A picture containing picture frame, rectangle, frame

Description automatically generated **BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

**PHÂN HIỆU TẠI THÀNH PHỐ HỒ CHÍ MINH**

BỘ MÔN: CÔNG NGHỆ THÔNG TIN

-----🙞🙜🕮🙞🙜-----



**ĐỒ ÁN TỐT NGHIỆP**

***Đề Tài* :**

XÂY DỰNG WEBSITE ĐỌC & QUẢN LÝ TRUYỆN ONLINE SỬ DỤNG CÔNG NGHỆ ASP.NET VÀ TÍCH HỢP THUẬT TOÁN RECOMMENDATION SYSTEM

***Giảng viên hướng dẫn*THS. TRẦN PHONG NHÃ**

***Sinh viên thực hiện*NGUYỄN THÀNH LIÊM  
MSSV : 6151071065**

*TPHCM, ngày tháng năm 2024*

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

**PHÂN HIỆU TẠI THÀNH PHỐ HỒ CHÍ MINH**

BỘ MÔN: CÔNG NGHỆ THÔNG TIN

-----🙞🙜🕮🙞🙜-----A picture containing picture frame, rectangle, frame

Description automatically generated



**ĐỒ ÁN TỐT NGHIỆP**

***Đề Tài* :**

XÂY DỰNG WEBSITE ĐỌC & QUẢN LÝ TRUYỆN ONLINE SỬ DỤNG CÔNG NGHỆ ASP.NET VÀ TÍCH HỢP THUẬT TOÁN RECOMMENDATION SYSTEM

Build a website to read and manage online stories using ASP.NET technology and integrated Recommendation system

***Giảng viên hướng dẫn*THS. TRẦN PHONG NHÃ  
*Sinh viên thực hiện*NGUYỄN THÀNH LIÊM  
MSSV : 6151071065**

*TPHCM, ngày tháng năm 2024*

|  |  |
| --- | --- |
| TRƯỜNG ĐH GIAO THÔNG VẬN TẢI PHÂN HIỆU TP.HCM  Bộ Môn Công Nghệ Thông Tin | CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM  Độc Lập – Tự Do -Hạnh Phúc |

**ĐỀ CƯƠNG CHI TIẾT ĐỒ ÁN TỐT NGHIỆP**

1. **Thông tin Sinh viên:**

Họ tên : Nguyễn Thành Liêm Mã sinh viên : 6151071065

Lớp : CQ\_CNTT - K61 Hệ : Chính Quy

Ngành đào tạo : Công nghệ thông tin Khoá : 61

Email : 6151071065@st.utc2.edu.vn Số điện thoại : 0327 780 481

1. **Thông tin Giảng viên hướng dẫn:**

Họ tên : Trần Phong Nhã Học vị : Thạc sĩ

Email : tpnha@utc2.edu.vn Số điện thoại : 0906 761 014

Đơn vị công tác: Trường Đại học Giao thông Vận tải PH tại TP Hồ Chí Minh

**LỜI CẢM ƠN**

* Sau quá trình học tập và rèn luyện tại Trường Đại học Giao thông vận tải Phân hiệu tại thành phố Hồ Chí Minh – UTC2 em đã trang bị được các kiến thức cơ bản , các kỹ năng thực tế để có thể hoàn thành Đồ án tốt nghiệp của mình.

Để hoàn thành Đồ án này , em xin cảm ơn chân thành đến :

* Ban giám hiệu trường Trường Đại học Giao thông vận tải Phân hiệu tại thành phố Hồ Chí Minh – UTC2 vì đã tạo điều kiện về cơ sở vật chất cũng như hệ thống thư viện hiện đại, đa dạng các loại sách và tài liệu thuận lợi cho việc tìm kiếm, nghiên cứu thông tin.
* Em xin cảm ơn chân thành và sâu sắc đến thầy THS. Trần Phong Nhã đã tận tình giúp đỡ, định hướng tư duy và cách làm việc khoa học. Đó là những góp ý hết sức quý báu không chỉ trong quá trình thực hiện Đồ án tốt nghiệp này mà còn là hành trang tiếp bước cho em trong quá trình học tập và lập nghiệp sau này.
* Và cuối cùng, em xin gửi lời cảm ơn đén gia đình, tất cả thầy cô trong khoa, bạn bè, tập thể lớp Công nghệ thông tin – K61 là những người luôn sẵn sàng sẽ chia và giúp đỡ trong học tập và cuộc sống.
* Trong quá trình làm Đồ án tốt nghiệp này em không tránh được những sai sót. Em kính mong nhận được sự chỉ dẫn và góp ý của quý thầy cô để được hoàn thiện hơn.

Em xin chân thành cảm ơn. Xin chúc những điều tốt đẹp nhất sẽ luôn đồng hành cùng mọi người.

**MỤC LỤC**

LỜI CẢM ƠN …………………………………………………………................

NHẬN XÉT CỦA GIẢNG VIÊN ….…………………………………................

DANH MỤC HÌNH ẢNH …………………………………………………….....

TỔNG QUAN ………………………………....…………………………….........

* Tên đề tài
* Giới thiệu
* Mục tiêu của đề tài

CHƯƠNG 1 : CƠ SỞ LÝ THUYẾT ……………...………………………….....

1.1 Giới thiệu mô hình MVC

1.2 Mô hình lập trình Web ASP.NET MVC

1.2.1 ASP.NET là gì ?

1.2.2 Cơ chế hoạt động của ASP.Net MVC

1.3 Kiến trúc MVC

1.3.1 Controller

1.3.2 View

1.3.3 Model

1.4 Lợi ích web dựa trên mô hình MVC

CHƯƠNG 2 : PHÂN TÍCH THIẾT KẾ HỆ THỐNG ...……………………..

2.1 Sơ đồ ERD

2.2 Sơ đồ Diagram

2.3 Sơ đồ Class

2.4 Sơ đồ Usercase

2.4.1 Sơ đồ Usercase tổng quát

2.4.2 Sơ đồ Usercase phân rã  
 2.5 Sơ đồ hoạt động

2.5.1 Sơ đồ hoạt động quản lý người dùng

2.5.2 Sơ đồ hoạt động quản lý thông tin truyện

2.5.3 Sơ đồ hoạt động quản lý thông tin thể loại

2.5.4 Sơ đồ hoạt động quản lý thông tin bài đăng

2.5.5 Sơ đồ hoạt động quản lý thông tin đánh giá

2.5.6 Sơ đồ hoạt động quản lý thông tin đọc truyện

2.6 Sơ đồ tuần tự

2.6.1 Sơ đồ tuần tự đăng nhập

2.6.2 Sơ đồ tuần tự quản lý người dùng

2.6.3 Sơ đồ tuần tự quản lý thông tin truyện

2.6.4 Sơ đồ tuần tự quản lý thông tin thể loại

2.6.5 Sơ đồ tuần tự quản lý thông tin bài đăng

2.6.6 Sơ đồ tuần tự quản lý thông tin đánh giá

2.6.7 Sơ đồ tuần tự quản lý thông tin đọc truyện

CHƯƠNG 3 : TRIỂN KHAI ỨNG DỤNG ………………………………….

3.1 Giao diện trang chủ

3.2 Giao diện trang chi tiết

3.2.1 Trang bộ lọc

3.2.2 Trang xem thông tin truyện

3.2.3 Trang đọc truyện

3.2.4 Trang xem thông tin người dùng

3.2.5 Trang bình luận

3.2.6 Trang đánh giá

3.2.7 Trang xem danh sách chương

3.2.8 Trang tìm kiếm

3.3 Giao diện Admin

3.3.1 Trang cá nhân

3.3.2 Trang truyện yêu thích

3.3.3 Trang đăng truyện

3.3.4 Trang truyện đã đăng

3.3.5Trang quản lý chương

3.3.6 Trang quản lý truyện

3.3.7 Trang quản lý người dùng

3.3.8 Trang quản lý thể loại

3.3.9 Trang quản lý thông báo

THUẬT TOÁN RECOMMENDATION ……………………………..…….

* Giới thiệu về hệ thống
* Hai nhóm chính của Recommendation Systems
* Utility Matrix
* Cosine Similarity
* TF-IDF

KẾT LUẬN ………………………………………………..……………………..

* Kết quả đạt được ……………………………...................................
* Tồn tại, hạn chế …..…………………………….................................
* Hướng phát triển ………………………………….............................

TÀI LIỆU THAM KHẢO ………………………………………………............

**DANH MỤC HÌNH ẢNH**

Hình 1.1 : Mô hình MVC …………………………………

Hình 1.2 : Mô hình lập trình ASP.NET MVC

Hình 1.3 : ASP.NET là gì?

Hình 1.4 : Cơ chế hoạt động của ASP.Net MVC

Hình 1.5 : Thêm mới Controller

Hình 1.6 : Route mặc định

Hình 1.7 : ActionResult

Hình 1.8 : ActionVerb

Hình 1.9 : Cơ chế Routing trong ASP.NET

Hình 1.10 : View

Hình 1.11 : Layout View

Hình 1.12 : ViewStart.cshtml

Hình 2.1 : Sơ đồ ERD

Hình 2.2 : Sơ đồ Diagram

Hình 2.3 : Sơ đồ Class

Hình 2.4 : Sơ đồ Usercase tổng quát

Hình 2.5 : Sơ đồ phân rã Usecase Quản lý người dùng

Hình 2.6 : Sơ đồ phân rã Usecase Quản lý truyện

Hình 2.7 : Sơ đồ phân rã Usecase Quản lý thể loại

Hình 2.8 : Sơ đồ phân rã Usecase Quản lý bài đăng

Hình 2.9 : Sơ đồ phân rã Usecase Quản lý đánh giá

Hình 2.10 : Sơ đồ phân rã Usecase Quản lý đọc truyện

Hình 2.11 : Sơ đồ hoạt động Quản lý người dùng

Hình 2.12 : Sơ đồ hoạt động Quản lý thông tin truyện

Hình 2.13 : Sơ đồ hoạt động Quản lý thông tin thể loại

Hình 2.14 : Sơ đồ hoạt động Quản lý thông tin bài đăng

Hình 2.15 : Sơ đồ hoạt động Quản lý thông tin đánh giá

Hình 2.16 : Sơ đồ hoạt động Quản lý đọc truyện

Hình 2.17 : Sơ đồ tuần tự đăng nhập

Hình 2.18 : Sơ đồ tuần tự xem danh sách người dùng

Hình 2.19 : Sơ đồ tuần tự xem danh sách truyện

Hình 2.20 : Sơ đồ tuần tự xem danh sách thể loại

Hình 2.21 : Sơ đồ tuần tự xem danh sách bài đăng

Hình 2.22 : Sơ đồ tuần tự xem danh sách đánh giá

Hình 2.23 : Sơ đồ tuần tự đọc truyện

Hình 3.1 : Giao diện trang chủ

Hình 3.2 : Trang bộ lọc

Hình 3.3 : Trang xem thông tin truyện

Hình 3.4 : Trang đọc truyện

Hình 3.5 : Trang xem thông tin người dùng

Hình 3.6 : Trang bình luận

Hình 3.7 : Trang đánh giá

Hình 3.8 : Trang xem danh sách chương

Hình 3.9 : Trang tìm kiếm

Hình 3.10 : Trang cá nhân

Hình 3.11 : Trang truyện yêu thích

Hình 3.12 : Trang đăng truyện

Hình 3.13 : Trang truyện đã đăng

Hình 3.14 : Trang quản lý chương

Hình 3.15 : Trang quản lý truyện

Hình 3.16 : Trang quản lý người dùng

Hình 3.17 : Trang quản lý thể loại

Hình 3.18 : Trang quản lý thông báo

Hình 4.1 : Hai nhóm chính của Recommendation Systems

Hình 4.2 : Utility matrix

Hình 4.3 : Consine similarity

Hình 4.4 : Công thức tích similarity giữa A và B

Hình 4.5 : Sự tương đồng của 2 vector

**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

*Tp. Hồ Chí Minh, ngày …... tháng 3 năm 2024*

**GIẢNG VIÊN HƯỚNG DẪN**

*( Ký và ghi rõ họ tên )*

……………………………………………....

# TỔNG QUAN VỀ ĐỀ TÀI

**Tên đề tài :**

Xây dựng Website đọc và quản lý truyện sử dụng công nghệ ASP.NET tích hợp thuật toán Recommendation system.

**Giới thiệu :**

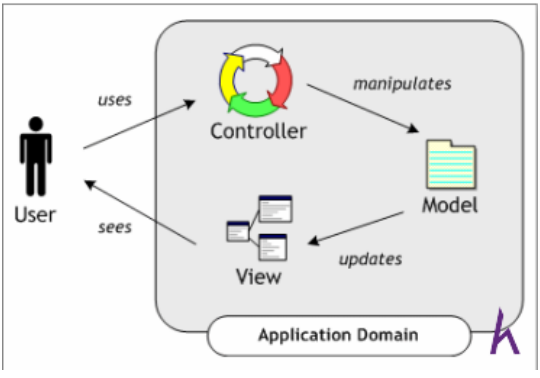
* ***Bối cảnh:***
* Truyện tranh và truyện chữ ngày càng trở thành một phương tiện giải trí phổ biến trên Internet. Đọc giả muốn đọc truyện một cách thuận tiện từ bất kỳ thiết bị nào có kết nối Internet.
* Xây dựng một trang web đọc và quản lý truyện online giúp cung cấp một nền tảng tốt để đọc giả có thể truy cập và đọc truyện một cách dễ dàng.
* ***Sự cần thiết :***
* Sự tiện lợi : Đọc giả có thể đọc truyện từ bất kỳ nơi nào cũng như bất kỳ thiết bị nào có kết nối Internet mà không cần phải mua sách hoặc tải về.
* Lưu trữ lớn : Trang web có thể lưu trữ lớn một lượng lớn truyện tranh và truyện chữ , cung cấp sự đa dạng và lựa chọn cho đọc giả.
* Tính tương tác : Trang web có thể cung cấp các tính năng như tìm kiếm bằng giọng nói hoặc tìm một nội dung truyện nào đó mà đọc giả nhớ mà không biết tên tác phẩm đó là gì , đồng thời trang web cũng gợi ý một số tác phẩm tươnng tự mà đọc giả có thể thích hoặc quan tâm. Ngoài ra trang web còn cung cấp các tính năng như cho đọc giả có thể đăng tác phẩm mà mình viết hoặc yêu thích, đánh giá , bình luận các tác phẩm cũng những đọc giả khác , tạo ra một cộng đồng đọc truyện trực tuyến.
* Tính logic : Xây dựng một trang web đọc và quản lý truyện online sử dụng công nghệ ASP.NET MVC cho phép sử dụng các công nghệ và khung công việc mạnh mẽ của ASP.NET để phát triển úng dụng trang web một cách linh hoạt và có hiệu suất cao. Nó cung cấp một kiến trúc phân lớp logic và rõ ràng sẽ giúp quản lý dễ dàng dữ liệu truyện và tương tác với đọc giả.

**Mục tiêu đề tài:**

* Tìm hiểu và làm việc với ngôn ngữ C#.
* Thiết kế web sử dụng HTML, CSS, JavaScript.
* Làm việc với CSDL dữ liệu SQL Server.
* Phát triển và xây dựng được các ứng dụng WEB với ASP.NET.
* Sử dụng Entity Framework để tương tác với CSDL.
* Xây dựng được ứng dụng ASP.NET theo mô hình MVC.

# CHƯƠNG 1. CƠ SỞ LÝ THUYẾT

* 1. **Giới thiệu mô hình MVC**

****

Hình 1.1 : Mô hình MVC

* Ví dụ: User yêu cầu hiển thị thông tin cá nhân của user
* User gửi một yêu cầu tới controller.
* Controller nhận yêu cầu, xử lý yêu cầu, nếu yêu cầu cần truy

xuất dữ liệu thì controller sẽ gửi yêu đó xuống tầng model để

truy xuất dữ liệu.

* Tầng model sẽ lấy dữ liệu từ database sau đó truyền dữ liệu qua

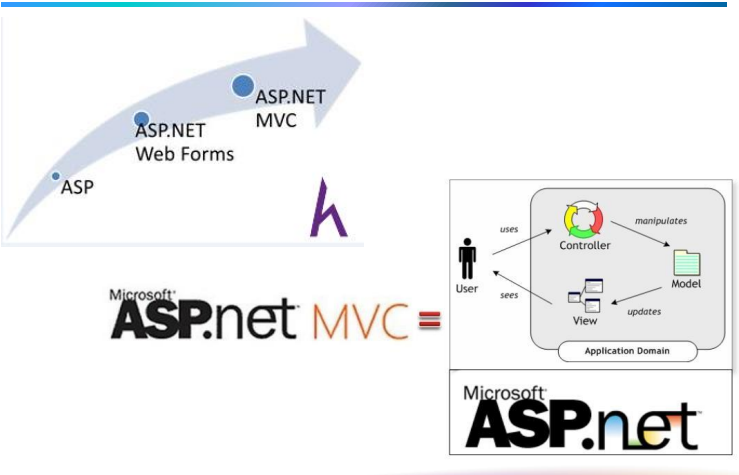
tầng view thông qua tầng controller để tầng view hiển thị dữ liệu

cho User.

* User sẽ thấy thông tin hiển thị ở giao diện và cụ thể ở đây là tầng

View.

* 1. **Mô hình lập trình Web ASP.NET MVC**

****

Hình 1.2 : Mô hình lập trình ASP.NET MVC

**1.2.1 ASP.NET là gì ?**

* ASP.NET là 1 framwork lập trình web được cung cấp bởi

Microsoft.

* Sự xuất hiện của ASP.NET vào năm 2002 đã thay đổi diện

mạo và cách thức phát triển web.



Hình 1.3 : ASP.NET là gì?

* ***Nhược điểm khi lập trình với ASP.Net web forms:***
* Web Forms không có sự tách biệt rõ ràng 3 tầng trong mô

hình lập trình web.

* Web Forms sử dụng đối tượng View State mà đối tượng

này làm cho trang web bị tăng kích thước vì vậy làm giảm

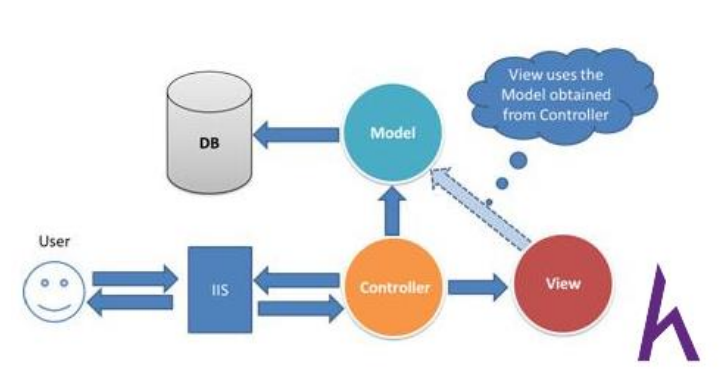
hiệu năng của ứng dụng.

* Giao diện của Web Forms được thiết kế dựa trên toolbox có

sẵn gây khó khăn trong việc thiết kế web hiện nay, việc

thiết kế trở nên mất linh hoạt.

**1.2.2 Cơ chế hoạt động của ASP.Net MVC**



Hình 1.4 : Cơ chế hoạt động của ASP.Net MVC

* ***MVC làm việc như thế nào?***
* User tương tác với View, bằng cách click vào button,

gửi yêu cầu đi.

* Controller nhận và điều hướng đến phương thức xử lý

ở Model.

* Model nhận thông tin và thực thi các yêu cầu, View sẽ

nhận kết quả từ Model và hiển thị lại cho người dùng.

* 1. **Kiến trúc MVC**
     1. **Controller**

**1.3.1.1 Khái niệm**

* Controller trong Framework MVC xử lý mọi yêu cầu

URL đến. Controller là một lớp trong namespace

System.Web.Mvc.Controller. Lớp Controller chứa các

phương thức public gọi là các phương thức Action.

Controller và phương thức action của nó xử lý các

yêu cầu trình duyệt đến, lấy dữ liệu mô hình cần thiết

và trả về cho trình duyệt.

* Trong ASP.NET MVC, mọi tên lớp của trình điều

khiển phải kết thúc bằng một từ "Controller".

* Ví dụ: bộ điều khiển cho trang chủ phải là

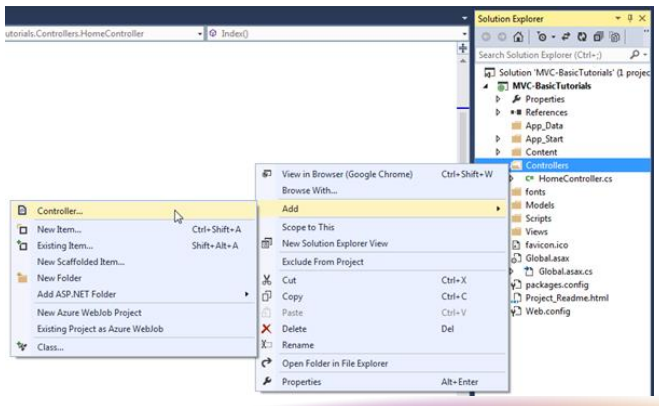
HomeController và bộ điều khiển cho truyện phải

là NovelController. Ngoài ra, mỗi lớp Controller

được đặt trong thư mục Controller của cấu trúc thư

mục MVC.

* Controller có 3 trách nhiệm chính:
* Nhận request
* Dựng model
* Gửi trả response
  + - 1. **Tạo và sử dụng Controller**
* ***Thêm mới Controller:***



Hình 1.5 : Thêm mới Controller

* + Ở hộp thoại Add Scaffold, chọn MVC 5 Controller –

Empty và nhấn Add. Ở bước này, do chúng ta mới

làm quen với Controller cho nên chúng ta sử dụng

mẫu Controller dạng Empty, tức là nội dung trống. Từ

đó, chúng ta sẽ viết 1 số đoạn code làm quen. Ở

những bài tiếp theo, các bạn sẽ làm quen các dạng

Controller khác.

* + Controller xử lý các yêu cầu của URL gửi đến.

Controller MVC gửi yêu cầu đến controller và

phương thức hành động dựa trên URL và các

Controller được định cấu hình.

* + Controller kế thừa từ System.Web.Mvc.Controller.
  + Một Controller mới có thể được tạo bằng các mẫu

scaffolding khác nhau. Bạn có thể tạo mẫu

scaffolding tùy ý.

**1.3.1.3 Action Methods**

* Tất cả các phương thức public của lớp Controller

được gọi là Phương thức hành động (Action). Chúng

giống như bất kỳ phương thức bình thường nào khác

với các hạn chế sau:

* Phương thức Action phải là public. Không thể private hay

Protected.

* Phương thức Action không thể nạp chồng (overloaded).
* Phương thức Action không thể là phương thức static.
* ***Phương thức Action mặc định:***
* Mỗi controller có phương thức action mặc định được

cài đặt trong lớp RouteConfig. Phương thức Index là

phương action mặc định, được cài đặt mặc định.

* Route mặc định :



Hình 1.6 : Route mặc định

* ***ActionResult:***
* MVC framework chứa nhiều lớp kết quả khác

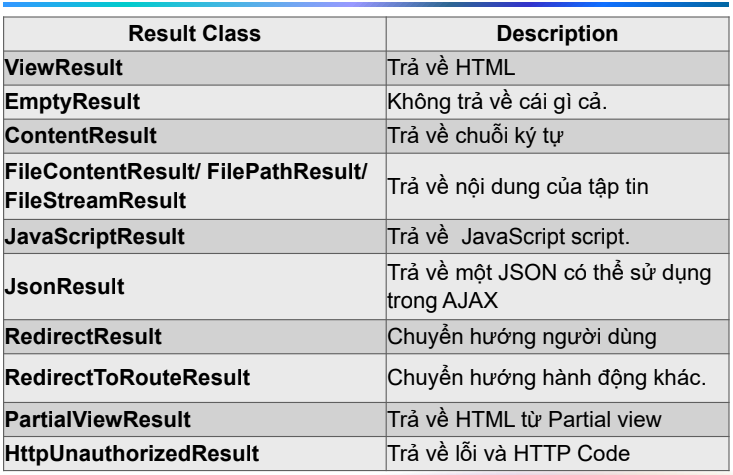
nhau, có thể được trả về từ một phương thức hành

động. Có các lớp kết quả đại diện cho các loại

phản hồi khác nhau, chẳng hạn như html, tệp,

chuỗi, json, javascript,... Bảng sau liệt kê tất cả

các lớp kết quả có sẵn trong ASP.NET MVC.



Hình 1.7 : ActionResult

* Lớp ActionResult là một lớp cơ sở của tất cả các lớp

kết quả ở trên, do đó, nó có thể là kiểu trả về của các

phương thức hành động trả về bất kỳ loại kết quả nào

được liệt kê ở trên. Tuy nhiên, bạn có thể chỉ định lớp

kết quả phù hợp làm kiểu trả về của phương thức

hành động.

* ***Các tham số của phương thức Action:***
* Mỗi phương thức hành động có thể có các tham số

đầu vào như các phương thức bình thường.

* Các tham số có thể là kiểu dữ liệu nguyên thủy

hoặc tham số kiểu phức tạp như trong ví dụ ở slide

sau.

* ***ActionVerb:***
* Một phương thức action khi được định nghĩa có thể được

gọi theo Get và Post. Cách này cho phép lập trình viên có

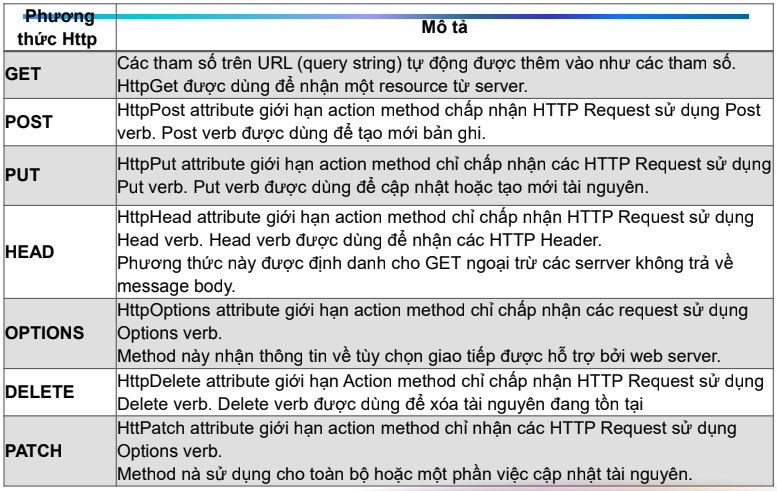
thể định nghĩa hai phương thức khác nhau nhưng cùng tên,

một phương thức có thể dùng Http Get và phương thức kia

thì đáp có thể dùng HttpPost.

* MVC Framework hỗ trợ nhiều ActionVerb, như HttpGet,

HttpPost, HttpPut, HttpDelete, HttpOptions & HttpPatch.



Hình 1.8 : ActionVerb

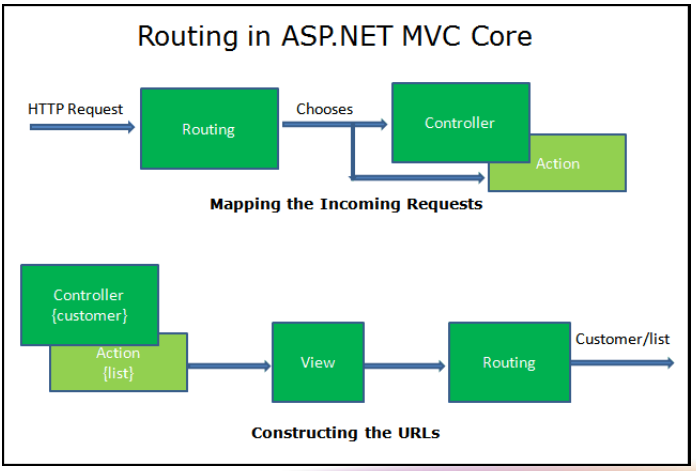
* + - 1. **Cơ chế Routing trong ASP.NET**
* Một trong các thành phần quan trọng nhất của kiến

trúc MVC là cơ chế routing (định tuyến). Nó là cơ

chế quyết định xem Controller nào sẽ được xử lý

request nào.

* Routing có 2 trách nhiệm chính:
* Nó map request đến vào Controller Action.
* Tạo ra URL đầu ra tương ứng với Controller action.



Hình 1.9 : Cơ chế Routing trong ASP.NET

* Mỗi Route bao gồm các thông tin như tên, mẫu URL

(URL pattern) hay còn gọi là template url, thông tin

controller action mặc định và ràng buộc (constraints).

URL pattern được so sánh với URL đến xem có đúng

mẫu không. Một ví dụ của URL pattern là:

{controller=Home}/{action=Index}/{id?}

* Route được định nghĩa trong Microsoft.AspNetCore.Routing.

**1.3.2 View**

**1.3.2.1 Khái niệm**

* View là một giao diện người dùng. View chính là các

thành phần chịu trách nhiệm hiển thị các thông tin lên cho

người dùng thông qua giao diện. Thông thường, các thông

tin cần hiển thị được lấy từ thành phần Models.

* Ví dụ, đối tượng Novel có một “Edit” view bao gồm các

textboxes, các dropdowns và checkboxes để chỉnh sửa các

thuộc tính của tác phẩm; có một “Display” view gồm 2

dòng, cột dòng là NovelID, dòng sau là Title... để

xem thông tin về truyên.

* View ASP.NET MVC được lưu trong thư mục Views. Các

phương thức hành động khác nhau của một lớp controller

duy nhất có thể hiển thị các Views khác nhau, do đó, thư

mục Views chứa một thư mục riêng cho mỗi controller có

cùng tên controller, để phù hợp với nhiều Views.

* Ví dụ: các Views sẽ được hiển thị từ bất kỳ phương thức

hành động nào của HomeController, nằm trong thư mục

Views -> Home. Tương tự các Views sẽ được hiển thị từ

NovelController, sẽ nằm trong thư mục Views -> Novel.

* Microsoft đã giới thiệu công cụ Razor được đóng gói

với MVC 3. Bạn có thể viết hỗn hợp các thẻ html và

mã phía máy chủ trong razor. Razor sử dụng ký tự @

cho mã phía máy chủ thay vì <%%>. Bạn có thể sử

ngôn ngữ C # hoặc Visual Basic để viết mã phía máy

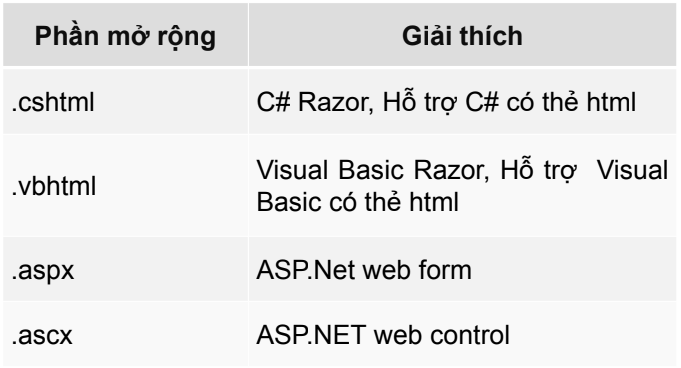
chủ bên trong Razor.

* Razor tối đa hóa tốc độ viết mã bằng cách giảm thiểu

số lượng ký tự và tổ hợp phím cần thiết khi viết trong

View. Các tập tin Razor có phần mở rộng .cshtml

hoặc .vbhtml.



Hình 1.10 : View

**1.3.2.2 Truyền dữ liệu từ Controller về View**

* ***ViewData trong MVC ASP.NET:***
* ViewData là một thuộc tính của Controller base class, nó

trả về một đối tượng ViewDataDictionary. ViewDataDictionary

là một đối tượng dictionary cho phép lưu dữ liệu dạng

key-value. Key phải là một chuỗi không phân biệt chữ hoa thường.

* ViewData truyền dữ liệu sang View từ Controller.
* Khi chúng ta gọi phương thức View trong Controller

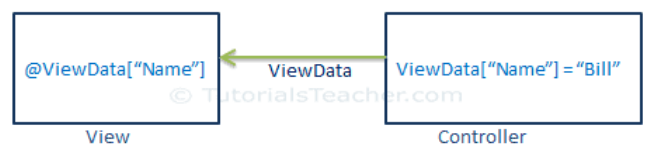
action, ViewData sẽ tự động gán vào View. Trong View

chúng ta có thể truy cập giá trị được lưu trong ViewData

cũng sử dụng key. Dữ liệu được lưu trong ViewData tồn tại

chỉ trong request đó. Khi View được tạo xong cho client thì

đối tượng ViewData đó cũng bị hủy.



* ***ViewBag trong MVC ASP.NET:***
* Là một DynamicViewData object, nó là một lớp bao bọc

(wrap) ViewData để cho phép truy cập vào object một cách

linh hoạt.

* ViewBag cũng cho phép chúng ta sử dụng dynamic

properties (dùng dấu chấm thay vì ngoặc vuông như

ViewData). Sử dụng ViewBag cũng tương tự như

ViewData nhưng sẽ tiện lợi hơn vì nó không cần phải ép

kiểu.

* ***TempData trong MVC ASP.NET:***
* TempData trong ASP.NET MVC có thể được sử dụng để

lưu trữ dữ liệu tạm thời có thể được sử dụng trong yêu cầu

tiếp theo. TempData sẽ bị xóa sau khi hoàn thành một yêu

cầu tiếp theo.

* TempData rất hữu ích khi chúng ta muốn chuyển dữ liệu từ

một phương thức hành động này sang một phương thức

hành động khác của cùng một controller. Đây là loại từ điển

có nguồn gốc từ TempDataDipedia.

* TempData có thể được sử dụng để lưu trữ dữ liệu giữa hai

yêu cầu liên tiếp.

* TemData là một kiểu TempDataDictionary.
* TempData sử dụng Session để lưu trữ dữ liệu.
* Giá trị của TempData phải được ép kiểu trước khi sử dụng.
* TempData chỉ có thể được sử dụng để lưu trữ các thông báo

một lần như thông báo lỗi, thông báo xác thực.

* Gọi TempData.Keep() để giữ lại các giá trị TempData trong

yêu cầu thứ ba tiếp theo.

* ***Sử dụng Partial View trong MVC ASP.NET:***
* Partial view trong ASP.NET MVC là một view sử

dụng lại, có thể được sử dụng như là một view con

trong nhiều view khác. Partial view giúp loại bỏ trùng

lặp mã bằng cách sử dụng lại cùng một partial view ở

nhiều view khác. Chúng ta có thể sử dụng partial

view là một phần trong layout view, cũng như nội

dung view khác.

**1.3.2.3 Layout View**

* Layout cho phép bạn xác định một mẫu trang web

chung, có thể được kế thừa trong nhiều view để cung

cấp giao diện nhất quán nhiều trang trong một ứng

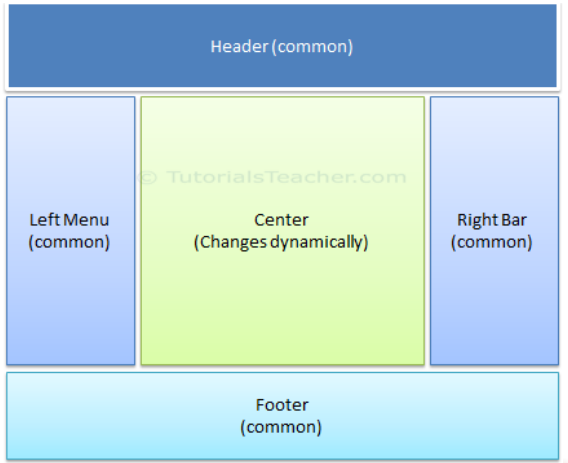
dụng. Layout bố trí giúp loại bỏ mã trùng lặp.

* Giao diện bố trí cho giao diện người dùng ở trên sẽ

chứa phần Header, Menu bên trái, Thanh bên phải và

Phần Footer. Nó chứa một placeholder nằm ở Center

để dễ dàng thay đổi nội dung.



Hình 1.11 : Layout View

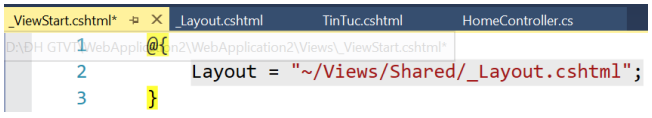
* ***Sử dụng Layout***
* Làm thế nào để View biết nên sử dụng Layout nào?
* Bạn có thể đặt layout theo nhiều cách:
* Sử dụng \_ViewStart.cshtml
* Hoặc thiết lập đường dẫn của layout bằng thuộc tính layout trong view riêng lẻ
* Hoặc gọi layout trong phương thức.
* ***Sử dụng \_ViewStart.cshtml***
* Theo mặc định, ViewStart.cshtml nằm trong thư mục

Views. Nó thiết lập trang layout mặc định cho tất cả các

view trong thư mục và các thư mục con bằng thuộc tính

Layout. Bạn có thể chỉ định đường dẫn hợp lệ của bất kỳ

trang Layout nào cho thuộc tính Layout.



Hình 1.12 : ViewStart.cshtml

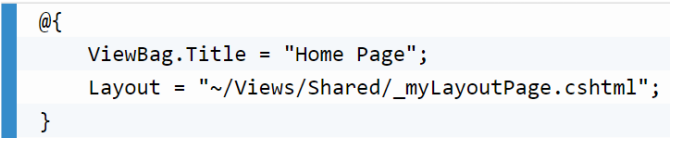
* ***Cài đặt thuộc tính Layout cho từng views:***
* Bạn cũng có thể ghi đè trang layout mặc định được đặt bởi

\_ViewStart.cshtml bằng cách đặt thuộc tính Layout trong

mỗi view .cshtml riêng lẻ. Ví dụ: view Index sử dụng

\_myLayoutPage.cshtml khi đó ta đổi \_ViewStart.cshtml

thành \_Layout.cshtml.



Hình 1.13

**1.3.2.4 Sử dụng View Razor**

* Razor View Engine là 1 ngôn ngữ cho phép bạn tạo ra

các giao diện cho ứng dụng ASP.NET MVC trong khi

vẫn giữ được sự phân chia rõ ràng, khả năng có thể

kiểm tra, và sự phát triển dựa trên pattern.

* ***Cú pháp Razor có các đặc điểm sau***
* Compact: Razor nhỏ gọn cho phép bạn giảm thiểu số

lượng ký tự và tổ hợp phím cần thiết để viết mã.

* Easy to Learn: Razor dễ học, bạn có thể dụng các

ngôn ngữ lập trình: C#, Visual Basic.

* Intellisense: Razor hỗ trợ các câu lệnh trong Visual

Studio.

* Unit Testable: Razor hỗ trợ khả năng unit test cho các

view mà không cần các controller hoặc web-server.

**1.3.3 Model**

**1.3.3.1 Khái niệm**

* Models trong các ứng dụng dựa trên MVC là những

thành phần có nhiệm vụ lưu trữ thông tin, trạng thái

của các đối tượng, thông thường nó là một lớp được

ánh xạ từ một bảng trong CSDL.

* Lấy ví dụ, chúng ta có lớp Novel được sử dụng để

mô tả dữ liệu từ bảng Novel trong SQL, bao gồm

NovelID, Title ,...

**1.3.3.2 Models Binding**

* Để hiểu ràng buộc mô hình trong MVC, trước tiên

hãy xem cách bạn có thể nhận các yêu cầu từ http

trong phương thức hành động bằng cách sử dụng kiểu

ASP.NET truyền thống.

* Với liên kết mô hình, framework MVC chuyển đổi

các giá trị yêu cầu từ http (từ chuỗi truy vấn hoặc

form) thành các tham số phương thức hành động. Các

tham số này có thể là kiểu nguyên thủy hoặc kiểu

phức tạp.

* ***Liên kết với kiểu nguyên thủy***
* Các httpGET nhúng dữ liệu vào chuỗi truy vấn. Framework

MVC tự động chuyển đổi một chuỗi truy vấn thành các

tham số của phương thức hành động.

* Ví dụ http://localhost/Novel/Edit?id=1&name=phamnhantutien sẽ

ánh xạ tới id và tham số tên của phương thức hành động

Edit

* ***Liên kết với kiểu phức tạp***
* Dữ liệu gửi qua POST có thể được ánh xạ tự động vào các tham số của phương thức action hoặc vào các đối tượng phức tạp. HTTP POST thường được sử dụng cho các thao tác gửi dữ liệu phức tạp hơn và có thể thay đổi trạng thái của server.

**1.4 Lợi ích web dựa trên mô hình MVC**

* Dễ dàng quản lý sự phức tạp của ứng dụng bằng

cách chia ứng dụng thành Model, View, Controller

* Không sử dụng view state hoặc server-based form.

Điều này tốt cho những lập trình viên muốn quản lý

hết các khía cạnh của một ứng dụng.

* Sử dụng mẫu Front Controller, mẫu này giúp quản lý

các requests (yêu cầu) chỉ thông qua một Controller.

* Hỗ trợ tốt hơn cho mô hình phát triển ứng dụng

hướng kiểm thử (TDD)

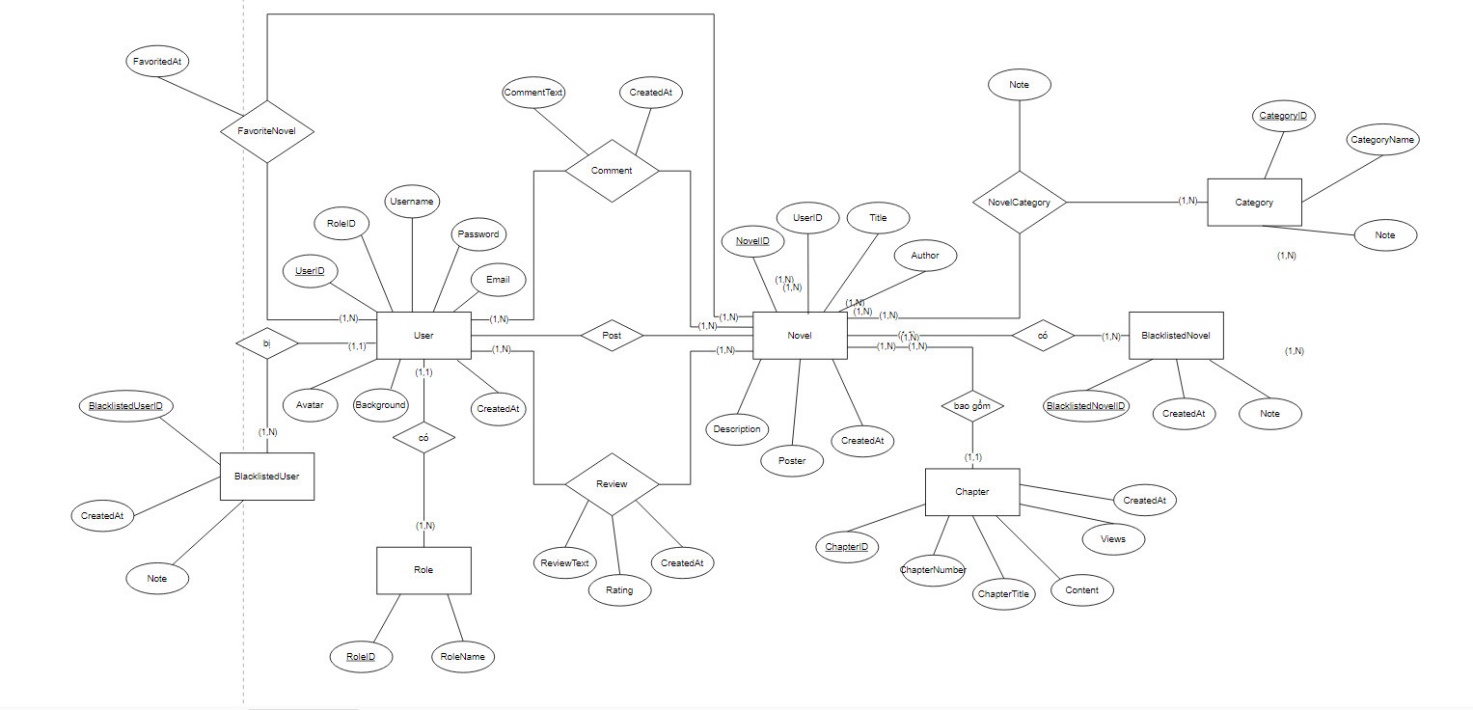
* Hỗ trợ tốt cho các ứng dụng được xây dựng bởi

hững đội có nhiều lập trình viên và thiết kế mà vẫn

quản lý được tính năng của ứng dụng

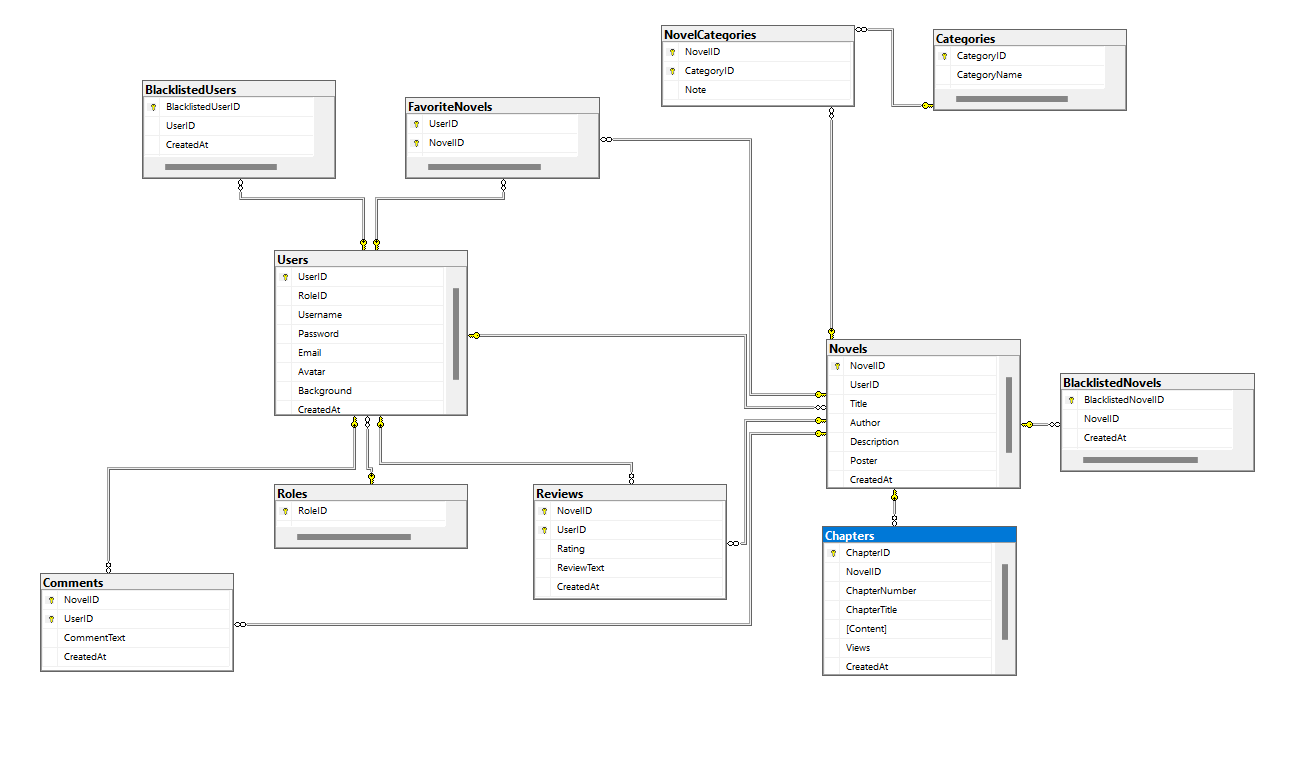
# CHƯƠNG 2. PHÂN TÍCH THIẾT KẾ HƯỚNG ĐỐI TƯỢNG

## 2.1 Sơ đồ ERD:



Hình 2.1 : Sơ đồ ERD

## 2.2 Sơ đồ Diagram:



Hình 2.2 : Sơ dồ Diagram

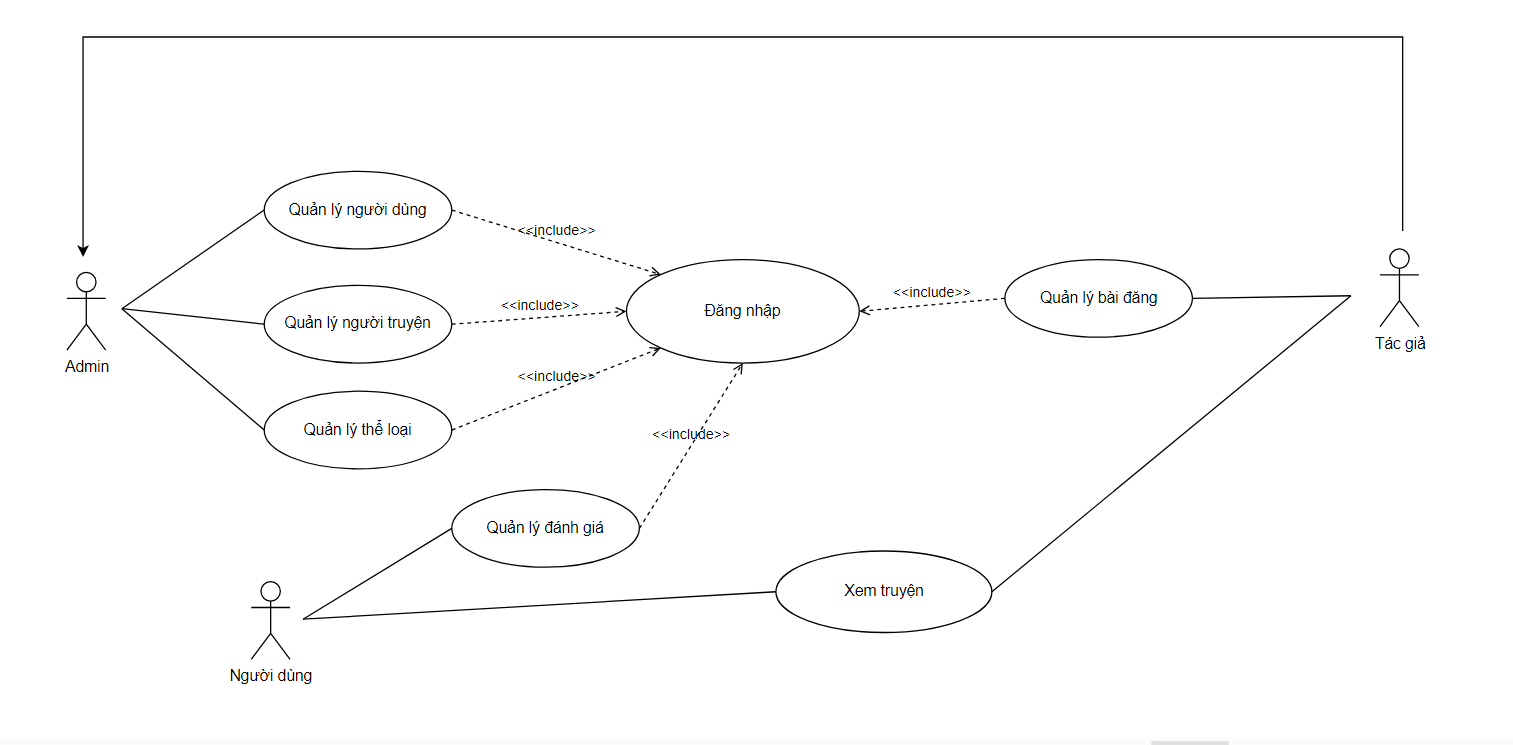
## 2.3 Sơ đồ Class :

## 

Hình 2.3 : Sơ đồ Class

## 2.4 Sơ đồ Usecase:

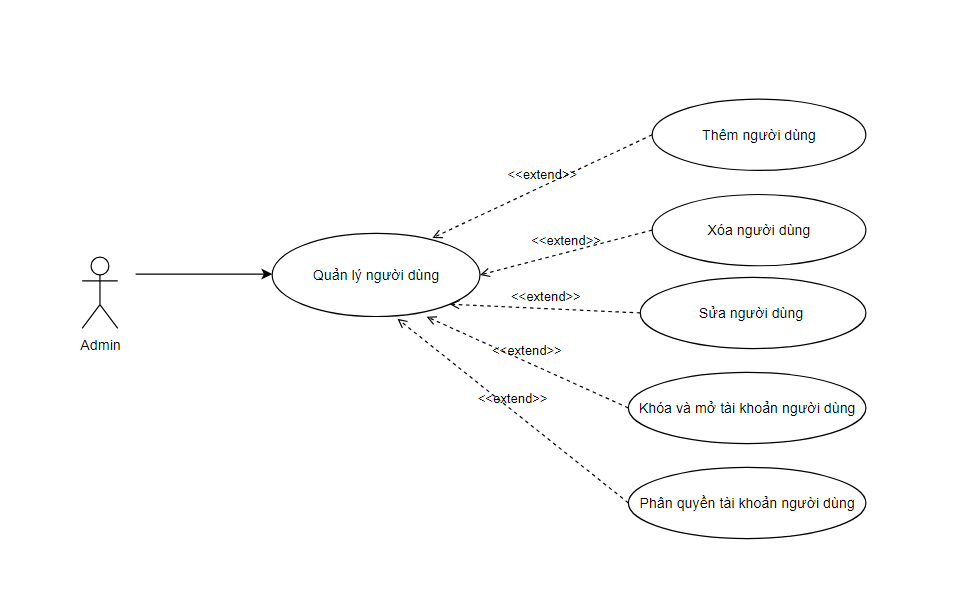
### 2.4.1 Sơ đồ Usecase Tổng Quát



Hình 2.4 : Sơ đồ Usecase Tổng Quát

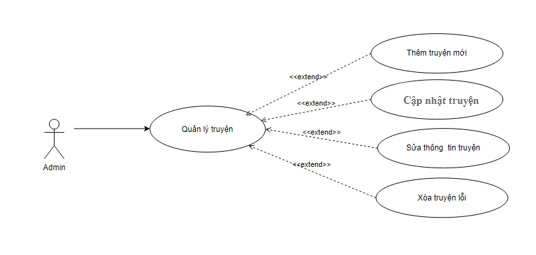
### 2.4.2 Sơ đồ Usecase Phân Rã

#### 2.4.2.1 Phân rã Usecase Quản Lý Người dùng



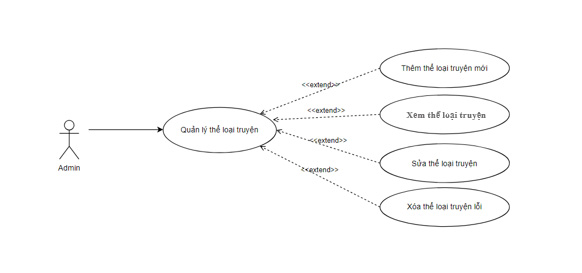
Hình 2.5 : Sơ đồ phân rã Usecase Quản lý người dùng

#### 2.4.2.2 Phân rã usecase Quản lý truyện

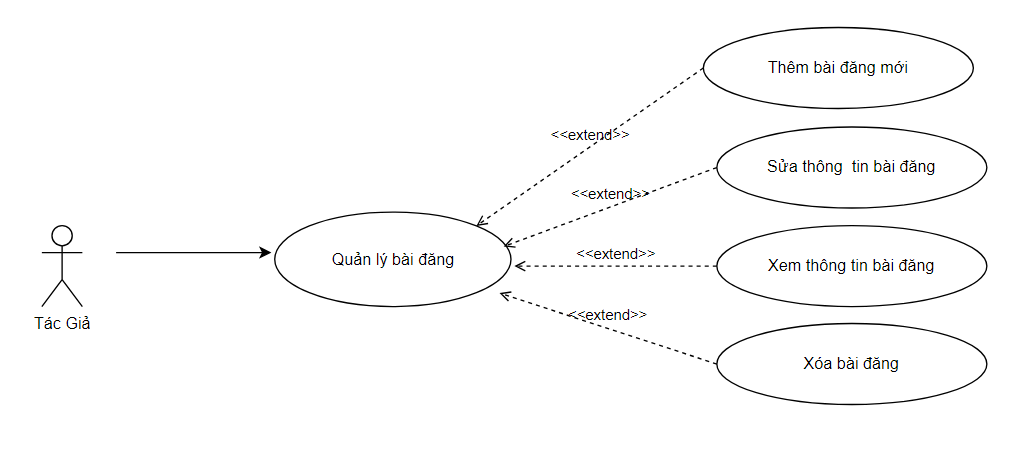


Hình 2.6 : Sơ đồ phân rã Usecase Quản lý truyện

#### 2.4.2.3 Phân rã usecase Quản lý thể loại

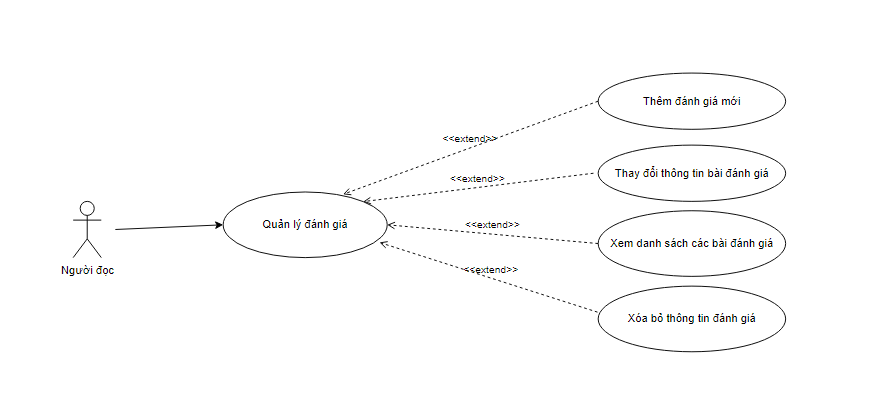


Hình 2.7 : Sơ đồ phân rã Usecase Quản lý thể loại



Hình 2.8 : Sơ đồ phân rã Usecase Quản lý bài đăng

#### 2.4.2.4 Phân rã usecase Quản lý đánh giá



Hình 2.9 : Sơ đồ phân rã Usecase Quản lý đánh giá

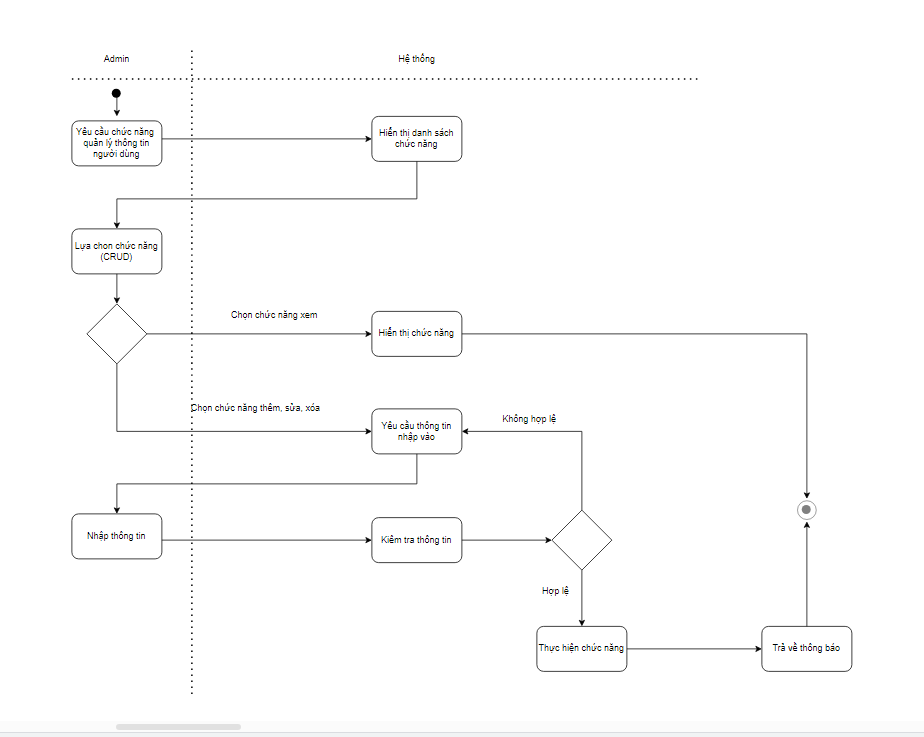
#### 2.4.2.5 Phân rã usecase Quản lý đọc truyện



Hình 2.10 : Sơ đồ phân rã Usecase Quản lý đọc truyện

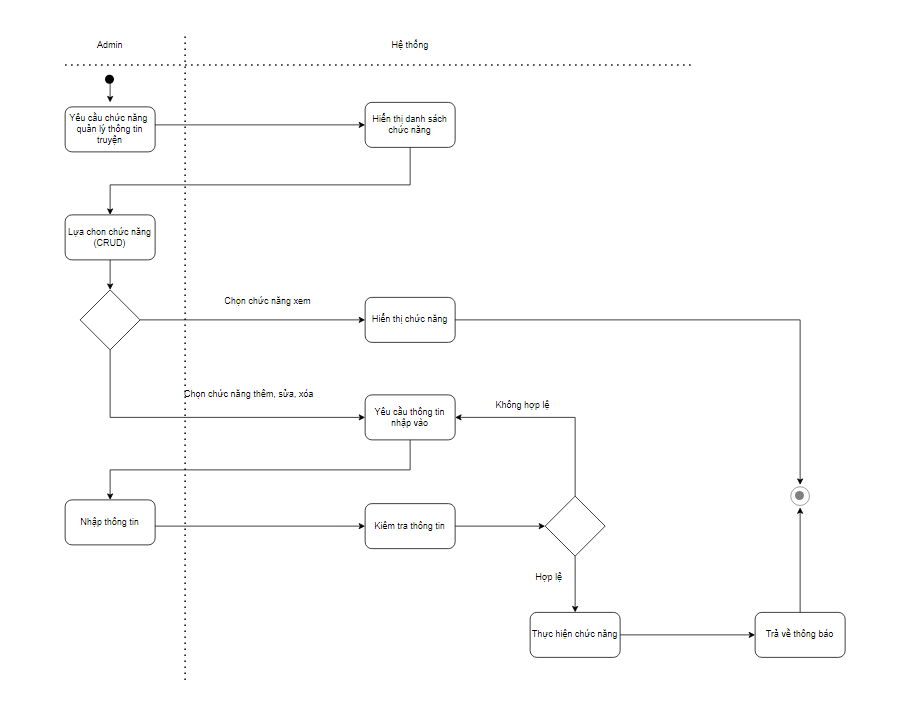
## 2.5 Thiết kế sơ đồ hoạt động

### 2.5.1 Quản lý người dùng



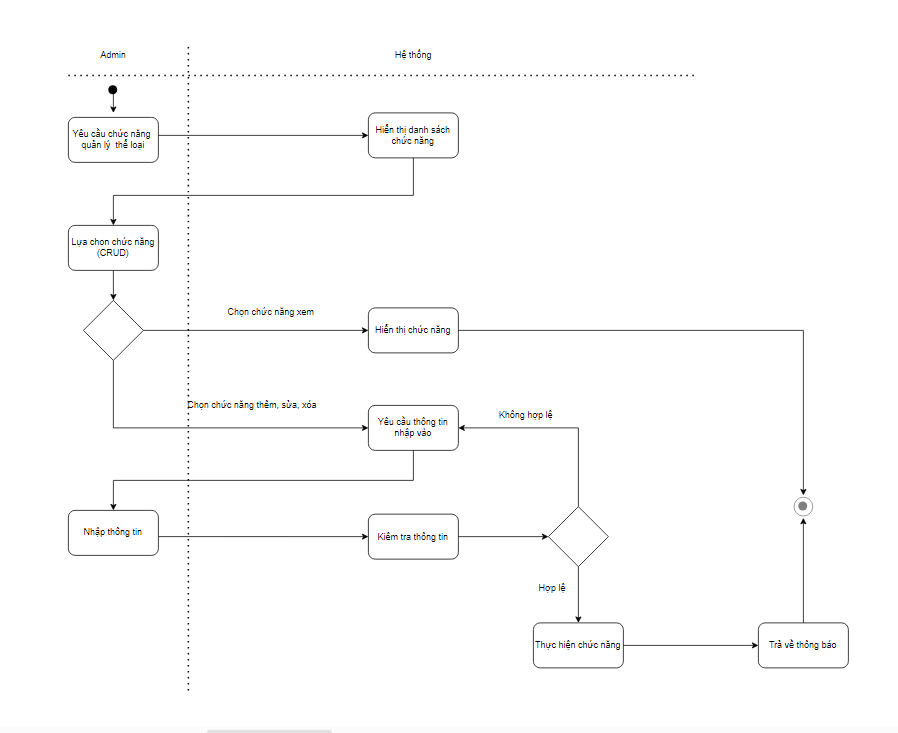
Hình 2.11 : Sơ đồ hoạt động quản lý người dùng

### 2.5.2 Quản lý thông tin truyện



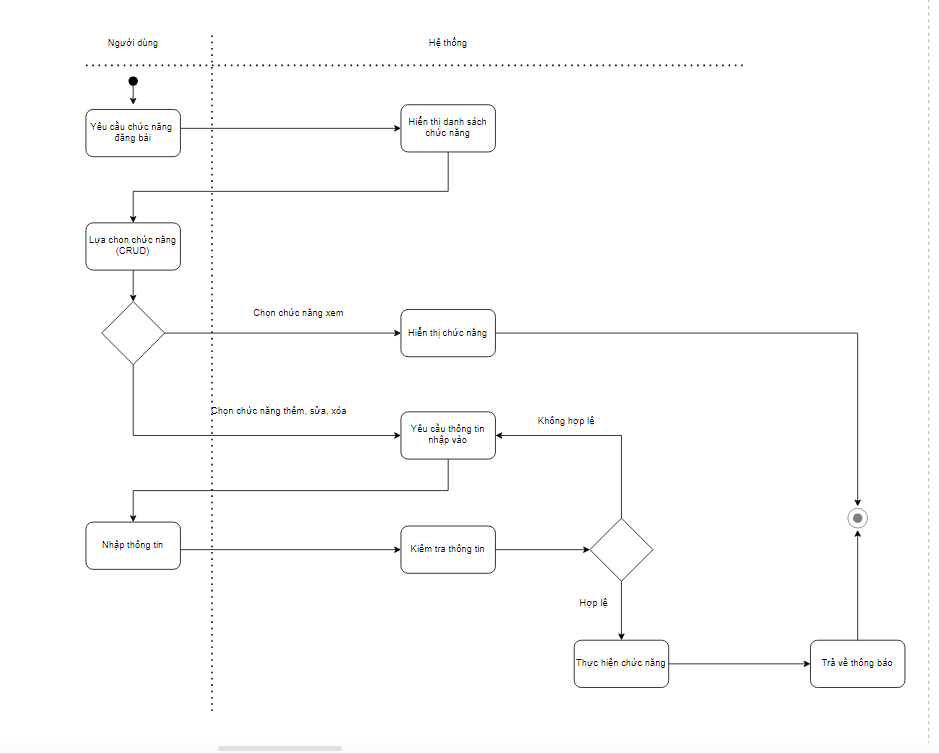
Hình 2.12 : Sơ đồ hoạt động quản lý thông tin truyện

### 2.5.3 Quản lý thông tin thể loại



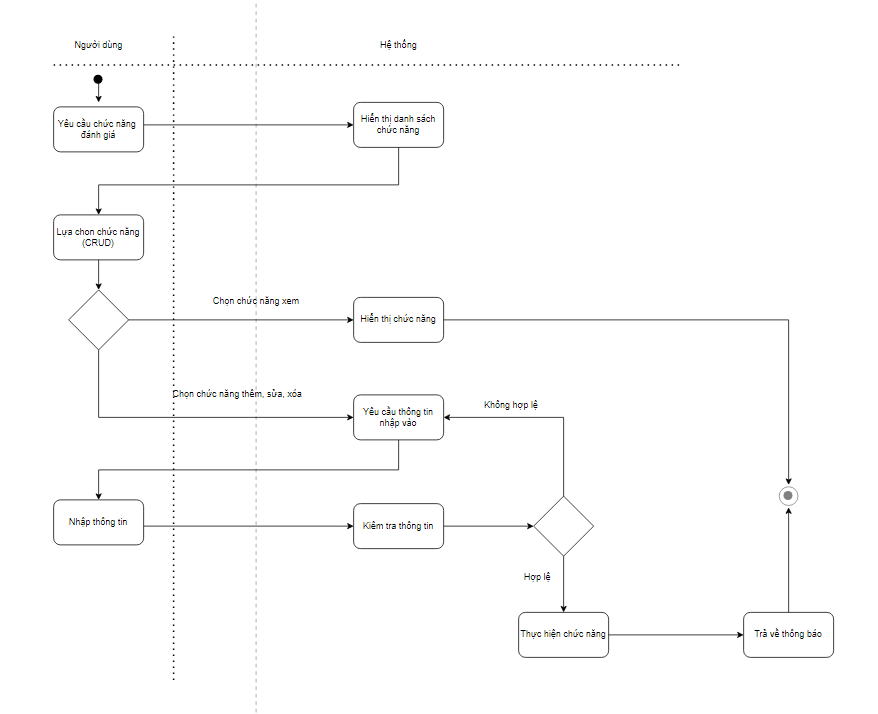
Hình 2.13 : Sơ đồ hoạt động quản lý thông tin thể loại

### 2.5.4 Quản lý thông tin bài đăng



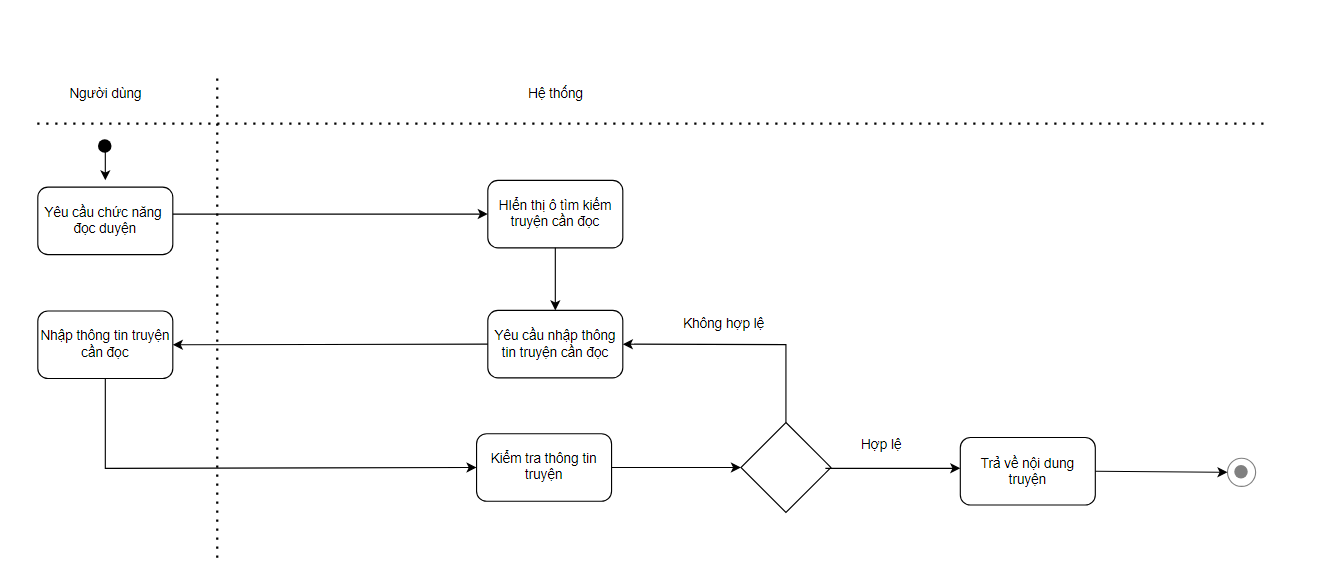
Hình 2.14 : Sơ đồ hoạt động quản lý thông tin bài đăng

### 2.5.5 Quản lý thông tin đánh giá



Hình 2.15 : Sơ đồ hoạt động quản lý thông tin đánh giá

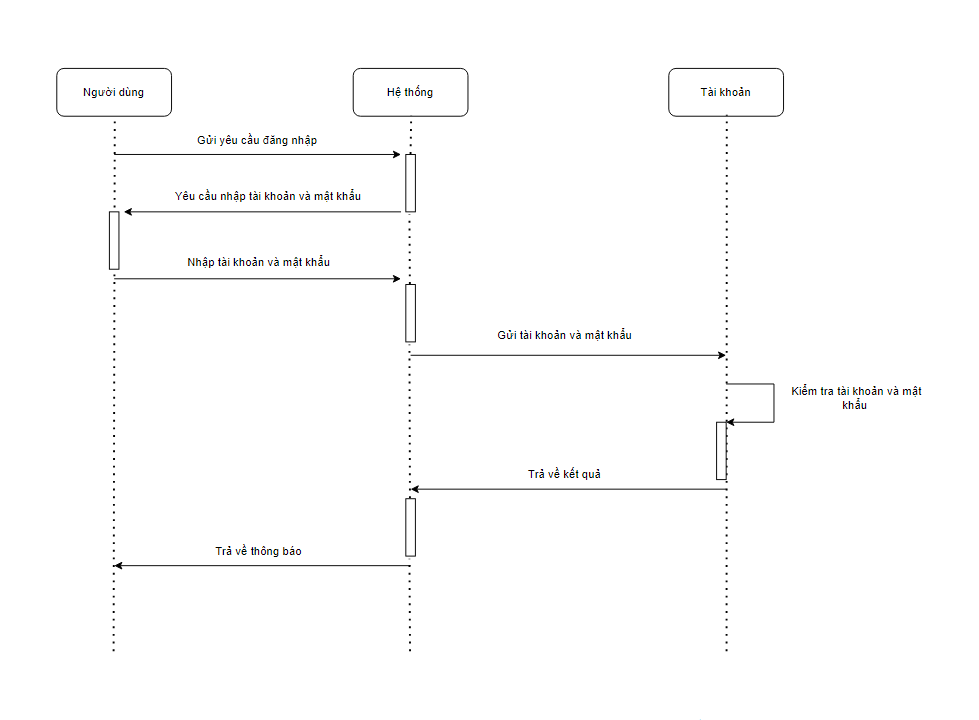
### 2.5.6 Quản lý đọc truyện



Hình 2.16 : Sơ đồ hoạt động quản lý đọc truyện

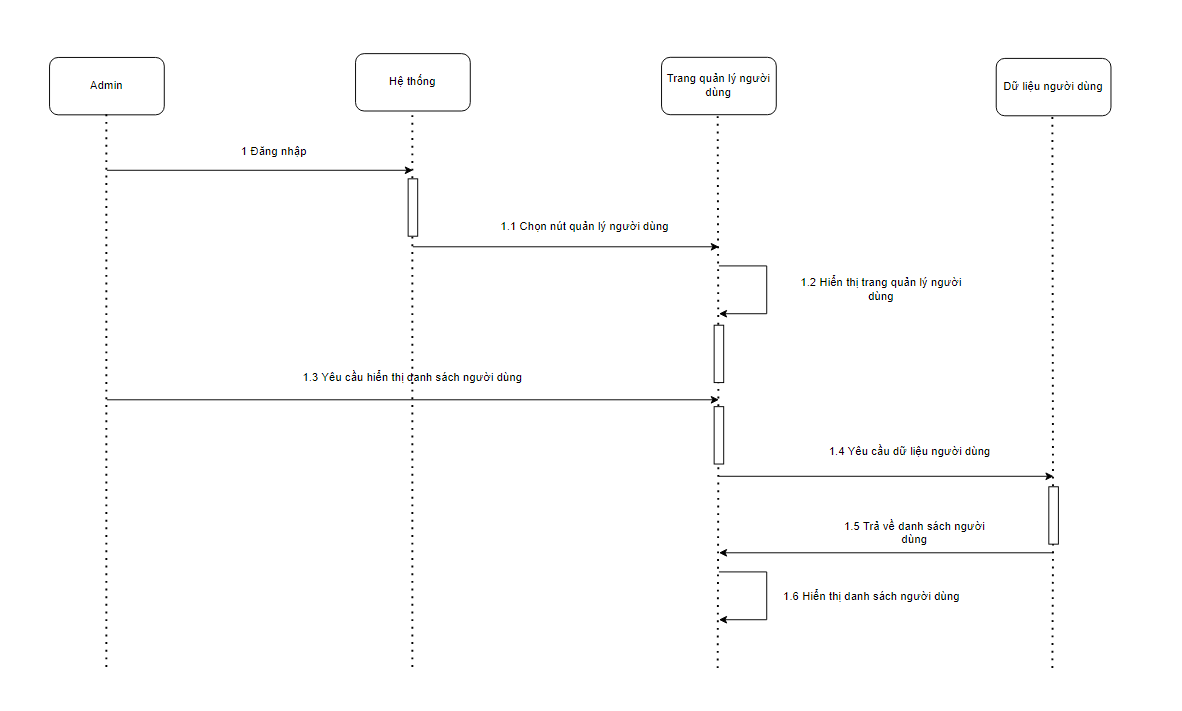
## 2.6 Thiết kế sơ đồ tuần tự

### 2.6.1 Đăng nhập



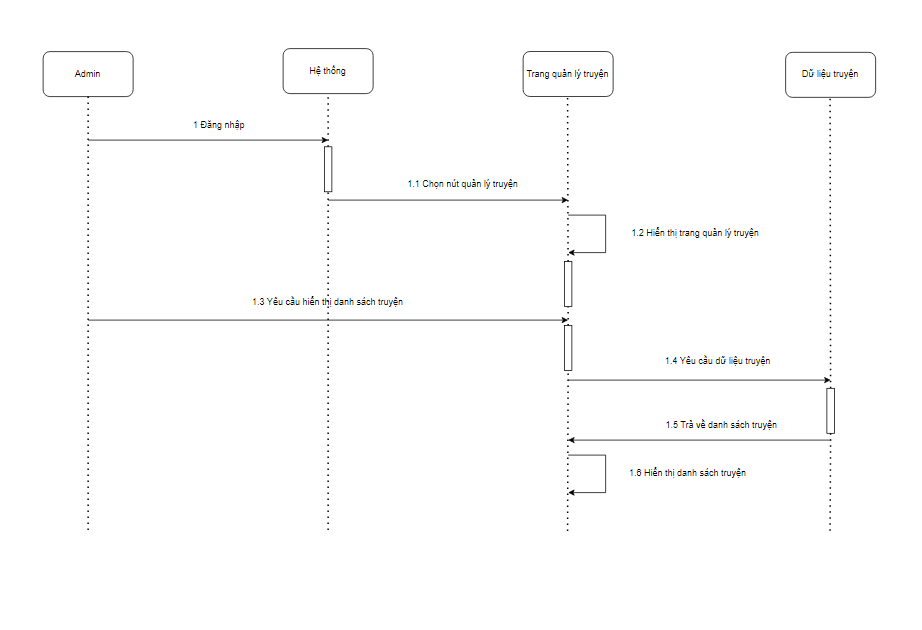
Hình 2.17 : Sơ đồ tuần tự đăng nhập

### 2.6.2 Quản lý người dùng



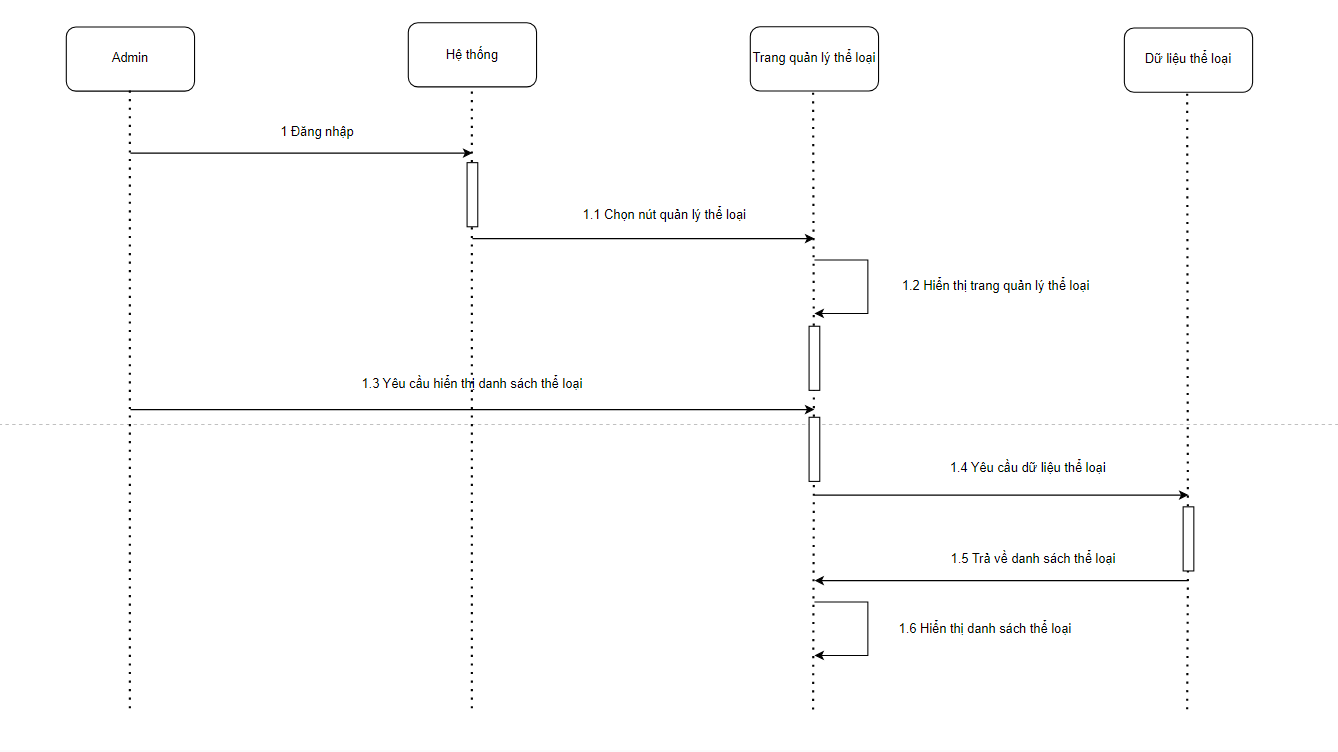
Hình 2.18 : Sơ đồ tuần tự xem danh sách người dùng

### 2.6.3 Quản lý thông tin truyện



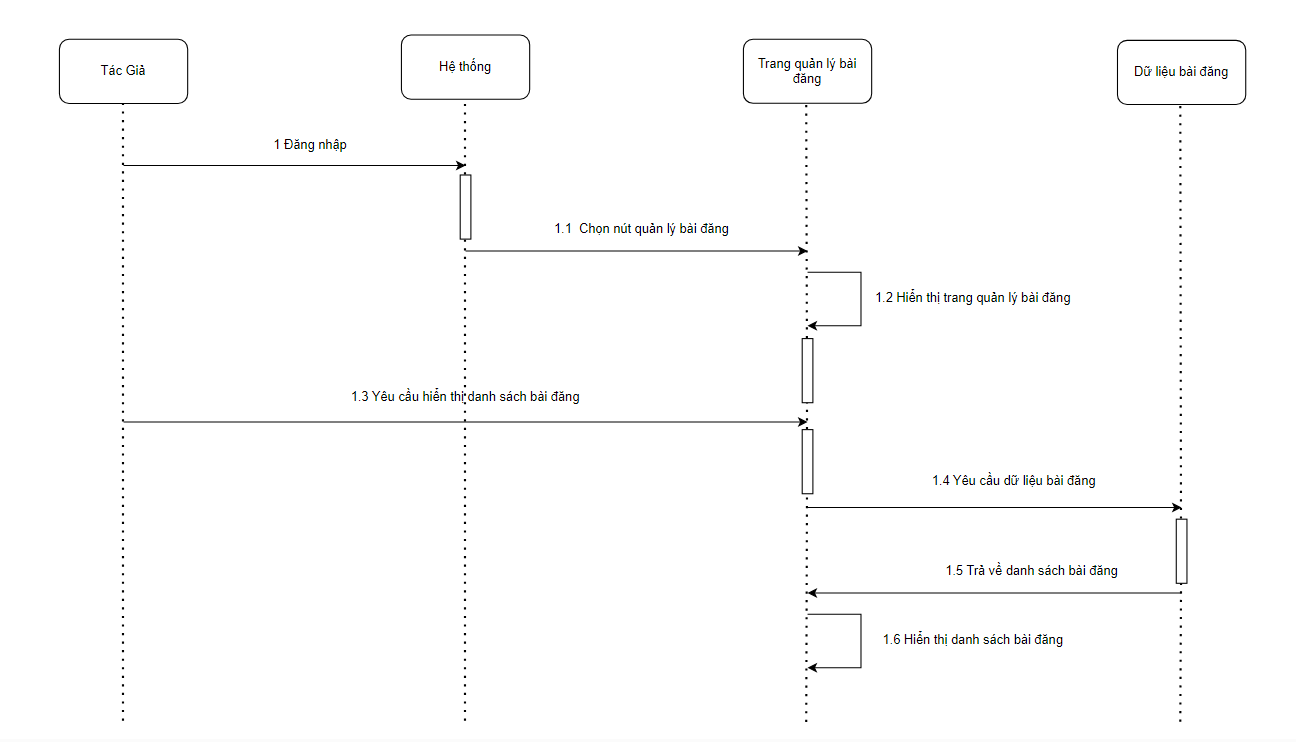
Hình 2.19 : Sơ đồ tuần tự xem danh sách truyện

### 2.6.4 Quản lý thông tin thể loại



Hình 2. 20 : Sơ đồ tuần tự xem danh sách thể loại

### 2.6.5 Quản lý thông tin bài đăng



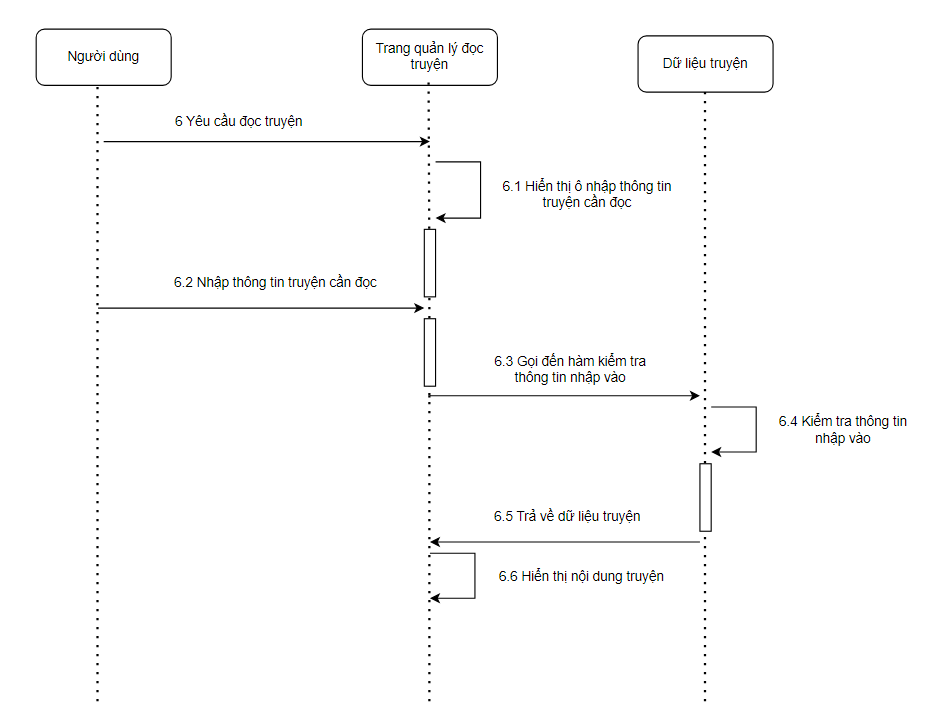
Hình 2.21 : Sơ đồ tuần tự xem danh sách bài đăng

### 2.6.6 Quản lý thông tin đánh giá



Hình 2.22 : Sơ đồ tuần tự xem danh sách đánh giá

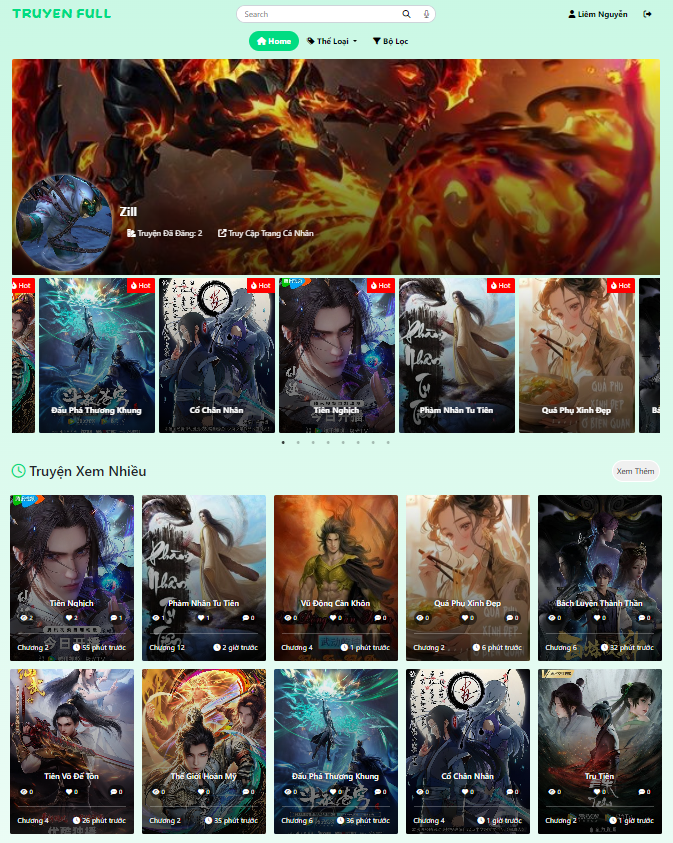
### 2.6.7 Quản lý đọc truyện



Hình 2.23 : Sơ đồ tuần tự đọc truyện

# CHƯƠNG 3. TRIỂN KHAI ỨNG DỤNG

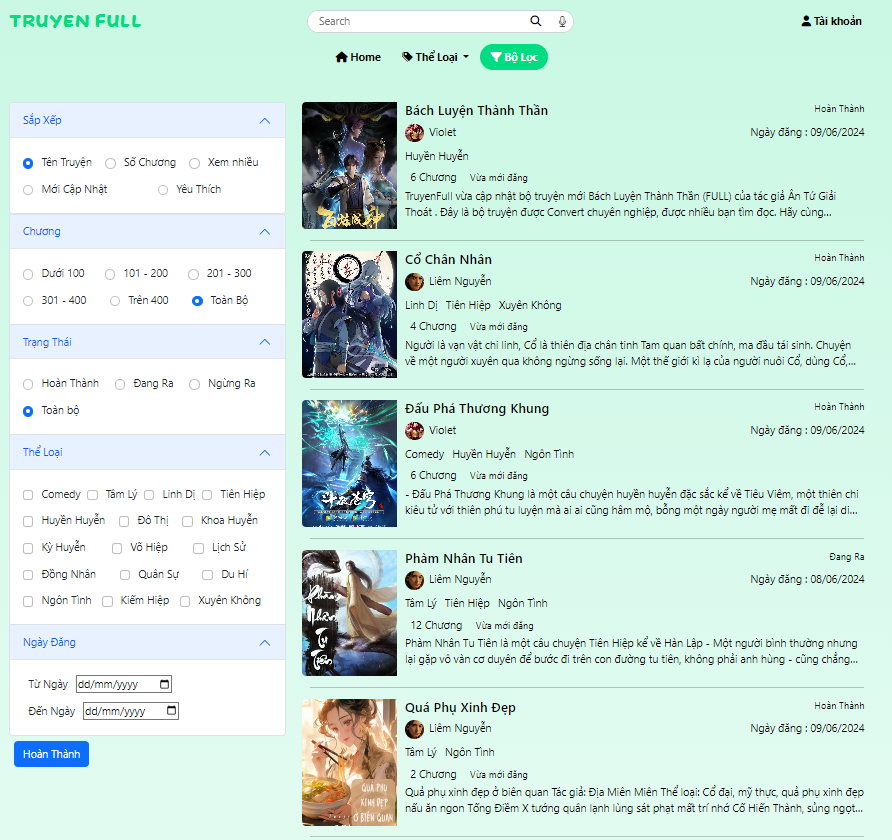
**3.1 Giao diện trang chủ**



Hình 3.1 : Giao diện trang chủ

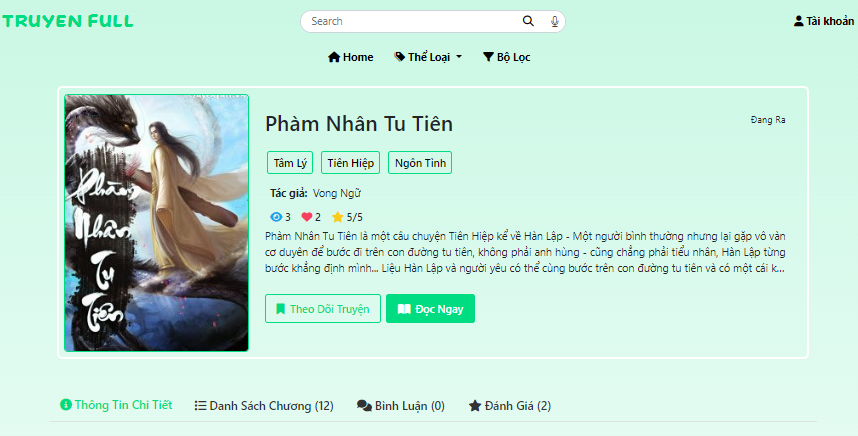
**3.2 Giao diện trang chi tiết**

**3.2.1 Giao diện trang bộ lọc**

****

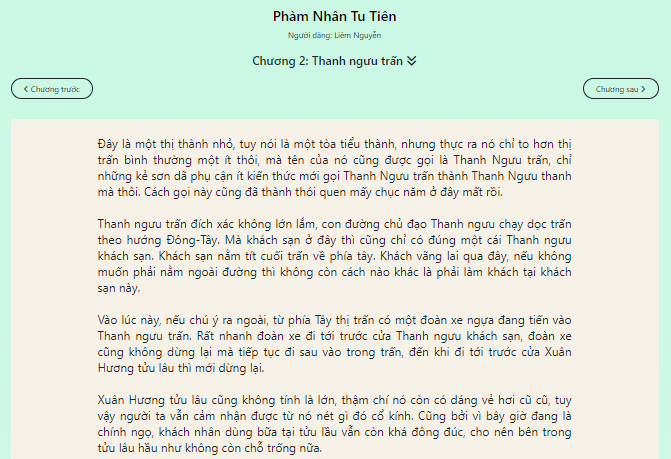
Hình 3.2 : Trang bộ lọc

**3.2.2 Giao diện trang xem thông tin truyện**

****

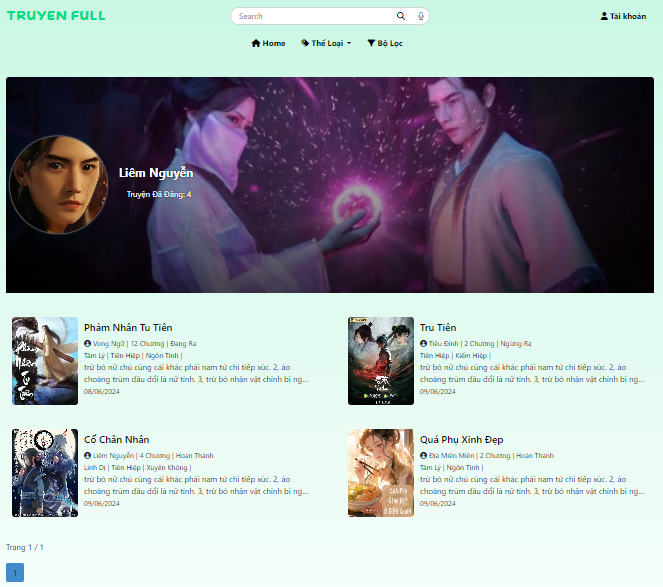
Hình 3.3 : Trang xem chi tiết truyện

**3.2.3 Giao diện trang đọc truyện**

****

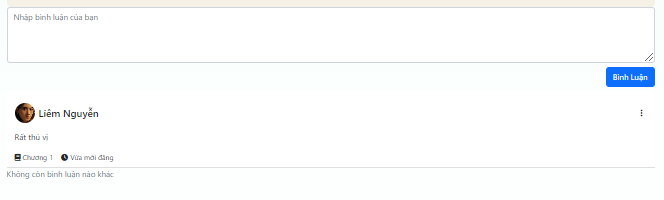
Hình 3.4 : Trang đọc truyện

**3.2.4 Giao diện trang thông tin người dùng**

****

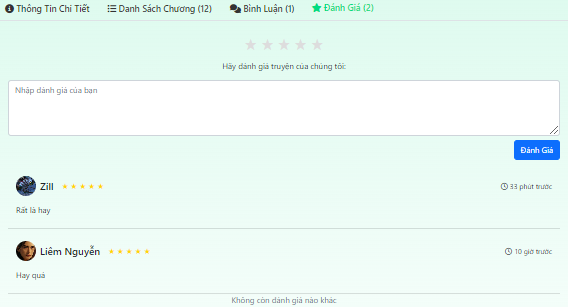
Hình 3.4 : Trang thông tin người dùng

**3.2.5 Giao diện bình luận truyện**

****

Hình 3.5 : Trang bình luận truyện

**3.2.6 Giao diện đánh giá truyện**



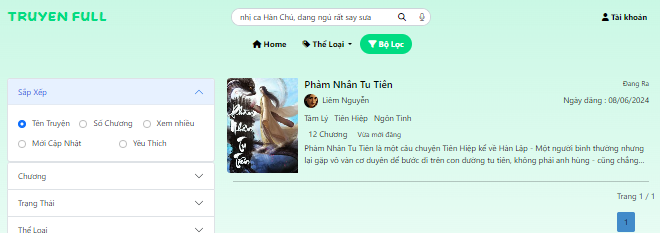
Hình 3.6 : Trang đánh giá truyện

**3.2.7 Giao diện danh sách chương**

****

Hình 3.7 : Trang danh sách chương

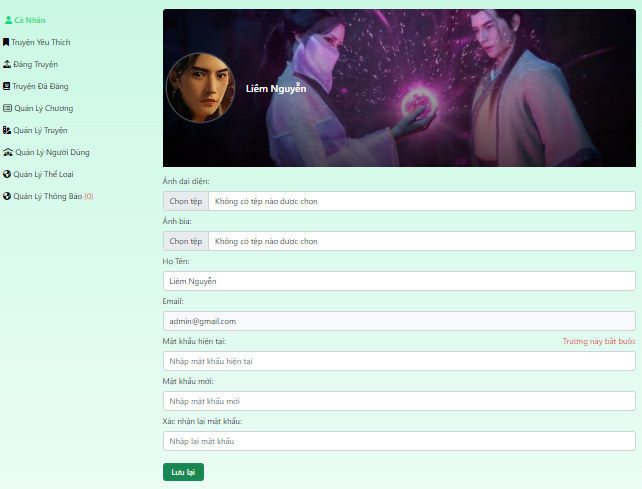
**3.2.8 Giao diện tìm kiếm**



Hình 3.8 : Giao diện tìm kiếm

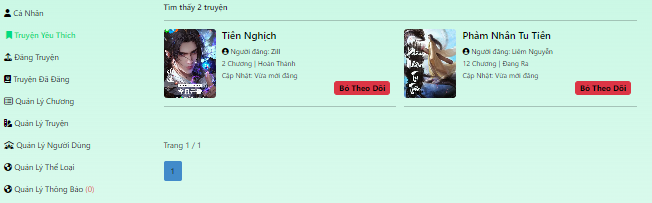
**3.3 Trang Admin**

**3.3.1 Trang cá nhân**

****

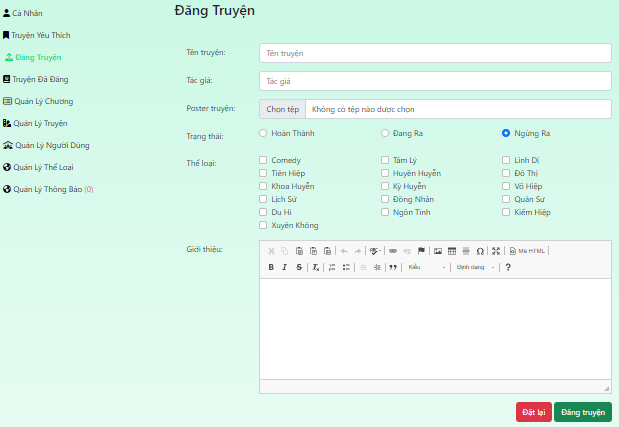
Hình 3.9 : Trang cá nhân Admin

**3.3.2 Trang truyện yêu thích**



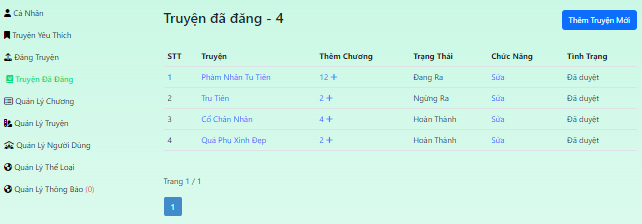
Hình 3.10 : Trang truyện yêu thích

**3.3.3 Trang đăng truyện**



Hình 3.11 : Đăng truyện

**3.3.4 Trang truyện đã đăng**

****

Hình 3.12 : Truyện đã đăng

**3.3.5 Trang quản lý chương**

****

Hình 3.13 : Quản lý chương

**3.3.6 Trang quản lý truyện**

****

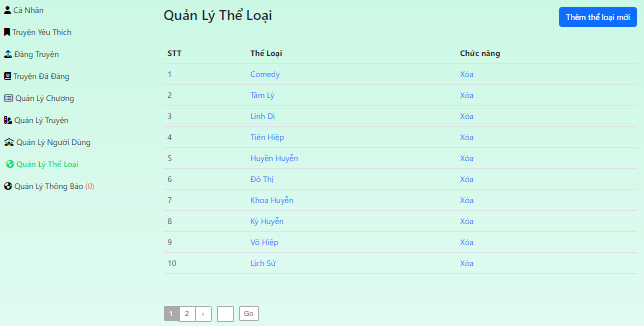
Hình 3.14 : Quản lý truyện

**3.3.7 Trang quản lý người dùng**

****

Hình 3.15 : Quản lý người dùng

**3.3.8 Trang quản lý thể loại**

****

Hình 3.16 : Quản lý thể loại

**3.3.9 Trang quản lý thông báo**

****

Hình 3.17 : Quản lý thông báo

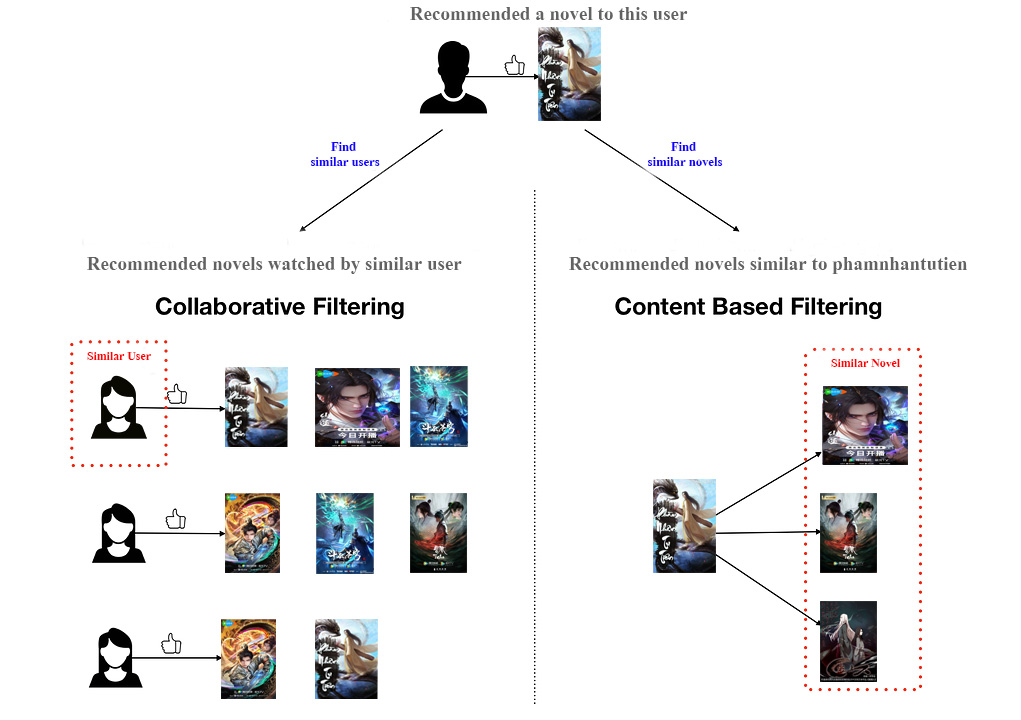
# THUẬT TOÁN RECOMMENDATION SYSTEM

## ****Giới thiệu về hệ thống****

* Recommemdation Systems là 1 trong những mảng tương đối lớn của Machine Learning và có tuổi đời đã khá lớn khi lần đầu tiên được đề cập trong 1 báo cáo kĩ thuật năm 1990 bởi Jussi Karlgren tại Đại Học Columbia và được triển khai quy mô lớn từ năm 1994 trở đi bởi Jussi Karlgren . RS là 1 giải pháp thay thế cho các thuật toán tìm kiếm ví chúng giúp người dùng có thế thấy được các item mà họ có thể thích kể cả họ có tìm tiếm hay không.
* Các thực thể chính trong hệ thống
* Nói đến hệ thống gợi ý thì chúng ta cần phải có ***tối thiểu*** 2 yếu tố sau:
* User : Người dùng là yếu tố không thể thiếu. Không có người dùng thì không thể gợi ý được. Các đặc điểm của người dùng thường được sử dụng trong gợi ý kết bạn hoặc các dịch vụ cá nhân hóa.
* Item :nếu không có 1 item cụ thể nào đó mà chỉ có người dùng thì chỉ có thể gợi ý người dùng khác cho nhau vd là Tinder nhưng trong trường hợp này User lại tính là 1 item trong bài toán gợi ý kết bạn hay kết nối với nhau.
* Rating : Thực ra đây là 1 yếu tố trong 1 vài hệ thống gợi ý sẽ không cần đến nhưng đa số thì đây là 1 yếu tố rất quan trọng để đưa ra gợi ý chính xác nhất cho người dùng.

**Hai nhóm chính của Recommendation Systems**

* Các hệ thống gợi ý (RS) thường được chia làm 2 nhóm chính lớn:
* Content-based systems: Hệ thống sẽ quan tâm đến ***đặc điểm*** của mục tiêu ví dụ như khi bạn đọc truyện Tru tiên, Phàm nhân tu tiên, Tiên nghịch thì hệ thống sẽ phân tích ra các bộ truyện bạn vừa đọc là thuộc thể loại tiên hiệp thì hệ thống sẽ gợi ý cho bạn các bộ truyện tiên hiệp liên quan.
* Collaborative filtering: Nhóm này thì đẳng cấp hơn khi gợi ý các item dựa trên sự tương quan (similarity) giữa các User và(hoặc) Item, hơi khó hiểu nhỉ để dễ hình dùng hơn là nhóm này gợi ý dựa trên ***rating*** của User với Item hay được recommended tới User dự trên những User có hành vi tương tự. Ví dụ: Có 4 anh A,B,C,D đều thích truyện của tác giả Vong Ngữ và là các đạo hữu chân chính luôn. Ngoài ra hệ thống còn có thông tin 2 anh A,B đều ghét truyện của tác giả Tiêu Đỉnh mà không biết 2 anh kia thế nào? Dựa trên phán đoán của hệ thống có thể đề xuất cả 2 anh còn lại đều ghét truyện của Tiêu Đỉnh bằng cách rating 1 sao hoặc dislike truyện và không bao giờ gợi ý truyện của Tiêu Đỉnh cho 2 anh còn lại.



Hình 4.1 : Hai nhóm chính của Recommendation Systems

**Utility matrix**

* Như đã đề cập, có hai thực thể chính trong các Recommendation Systems là *users* và *items*. Mỗi *user* sẽ có *mức độ quan tâm* (*degree of preference*) tới từng *item* khác nhau. Mức độ quan tâm này, *nếu đã biết trước*, được gán cho một giá trị ứng với mỗi cặp *user-item*. Giả sử rằng *mức độ quan tâm* được đo bằng giá trị *user* rate cho *item*, ta tạm gọi giá trị này là *rating*. Tập hợp tất cả các *ratings*, bao gồm cả những giá trị chưa biết cần được dự đoán, tạo nên một ma trận gọi là Utility Matrix.

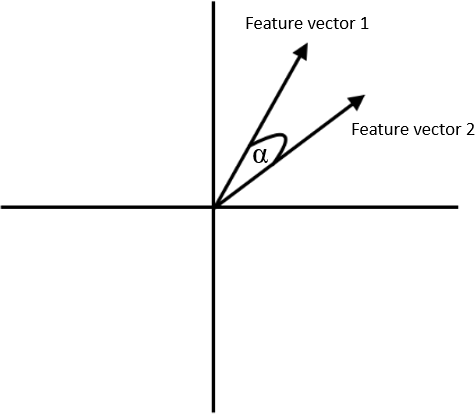


Hình 4.2 : Utility matrix

* Hình trên là ví dụ về utility matrix. Trong đó, *u0*-> *u6* tương ứng với các *users*, *i0* -> *i6* tương ứng với các items. Giá trị từng cặp (u, i) là giá trị rating của user *u* đối với item *i*.
* Xây dựng utility matrix: Không có Utility matrix, gần như không thể gợi ý được sản phẩm tới người dùng, ngoài cách luôn luôn gợi ý các sản phẩm phổ biến nhất. Vì vậy, trong các Recommender Systems, việc xây dựng Utility Matrix là tối quan trọng.

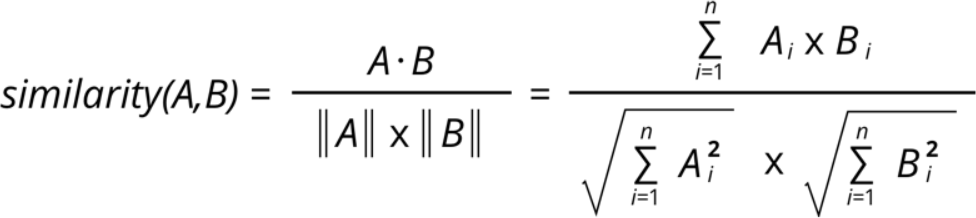
**Cosine Similarity**

* Sau khi biểu diễn các rating thành utility matrix, mỗi hàng(item) là một vector với độ dài là số lượng user. Để tính độ tương tự giữa các user ta sẽ dùng Cosine similarity.



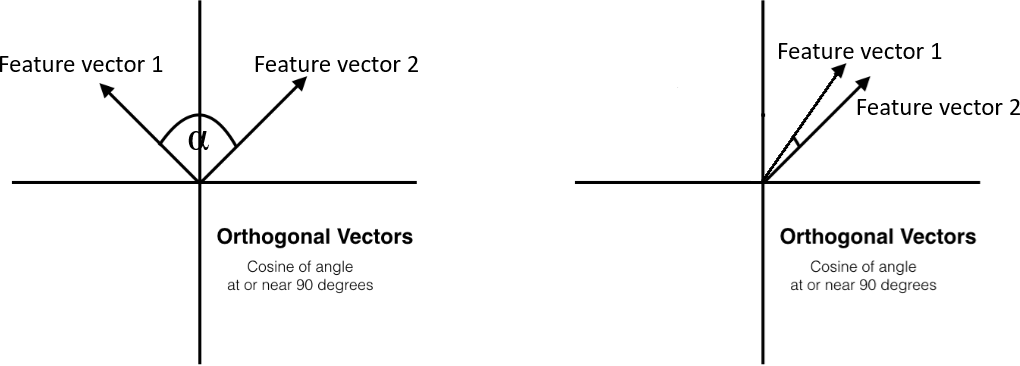
Hình 4.3 : Consine similarity

* Để tính độ tương quan giữa 2 vector, ta thực hiện phép tính tích vô hướng giữa 2 vector, hay nói cách khác là tính giá trị góc cosin được hợp bởi 2 vector, với 0° < α < 180°:



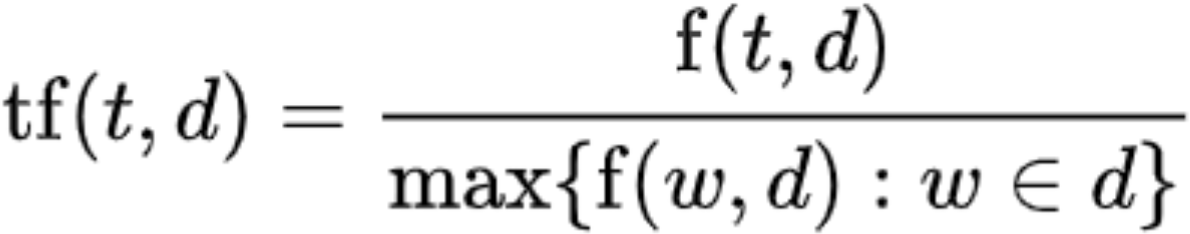
Hình 4.4 : Công thức tích similarity giữa A và B

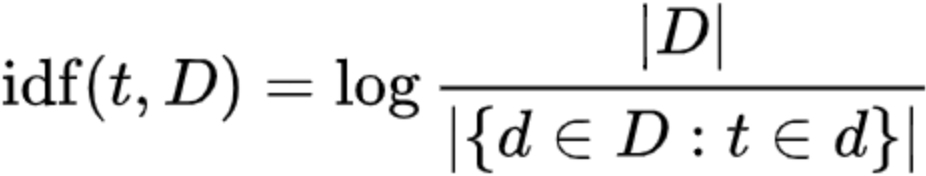
* Nếu tích vô hướng cho ra kết quả càng gần về 0, khi đó 2 vector gần như vuông góc với nhau, đồng nghĩa với việc 2 vector không có sự tương đồng với nhau. Ngược lại, nếu tích vô hướng càng tiến về 1, khi đó 2 vector gần như trùng nhau, nghĩa là sự tương đồng giữa 2 vector rất cao.



Hình 4.5 : Sự tương đồng của 2 vector

**TF-IDF**

* **TF-IDF** (Term Frequency – Inverse Document Frequency) là 1 kĩ thuật sử dụng trong khai phá dữ liệu văn bản. Trọng số này được sử dụng để đánh giá tầm quan trọng của một từ trong một văn bản. Giá trị cao thể hiện độ quan trọng cao và nó phụ thuộc vào số lần từ xuất hiện trong văn bản nhưng bù lại bởi tần suất của từ đó trong tập dữ liệu. Một vài biến thể của TF-IDF thường được sử dụng trong các hệ thống tìm kiếm như một công cụ chính để đánh giá và sắp xếp văn bản dựa vào truy vấn của người dùng. TF-IDF cũng được sử dụng để lọc những từ stopwords trong các bài toán như tóm tắt văn bản và phân loại văn bản.
* **TF**: Term Frequency (Tần suất xuất hiện của từ) là số lần từ xuất hiện trong văn bản. Vì các văn bản có thể có độ dài ngắn khác nhau nên một số từ có thể xuất hiện nhiều lần trong một văn bản dài hơn là một văn bản ngắn. Như vậy, term frequency thường được chia cho độ dài văn bản( tổng số từ trong một văn bản).
* Trong đó:
* tf(t, d): tần suất xuất hiện của từ t trong văn bản d.
* f(t, d): Số lần xuất hiện của từ t trong văn bản d.
* max({f(w, d) : w ∈ d}): Số lần xuất hiện của từ có số lần xuất hiện nhiều nhất trong văn bản d.
* **IDF**: Inverse Document Frequency (Nghịch đảo tần suất của văn bản), giúp đánh giá tầm quan trọng của một từ . Khi tính toán TF , tất cả các từ được coi như có độ quan trọng bằng nhau. Nhưng một số từ như “is”, “of” và “that” thường xuất hiện rất nhiều lần nhưng độ quan trọng là không cao. Như thế cần giảm độ quan trọng của những từ này xuống.



* Trong đó:
* idf(t, D): giá trị IDF của từ t trong tập văn bản.
* |D|: Tổng số văn bản trong tập D.
* |{d ∈ D : t ∈ d}|: thể hiện số văn bản trong tập D có chứa từ t.

**Áp dụng vào bài toán**

* ***D =*** [

“ Trên ngọn núi, có người đàn ông tu luyện từ thuở nhỏ ”

“ Trong hang động, một nhà tu hành tập trung vào việc hoàn thiện linh hồn của mình. ”

“ Ở bờ biển xa xôi, một tu sĩ tìm kiếm bình an và sự giải thoát.”

“ Trên ngọn đồi hoang, một phù thủy sáng tạo nỗ lực để bảo vệ vùng đất của mình.”

“ Trong khu rừng rậm, một cung điện bí ẩn là nơi các tu sĩ rèn luyện trí tuệ và sức mạnh. ”

]

* **xét " Trên ngọn núi, có người đàn ông tu luyện từ thuở nhỏ "**
* ***Tính toán TF (Term Frequency)***

có d=12

với t = trên => TF(t,d) = (số lần từ t xuất hiện trong văn bản d)/(tổng số từ trong văn bản d) = 1/12

với t = ngọn => TF(t,d) = (số lần từ t xuất hiện trong văn bản d)/(tổng số từ trong văn bản d) = 1/12

với t = núi => TF(t,d) = (số lần từ t xuất hiện trong văn bản d)/(tổng số từ trong văn bản d) = 1/12

với t = có => TF(t,d) = (số lần từ t xuất hiện trong văn bản d)/(tổng số từ trong văn bản d) = 1/12

với t = người => TF(t,d) = (số lần từ t xuất hiện trong văn bản d)/(tổng số từ trong văn bản d) = 1/12

với t = đàn => TF(t,d) = (số lần từ t xuất hiện trong văn bản d)/(tổng số từ trong văn bản d) = 1/12

với t = ông => TF(t,d) = (số lần từ t xuất hiện trong văn bản d)/(tổng số từ trong văn bản d) = 1/12

với t = tu => TF(t,d) = (số lần từ t xuất hiện trong văn bản d)/(tổng số từ trong văn bản d) = 1/12

với t = luyện => TF(t,d) = (số lần từ t xuất hiện trong văn bản d)/(tổng số từ trong văn bản d) = 1/12

với t = từ => TF(t,d) = (số lần từ t xuất hiện trong văn bản d)/(tổng số từ trong văn bản d) = 1/12

với t = thuở => TF(t,d) = (số lần từ t xuất hiện trong văn bản d)/(tổng số từ trong văn bản d) = 1/12

với t = nhỏ => TF(t,d) = (số lần từ t xuất hiện trong văn bản d)/(tổng số từ trong văn bản d) = 1/12

* ***Tính toán IDF (Inverse Document Frequency)***

cho N = 10

với t = trên => IDF(t,D) = log(N/số văn bản chứa từ t) = log(10/1)

với t = ngọn => IDF(t,D) = log(N/số văn bản chứa từ t) = log(10/1)

với t = núi => IDF(t,D) = log(N/số văn bản chứa từ t) = log(10/2)

với t = có => IDF(t,D) = log(N/số văn bản chứa từ t) = log(10/2)

với t = người => IDF(t,D) = log(N/số văn bản chứa từ t) = log(10/3)

với t = đàn => IDF(t,D) = log(N/số văn bản chứa từ t) = log(10/1)

với t = ông => IDF(t,D) = log(N/số văn bản chứa từ t) = log(10/1)

với t = tu => IDF(t,D) = log(N/số văn bản chứa từ t) = log(10/2)

với t = luyện => IDF(t,D) = log(N/số văn bản chứa từ t) = log(10/3)

với t = từ => IDF(t,D) = log(N/số văn bản chứa từ t) = log(10/3)

với t = thuở => IDF(t,D) = log(N/số văn bản chứa từ t) = log(10/3)

với t = nhỏ => IDF(t,D) = log(N/số văn bản chứa từ t) = log(10/1)

* ***Tính toán trọng số TF-IDF cho mỗi từ trong từ điển***

với t = trên => TF-IDF(t,d,D)= TF(t,d) x IDF(t,D) = 1/12 x log(10/1)

với t = ngọn => TF-IDF(t,d,D)= TF(t,d) x IDF(t,D) = 1/12 x log(10/1)

với t = núi => TF-IDF(t,d,D)= TF(t,d) x IDF(t,D) = 1/12 x log(10/2)

với t = có => TF-IDF(t,d,D)= TF(t,d) x IDF(t,D) = 1/12 x log(10/2)

với t = người => TF-IDF(t,d,D)= TF(t,d) x IDF(t,D) = 1/12 x log(10/3)

với t = đàn => TF-IDF(t,d,D)= TF(t,d) x IDF(t,D) = 1/12 x log(10/1)

với t = ông => TF-IDF(t,d,D)= TF(t,d) x IDF(t,D) = 1/12 x log(10/1)

với t = tu => TF-IDF(t,d,D)= TF(t,d) x IDF(t,D) = 1/12 x log(10/2)

với t = luyện => TF-IDF(t,d,D)= TF(t,d) x IDF(t,D) = 1/12 x log(10/3)

với t = từ => TF-IDF(t,d,D)= TF(t,d) x IDF(t,D) = 1/12 x log(10/3)

với t = thuở => TF-IDF(t,d,D)= TF(t,d) x IDF(t,D) = 1/12 x log(10/3)

với t = nhỏ => TF-IDF(t,d,D)= TF(t,d) x IDF(t,D) = 1/12 x log(10/1)

# KẾT LUẬN

* **Kết quả đạt được**
* **Tồn tại, hạn chế**
* **Hướng phát triển**

# TÀI LIỆU THAM KHẢO

* **Giáo trình và sách ASP.NET MVC**

1. Pro ASP.NET MVC **5** by Adam Freeman
2. ASP.NET MVC 5 with Bootstrap and Knockout.js by Jamie Munro
3. Professional ASP.NET MVC 5 by Jon Galloway, Brad Wilson, K. Scott Allen, và David Matson
4. Learning ASP.NET Core MVC Programming by Mugilan T. S. Ragupathi
5. Programming ASP.NET MVC 4 by Jess Chadwick, Todd Snyder, Hrusikesh Panda
6. ASP.NET MVC 5 - Building a Website with Visual Studio 2015 and C# của Jonas Fagerberg
7. ASP.NET MVC 5: Building Web Applications by Mahesh Sabnis
8. Beginning ASP.NET MVC 4 by Jose Rolando Guay Paz
9. ASP.NET MVC Framework Unleashed by Stephen Walther
10. ASP.NET MVC 5 with Entity Framework 6 by Lee Naylor
11. Pro ASP.NET Core 3 by Adam Freeman
12. Building Single Page Applications in .NET Core 3 by Michele Aponte
13. Building Web Applications with Visual Studio 2017 by Philip Japikse, Kevin Grossnicklaus, Ben Dewey
14. **Building Microservices with .NET Core 3.1** by Gaurav Aroraa, Lalit Kale, Kanwar Manish

* **Tài liệu và khóa học trực tuyến ASP.NET MVC**

1. **Trang web Microsoft Learn**

* ***Link truy cập:*** *https://learn.microsoft.com/enus/aspnet/core/mvc/overview?view=aspnetcore-6.0*

1. **Trang web w3School**

* ***Link truy cập*:** *https://www.w3schools.com/asp/default.ASP*

1. Codecademy

* ***Link truy cập:*** [*https://www.codecademy.com/learn/learn-asp-net*](https://www.codecademy.com/learn/learn-asp-net)

1. Pluralsight

* ***Link truy cập:*** *https://www.pluralsight.com/paths/aspnet-core-6*
* **Giáo trình và tài liệu về thuật toán Recommendation**

**1. Building Recommendation Engines by Suresh Kumar Gorakala**

**2. Recommender Systems Handbook by Ricci, Rokach, Shapira**

**3. Practical Recommender Systems by Kim Falk**

**4. Machine Learning Yearning by Andrew Ng**