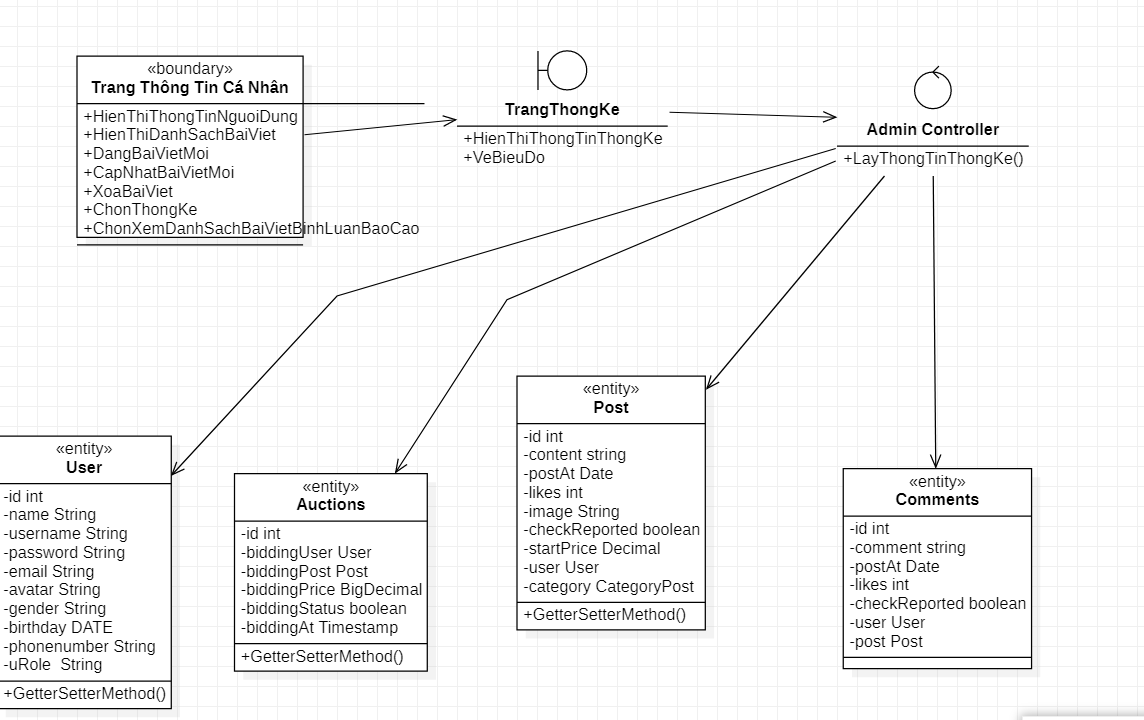
Hình 3.3.6.13 Class Diagram Chức năng đấu giá

#### Hệ thống thông báo



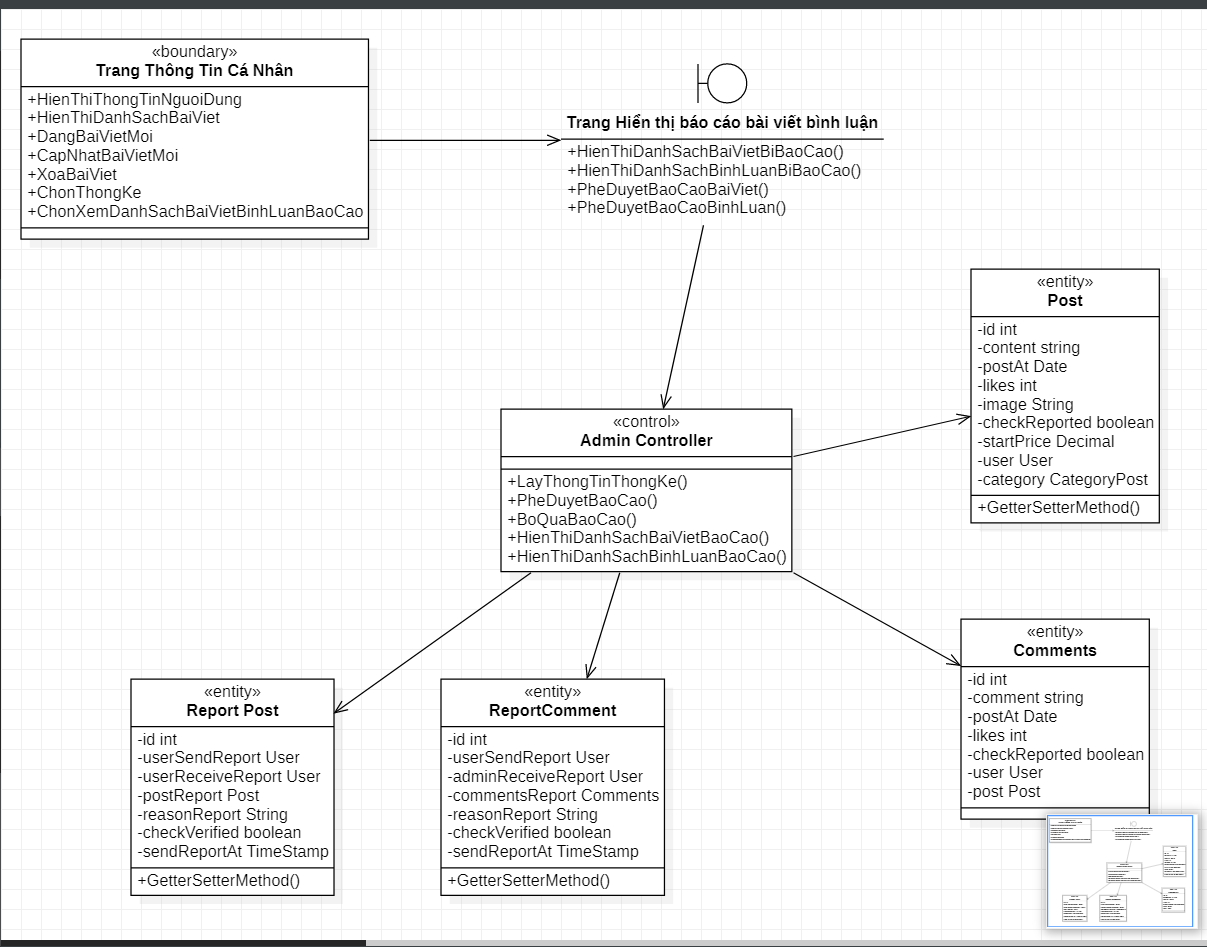
Hình 3.3.6.14 Class Diagram hệ thống thông báo

#### Thống kê dữ liệu



Hình 3.3.6.15 Class Diagram thống kê dữ liệu

#### Phê Duyệt Báo Cáo Bài Viết Bình Luận



Hình 3.3.6.16 Class Diagram phê duyệt báo cáo bài viết bình luận

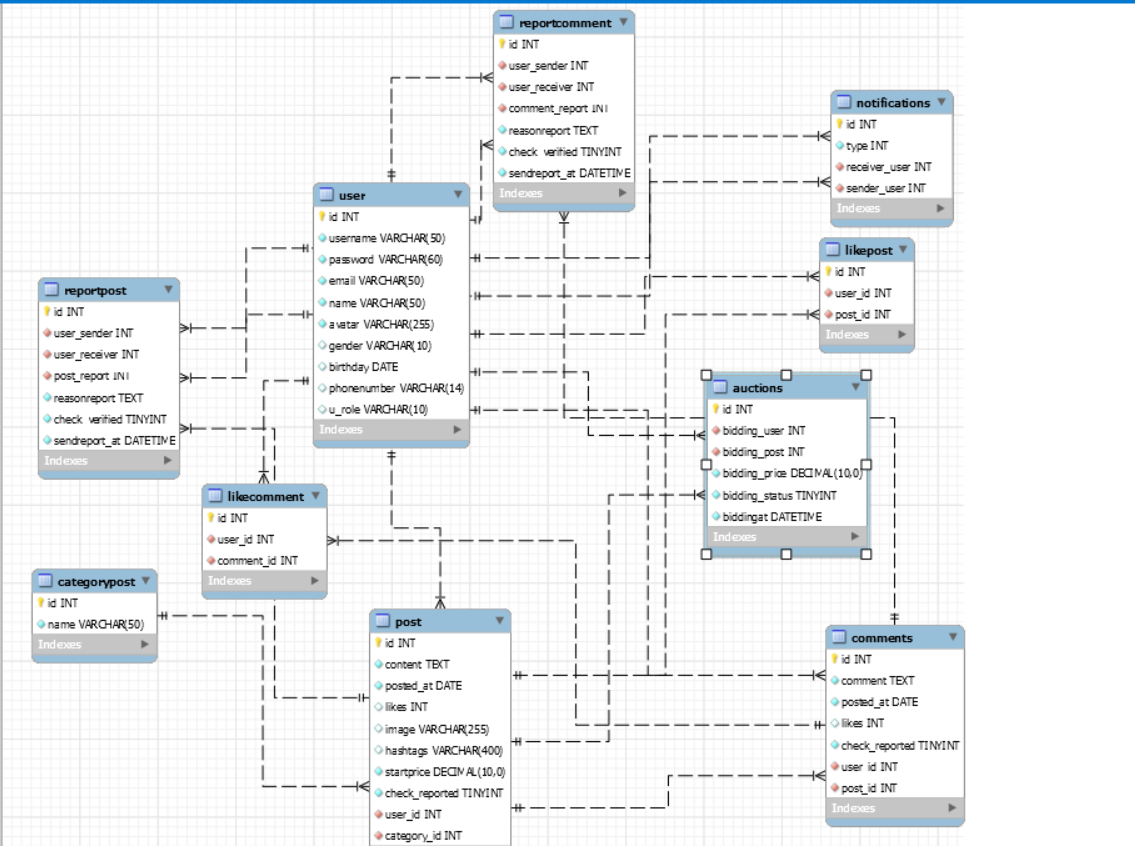
#### Biểu đồ Class Diagram mối quan hệ giữa các thực thể trong Hệ Thống Chương Trình



Hình 3.3.6.17 Class Diagram Biểu đồ mối quan hệ giữa các lớp

## Thiết kế cơ sở dữ liệu

### Cấu trúc và mối quan hệ của các bảng



## Hiện thực hệ thống

Bảng 2.1: Tên bảng 1

Bảng 2.2: Tên bảng 2

Bảng 2.3: Tên bảng 3

## Chủ đề cấp độ 2

### Chủ đề cấp độ 3

# TÊN CHƯƠNG 3

## Chủ đề cấp độ 2

Nội dung …………………

Nội dung………………….

### Chủ đề cấp độ 3

#### Chủ đề cấp độ 4

## Chủ đề cấp độ 2

TÀI LIỆU THAM KHẢO

(Theo chuẩn IEEE)

PHỤ LỤC