

MỤC LỤC

I. Giới thiệu	2
1. Giới thiệu đê tài	2
2. Giới thiệu tóm tắt chức năng của đê tài	2
II. Xây dựng và phát triển ứng dụng	3
1. Thiết kế cơ sở dữ liệu	3
1.1. Sơ đồ ERD	3
1.2. Mô hình hóa lớp thực thể trong ORM với JPA	3
2. Trang quản trị và cấu trúc source code	12
2.1. Controller dashboard	12
2.2. Controller manage commission	13
2.3. Controller manage discount	14
2.3. Controller manage document	15
2.2. Controller manage notification	15
2.6. Controller manage transaction	16
2.7. Controller manage user	16
3. Trang người dùng và cấu trúc source code	17
3.1. Controller document user	17
3.2. Controller navigation	18
3.3. Controller profile	18
3. Service và cấu trúc source code	19
3.1. Comment Service	19

3.2. Commission Service	20
3.3. Document Service	21
3.3. Downloads Service	22
3.2. Follower Service	23
3.6. Likes Service	24
3.7. Notification Service	24
3.8. Promo Service	25
3.9. Subscription Service	26
3.10. UserServiceImpl	26
3.11. User Service	27
3.12. View history Service	28
2. Repository	28
2.1. Comment Repository	28
2.2. Commission Repository	28
2.3. Document Repository	29
2.3. Download Repository	29
2.2. Follower Repository	29
2.6. Likes Repository	30
2.7. Notification Repository	30
2.8. Promo Repository	30
2.9. Subscription Repository	30
2.10. User Repository	31
2.11. View history Repository	31
6. Html	32

6.1. Admin	32
6.2. User	33
7. Security Config	34
III. Demo chương trình	35
1. Trang quản trị	35
1.1. Trang dashboard	35
1.2. Trang quản lý hoa hồng	35
1.3. Trang quản lý voucher	36
1.3. Trang quản lý tài liệu	36
1.2. Trang quản lý thông báo	37
1.6. Trang quản lý giao dịch	37
1.7. Trang quản lý người dùng	38
2. Trang người dùng	38
IV. Kết luận và hướng phát triển	39
1. Kết quả đạt được	39
2. Hướng phát triển	39

I. Giới thiệu

1. Giới thiệu đề tài

Trong thời đại công nghệ 4.0, việc số hóa và chia sẻ tài liệu đang trở thành xu hướng phổ biến, không chỉ trong môi trường học thuật mà còn trong các lĩnh vực nghiên cứu, kinh doanh và giáo dục. Với sự phát triển nhanh chóng của internet và các nền tảng số, nhu cầu về một hệ thống quản lý và chia sẻ tài liệu điện tử chuyên nghiệp, dễ sử dụng và bảo mật ngày càng trở nên cần thiết.

Đề tài "Xây dựng website tài liệu điện tử" được nhóm em chọn nhằm tạo ra một nền tảng trực tuyến cho phép người dùng dễ dàng tìm kiếm, tải lên, quản lý và chia sẻ các tài liệu dưới dạng số hóa. Hệ thống được thiết kế để đáp ứng nhu cầu của nhiều nhóm người dùng, từ sinh viên, giảng viên đến các nhà nghiên cứu và người dùng phổ thông.

2. Giới thiệu tóm tắt chức năng của đề tài

Đề tài được xây dựng một số chức năng chính như:

- Người dùng:
 - + Đăng nhập
 - + Đăng ký
 - + Xem, tải các tài liệu bình thường
 - + Đăng ký VIP để tải các tài liệu VIP
 - + Nhận hoa hồng từ các lượt tải với tài liệu được gán nhãn VIP(chỉ khi là người dùng VIP)
 - + Like, comment, follow
 - + Sử dụng voucher để giảm giá khi đăng ký VIP
 - + Nhận thông báo từ người quản trị
- Người quản trị:
 - + Đăng nhập
 - + Quản lý sơ bộ thông qua trang dashboard

- + Quản lý tài liệu
- + Quản lý người dùng
- + Quản lý voucher
- + Quản lý hoa hồng
- + Quản lý thông báo

II. Phân tích và thiết kế hệ thống

1. Phân tích chức năng

1.1. Phía khách (Guest)

ST T	Chức năng	Mô tả
1	Đăng ký	Đăng ký tài khoản
2	Xem about us	Xem các thông tin về trang web
3	Xem trang index	Xem trang giới thiệu về trang web

1.2. Phía người dùng (User)

ST T	Chức năng	Mô tả
1	Đăng nhập	Đăng nhập vào hệ thống
2	Đăng xuất	Đăng xuất khỏi hệ thống
3	Xem trang home	Xem tổng quan về các tài liệu đã xem, đề xuất tài liệu, số lượng tài liệu đã đăng, số tiền kiếm được, số người follow trong tháng
4	Xem trang my docs	Xem các tài liệu đã đăng

5	Xem trang Setting	Chỉnh sửa thông tin cá nhân
6	Xem trang Subscription	Đăng ký VIP để download được các tài liệu VIP và hưởng chiết khấu từ các tài liệu mình đã đăng
7	Xem trang Recent	Xem các tài liệu mình đã xem
8	Xem trang Recommend	Xem các tài liệu được đề xuất
9	Tìm kiếm theo đại học	Xem các tài liệu liên quan đến trường đại học
10	Tìm kiếm theo categories	Xem các tài liệu liên quan đến các category
11	Xem trang Setting	Chỉnh sửa thông tin cá nhân, chỉnh dark/light mode, ngôn ngữ

1.3. Phía quản trị (Admin)

ST T	Chức năng	Mô tả
1	Xem trang dashboard	Quản lý khái quát về hệ thống như số lượng user, số lượng VIP, số người online, số lượng document, hôm nay

		upload được bao nhiêu, doanh thu theo ngày, tháng, năm
2	Xem trang quản lý commission	Xem các chiết khấu của các người dùng VIP, có thể thanh toán cho từng người dùng hay thanh toán cho tất cả, xem tổng chiết khấu, tổng chi.
3	Xem trang quản lý documents	Xem thông tin chi tiết về tất cả các tài liệu được đăng trên website, có thể chỉnh sửa hay xóa các tài liệu và xem các categories hiện có
4	Xem trang quản lý notifications	Xem tất cả các thông báo đã được đăng, có thể xóa các thông báo đã đăng hay thêm các thông báo mới
5	Xem trang quản lý transaction	Xem các thông tin giao dịch của người dùng và thống kê tổng người dùng VIP, tổng số tiền thu được
6	Xem trang quản lý discount	Xem các voucher đã được phát hành, có thể

		thêm, chỉnh sửa, xóa các voucher
7	Xem trang quản lý người dùng	Xem thông tin các người dùng, bao gồm việc thêm và chỉnh sửa người dùng.

2. Biểu đồ Use case

2.1. Đăng ký (Sign up)

Name	Sign up
Goal	Đăng ký tạo tài khoản người dùng
Actors	Khách hàng (Guest)
Pre-conditions	N/A
Post-conditions	<ul style="list-style-type: none"> - Nếu thành công, sẽ có tài khoản mới được tạo - Nếu thất bại, hiển thị thông báo thất bại
Main Flow	<ol style="list-style-type: none"> 1. Vào hệ thống website 2. Chọn nút đăng nhập 3. Chọn chức năng đăng ký 4. Nhập đầy đủ thông tin tài khoản 5. Bấm vào nút đăng ký 6. Nếu thành công chuyển sang trang đăng nhập
Alternative	N/A
Exception	<ol style="list-style-type: none"> 6a. Thông tin tài khoản không hợp lệ email đã tồn tại 6b. Thông báo đăng ký thất bại

2.2. Đăng nhập (Sign in)

Name	Sign in
Goal	Đăng nhập vào hệ thống

Actors	Người dùng (User), Quản trị viên (Admin)
Pre-conditions	Đã có tài khoản trong hệ thống
Post-conditions	<ul style="list-style-type: none"> - Nếu thành công, hệ thống sẽ dẫn đến trang tương ứng với vai trò người dùng - Nếu thất bại, hiển thị thông báo thất bại
Main Flow	<ol style="list-style-type: none"> 1. Vào hệ thống website 2. Chọn nút Sign in 3. Nhập email và mật khẩu 4. Bấm vào nút Sign in 5. Đăng nhập thành công 6. Hiển thị giao diện dành cho Người mua, hoặc chuyển tới trang quản lý đối với Admin
Alternative	N/A
Exception	<p>5a. Email hoặc điện thoại không hợp lệ, mật khẩu không trùng khớp</p> <p>5a1. Đăng nhập thất bại</p> <p>6a. Hiển thị thông báo đăng nhập thất bại</p>

2.3. Đăng xuất (Sign out)

Name	Sign out
Goal	Đăng xuất khỏi hệ thống
Actors	Người dùng(User), Quản trị viên (Admin)
Pre-conditions	Đã đăng nhập thành công vào hệ thống
Post-conditions	<ul style="list-style-type: none"> - Nếu thành công, đăng xuất khỏi hệ thống, trở về giao diện dành cho Khách (Guest) - Nếu thất bại, thông báo lỗi
Main Flow	<ol style="list-style-type: none"> 1. Chọn biểu tượng Account, hiển thị dropdown 2. Chọn nút Sign out

	3. Đăng xuất thành công, hiển thị giao diện dành cho Khách (Guest)
Alternative	N/A
Exception	N/A

2.3. Tìm kiếm/ xem tài liệu

Name	Search/View documents
Goal	Tìm kiếm, xem danh sách, chi tiết documents
Actors	Người dùng (User)
Pre-conditions	N/A
Post-conditions	<ul style="list-style-type: none"> - Nếu thành công, hiển thị danh sách sản phẩm theo từ khóa tìm kiếm (keyword) hoặc theo loại sản phẩm (category) và bộ lọc (filter) - Nếu thất bại, thông báo lỗi
Main Flow	<ol style="list-style-type: none"> 1. Vào hệ thống website 2. Đăng nhập 3. Chọn chức năng tìm kiếm theo sản phẩm trên thanh tìm kiếm 4. Nhập từ khóa 5. Nhấn Enter 6. Hiển thị danh sách tài liệu theo từ khóa 7. Có thể bấm view ở các tài liệu để xem chi tiết tài liệu
Alternative	N/A
Exception	N/A

2.2. Upload tài liệu

Name	Upload document
Goal	Tải các tài liệu lên trang web
Actors	Người dùng (User), quản trị viên (Admin)

Pre-conditions	N/A
Post-conditions	<ul style="list-style-type: none"> - Nếu thành công, tài liệu sẽ được upload lên trang web - Nếu thất bại, thông báo lỗi
Main Flow	<ol style="list-style-type: none"> 1. Vào hệ thống website 2. Đăng nhập 3. Chọn chức năng upload 4. Chọn tệp để tải lên 5. Chọn thumbnail 6. Điền title document 7. Điền hashtag 8. Chọn trường đại học 9. Án nút upload document
Alternative	N/A
Exception	<p>4a. Nếu tải tệp khác PDF hay Word thì sẽ xuất hiện thông báo lỗi</p> <p>9a. Nếu có lỗi trong quá trình upload thì sẽ thông báo lỗi</p>

2.6. Chính sửa thông tin cá nhân

Name	Edit profile
Goal	Thay đổi thông tin người dùng
Actors	Người dùng (User)
Pre-conditions	N/A
Post-conditions	<ul style="list-style-type: none"> - Nếu thành công, thông tin người dùng sẽ được thay đổi - Nếu thất bại, thông báo lỗi
Main Flow	<ol style="list-style-type: none"> 1. Vào hệ thống website 2. Đăng nhập 3. Chọn chức năng My profile 4. Án nút Edit profile 5. Chọn hình ảnh để thay đổi avatar

	<p>6. Điền thông tin username, email, Bio</p> <p>7. Án nút Save Changes</p>
Alternative	N/A
Exception	<p>5a. Nếu hình ảnh có vấn đề thì sẽ thông báo lỗi</p> <p>7a. Nếu có vấn đề trong quá trình thay đổi sẽ thông báo lỗi</p>

2.7. Tìm kiếm tài liệu theo đại học hoặc category

Name	Find by university or categories
Goal	Tìm kiếm tài liệu theo trường đại học hoặc categories
Actors	Người dùng (User)
Pre-conditions	N/A
Post-conditions	<ul style="list-style-type: none"> - Nếu thành công, hiển thị danh sách tài liệu theo trường đại học hay categories - Nếu thất bại, thông báo lỗi
Main Flow	<ol style="list-style-type: none"> 1. Vào hệ thống website 2. Đăng nhập 3. Chọn chức năng tìm kiếm tài liệu theo University hay Categories trên thanh nav 4. Nhập tên vào thanh tìm kiếm 5. Nhấn Enter 6. Hiển thị danh sách tài liệu theo tìm kiếm 7. Có thể bấm view ở các tài liệu để xem chi tiết tài liệu
Alternative	N/A
Exception	N/A

2.8. Quản lý tài khoản

Name	Account Management
Goal	Quản lý các tài khoản của người dùng
Actors	Người quản trị (Admin)

Pre-conditions	N/A
Post-conditions	N/A
Main Flow	<ol style="list-style-type: none"> 1. Vào hệ thống website 2. Đăng nhập với tài khoản của admin 3. Chọn chức năng Users trên thanh nav 4. Thấy được toàn bộ thông tin của các người dùng 5. Án nút edit để chỉnh sửa, delete để xóa hoặc add để thêm user
Alternative	N/A
Exception	N/A

2.9. Bình luận, đánh giá

Name	Comment/Like
Goal	Bình luận và like các tài liệu
Actors	Người dùng (User), người quản trị (Admin)
Pre-conditions	N/A
Post-conditions	<ul style="list-style-type: none"> - Nếu thành công, bình luận và đánh giá sẽ được lưu lại - Nếu thất bại, thông báo lỗi
Main Flow	<ol style="list-style-type: none"> 1. Vào hệ thống website 2. Đăng nhập 3. Bấm nút view ở tài liệu muốn bình luận và đánh giá 4. Bấm nút like 5. Điền nội dung bình luận vào ô comment 6. Án submit
Alternative	N/A
Exception	N/A

2.10. Quản lý các tài liệu

Name	Documents Management
------	----------------------

Goal	Quản lý tài liệu trên hệ thống website
Actors	Người quản trị (Admin)
Pre-conditions	N/A
Post-conditions	<ul style="list-style-type: none"> - Nếu thành công, hiển thị trang quản lý các tài liệu - Nếu thất bại, thông báo lỗi
Main Flow	<ol style="list-style-type: none"> 1. Vào hệ thống website 2. Đăng nhập với tài khoản của admin 3. Chọn nút document trên thanh nav 4. Hiển thị trang quản lý các thông tin của các tài liệu 5. Có thể chỉnh sửa hoặc xóa các tài liệu 6. Nhấn Enter 7. Hiển thị danh sách tài liệu theo từ khóa 8. Có thể bấm view ở các tài liệu để xem chi tiết tài liệu
Alternative	N/A
Exception	N/A

2.11. Quản lý các thông báo

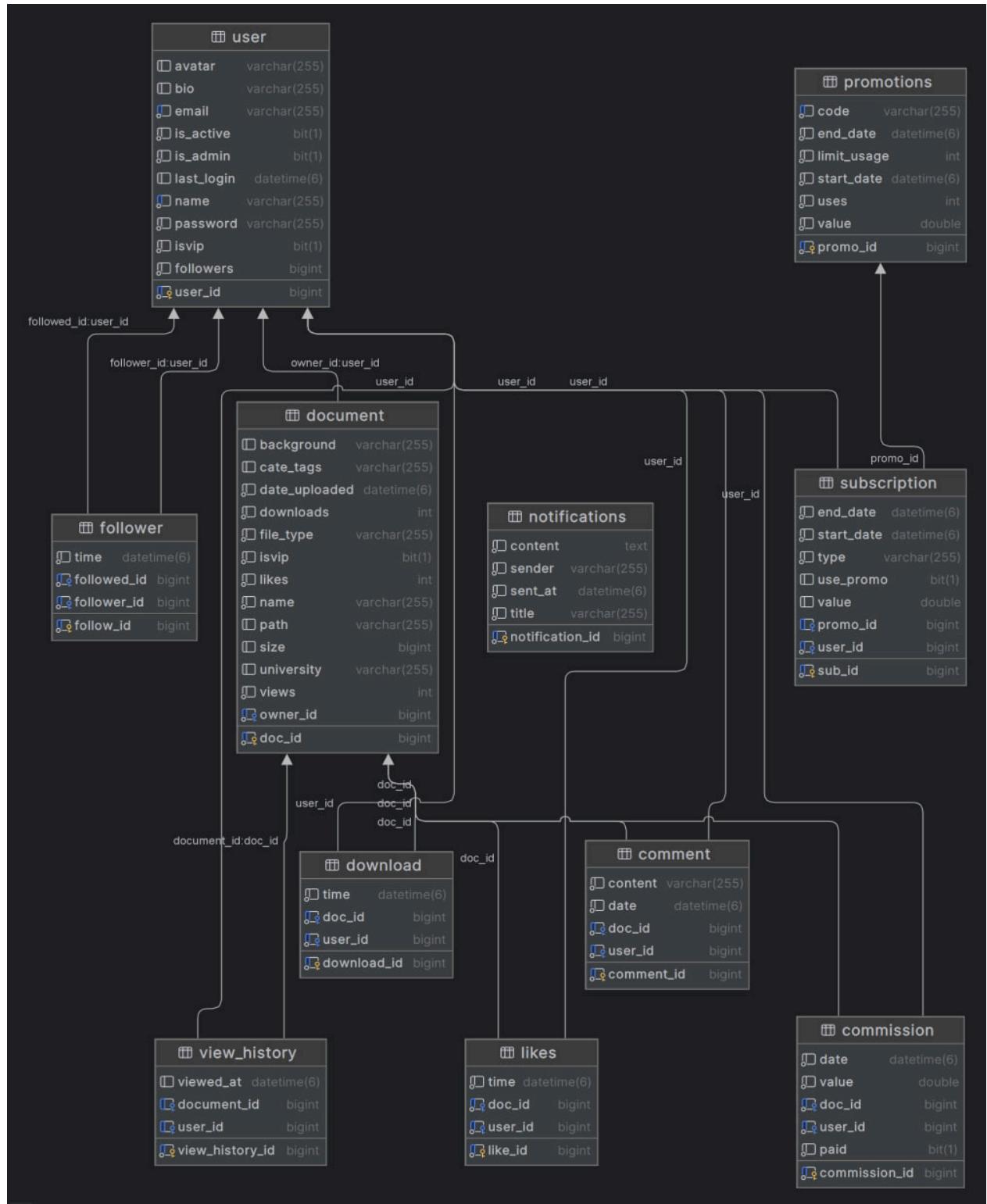
Name	Notifications Management
Goal	Quản lý các thông báo của hệ thống
Actors	Người quản trị (Admin)
Pre-conditions	N/A
Post-conditions	<ul style="list-style-type: none"> - Nếu thành công, hiển thị danh sách sản phẩm theo từ khóa tìm kiếm (keyword) hoặc theo loại sản phẩm (category) và bộ lọc (filter) - Nếu thất bại, thông báo lỗi
Main Flow	<ol style="list-style-type: none"> 1. Vào hệ thống website 2. Đăng nhập 8. Chọn chức năng tìm kiếm theo sản phẩm trên thanh tìm kiếm

	<p>9. Nhập từ khóa</p> <p>10.Nhấn Enter</p> <p>11.Hiển thị danh sách tài liệu theo từ khóa</p> <p>12.Có thể bấm view ở các tài liệu để xem chi tiết tài liệu</p>
Alternative	N/A
Exception	N/A

III. Xây dựng và phát triển ứng dụng

1. Thiết kế cơ sở dữ liệu

1.1. Sơ đồ ERD



Hình 1: Sơ đồ ERD

Sơ đồ ERD (Entity Relationship Diagram) thể hiện các thực thể chính trong hệ thống và mối quan hệ giữa chúng. Ví dụ: thực thể 'User' liên kết với 'Document' qua mối quan hệ 'Upload', 'Likes', và 'Comments'. Điều này giúp xác định rõ vai trò và mối liên kết giữa các bảng trong cơ sở dữ liệu.

1.2. Mô hình hóa lớp thực thể trong ORM với JPA

a) Bảng user

```
public class User {  
  
    @Id  
    @GeneratedValue(strategy = GenerationType.IDENTITY)  
    private Long userId;  
  
    @Column(nullable = false, unique = true)  
    private String name;  
  
    @Column(nullable = false, unique = true)  
    private String email;  
  
    @Column(nullable = false)  
    private String password;  
  
    private String avatar;  
    private String bio;  
    private boolean isAdmin;  
    private boolean isActive;  
    private boolean isVIP;  
  
    @Column(name = "last_login")  
    private LocalDateTime lastLogin;  
}
```

Hình 2: Entity User

Bảng này lưu trữ thông tin về người dùng, bao gồm: ID, tên, email, vai trò (User, Admin), và trạng thái VIP. Thông tin này cần thiết cho việc xác thực và phân quyền người dùng.

b) Bảng document

```

public class Document {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long docId;

    @Column(nullable = false)
    private String fileType;

    @Column(nullable = false)
    private String name;

    private Long size;
    private String path;

    @Column(name = "date_uploaded", nullable = false)
    private LocalDateTime dateUploaded;

    private int likes;
    private int downloads;
    private boolean isVIP;
    private String background;
    private String university;
    private int views;

    @Column(name = "cate_tags")
    private String cateTags;

    @ManyToOne
    @JoinColumn(name = "owner_id", nullable = false)
    private User owner;
}

```

Hình 3: Entity Document

Lớp `Document` đại diện cho một tài liệu trong hệ thống, được ánh xạ tới cơ sở dữ liệu bằng JPA. Lớp này chứa các thuộc tính quan trọng như `fileType`, `name`, `size`, `path`, và `dateUploaded` để quản lý thông tin cơ bản và trạng thái của tài liệu. Ngoài ra, nó hỗ trợ các trường thống kê như `likes`, `downloads`, `views` cùng với thuộc tính phân loại như `cateTags` và `university`. Quan hệ nhiều-tới-một với `User` cho phép gắn tài liệu với người

sở hữu. Đây là thành phần cốt lõi giúp hệ thống quản lý và tìm kiếm tài liệu hiệu quả.

c) Bảng comment

```
public class Comment {  
  
    @Id  
    @GeneratedValue(strategy = GenerationType.IDENTITY)  
    private Long commentId;  
  
    @Column(nullable = false)  
    private String content;  
  
    @Column(nullable = false)  
    private LocalDateTime date;  
  
    @ManyToOne  
    @JoinColumn(name = "user_id", nullable = false)  
    private User user;  
  
    @ManyToOne  
    @JoinColumn(name = "doc_id", nullable = false)  
    private Document document;  
}
```

Hình 4: Entity Comment

Lớp `Comment` đại diện cho một bình luận trong hệ thống, được ánh xạ với cơ sở dữ liệu bằng JPA. Lớp này bao gồm các thuộc tính:

- `commentId`: Khóa chính, tự động tăng, dùng để định danh duy nhất từng bình luận.
- `content`: Nội dung của bình luận, không được để trống.
- `date`: Thời gian tạo bình luận, lưu dưới dạng `LocalDateTime`, không được để trống.

- `user`: Quan hệ nhiều-tới-một với lớp `User`, biểu thị người đã tạo bình luận, được ánh xạ thông qua cột `user_id`.
- `document`: Quan hệ nhiều-tới-một với lớp `Document`, biểu thị tài liệu mà bình luận này liên quan, được ánh xạ thông qua cột `doc_id`.

Lớp này đảm bảo quản lý thông tin bình luận một cách đầy đủ, đồng thời hỗ trợ mối quan hệ giữa người dùng, tài liệu và bình luận để nâng cao khả năng quản lý nội dung trên hệ thống.

d) Bảng Commission

```
public class Commission {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long commissionId;

    @Column(nullable = false)
    private Double value;

    @Column(nullable = false)
    private Boolean paid;// Amount already paid (default to 0)

    @Column(nullable = false)
    private LocalDateTime date;

    @ManyToOne
    @JoinColumn(name = "user_id", nullable = false)
    private User user;

    @ManyToOne
    @JoinColumn(name = "doc_id", nullable = false)
    private Document document;
}
```

Hình 5: Entity Commission

Lớp `Commission` đại diện cho thông tin hoa hồng trong hệ thống, bao gồm các thuộc tính `commissionId` (ID), `value` (giá trị hoa hồng), `paid` (trạng thái đã thanh toán), và `date` (ngày tạo). Quan hệ nhiều-tới-một với `User` và `Document` giúp xác định người nhận hoa hồng và tài liệu liên quan, hỗ trợ quản lý minh bạch các giao dịch trong hệ thống.

e) Bảng download

```
public class Download {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long downloadId;

    @Column(nullable = false)
    private LocalDateTime time;
    @ManyToOne
    @JoinColumn(name = "user_id", nullable = false)
    private User user;

    @ManyToOne
    @JoinColumn(name = "doc_id", nullable = false)
    private Document document;
}
```

Hình 6: Entity Download

Lớp `Download` đại diện cho một lượt tải tài liệu trong hệ thống. Lớp này bao gồm các thuộc tính chính: `downloadId` (ID của lượt tải), `time` (thời gian tải). Quan hệ nhiều-tới-một với `User` và `Document` giúp xác định người dùng thực hiện lượt tải và tài liệu được tải. Lớp này hỗ trợ theo dõi và thống kê hoạt động tải tài liệu một cách hiệu quả.

f) Bảng follower

```

public class Follower {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long followId;

    @Column(nullable = false)
    private LocalDateTime time;

    @ManyToOne
    @JoinColumn(name = "follower_id", nullable = false)
    private User follower;

    @ManyToOne
    @JoinColumn(name = "followed_id", nullable = false)
    private User followed;
}

```

Hình 7: Entity Follower

Lớp Follower đại diện cho mối quan hệ theo dõi giữa hai người dùng trong hệ thống. Lớp này bao gồm các thuộc tính: followId (ID của mối quan hệ), time (thời gian bắt đầu theo dõi). Quan hệ nhiều-tới-một với follower (người theo dõi) và followed (người được theo dõi) giúp quản lý mối liên kết giữa các người dùng một cách rõ ràng và hiệu quả.

g) Bảng likes

```

public class Likes {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long likeId;

    @ManyToOne
    @JoinColumn(name = "user_id", nullable = false)
    private User user;

    @ManyToOne
    @JoinColumn(name = "doc_id", nullable = false)
    private Document document;

    @Column(nullable = false)
    private LocalDateTime time;
}

```

Hình 8: Likes Entity

Lớp Likes đại diện cho lượt thích của người dùng đối với tài liệu trong hệ thống. Lớp bao gồm các thuộc tính: likeId (ID của lượt thích), time (thời gian thực hiện lượt thích). Quan hệ nhiều-tới-một với User (người thích) và Document (tài liệu được thích) giúp theo dõi và quản lý dữ liệu về lượt thích một cách rõ ràng và chi tiết.

h) Bảng notification

```
public class Notification {  
    @Id  
    @GeneratedValue(strategy = GenerationType.IDENTITY)  
    private Long notificationId; // Notification ID  
  
    @Column(nullable = false)  
    private String title; // Title of the notification  
  
    @Column(nullable = false, columnDefinition = "TEXT")  
    private String content; // Content of the notification  
  
    @Column(nullable = false)  
    private String sender; // Sender of the notification  
  
    @Column(nullable = false)  
    private LocalDateTime sentAt; // Timestamp when the notification was sent
```

Hình 9: Notification Entity

Lớp Notification đại diện cho thông báo trong hệ thống. Các thuộc tính chính bao gồm:

- **notificationId**: ID duy nhất của thông báo.
- **title**: Tiêu đề của thông báo, không được để trống.
- **content**: Nội dung chi tiết của thông báo, định nghĩa kiểu TEXT để lưu trữ nội dung dài.
- **sender**: Người gửi thông báo.

- **sentAt**: Thời gian gửi thông báo, được lưu dưới dạng LocalDateTime.

Lớp này cho phép quản lý và gửi thông báo đến người dùng một cách hiệu quả và rõ ràng.

i) Bảng promo (voucher)

```
public class Promo {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long promoid;

    @Column(nullable = false)
    private LocalDateTime startDate;

    @Column(nullable = false)
    private LocalDateTime endDate;

    @Column(nullable = false)
    private Double value;

    private int uses;
    private int limitUsage;

    @Column(nullable = false, unique = true)
    private String code;
}
```

Hình 10: Promo Entity

Lớp Promo đại diện cho mã khuyến mãi trong hệ thống, với các thuộc tính chính:

- **promoid**: ID duy nhất của mã khuyến mãi.
- **startDate** và **endDate**: Thời gian bắt đầu và kết thúc hiệu lực của mã.
- **value**: Giá trị khuyến mãi (ví dụ: giảm giá phần trăm hoặc số tiền cố định).

- **uses**: Số lần mã đã được sử dụng.
- **limitUsage**: Giới hạn số lần mã có thể được sử dụng.
- **code**: Mã khuyến mãi, duy nhất và không được để trống.

Lớp này giúp quản lý các chương trình khuyến mãi hiệu quả, đảm bảo tính năng kiểm soát và sử dụng mã khuyến mãi trong hệ thống.

j) Bảng subscription

```
public class Subscription {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long subId;

    @ManyToOne
    @JoinColumn(name = "user_id", nullable = false)
    private User user;

    @Column(nullable = false)
    private LocalDateTime startDate;

    @Column(nullable = false)
    private LocalDateTime endDate;

    @Column(nullable = false)
    private String type;

    private Double value;

    @Column(name = "use_promo")
    private boolean usePromo;

    @ManyToOne
    @JoinColumn(name = "promo_id")
    private Promo promo;
}
```

Hình 11: Subscription Entity

Lớp Subscription đại diện cho thông tin gói đăng ký của người dùng trong hệ thống, với các thuộc tính chính:

- **subId**: ID duy nhất của gói đăng ký.
- **user**: Quan hệ nhiều-tới-một với lớp User, xác định người dùng sở hữu gói đăng ký.
- **startDate** và **endDate**: Ngày bắt đầu và kết thúc hiệu lực của gói đăng ký.
- **type**: Loại gói đăng ký (ví dụ: 1 tháng, 1 năm).
- **value**: Giá trị của gói đăng ký.
- **usePromo**: Biến boolean cho biết liệu có áp dụng mã khuyến mãi khi đăng ký hay không.
- **promo**: Quan hệ nhiều-tới-một với lớp Promo, liên kết mã khuyến mãi được sử dụng (nếu có).

Lớp này hỗ trợ quản lý chi tiết gói đăng ký của người dùng, bao gồm thời hạn, loại, giá trị, và các ưu đãi áp dụng.

2. Xây dựng các repositories

2.1. Comment Repository

```
@Repository 3 usages ▾ Winz18
public interface CommentRepository extends JpaRepository<Comment, Long> {
    List<Comment> findByDocument(Document document); 1 usage ▾ Winz18
    long countByDocument(Document document); 1 usage ▾ Winz18

}
```

Hình 34: Comment Repository

CommentRepository

CommentRepository quản lý các truy vấn liên quan đến bình luận của tài liệu. Repository cung cấp các phương thức:

- Tìm danh sách bình luận theo tài liệu (findByDocument).
- Đếm số lượng bình luận cho một tài liệu (countByDocument).

Repository này hỗ trợ việc theo dõi và xử lý các bình luận, giúp quản lý nội dung tương tác hiệu quả trong hệ thống.

2.2. Commission Repository

```
@Repository 7 usages ▾ Winz18+1*
public interface CommissionRepository extends JpaRepository<Commission, Long> {
    List<Commission> findByUser(User user); 1 usage ▾ Winz18
    List<Commission> findByDocument(Document document); 1 usage ▾ Winz18
    List<Commission> findByUserAndDateBetween(User user, LocalDate start, LocalDate end); 1 usage ▾ Winz18

    // Lấy danh sách hoa hồng theo khoảng thời gian
    @Query("SELECT SUM(c.value) FROM Commission c WHERE c.user = :user AND c.date BETWEEN :startDate AND :endDate") 1 usage ▾ Winz18
    Double sumValueByUserAndDateBetween(
        @Param("user") User user,
        @Param("startDate") LocalDate startDate,
        @Param("endDate") LocalDate endDate
    );

    List<Commission> findByPaid(boolean b); 1 usage ▾ truongvip1
    @Query("SELECT COALESCE(SUM(c.value), 0) FROM Commission c WHERE c.date = CURRENT_DATE") 1 usage ▾ truongvip1
    long getDailyRevenue();

    @Query("SELECT SUM(c.value) FROM Commission c WHERE MONTH(c.date) = MONTH(CURRENT_DATE) AND YEAR(c.date) = YEAR(CURRENT_DATE)") 1 usage ▾ truongvip1
    long getMonthlyRevenue();

    @Query("SELECT SUM(c.value) FROM Commission c WHERE YEAR(c.date) = YEAR(CURRENT_DATE)") 1 usage ▾ truongvip1
    long getYearlyRevenue();
}
```

Hình 35: Commission Repository

CommissionRepository

CommissionRepository cung cấp các truy vấn quản lý hoa hồng, bao gồm:

- Lấy danh sách hoa hồng theo người dùng hoặc tài liệu (findByUser, findByDocument).
- Tính tổng giá trị hoa hồng trong khoảng thời gian cụ thể (sumValueByUserAndDateBetween).
- Thống kê doanh thu theo ngày, tháng, năm (getDailyRevenue, getMonthlyRevenue, getYearlyRevenue).

Repository này đóng vai trò quan trọng trong việc tính toán và theo dõi thu nhập của người dùng VIP từ tài liệu của họ.

2.3. Document Repository

```
@Repository 9 usages  ↳ Winz18 +1
public interface DocumentRepository extends JpaRepository<Document, Long> {
    List<Document> findByUniversity(String university); 1 usage  ↳ Winz18
    List<Document> findByCateTagsContaining(String tag); // Tìm tài liệu theo hashtag 1 usage  ↳ Winz18
    List<Document> findByOwner(User owner); // Lấy tài liệu của user 1 usage  ↳ Winz18
    List<Document> findByIsVIP(boolean isVIP); // Lấy tài liệu chỉ dành cho VIP 2 usages  ↳ Winz18
    List<Document> findByNameContaining(String keyword); // Tìm kiếm tài liệu theo tên 1 usage  ↳ Winz18
    List<Document> findByOrderByDateUploadedDesc(); // Tài liệu mới nhất no usages  ↳ Winz18
    List<Document> findByOrderByViewsDesc(); // Tài liệu nhiều lượt xem nhất no usages  ↳ Winz18

    List<Document> findByOrderByDownloadsDesc(); // Tài liệu được tải xuống nhiều nhất no usages  ↳ Winz18
    @Query("SELECT d.cateTags FROM Document d") 1 usage  ↳ truongvip1
    List<String> findAllTags();
    @Query("SELECT COUNT(d) FROM Document d WHERE DATE(d.dateUploaded) = CURRENT_DATE") 1 usage  ↳ truongvip1
    long countDocumentsUploadedToday();

    List<Document> findTop3ByOrderByDownloadsDesc(); // Tài liệu được tải xuống nhiều nhất 1 usage  ↳ Winz18
    List<Document> findTop3ByOrderByViewsDesc(); no usages  ↳ Winz18
}
```

Hình 36: Document Repository

DocumentRepository

DocumentRepository quản lý các tài liệu trong hệ thống, cung cấp các chức năng như:

- Tìm kiếm tài liệu theo trường đại học, hashtag, hoặc tên (findByUniversity, findByCateTagsContaining, findByNameContaining).
- Lấy danh sách tài liệu phổ biến nhất theo lượt tải xuống hoặc lượt xem (findByOrderByDownloadsDesc, findByOrderByViewsDesc).
- Đếm số tài liệu được upload trong ngày (countDocumentsUploadedToday).
- Lấy danh sách hashtag hoặc trường đại học phổ biến (findPopularTags, findPopularUniversities).

Repository này hỗ trợ việc tổ chức, tìm kiếm và phân tích dữ liệu tài liệu hiệu quả.

2.4. Download Repository

```
@Repository 3 usages ▾ Winz18
public interface DownloadRepository extends JpaRepository<Download, Long> {
    List<Download> findByUser(User user); 1 usage ▾ Winz18
    List<Download> findByDocument(Document document); 1 usage ▾ Winz18

    long countByDocument(Document document); // Số lượt tải của một tài liệu 1 usage ▾ Winz18
    long countByUser(User user); // Số lượt tải của một user 1 usage ▾ Winz18
}
```

Hình 37: Download Repository

DownloadRepository

DownloadRepository quản lý thông tin liên quan đến lượt tải tài liệu, với các chức năng:

- Lấy danh sách lượt tải theo người dùng hoặc tài liệu (findByUser, findByDocument).
- Đếm số lượt tải theo tài liệu hoặc người dùng (countByDocument, countByUser).
- Kiểm tra liệu người dùng đã tải tài liệu hay chưa (existsByUserAndDocument).

Repository này hỗ trợ thống kê và theo dõi dữ liệu tải xuống, góp phần vào các phân tích liên quan đến hành vi người dùng.

2.5. Follower Repository

```
@Repository 3 usages ▾ Winz18
public interface FollowerRepository extends JpaRepository<Follower, Long> {

    // Tìm tất cả những người dùng mà user đang theo dõi
    List<Follower> findByFollower(User follower); 2 usages ▾ Winz18

    // Tìm tất cả những người theo dõi user
    List<Follower> findByFollowed(User followed); 1 usage ▾ Winz18

    long countByFollower(User follower); // Số lượng người user đang theo dõi 1 usage ▾ Winz18
    long countByFollowed(User followed); // Số lượng người theo dõi user 1 usage ▾ Winz18

    // Kiểm tra xem user có đang theo dõi người khác không
    boolean existsByFollowerAndFollowed(User follower, User followed); 1 usage ▾ Winz18

}
```

Hình 38: Follower Repository

FollowerRepository

FollowerRepository cung cấp các chức năng quản lý quan hệ theo dõi giữa người dùng:

- Lấy danh sách người theo dõi và được theo dõi (findByFollower, findByFollowed).
- Đếm số lượng người dùng đang theo dõi hoặc được theo dõi (countByFollower, countByFollowed).
- Kiểm tra quan hệ theo dõi giữa hai người dùng (existsByFollowerAndFollowed).

Repository này đảm bảo việc quản lý mạng lưới kết nối người dùng trên nền tảng.

2.6. Likes Repository

```
@Repository
public interface LikesRepository extends JpaRepository<Likes, Long> {
    List<Likes> findByUser(User user);
    List<Likes> findByDocument(Document document);

    long countByDocument(Document document); // Số lượng like của một tài liệu
    boolean existsByUserAndDocument(User user, Document document); // Kiểm tra xem user đã like tài liệu chưa

}
```

Hình 39: Likes Repository

LikesRepository

LikesRepository quản lý các lượt thích tài liệu, bao gồm:

- Lấy danh sách lượt thích theo người dùng hoặc tài liệu (findByUser, findByDocument).
- Đếm số lượt thích của một tài liệu (countByDocument).
- Kiểm tra người dùng đã thích tài liệu hay chưa (existsByUserAndDocument).

Repository này hỗ trợ ghi nhận và phân tích tương tác của người dùng đối với tài liệu.

2.7. Notification Repository

```
@Repository 2 usages  ✎ truongvip1
public interface NotificationRepository extends JpaRepository<Notification, Long> { }
```

Hình 40: Notification Repository

2.8. Promo Repository

```
@Repository 3 usages ▲ Winz18
public interface PromoRepository extends JpaRepository<Promo, Long> {
    Promo findByCode(String code); // Tìm promo bằng mã 2 usages ▲ Winz18
    List<Promo> findByEndDateAfterAndStartDateBefore(LocalDateTime endDate, LocalDateTime startDate); // Lấy các mã khuyến mãi đang áp dụng
}
```

Hình 41: Promo Repository

PromoRepository

PromoRepository quản lý các mã khuyến mãi, với các chức năng:

- Tìm mã khuyến mãi theo code (findByCode).
- Lấy danh sách mã khuyến mãi đang áp dụng (findByEndDateAfterAndStartDateBefore).

Repository này giúp quản lý và triển khai các chương trình khuyến mãi hiệu quả

2.9. Subscription Repository

```
@Repository 5 usages ▲ Winz18
public interface SubscriptionRepository extends JpaRepository<Subscription, Long> {
    Subscription findByUser(User user); // Lấy gói subscription của user 2 usages ▲ Winz18
    List<Subscription> findByEndDateAfter(LocalDateTime now); // Tìm các user đang trong gói VIP 1 usage ▲ Winz18
}
```

Hình 42: Subscription Repository

SubscriptionRepository

SubscriptionRepository quản lý gói đăng ký của người dùng, cung cấp các chức năng:

- Tìm gói đăng ký theo người dùng (findByUser).
- Lấy danh sách gói VIP còn hiệu lực (findByEndDateAfter).

Repository này hỗ trợ quản lý chi tiết thông tin đăng ký VIP của người dùng.

2.10. User Repository

```
@Repository 16 usages ▾ Winz18
public interface UserRepository extends JpaRepository<User, Long> {
    User findByEmail(String email); // Tìm user qua email 3 usages ▾ Winz18
    List<User> findByNameContaining(String keyword); // Tìm user theo tên 1 usage ▾ Winz18
    long countByIsActiveTrue(); // Đếm số user đang hoạt động 1 usage ▾ Winz18
    long countByIsVIPTrue(); // Đếm số user VIP 1 usage ▾ Winz18
    long countByIsActive(boolean b); 1 usage ▾ Winz18

    // Lấy danh thu trong tháng của người dùng hiện tại
    @Query("SELECT SUM(c.value) FROM Commission c WHERE c.user.userId = :userId " + 1 usage ▾ Winz18
           "AND c.date BETWEEN :startDate AND :endDate")
    Double sumCommissionValue(@Param("userId") Long userId,
                             @Param("startDate") LocalDateTime startDate,
                             @Param("endDate") LocalDateTime endDate);

    // Đếm số lượng follower của người dùng
    @Query("SELECT COUNT(f) FROM Follower f WHERE f.followed.userId = :userId") 1 usage ▾ Winz18
    Long countFollowers(@Param("userId") Long userId);

    // Đếm số lượng follower mới của người dùng trong tháng
    @Query("SELECT COUNT(f) FROM Follower f WHERE f.followed.userId = :userId " + 1 usage ▾ Winz18
           "AND f.time BETWEEN :startOfMonth AND :endOfMonth")
    long countNewFollowers(Long userId, LocalDateTime startOfMonth, LocalDateTime endOfMonth);

    // Đếm số lượng tài liệu đã upload trong tháng
    @Query("SELECT COUNT(d) FROM Document d WHERE d.owner.userId = :userId " + 1 usage ▾ Winz18
           "AND d.dateUploaded BETWEEN :startOfMonth AND :endOfMonth")
    long countDocumentUploaded(Long userId, LocalDateTime startOfMonth, LocalDateTime endOfMonth);

    // Đếm số lượng tài liệu mà người dùng đã upload
    @Query("SELECT COUNT(d) FROM Document d WHERE d.owner.userId = :userId") no usages ▾ Winz18
    long countByDocumentsUploaded(Long userId);
}
```

Hình 43: User Repository

UserRepository

UserRepository cung cấp các truy vấn quản lý thông tin người dùng:

- Tìm người dùng qua email hoặc tên (findByEmail, findByNameContaining).
- Đếm số người dùng đang hoạt động hoặc VIP (countByIsActiveTrue, countByIsVIPTrue).
- Thống kê thu nhập, lượt theo dõi, và tài liệu đã upload của người dùng.

Repository này là trung tâm quản lý người dùng, hỗ trợ các chức năng xác thực và phân tích hành vi.

2.11. View history Repository

```
public interface ViewHistoryRepository extends JpaRepository<ViewHistory, Long> { 3 usages ± Winz18
    @Query("SELECT vh FROM ViewHistory vh " + 1 usage ± Winz18
        "WHERE vh.user = :user " +
        "AND vh.viewedAt IN (SELECT MAX(vhSub.viewedAt) FROM ViewHistory vhSub WHERE vhSub.user = :user GROUP BY vhSub.document) " +
        "ORDER BY vh.viewedAt DESC")
    List<ViewHistory> findDistinctDocumentsByUserOrderByViewedAtDesc(@Param("user") User user);

    Optional<ViewHistory> findByUserAndDocument(User currentUser, Document doc); 1 usage ± Winz18
}
```

Hình 44: View history Repository

ViewHistoryRepository

ViewHistoryRepository quản lý lịch sử xem tài liệu của người dùng:

- Lấy danh sách các tài liệu đã xem gần đây
(findDistinctDocumentsByUserOrderByViewedAtDesc).
- Kiểm tra lịch sử xem của người dùng đối với tài liệu
(findByUserAndDocument).

Repository này giúp theo dõi hành vi người dùng và tối ưu hóa để xuất tài liệu.

3. Xây dựng các services

3.1. Comment Service

```
@Service 2 usages ▲ Winz18
public class CommentService {

    private final CommentRepository commentRepository; 8 usages

    @Autowired ▲ Winz18
    public CommentService(CommentRepository commentRepository) { this.commentRepository = commentRepository; }

    // Lấy tất cả các bình luận
    > public List<Comment> getAllComments() { return commentRepository.findAll(); }

    // Lấy bình luận theo ID
    > public Optional<Comment> getCommentById(Long commentId) { ... }

    // Lấy danh sách bình luận của một tài liệu
    > public List<Comment> getCommentsByDocument(Document document) { ... }

    // Đếm số bình luận của một tài liệu
    > public long countCommentsByDocument(Document document) { ... }

    // Thêm bình luận mới
    > public Comment addComment(User user, Document document, String content) { ... }

    // Xóa bình luận theo ID
    > public void deleteCommentById(Long commentId) { ... }

    // Kiểm tra tính hợp lệ của một bình luận
    > private void validateComment(Comment comment) { ... }
}
```

Hình 22: Comment Service

CommentService

CommentService xử lý các chức năng liên quan đến quản lý bình luận:

- Cung cấp các phương thức lấy danh sách bình luận, tìm kiếm theo ID hoặc tài liệu, và đếm số lượng bình luận của một tài liệu.
- Cho phép thêm mới bình luận với các thông tin như người dùng, tài liệu, và nội dung.
- Hỗ trợ xóa bình luận theo ID và kiểm tra tính hợp lệ của bình luận trước khi lưu.

3.2. Commission Service

```
@Service 2 usages ▲ Winz18
public class CommissionService {

    private final CommissionRepository commissionRepository; 10 usages

    @Autowired ▲ Winz18
    public CommissionService(CommissionRepository commissionRepository) {...}

    // Lấy tất cả hoa hồng
    public List<Commission> getAllCommissions() { return commissionRepository.findAll(); }

    // Tìm hoa hồng theo ID
    public Optional<Commission> getCommissionById(Long id) {...}

    // Lấy danh sách hoa hồng theo User
    public List<Commission> getCommissionsByUser(User user) {...}

    // Lấy danh sách hoa hồng theo Document
    public List<Commission> getCommissionsByDocument(Document document) {...}

    // Lấy danh sách hoa hồng theo User trong một khoảng thời gian
    public List<Commission> getCommissionsByUserAndDateRange(User user, LocalDate start, LocalDate end) {...}

    // Tính tổng giá trị hoa hồng của User trong một khoảng thời gian
    public double getTotalCommissionValueByUserAndDateRange(User user, LocalDate start, LocalDate end) {...}

    // Tạo hoặc cập nhật hoa hồng
    public Commission saveOrUpdateCommission(Commission commission) {...}

    // Xóa hoa hồng theo ID
    public void deleteCommissionById(Long id) {...}

    // Kiểm tra tính hợp lệ của Commission
    private void validateCommission(Commission commission) {...}
}
```

Hình 23: Commission Service

CommissionService

CommissionService quản lý thông tin hoa hồng trong hệ thống:

- Cung cấp danh sách hoa hồng theo người dùng, tài liệu, hoặc khoảng thời gian cụ thể.
- Hỗ trợ tính tổng giá trị hoa hồng của người dùng trong một khoảng thời gian.
- Cho phép thêm mới hoặc cập nhật hoa hồng, đồng thời kiểm tra dữ liệu trước khi lưu.

- Hỗ trợ xóa hoa hồng theo ID và tìm kiếm hoa hồng dựa trên các tiêu chí như tài liệu hoặc người dùng.

3.3. Document Service

```

@Service 9 usages + Winz18 +
public class DocumentService {

    private final DocumentRepository documentRepository; 21 usages
    private final UserRepository userRepository; 3 usages

    @Autowired + Winz18
    public DocumentService(DocumentRepository documentRepository, UserRepository userRepository) {...}

    // Lấy tất cả tài liệu
    > public List<Document> getAllDocuments() { return documentRepository.findAll(); }

    // Tìm tài liệu theo ID
    > public Optional<Document> getDocumentById(Long id) {...}

    // Lấy tài liệu theo owner
    > public List<Document> getDocumentsByOwner(User owner) {...}

    // Tìm tài liệu theo tên
    > public List<Document> searchDocumentsByName(String keyword) {...}

    // Lấy tài liệu theo trường đại học
    > public List<Document> getDocumentsByUniversity(String university) {...}

    // Lấy tài liệu theo danh mục hashtag
    > public List<Document> getDocumentsByCategoryTag(String tag) {...}

    > public List<Document> getVIPDocuments() {...}

    > public List<Document> getNonVIPDocuments() {...}

    // Tạo hoặc cập nhật tài liệu
    > public Document saveOrUpdateDocument(Document document) {...}

    // Xóa tài liệu theo ID
    > public void deleteDocumentById(Long id) {...}

    // Cập nhật lượt xem tài liệu
    > public void incrementViews(Long documentId) {...}

```

Hình 24: Document Service

DocumentService

Quản lý các tài liệu trong hệ thống, với các chức năng:

- Lấy danh sách tài liệu theo tiêu chí như tên, trường đại học, hoặc danh mục hashtag.

- Quản lý tài liệu VIP, lượt xem, lượt tải, và lượt thích.
- Hỗ trợ tìm kiếm, tạo mới, cập nhật, và xóa tài liệu.
- Cung cấp các dữ liệu thống kê như danh mục và trường đại học phổ biến.

3.3. Downloads Service

```
@Service 9 usages ▾ Winz18
public class DownloadService {

    private final DownloadRepository downloadRepository; 10 usages

    @Autowired ▾ Winz18
    public DownloadService(DownloadRepository downloadRepository) { this.downloadRepository = downloadRepository; }

    // Lấy tất cả các lượt tải xuống
    > public List<Download> getAllDownloads() { return downloadRepository.findAll(); }

    // Tìm lượt tải xuống bằng ID
    > public Optional<Download> getDownloadById(Long downloadId) { ... }

    // Lấy danh sách lượt tải của một user
    > public List<Download> getDownloadsByUser(User user) { ... }

    // Lấy danh sách lượt tải của một tài liệu
    > public List<Download> getDownloadsByDocument(Document document) { ... }

    // Đếm số lượt tải của một tài liệu
    > public long countDownloadsByDocument(Document document) { ... }

    // Đếm số lượt tải của một user
    > public long countDownloadsByUser(User user) { ... }

    // Thêm một lượt tải xuống
    > public Download addDownload(User user, Document document) { ... }

    // Xóa lượt tải xuống theo ID
    > public void deleteDownloadById(Long downloadId) { ... }

    // Kiểm tra tính hợp lệ của một lượt tải xuống
    > private void validateDownload(Download download) { ... }
}
```

Hình 25: Downloads Service

DownloadService

Theo dõi và quản lý các lượt tải tài liệu:

- Lấy danh sách lượt tải theo người dùng hoặc tài liệu.
- Thông kê số lượt tải cho mỗi tài liệu hoặc người dùng.

- Hỗ trợ kiểm tra và thêm mới lượt tải xuống, đảm bảo dữ liệu chính xác và hợp lệ.

3.2. Follower Service

```

@Service 7 usages ▲ Winz18
public class FollowerService {

    private final FollowerRepository followerRepository; 11 usages

    @Autowired ▲ Winz18
    public FollowerService(FollowerRepository followerRepository) {this.followerRepository = followerRepository; }

    // Lấy tất cả các lượt theo dõi
    public List<Follower> getAllFollowers() { return followerRepository.findAll(); }

    // Tìm lượt theo dõi bằng ID
    public Optional<Follower> getFollowerById(Long followId) {...}

    // Lấy danh sách người dùng mà user đang theo dõi
    public List<Follower> getFollowings(User user) {...}

    // Lấy danh sách người đang theo dõi user
    public List<Follower> getFollowers(User user) {...}

    // Đếm số người dùng mà user đang theo dõi
    public long countFollowings(User user) {...}

    // Đếm số người đang theo dõi user
    public long countFollowers(User user) {...}

    // Kiểm tra xem user có theo dõi người khác không
    public boolean isFollowing(User follower, User followed) {...}

    // Thêm lượt theo dõi
    public Follower addFollower(User follower, User followed) {...}

    // Hủy theo dõi
    public void removeFollower(User follower, User followed) {...}
}

```

Hình 26: Follower Service

FollowerService

Quản lý quan hệ theo dõi giữa người dùng:

- Lấy danh sách người dùng mà user theo dõi và người đang theo dõi user.
- Đếm số lượng người theo dõi hoặc đang được theo dõi.
- Thêm hoặc hủy bỏ quan hệ theo dõi với các kiểm tra hợp lệ.

3.6. Likes Service

```
@Service 9 usages ▾ Winz18
public class LikesService {

    private final LikesRepository likesRepository; 10 usages

    @Autowired ▾ Winz18
    > public LikesService(LikesRepository likesRepository) { this.likesRepository = likesRepository; }

    // Lấy tất cả các lượt like
    > public List<Likes> getAllLikes() { return likesRepository.findAll(); }

    // Tìm lượt like bằng ID
    > public Optional<Likes> getLikeById(Long likeId) { ... }

    // Lấy danh sách lượt like của một user
    > public List<Likes> getLikesByUser(User user) { ... }

    // Lấy danh sách lượt like của một tài liệu
    > public List<Likes> getLikesByDocument(Document document) { ... }

    // Đếm số lượt like của một tài liệu
    > public long countLikesByDocument(Document document) { ... }

    // Kiểm tra xem một user đã like tài liệu chưa
    > public boolean userHasLikedDocument(User user, Document document) { ... }

    // Thêm một lượt like
    > public Likes addLike(User user, Document document) { ... }

    // Xóa một lượt like
    > public void removeLike(User user, Document document) { ... }
}
```

Hình 27: Likes Service

LikesService

Xử lý các lượt thích tài liệu:

- Lấy danh sách lượt thích theo người dùng hoặc tài liệu.
- Đếm số lượt thích và kiểm tra trạng thái đã thích.
- Thêm mới hoặc xóa lượt thích một cách linh hoạt.

3.7. Notification Service

```
@Service 1usage  ± truongvip1
public class NotificationService {
    @Autowired
    private NotificationRepository notificationRepository;

    /** Get all notifications ...*/
    public List<Notification> getAllNotifications() { return notificationRepository.findAll(); }

    /** Get a notification by its ID ...*/
    public Optional<Notification> getNotificationById(Long id) { return notificationRepository.findById(id); }

    /** Create a new notification ...*/
    public Notification createNotification(Notification notification) { 1usage  ± truongvip1
        notification.setSentAt(LocalDateTime.now()); // Set current timestamp for 'sentAt'
        return notificationRepository.save(notification);
    }

    /** Update an existing notification ...*/
    public Notification updateNotification(Long id, Notification notification) {...}

    /** Delete a notification by its ID ...*/
    public void deleteNotificationById(Long id) {...}
}
```

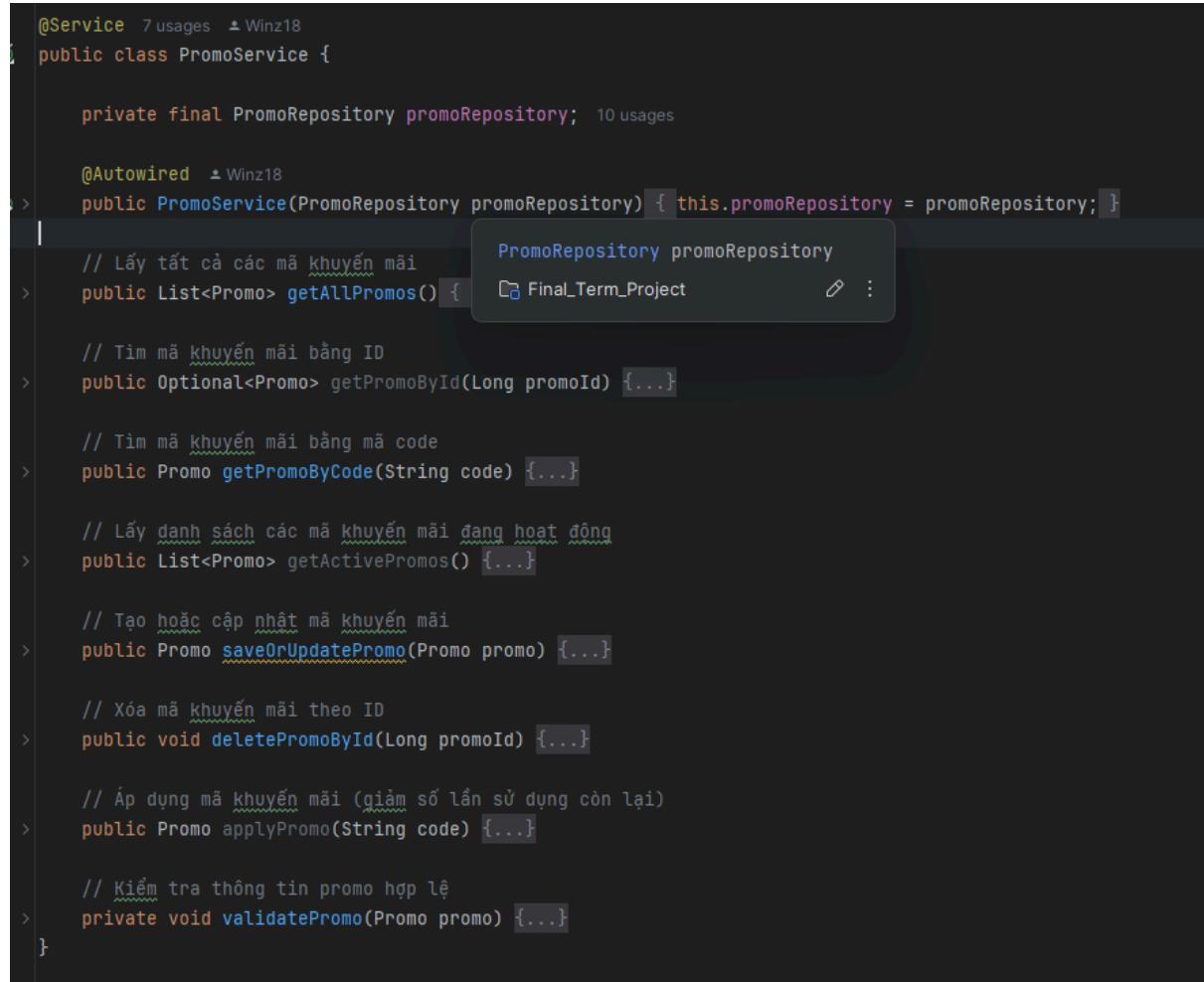
Hình 28: Notification Service

NotificationService

Quản lý thông báo trong hệ thống:

- Tạo mới, cập nhật, hoặc xóa thông báo.
- Cung cấp danh sách thông báo và tìm kiếm theo ID.

3.8. Promo Service



```
@Service 7 usages ▾ Winz18
public class PromoService {

    private final PromoRepository promoRepository; 10 usages

    @Autowired ▾ Winz18
    public PromoService(PromoRepository promoRepository) { this.promoRepository = promoRepository; }

    // Lấy tất cả các mã khuyến mãi
    public List<Promo> getAllPromos() {
        // Tìm mã khuyến mãi bằng ID
        public Optional<Promo> getPromoById(Long promoId) {...}

        // Tìm mã khuyến mãi bằng mã code
        public Promo getPromoByCode(String code) {...}

        // Lấy danh sách các mã khuyến mãi đang hoạt động
        public List<Promo> getActivePromos() {...}

        // Tạo hoặc cập nhật mã khuyến mãi
        public Promo saveOrUpdatePromo(Promo promo) {...}

        // Xóa mã khuyến mãi theo ID
        public void deletePromoById(Long promoId) {...}

        // Áp dụng mã khuyến mãi (giảm số lần sử dụng còn lại)
        public Promo applyPromo(String code) {...}

        // Kiểm tra thông tin promo hợp lệ
        private void validatePromo(Promo promo) {...}
    }
}
```

Hình 29: Promo Service

PromoService

Quản lý mã khuyến mãi:

- Lấy danh sách mã khuyến mãi hiện có và đang hoạt động.
- Tạo mới, cập nhật, hoặc xóa mã khuyến mãi.
- Hỗ trợ áp dụng mã khuyến mãi và kiểm tra các điều kiện hợp lệ.

3.9. Subscription Service

```
@Service 1 usage ▲ Winz18
public class SubscriptionService {

    private final SubscriptionRepository subscriptionRepository; 11 usages
    private final UserRepository userRepository; 1 usage

    @Autowired ▲ Winz18
    public SubscriptionService(SubscriptionRepository subscriptionRepository, UserRepository userRepository) {...}

    // Lấy tất cả subscription
    public List<Subscription> getAllSubscriptions() { return subscriptionRepository.findAll(); }

    // Lấy subscription của một user
    public Subscription getSubscriptionByUser(User user) {...}

    // Tìm subscription theo ID
    public Optional<Subscription> getSubscriptionById(Long id) {...}

    // Tạo mới hoặc cập nhật subscription
    public Subscription saveOrUpdateSubscription(Subscription subscription) {...}

    // Xóa subscription theo ID
    public void deleteSubscriptionById(Long id) {...}

    // Tìm các subscription hiện đang hoạt động
    public List<Subscription> getActiveSubscriptions() {...}

    // Kiểm tra xem một user có đang trong gói VIP hay không
    public boolean isUserVIP(User user) {...}

    // Gia hạn gói subscription
    public Subscription extendSubscription(Long subscriptionId, int additionalMonths) {...}

    // Validate subscription
    private void validateSubscription(Subscription subscription) {...}
}
```

Hình 30: Subscription Service

SubscriptionService

Quản lý gói đăng ký của người dùng:

- Lấy danh sách gói VIP hiện tại hoặc theo người dùng cụ thể.
- Tạo mới, gia hạn, hoặc xóa gói đăng ký.
- Kiểm tra trạng thái VIP của người dùng một cách nhanh chóng.

3.10. UserServiceImpl

```
@Service ▲ Winz18 *
public class UserDetailsServiceImpl implements UserDetailsService {

    private final UserRepository userRepository;  2 usages

    public UserDetailsServiceImpl(UserRepository userRepository) { this.userRepository = userRepository; }

    @Override no usages ▲ Winz18 *
    public UserDetails loadUserByUsername(String email) throws UsernameNotFoundException {
        hcmute.uni.final_term_project.entity.User user = userRepository.findByEmail(email);
        System.out.print("AAA");
        if (user == null) {
            throw new UsernameNotFoundException("User not found with email: " + email);
        }

        return org.springframework.security.core.userdetails.User.withUsername(user.getEmail())
            .password(user.getPassword())
            .roles(user.isAdmin() ? "ADMIN" : "USER")
            .build();
    }
}
```

Hình 31: UserServiceImpl

UserDetailsServiceImpl

Hỗ trợ tích hợp bảo mật:

- Cung cấp thông tin xác thực cho Spring Security, dựa trên email người dùng.
- Phân quyền giữa ADMIN và USER.

3.11. User Service

```
// Lấy User ID của người dùng hiện tại
public Long getCurrentUserId() {...}

💡 // Lấy entity User hiện tại
public User getCurrentUser() {...}

// Lấy tất cả người dùng
public List<User> getAllUsers() { 1 usage  ± Winz18
    return userRepository.findAll();
}

// Tìm người dùng theo ID
public Optional<User> getUserById(Long id) {...}

// Tìm người dùng theo email
public User getUserByEmail(String email) {...}

// Tìm kiếm người dùng theo tên (keyword)
public List<User> searchUsersByName(String keyword) {...}

// Tạo mới hoặc cập nhật thông tin người dùng
public void saveOrUpdateUser(User user) {...}

// Xóa người dùng theo ID
public void deleteUserById(Long id) {...}

// Đếm số người dùng đang hoạt động
public long countActiveUsers() { no usages  ± Winz18
    return userRepository.countByIsActiveTrue();
}

// Kích hoạt hoặc vô hiệu hóa người dùng
public void setActiveStatus(Long userId, boolean isActive) {...}

// Cập nhật thời gian đăng nhập cuối cùng
public void updateLastLogin(Long userId) {...}

// Kiểm tra thông tin người dùng hợp lệ
private void validateUser(User user) { 1 usage  ± Winz18}
```

Hình 32: User Service

UserService

Quản lý người dùng và các hoạt động liên quan:

- Hỗ trợ tìm kiếm, cập nhật, và xóa thông tin người dùng.
- Thống kê dữ liệu như số lượng người dùng VIP, đang hoạt động, và doanh thu tháng.

- Hỗ trợ quản lý profile, thay đổi mật khẩu, và xóa tài khoản.

3.12. View history Service

```
@Service 5 usages ▾ Winz18
public class ViewHistoryService {
    private final ViewHistoryRepository viewHistoryRepository; 5 usages

    @Autowired ▾ Winz18
    public ViewHistoryService(ViewHistoryRepository viewHistoryRepository) {...}

    // Lấy danh sách tài liệu đã xem gần đây, không trùng lặp
    public List<ViewHistory> getDistinctViewHistoryByUser(User user) {...}

    // Lấy top 3 tài liệu đã xem gần đây, không trùng lặp
    public List<ViewHistory> getTop3DistinctViewHistoryByUser(User user) {...}

    public void saveViewHistory(User currentUser, Document doc) {...}
}
```

Hình 33: View history Service

ViewHistoryService

Theo dõi lịch sử xem tài liệu:

- Cung cấp danh sách các tài liệu đã xem gần đây, không trùng lặp.
- Lưu hoặc cập nhật lịch sử xem tài liệu của người dùng.

4. Xây dựng các controller Admin

4.1. Controller dashboard

```
public String GotoDashboard() { return "admin/dashboard"; // file templates/index.html }
@GetMapping("/admin/dashboard") ✎ truongvip1
public String Dashboard(Model model) {
    // Thống kê số lượng Users
    long totalUsers = userService.countAllUsers();
    long vipUsers = userService.countVipUsers();
    long onlineUsers = userService.countOnlineUsers();

    // Thống kê số lượng Documents
    long totalDocuments = documentRepository.count();
    long todaysUploads = documentRepository.countDocumentsUploadedToday();

    // Thống kê doanh thu
    long dailyRevenue = commissionRepository.getDailyRevenue();
    long monthlyRevenue = commissionRepository.getMonthlyRevenue();
    long yearlyRevenue = commissionRepository.getYearlyRevenue();

    // Đưa dữ liệu vào model
    model.addAttribute(attributeName: "totalUsers", totalUsers);
    model.addAttribute(attributeName: "vipUsers", vipUsers);
    model.addAttribute(attributeName: "onlineUsers", onlineUsers);
    model.addAttribute(attributeName: "totalDocuments", totalDocuments);
    model.addAttribute(attributeName: "todaysUploads", todaysUploads);
    model.addAttribute(attributeName: "dailyRevenue", dailyRevenue);
    model.addAttribute(attributeName: "monthlyRevenue", monthlyRevenue);
    model.addAttribute(attributeName: "yearlyRevenue", yearlyRevenue);

    return "admin/dashboard"; // file templates/index.html
}
```

Hình 12: Controller dashboard

4.2. Controller manage commission

```
@GetMapping("/admin/commission")  ✎ truongvip1
public String adminCommission(Model model) {
    List<Commission> commissions = commissionService.getAllCommissions();
    List<Download> downloads = downloadService.getAllDownloads();
    model.addAttribute( attributeName: "commissions", commissions);

    // Tinh toán các giá trị tổng hợp
    double totalCommission = commissions.stream() Stream<Commission>
        .mapToDouble(Commission::getValue) DoubleStream
        .sum();

    double approvedCommission = commissions.stream() Stream<Commission>
        .filter(Commission::getPaid)
        .mapToDouble(Commission::getValue) DoubleStream
        .sum();

    double pendingCommission = totalCommission - approvedCommission;

    // Truyền các giá trị vào model
    model.addAttribute( attributeName: "totalCommission", totalCommission);
    model.addAttribute( attributeName: "approvedCommission", approvedCommission);
    model.addAttribute( attributeName: "pendingCommission", pendingCommission);
    return "admin/manage-commission";
}
@PostMapping("/admin/commission/approve")  ✎ truongvip1
public String approvePayment(@RequestParam Long commisionId){
    try {
        Commission commission = commissionRepository.findById(commisionId).get();
        commission.setPaid(true);
        commissionService.saveOrUpdateCommission(commission);
        return "redirect:/admin/commission";
    } catch (Exception e) {
        return "redirect:/admin/commission";
    }
}
@PostMapping("/admin/commission/payall")  ✎ truongvip1
```

Hình 13: Controller manage commission

4.3. Controller manage discount

```
@GetMapping("/admin/discount")  ✎ truongvip1
public String discount(Model model) {
    List<Promo> promos = promoService.getAllPromos();

    model.addAttribute( attributeName: "promos", promos);
    return "admin/manage-discount";
}

@PostMapping("/admin/discount/edit")  ✎ truongvip1
public String editVoucher(
    @RequestParam("code") String code,
    @RequestParam("value") Double value,
    @RequestParam("uses") int uses,
    @RequestParam("limitUsage") int limitUsage,
    @RequestParam("startDate") @DateTimeFormat(pattern = "yyyy-MM-dd") LocalDate startDate,
    @RequestParam("endDate") @DateTimeFormat(pattern = "yyyy-MM-dd") LocalDate endDate) {

    // Tìm voucher (promo) theo code
    Promo promo = promoService.getPromoByCode(code);
    if (promo != null) {
        // Cập nhật các trường
        promo.setValue(value);
        promo.setUses(uses);
        promo.setLimitUsage(limitUsage);
        promo.setStartDate(startDate.atStartOfDay()); // Chuyển LocalDate thành LocalDateTime
        promo.setEndDate(endDate.atStartOfDay());

        // Lưu thay đổi
        promoService.saveOrUpdatePromo(promo);
    }
    // Chuyển hướng về trang danh sách vouchers
    return "redirect:/admin/discount";
}

@PostMapping("/admin/discount/add")  ✎ truongvip1
public String addVoucher(
```

Hình 14: Controller manage discount

4.4. Controller manage document

```
public String adminDocuments(Model model) {
    List<Document> documents = documentService.getAllDocuments();

    // Truyền danh sách tài liệu vào model để hiển thị trên Thymeleaf
    model.addAttribute("documents", documents);
    List<String> tags = documentService.getAllTags();

    // Chuyển danh sách các tag vào model
    model.addAttribute("tags", tags);

    return "admin/manage-document";
}
@PostMapping(value = "/admin/documents/edit") @truongvip1
public String editDocument(Model model,
    @RequestParam Long docId,
    @RequestParam String background,
    @RequestParam String university,
    @RequestParam String cateTags,
    @RequestParam int views,
    RedirectAttributes redirectAttributes) {
try {
    // Gọi service để cập nhật tài liệu
    Optional<Document> documentOptional = documentService.getDocumentById(docId);
    documentOptional.get().setBackground(background);
    documentOptional.get().setUniversity(university);
    documentOptional.get().setCateTags(cateTags);
    documentOptional.get().setViews(views);
    documentService.saveOrUpdateDocument(documentOptional.get());
    redirectAttributes.addFlashAttribute("successMessage", "Document updated successfully.");
} catch (Exception e) {
    redirectAttributes.addFlashAttribute("errorMessage", "Failed to update document. Please try again.");
}
return "redirect:/admin/documents";
}
@PostMapping(value = "/admin/documents/delete") @truongvip1
```

Hình 15: Controller manage document

4.5. Controller manage notification

```
@GetMapping("/admin/notification")  ✎ truongvip1
public String notification(Model model) {
    List<Notification> notifications = notificationService.getAllNotifications();
    // Add the notifications list to the model to pass to the view
    model.addAttribute( attributeName: "notifications", notifications);
    return "admin/manage-notification";
}

@PostMapping("/admin/notification/add")  ✎ truongvip1
public String addNotification(@RequestParam("title") String title,
                               @RequestParam("content") String content){
    Notification notification = new Notification();
    notification.setTitle(title);
    notification.setContent(content);
    notification.setSender("Admin"); // Default sender
    notification.setSentAt(LocalDateTime.now()); // Current timestamp
    notificationService.createNotification(notification);
    return "redirect:/admin/notification";
}

@PostMapping("/admin/notification/delete")  ✎ truongvip1
public String deleteNotification(@RequestParam("notificationId") Long notificationId) {
    // Delete the notification by ID
    notificationService.deleteNotificationById(notificationId);
    return "redirect:/admin/notification"; // Redirect to the notifications page
}
```

Hình 16: Controller manage notification

4.6. Controller manage transaction

```
@GetMapping("/admin/transaction")  ✎ truongvip1
public String adminTransaction(Model model) {
    List<Subscription> subscriptions = subscriptionService.getAllSubscriptions();

    // Đổ dữ liệu vào model để hiển thị trong Thymeleaf
    model.addAttribute( attributeName: "subscriptions", subscriptions);
    long totalSubscribers = subscriptions.size();

    // Tính tổng doanh thu
    double totalRevenue = subscriptions.stream()
        .mapToDouble(Subscription::getValue) // Lấy giá trị value của mỗi subscription
        .sum();

    // Đổ dữ liệu vào model
    model.addAttribute( attributeName: "totalSubscribers", totalSubscribers);
    model.addAttribute( attributeName: "totalRevenue", totalRevenue);

    return "admin/manage-transaction";
}
```

Hình 17: Controller manage transaction

4.7. Controller manage user

```
@GetMapping(@RequestMapping("/admin/users"))  ± truongvip1
public String ManageUser(Model model) {
    List<User> users = userService.getAllUsers();
    List<Download> downloads = downloadService.getAllDownloads();
    List<Likes> likes = likesService.getAllLikes();
    List<Follower> followers = followerService.getAllFollowers();
    download list
    Map<Long, Long> downloadCountByUserId = downloads.stream()
        .collect(Collectors.groupingBy(download -> download.getUser().getUserId(), Collectors.counting()));
    List<Long> listDownload = users.stream() Stream<User>
        .map(user -> downloadCountByUserId.getOrDefault(user.getUserId(), defaultValue: 0L)) Stream<Long>
        .collect(Collectors.toList());
    like list
    Map<Long, Long> likesCountByUserId = likes.stream()
        .collect(Collectors.groupingBy(like -> like.getUser().getUserId(), Collectors.counting()));
    List<Long> listLikes = users.stream() Stream<User>
        .map(user -> likesCountByUserId.getOrDefault(user.getUserId(), defaultValue: 0L)) Stream<Long>
        .collect(Collectors.toList());
    follow list
    Map<Long, Long> followersCountByUserId = followers.stream()
        .collect(Collectors.groupingBy(follower -> follower.getFollowed().getUserId(), Collectors.counting()));

    List<Long> listFollowers = users.stream() Stream<User>
        .map(user -> followersCountByUserId.getOrDefault(user.getUserId(), defaultValue: 0L)) Stream<Long>
        .collect(Collectors.toList());

    System.out.println(listDownload);
    List<Map<String, Object>> userData = users.stream() Stream<User>
        .map(user -> {
            Map<String, Object> data = new HashMap<>();
            data.put("user", user);
            data.put("downloads", downloadCountByUserId.getOrDefault(user.getUserId(), defaultValue: 0L));
            data.put("likes", likesCountByUserId.getOrDefault(user.getUserId(), defaultValue: 0L));
            data.put("followers", followersCountByUserId.getOrDefault(user.getUserId(), defaultValue: 0L));
            return data;
        }) Stream<Map<...>>
```

Hình 18: Controller manage user

5. Xây dựng các controller User

5.1. UserDocumentController

```
@GetMapping("/my-documents") ʌ Winz18
public String showMyDocumentPage(Model model) {
    // Lấy danh sách tài liệu do user hiện tại sở hữu
    List<Document> myDocuments = documentService.getDocumentsByOwner(userService.getCurrentUser());
    model.addAttribute( attributeName: "myDocuments", myDocuments);

    // Get info for sidebar
    model.addAttribute( attributeName: "currentUser", userService.getCurrentUser().getName());
    model.addAttribute( attributeName: "isVIP", userService.getCurrentUser().isVIP());
    model.addAttribute( attributeName: "followers", userService.getCurrentUser().getFollowers());
    model.addAttribute( attributeName: "documentsUploaded", documentService.getDocumentsByOwner(userService.getCurrentUser()).size());

    return "user/my-documents";
}

// endpoint for view detail of a document
@GetMapping("/view-detail/{id}") ʌ Winz18
public String showDocumentDetailPage(@PathVariable("id") Long documentId, Model model) {
    // Lấy tài liệu theo ID
    Optional<Document> document = documentService.getDocumentById(documentId);
    Document doc = document.orElseThrow(() -> new IllegalArgumentException("Document not found"));

    model.addAttribute( attributeName: "document", doc);

    // Lấy danh sách người dùng đã like tài liệu này
    List<Likes> likes = likesService.getLikesByDocument(doc);
    model.addAttribute( attributeName: "likes", likes.size());

    // Lấy danh sách bình luận của tài liệu
    List<Comment> comments = commentService.getCommentsByDocument(doc);
    model.addAttribute( attributeName: "comments", comments);

    // Lấy danh sách người dùng đã tải tài liệu này
    List<Download> downloads = downloadService.getDownloadsByDocument(doc);
    model.addAttribute( attributeName: "downloads", downloads.size());
}
```

Hình 19: UserDocumentController

UserDocumentController chịu trách nhiệm quản lý các chức năng liên quan đến tài liệu của người dùng trong hệ thống, hỗ trợ đầy đủ các thao tác từ hiển thị, tạo mới, chỉnh sửa đến xóa tài liệu.

Chức năng chính

- **Hiển thị danh sách tài liệu đã upload:**
 - Endpoint: GET /user/my-documents.
 - Lấy danh sách các tài liệu do người dùng hiện tại sở hữu, loại bỏ các tài liệu bị xóa hoặc chưa được duyệt.

- Hiển thị thông tin sidebar bao gồm tên người dùng, trạng thái VIP, số lượt theo dõi, và số tài liệu đã upload.

- **Xem chi tiết tài liệu:**

- Endpoint: GET /user/view-detail/{id}.
- Lấy thông tin chi tiết tài liệu theo ID, chỉ cho phép tài liệu VIP được xem bởi người dùng VIP.
- Tăng lượt xem và lưu lịch sử xem của tài liệu.
- Hiển thị danh sách bình luận và thông tin quyền sở hữu của tài liệu.

- **Thêm mới tài liệu:**

- Endpoint: POST /user/upload-document.
- Kiểm tra định dạng file tải lên (chỉ chấp nhận PDF và DOCX).
- Lưu file tài liệu và thumbnail vào thư mục tương ứng, sau đó lưu thông tin vào cơ sở dữ liệu.
- Hỗ trợ chế độ kiểm tiền cho người dùng VIP.

- **Chỉnh sửa tài liệu:**

- Endpoint: POST /user/edit-document/{id}.
- Cho phép người dùng cập nhật thông tin và file liên quan của tài liệu.
- Kiểm tra quyền sở hữu trước khi thực hiện chỉnh sửa.

- **Xóa tài liệu:**

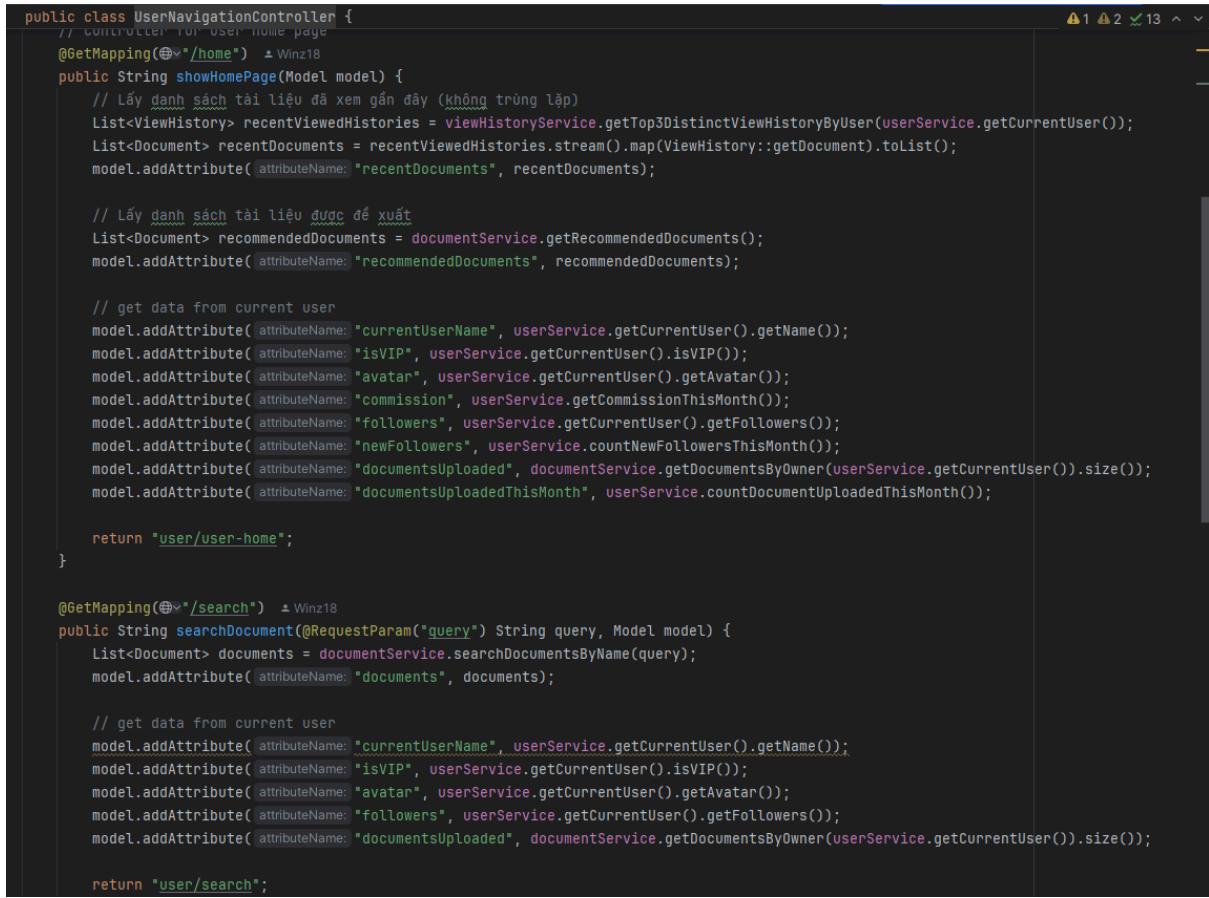
- Endpoint: POST /user/delete-document/{id}.

- Loại bỏ file tài liệu và thumbnail khỏi hệ thống, đồng thời đánh dấu tài liệu là "đã xóa" trong cơ sở dữ liệu.

- Thao tác tương tác:**

- Like tài liệu: POST /user/like-document/{id}.
- Tải xuống tài liệu: GET /user/download-document/{id}, đồng thời xử lý việc tăng lượt tải xuống và chi trả hoa hồng cho tài liệu VIP.
- Bình luận tài liệu: POST /user/comment-document/{id}.

5.2. UserNavController



```

public class UserNavController {
    // Controller for user home page
    @GetMapping(path="/home")  ↳ Winz18
    public String showHomePage(Model model) {
        // Lấy danh sách tài liệu đã xem gần đây (không trùng lặp)
        List<ViewHistory> recentViewedHistories = viewHistoryService.getTop3DistinctViewHistoryByUser(userService.getCurrentUser());
        List<Document> recentDocuments = recentViewedHistories.stream().map(ViewHistory::getDocument).toList();
        model.addAttribute(attributeName: "recentDocuments", recentDocuments);

        // Lấy danh sách tài liệu được đề xuất
        List<Document> recommendedDocuments = documentService.getRecommendedDocuments();
        model.addAttribute(attributeName: "recommendedDocuments", recommendedDocuments);

        // get data from current user
        model.addAttribute(attributeName: "currentUserName", userService.getCurrentUser().getName());
        model.addAttribute(attributeName: "isVIP", userService.getCurrentUser().isVIP());
        model.addAttribute(attributeName: "avatar", userService.getCurrentUser().getAvatar());
        model.addAttribute(attributeName: "commission", userService.getCommissionThisMonth());
        model.addAttribute(attributeName: "followers", userService.getCurrentUser().getFollowers());
        model.addAttribute(attributeName: "newFollowers", userService.countNewFollowersThisMonth());
        model.addAttribute(attributeName: "documentsUploaded", documentService.getDocumentsByOwner(userService.getCurrentUser()).size());
        model.addAttribute(attributeName: "documentsUploadedThisMonth", userService.countDocumentUploadedThisMonth());

        return "user/user-home";
    }

    @GetMapping(path="/search")  ↳ Winz18
    public String searchDocument(@RequestParam("query") String query, Model model) {
        List<Document> documents = documentService.searchDocumentsByName(query);
        model.addAttribute(attributeName: "documents", documents);

        // get data from current user
        model.addAttribute(attributeName: "currentUserName", userService.getCurrentUser().getName());
        model.addAttribute(attributeName: "isVIP", userService.getCurrentUser().isVIP());
        model.addAttribute(attributeName: "avatar", userService.getCurrentUser().getAvatar());
        model.addAttribute(attributeName: "followers", userService.getCurrentUser().getFollowers());
        model.addAttribute(attributeName: "documentsUploaded", documentService.getDocumentsByOwner(userService.getCurrentUser()).size());

        return "user/search";
    }
}

```

Hình 20: UserNavController

UserNavController tập trung vào điều hướng và hiển thị các trang thông tin quan trọng trong hệ thống.

Chức năng chính

- **Trang chủ người dùng:**
 - Endpoint: GET /user/home.
 - Hiển thị danh sách các tài liệu đã xem gần đây (không trùng lặp) và tài liệu được đề xuất, loại bỏ các tài liệu bị xóa hoặc chưa được duyệt.
 - Hiển thị thông tin người dùng hiện tại như trạng thái VIP, doanh thu tháng, số lượt theo dõi, và số tài liệu đã upload.
- **Tìm kiếm tài liệu:**
 - Endpoint: GET /user/search.
 - Tìm kiếm tài liệu theo tên, loại bỏ các tài liệu bị xóa hoặc chưa được duyệt.
- **Xem tài liệu gần đây:**
 - Endpoint: GET /user/recent-documents.
 - Hiển thị danh sách các tài liệu mà người dùng đã xem gần đây.
- **Tài liệu được đề xuất:**
 - Endpoint: GET /user/recommended-documents.
 - Lấy danh sách tài liệu được đề xuất dựa trên hệ thống gợi ý.
- **Tìm kiếm theo tag:**
 - Endpoint: GET /user/cate-doc (hiển thị tag phổ biến) và GET /user/cate-search (tìm kiếm tài liệu theo tag).
 - Lọc tài liệu theo tag, chỉ hiển thị các tài liệu đã được duyệt.

- **Tìm kiếm theo trường đại học:**

- Endpoint: GET /user/uni-doc (hiển thị các trường phô biến) và GET /user/uni-search (tìm kiếm tài liệu theo trường).
- Hiển thị danh sách tài liệu của từng trường đại học, hỗ trợ phân loại.

5.3 UserPaymentController

```
@Controller  ✎ Winz18
@RequestMapping(@"/user")
public class UserPaymentController {
    private final UserService userService;  18 usages
    private final DocumentService documentService;  3 usages
    private final SubscriptionService subscriptionService;  2 usages

    @Autowired  ✎ Winz18
    public UserPaymentController(UserService userService , DocumentService documentService, SubscriptionService subscriptionService) {
        this.userService = userService;
        this.documentService = documentService;
        this.subscriptionService = subscriptionService;
    }

    @GetMapping(@"/vip-member")  ✎ Winz18
    public String showVipMemberPage(Model model) {
        // get vip plan of current user
        Subscription currentSubscription = subscriptionService.getSubscriptionByUser(userService.getCurrentUser());
        model.addAttribute( attributeName: "currentSubscription", currentSubscription);

        // get data from current user
        model.addAttribute( attributeName: "currentUser", userService.getCurrentUser().getName());
        model.addAttribute( attributeName: "isVIP", userService.getCurrentUser().isVIP());
        model.addAttribute( attributeName: "avatar", userService.getCurrentUser().getAvatar());
        model.addAttribute( attributeName: "followers", userService.getCurrentUser().getFollowers());
        model.addAttribute( attributeName: "bio", userService.getCurrentUser().getBio());
        model.addAttribute( attributeName: "email", userService.getCurrentUser().getEmail());
        model.addAttribute( attributeName: "documentsUploaded", documentService.getDocumentsByOwner(userService.getCurrentUser()).size());
    }
}
```

Hình 21: UserPaymentController

UserPaymentController quản lý các thao tác liên quan đến thanh toán và nâng cấp tài khoản VIP.

Chức năng chính

- **Xem gói VIP hiện tại:**
 - Endpoint: GET /user/vip-member.
 - Hiển thị thông tin gói VIP của người dùng hiện tại, bao gồm ngày bắt đầu và kết thúc, doanh thu từ hoa hồng, và trạng thái VIP.
- **Nâng cấp VIP:**
 - Endpoint: GET /user/payment.

- Cung cấp giao diện thanh toán để nâng cấp lên VIP, đồng thời hiển thị thông tin tài khoản hiện tại như tên, email, và bio.

5.4 UserProfileController

```

@GetMapping("user/profile")  ^ Winz18
public String showProfilePage(Model model) {
    // get data from current user
    model.addAttribute("currentUser", userService.getCurrentUser());
    model.addAttribute("isVIP", userService.getCurrentUser().isVIP());
    model.addAttribute("avatar", userService.getCurrentUser().getAvatar());
    model.addAttribute("followers", userService.getCurrentUser().getFollowers());
    model.addAttribute("bio", userService.getCurrentUser().getBio());
    model.addAttribute("email", userService.getCurrentUser().getEmail());
    model.addAttribute("documentsUploaded", documentService.getDocumentsByOwner(userService.getCurrentUser()).size());

    return "user/my-profile";
}

@PostMapping("user/update-profile")  ^ Winz18
public String updateProfile(@RequestParam("name") String name, @RequestParam("bio") String bio, @RequestParam("email") String email, Model model) {
    userService.updateProfile(name, bio, email);
    return "redirect:/user/profile";
}

```

Hình 21: UserProfileController

UserProfileController

Mô tả

UserProfileController tập trung vào việc quản lý hồ sơ cá nhân và các cài đặt tài khoản.

Chức năng chính

- **Xem hồ sơ cá nhân:**
 - Endpoint: GET /user/profile.
 - Hiển thị thông tin cá nhân của người dùng như tên, bio, email, và số lượt tài liệu đã upload.
- **Cập nhật hồ sơ:**
 - Endpoint: POST /user/update-profile.

- Hỗ trợ người dùng cập nhật thông tin như tên, bio, email, và ảnh đại diện.

- **Trang cài đặt tài khoản:**

- Endpoint: GET /user/setting.
- Hiển thị các cài đặt tài khoản, bao gồm thay đổi mật khẩu và xóa tài khoản.

- **Thay đổi mật khẩu:**

- Endpoint: POST /user/setting/change-password.
- Hỗ trợ kiểm tra và thay đổi mật khẩu mới với xác nhận mật khẩu cũ.

- **Xóa tài khoản:**

- Endpoint: POST /user/setting/delete-account.
- Cho phép người dùng xóa tài khoản của mình, hệ thống sẽ chuyển hướng về trang đăng xuất.

6. Các controller cho Guest

6.1. AboutController

AboutController đơn giản, quản lý việc điều hướng đến trang "About Us" của hệ thống.

Chức năng chính

- **Hiển thị thông tin về hệ thống:**

- Endpoint: GET /about.
- Điều hướng người dùng đến giao diện mô tả thông tin về hệ thống, lịch sử hình thành, và mục tiêu phát triển.
- Trang liên quan: about_us.html.

6.2. AuthController

```
package hcmute.uni.final_term_project.controller;

> import ...;

@Controller @Winz18
public class AuthController {

    private final UserService userService; 4 usages
    private final AuthenticationManager authenticationManager; 2 usages
    private final PasswordEncoder passwordEncoder; 2 usages

    @Autowired @Winz18
    public AuthController(UserService userService, AuthenticationManager authenticationManager, PasswordEncoder passwordEncoder) {
        this.userService = userService;
        this.authenticationManager = authenticationManager;
        this.passwordEncoder = passwordEncoder;
    }

    // Hiển thị trang login
    @GetMapping("/login") @Winz18
    public String showLoginPage(Model model) {
        model.addAttribute("attributeName: "error", "attributeValue: null); // Để thông báo lỗi (nếu có)
        return "login"; // templates/login.html
    }

    // Xử lý đăng nhập
    @PostMapping("/Login") @Winz18
    public String handleLogin(@ModelAttribute("email") String email,
                             @ModelAttribute("password") String password,
                             Model model) {
```

AuthController chịu trách nhiệm quản lý các thao tác liên quan đến xác thực và đăng ký tài khoản.

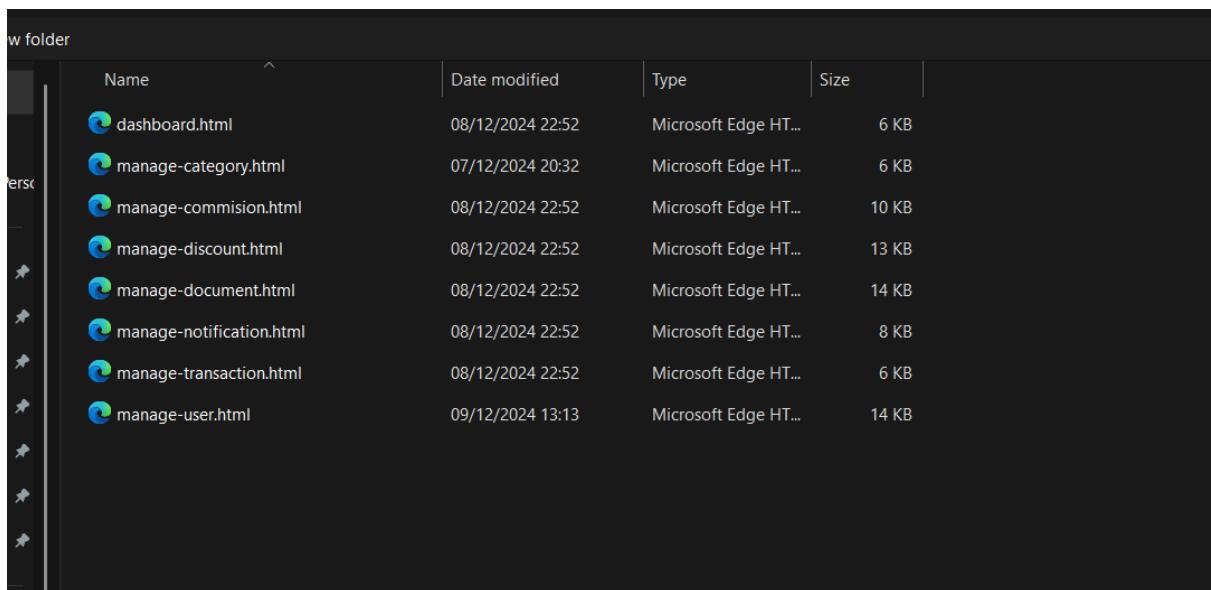
Chức năng chính

- **Hiển thị trang đăng nhập:**
 - Endpoint: GET /login.
 - Hiển thị giao diện đăng nhập, hỗ trợ hiển thị lỗi nếu đăng nhập thất bại.
- **Xử lý đăng nhập:**
 - Endpoint: POST /login.
 - Xác thực thông tin đăng nhập của người dùng bằng AuthenticationManager.
 - Cập nhật thời gian đăng nhập cuối cùng nếu thành công.
 - Điều hướng đến trang chủ hoặc hiển thị lỗi khi thất bại.
- **Hiển thị trang đăng ký:**
 - Endpoint: GET /register.

- Chuẩn bị một đối tượng User rỗng để binding dữ liệu từ form đăng ký.
- **Xử lý đăng ký:**
 - Endpoint: POST /register.
 - Tạo tài khoản mới với thông tin như tên, email, mật khẩu, số điện thoại, và ngày sinh.
 - Mã hóa mật khẩu trước khi lưu vào cơ sở dữ liệu bằng PasswordEncoder.
 - Thiết lập các giá trị mặc định (kích hoạt, không phải admin, không phải VIP).
 - Điều hướng đến trang đăng nhập nếu thành công hoặc hiển thị lỗi khi thất bại.
- **Xử lý đăng xuất:**
 - Endpoint: GET /logout.
 - Chuyển hướng người dùng về trang đăng nhập với thông báo đã đăng xuất thành công.

7. Các web-view của website

7.1. Các giao diện admin



Name	Date modified	Type	Size
dashboard.html	08/12/2024 22:52	Microsoft Edge HT...	6 KB
manage-category.html	07/12/2024 20:32	Microsoft Edge HT...	6 KB
manage-commision.html	08/12/2024 22:52	Microsoft Edge HT...	10 KB
manage-discount.html	08/12/2024 22:52	Microsoft Edge HT...	13 KB
manage-document.html	08/12/2024 22:52	Microsoft Edge HT...	14 KB
manage-notification.html	08/12/2024 22:52	Microsoft Edge HT...	8 KB
manage-transaction.html	08/12/2024 22:52	Microsoft Edge HT...	6 KB
manage-user.html	09/12/2024 13:13	Microsoft Edge HT...	14 KB

1. dashboard.html:

- Hiển thị bảng điều khiển chính với các thống kê về hệ thống.
- Cung cấp cái nhìn tổng quan về trạng thái hoạt động.

2. manage-category.html:

- Quản lý danh mục của tài liệu, cho phép thêm, sửa, xóa các danh mục.

3. manage-commission.html:

- Quản lý hoa hồng từ các giao dịch tải tài liệu VIP.
- Hiển thị chi tiết về số tiền hoa hồng và trạng thái thanh toán.

4. manage-discount.html:

- Quản lý các mã khuyến mãi, bao gồm thêm, sửa, và theo dõi trạng thái.

5. manage-document.html:

- Quản lý các tài liệu được upload lên hệ thống, bao gồm duyệt hoặc từ chối tài liệu.

6. manage-notification.html:

- Gửi và quản lý các thông báo hệ thống đến người dùng.

7. manage-transaction.html:

- Theo dõi và quản lý các giao dịch liên quan đến tài khoản VIP hoặc tài liệu tải xuống.

8. manage-user.html:

- Quản lý danh sách người dùng, bao gồm trạng thái kích hoạt, quyền admin hoặc VIP.

Chức năng chung

- Mỗi giao diện hỗ trợ quản trị viên dễ dàng theo dõi, kiểm soát và quản lý các tài nguyên của hệ thống.
- Các file giao diện được liên kết với các controller quản trị tương ứng.

7.2. Các giao diện user

Name	Date modified	Type	Size
uni-search.html	09/12/2024 10:21	Microsoft Edge HT...	1 KB
cate-search.html	09/12/2024 9:55	Microsoft Edge HT...	1 KB
search.html	08/12/2024 12:36	Microsoft Edge HT...	1 KB
cate-doc.html	07/12/2024 20:32	Microsoft Edge HT...	1 KB
edit-doc.html	07/12/2024 20:32	Microsoft Edge HT...	1 KB
my-documents.html	07/12/2024 20:32	Microsoft Edge HT...	1 KB
my-profile.html	07/12/2024 20:32	Microsoft Edge HT...	1 KB
payment.html	07/12/2024 20:32	Microsoft Edge HT...	1 KB
recent-documents.html	07/12/2024 20:32	Microsoft Edge HT...	1 KB
recommend-documents.html	07/12/2024 20:32	Microsoft Edge HT...	1 KB
setting.html	07/12/2024 20:32	Microsoft Edge HT...	1 KB
uni-doc.html	07/12/2024 20:32	Microsoft Edge HT...	1 KB
upload.html	07/12/2024 20:32	Microsoft Edge HT...	1 KB
user-home.html	07/12/2024 20:32	Microsoft Edge HT...	1 KB
view-doc.html	07/12/2024 20:32	Microsoft Edge HT...	1 KB
vip-member.html	07/12/2024 20:32	Microsoft Edge HT...	1 KB

1. uni-search.html & cate-search.html:

- o Tìm kiếm tài liệu theo trường đại học hoặc danh mục tag.

2. search.html:

- o Giao diện tìm kiếm tổng hợp tài liệu theo từ khóa.

3. cate-doc.html & uni-doc.html:

- o Hiển thị danh sách tag phổ biến hoặc trường đại học để người dùng dễ dàng khám phá tài liệu.

4. edit-doc.html & my-documents.html:

- o Giao diện chỉnh sửa và danh sách tài liệu của người dùng.

5. my-profile.html & setting.html:

- o Quản lý thông tin cá nhân và các cài đặt tài khoản như thay đổi mật khẩu hoặc xóa tài khoản.

6. recent-documents.html & recommend-documents.html:

- o Hiển thị các tài liệu đã xem gần đây hoặc được đề xuất.

7. upload.html:

- o Giao diện cho phép người dùng upload tài liệu mới.

8. user-home.html:

- Trang chủ cá nhân, hiển thị thông tin tài khoản và tài liệu nổi bật.

9. view-doc.html:

- Hiển thị nội dung chi tiết của tài liệu, hỗ trợ các thao tác like, tải xuống, và bình luận.

10. vip-member.html:

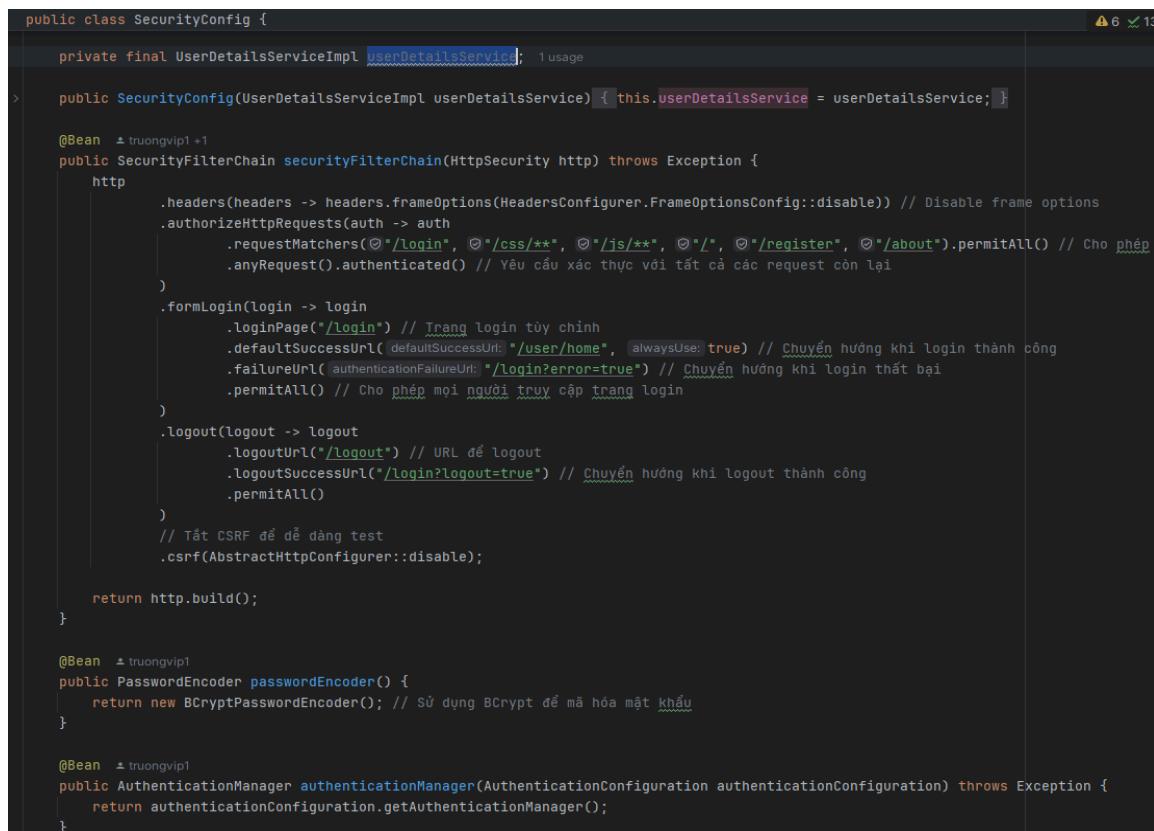
- Hiển thị thông tin và hướng dẫn nâng cấp lên tài khoản VIP.

Chức năng chung

- Giao diện người dùng tập trung vào trải nghiệm trực quan, hỗ trợ các tính năng như tìm kiếm, quản lý tài liệu, và tương tác với nội dung.
- Liên kết chặt chẽ với các controller của người dùng để xử lý dữ liệu.

8. Các configs

8.1. SecurityConfig



```

public class SecurityConfig {

    private final UserDetailsServiceImpl userDetailsService; 1 usage

    public SecurityConfig(UserDetailsServiceImpl userDetailsService) { this.userDetailsService = userDetailsService; }

    @Bean @truongvip1
    public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
        http
            .headers(headers -> headers.frameOptions(HeadersConfigurer.FrameOptionsConfig::disable)) // Disable frame options
            .authorizeHttpRequests(auth -> auth
                .requestMatchers(ignoredPathMatchers("/login", "/css/**", "/js/**", "/*", "/register", "/about")).permitAll() // Cho phép tất cả các request
                .anyRequest().authenticated() // Yêu cầu xác thực với tất cả các request còn lại
            )
            .formLogin(login -> login
                .loginPage("/login") // Trang login tùy chỉnh
                .defaultSuccessUrl(defaultSuccessUrl: "/user/home", alwaysUse: true) // Chuyển hướng khi login thành công
                .failureUrl(authenticationFailureUrl: "/login?error=true") // Chuyển hướng khi login thất bại
                .permitAll() // Cho phép mọi người truy cập trang login
            )
            .logout(logout -> logout
                .logoutUrl("/logout") // URL để logout
                .logoutSuccessUrl("/login?logout=true") // Chuyển hướng khi logout thành công
                .permitAll()
            )
            // Tắt CSRF để dễ dàng test
            .csrf(AbstractHttpConfigurer::disable);
    }

    return http.build();
}

@Bean @truongvip1
public PasswordEncoder passwordEncoder() {
    return new BCryptPasswordEncoder(); // Sử dụng BCrypt để mã hóa mật khẩu
}

@Bean @truongvip1
public AuthenticationManager authenticationManager(AuthenticationConfiguration authenticationConfiguration) throws Exception {
    return authenticationConfiguration.getAuthenticationManager();
}

```

Hình 47: Security Config

SecurityConfig là lớp cấu hình bảo mật của dự án, sử dụng Spring Security để đảm bảo an toàn cho các endpoint trong hệ thống.

Các cấu hình chính

1. Bảo vệ các endpoint:

- Các endpoint như /login, /register, /about, và các tài nguyên tĩnh (/css/**, /js/**) được cho phép truy cập mà không cần xác thực.
- Các request khác đều yêu cầu xác thực người dùng.

2. Cấu hình đăng nhập:

- Trang đăng nhập tùy chỉnh tại /login.
- Sau khi đăng nhập thành công, người dùng sẽ được chuyển hướng đến /user/home.
- Nếu đăng nhập thất bại, người dùng sẽ được chuyển hướng đến /login?error=true.

3. Cấu hình đăng xuất:

- URL đăng xuất được định nghĩa tại /logout.
- Sau khi đăng xuất thành công, người dùng sẽ được chuyển hướng đến /login?logout=true.

4. Mã hóa mật khẩu:

- Sử dụng BCryptPasswordEncoder để mã hóa mật khẩu, đảm bảo tính an toàn cao.

8.2. WebConfig

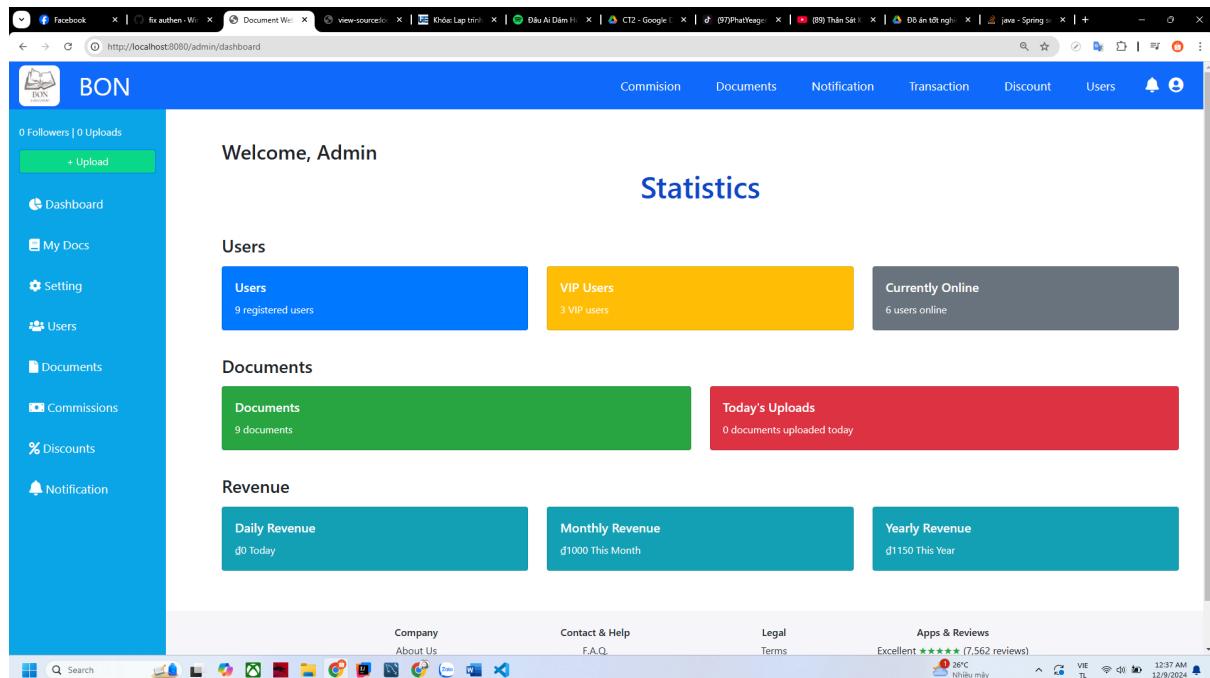
WebConfig là lớp cấu hình để xử lý các tài nguyên tĩnh và upload file trong hệ thống.

- Định nghĩa URL /uploads/** để truy cập các file được upload lên hệ thống.
- Liên kết với thư mục vật lý uploads/ trong hệ thống file.
- Ví dụ: Nếu một tài liệu được lưu trong uploads/documents/123.pdf, người dùng có thể truy cập tại URL /uploads/documents/123.pdf.

IV. Demo chương trình

1. Trang quản trị

1.1. Trang dashboard



Hình 48: Trang dashboard

1.2. Trang quản lý hoa hồng

The screenshot shows the 'Commission Table' section of the BON application. It displays a table with 10 rows of data, each representing a commission entry. The columns are: #, Commission ID, User Name, Email, Document ID, Amount Earned (VND), Paid, Date, and Actions. The 'Actions' column contains green 'Approve Payment' buttons. A 'Pay All' button is located at the bottom right of the table.

#	Commission ID	User Name	Email	Document ID	Amount Earned (VND)	Paid	Date	Actions
1	1	User123	user1@example.com	2	100.0	Paid	2024-12-08	<button>Approve Payment</button>
2	2	User123	user1@example.com	5	150.0	Paid	2024-12-06	<button>Approve Payment</button>
3	3	User3	user3@example.com	2	200.0	Paid	2024-12-03	<button>Approve Payment</button>
4	4	User3	user3@example.com	5	75.0	Paid	2024-11-28	<button>Approve Payment</button>
5	5	User123	user1@example.com	2	50.0	Paid	2024-12-07	<button>Approve Payment</button>
6	6	User123	user1@example.com	2	100.0	Paid	2024-12-08	<button>Approve Payment</button>
7	7	User123	user1@example.com	5	150.0	Paid	2024-12-06	<button>Approve Payment</button>
8	8	User3	user3@example.com	2	200.0	Paid	2024-12-03	<button>Approve Payment</button>
9	9	User3	user3@example.com	5	75.0	Paid	2024-11-28	<button>Approve Payment</button>
10	10	User123	user1@example.com	2	50.0	Paid	2024-12-07	<button>Approve Payment</button>

Commission Summary

Total Commission (VND)	Pending (VND)	Approved (VND)
1150.0	0.0	1150.0

Hình 49: Trang quản lý hoa hồng

1.3. Trang quản lý voucher

The screenshot shows the 'Add Voucher' form and 'Existing Vouchers' table. The 'Add Voucher' form fields include: Code (input: Enter promo code), Discount (%) (input: Enter discount value), Limit (input: Enter usage limit), Start Date (input: dd/mm/yyyy), and End Date (input: dd/mm/yyyy). A 'Save' button is at the bottom. Below the form is a table titled 'Existing Vouchers' with 5 rows of data. The columns are: #, Promo Code, Discount (%), Used, Limit, Start Date, End Date, and Actions. Each row has 'Edit' and 'Delete' buttons in the 'Actions' column.

#	Promo Code	Discount (%)	Used	Limit	Start Date	End Date	Actions
1	PROMO10	10.0	10	100	2024-12-07	2025-01-06	<button>Edit</button> <button>Delete</button>
2	PROMO15	15.0	4	50	2024-12-07	2024-12-22	<button>Edit</button> <button>Delete</button>
3	PROMO20	20.0	10	200	2024-12-07	2025-02-05	<button>Edit</button> <button>Delete</button>
4	PROMO25	25.0	2	30	2024-12-07	2025-03-07	<button>Edit</button> <button>Delete</button>
5	PROMOS	5.0	1	10	2024-12-07	2025-04-06	<button>Edit</button> <button>Delete</button>

Hình 50: Trang quản lý voucher

1.3. Trang quản lý tài liệu

The screenshot shows a list of uploaded documents:

#	Document ID	File Info	Document Details	Statistics	Owner ID	Is VIP	Actions
1	1	Type: PDF Size: 1 MB Path: /path/doc1.pdf	Background: Blue University: University A Category: #science,#technology	Downloads: 5 Views: 50 Likes: 10 Comments: 0	1	No	<button>Edit</button> <button>Delete</button>
2	2	Type: DOCX Size: 2 MB Path: /path/doc2.docx	Background: Green University: University B Category: #programming,#algorithms	Downloads: 10 Views: 100 Likes: 20 Comments: 0	1	Yes	<button>Edit</button> <button>Delete</button>
3	3	Type: PDF Size: 3 MB Path: /path/doc3.pdf	Background: Red University: University A Category: #database,#design	Downloads: 8 Views: 75 Likes: 15 Comments: 0	1	No	<button>Edit</button> <button>Delete</button>
4	4	Type: DOCX Size: 4 MB Path: /path/doc4.docx	Background: Yellow University: University C Category: #web,#frontend	Downloads: 3 Views: 25 Likes: 5 Comments: 0	1	No	<button>Edit</button> <button>Delete</button>
5	5	Type: PDF Size: 5 MB Path: /path/doc5.pdf	Background: Black1 University: University B Category: #ai,#machinelearning	Downloads: 15 Views: 150 Likes: 25 Comments: 0	1	Yes	<button>Edit</button> <button>Delete</button>
6	6	Type: PDF Size: 1 MB Path: /documents/science/physics_basics.pdf	Background: Introduction to Physics University: MIT Category: #physics,#education	Downloads: 50 Views: 120 Likes: 75 Comments: 0	1	Yes	<button>Edit</button> <button>Delete</button>
7	7	Type: DOCX Size: 2 MB Path: /documents/tech/algorithm_guide.docx	Background: Advanced Algorithms University: Stanford Category: #algorithms,#technology	Downloads: 30 Views: 80 Likes: 45 Comments: 0	2	No	<button>Edit</button> <button>Delete</button>
8	8	Type: PDF Size: 1 MB	Background: Database Design University: Harvard	Downloads: 70 Views: 150	3	Yes	<button>Edit</button> <button>Delete</button>

Hình 51: Trang quản lý tài liệu

1.2. Trang quản lý thông báo

The screenshot shows a list of notifications:

#	Notification ID	Title	Content	Sender	Sent At	Actions
1	1	Welcome!	Welcome to our platform. We're glad to have you!	Admin	2024-12-08 01:42	<button>Delete</button>
2	2	Reminder	Your subscription will expire in 3 days. Renew now!	Admin	2024-12-08 01:42	<button>Delete</button>
3	3	Update	We've updated our privacy policy. Please take a moment to review it.	Admin	2024-12-08 01:42	<button>Delete</button>
4	5	as	sss	Admin	2024-12-08 01:51	<button>Delete</button>
5	6	ss	aaa	Admin	2024-12-08 01:51	<button>Delete</button>

Buttons at the bottom: Add Notification Export Excel

Hình 52: Trang quản lý thông báo

1.6. Trang quản lý giao dịch

The screenshot shows a web application interface titled "Subscription Management". On the left, there is a sidebar with navigation links: Dashboard, My Docs, Setting, Users, Documents, Commissions, Discounts, and Notification. The main content area has a search bar and a table with columns: #, Sub ID, User ID, Start Date, End Date, Type, Value (VND), Use Promo, and Promo ID. There are two rows of data in the table. Below the table is a "Statistics" section with a table showing Total Subscribers (2) and Total Revenue (VND) (250000.0). At the bottom of the page, there is a footer with links to Company (About Us), Contact & Help (F.A.Q., Contact), Legal (Terms, Privacy Policy), and Apps & Reviews (Excellent, 7.562 reviews). The status bar at the bottom shows system information like temperature (26°C), battery level (Nhiều máy), and date/time (12/9/2024).

Hình 53: Trang quản lý giao dịch

1.7. Trang quản lý người dùng

The screenshot shows a web application interface titled "User List". On the left, there is a sidebar with navigation links: Dashboard, My Docs, Setting, Users, Documents, Commissions, Discounts, and Notification. The main content area has a table with columns: #, Avatar, Name, Email, Password, isAdmin, Last Login, isActive, isVIP, Downloads, Likes, Followers, and Actions. There are nine rows of user data in the table, each with edit and delete buttons. At the bottom of the page, there are buttons for "Add User" and "Export to Excel". The status bar at the bottom shows system information like temperature (26°C), battery level (Nhiều máy), and date/time (12/9/2024).

Hình 54: Trang quản lý người dùng

2. Trang người dùng

2.1. Trang user home

Xuất hiện ngay sau khi vừa login, hiển thị 1 số thông kê về người dùng

The screenshot shows the BON user home page. On the left sidebar, there is a profile section for 'Nguyen Thang Loi' with 0 Followers and 20 Uploads, and a '+ Upload' button. Below it are navigation links: Home, My Docs, My Profile, Setting, Subscription, Recent, and Recommend. The main content area has a search bar at the top. Below it is a 'Recently Viewed Documents' section with three items: 'TestNew' (author: Nguyen Thang Loi, views: 5, uploaded: 2024-12-09T15:28:05.173071), 'Test011' (author: Nguyen Thang Loi, views: 4, uploaded: 2024-12-09T14:17:50.459574), and 'test003' (author: Nguyen Thang Loi, views: 11, uploaded: 2024-12-09T14:23:10.568096). Each item has a 'View' button. Below this is a 'Recommended Documents' section with one item: 'Physics Lab Report' (author: user1, views: 80, uploaded: 2024-12-02T14:00), with a 'View' button. At the bottom of the page are links for Company, Contact & Help, Legal, and Apps & Reviews.

This screenshot shows the BON user home page with a different set of recommended documents. The sidebar and top navigation are identical to the previous screenshot. The 'Recommended Documents' section now features a single item: 'Physics Lab Report' (author: user1, views: 80, uploaded: 2024-12-02T14:00). Below this are three boxes showing monthly statistics: 'You've uploaded this month: 19 documents', 'You've earned this month: 0.02 \$', and 'Followers increased this month: 0 people'. The bottom navigation links are also present.

Hình 55: Trang user home

2.2. Trang my documents

Hiển thị các tài liệu do người dùng upload

The screenshot shows the BON platform interface. On the left, there is a sidebar with a blue header "BON". Below it, the user's profile information is displayed: "Nguyen Thang Loi", "0 Followers | 20 Uploads", and a green "+ Upload" button. The sidebar also contains links for Home, My Docs, My Profile, Setting, Subscription, Recent, and Recommend. The main content area features a search bar at the top with the placeholder "Search for documents...". Below the search bar, three uploaded documents are listed in cards:

- Test011**: Author: Nguyen Thang Loi, Views: 4, Uploaded: 2024-12-09T14:17:50.459574. Buttons: View (green), Delete (red).
- test003**: Author: Nguyen Thang Loi, Views: 11, Uploaded: 2024-12-09T14:23:10.568096. Buttons: View (green), Delete (red).
- VipDoc**: Author: Nguyen Thang Loi, Views: 5, Uploaded: 2024-12-09T14:28:23.376581. Buttons: View (green), Delete (red).

Below the cards, there is a small thumbnail image of a technical diagram or graph.

Hình 56: Trang my documents

2.3. Trang my profile

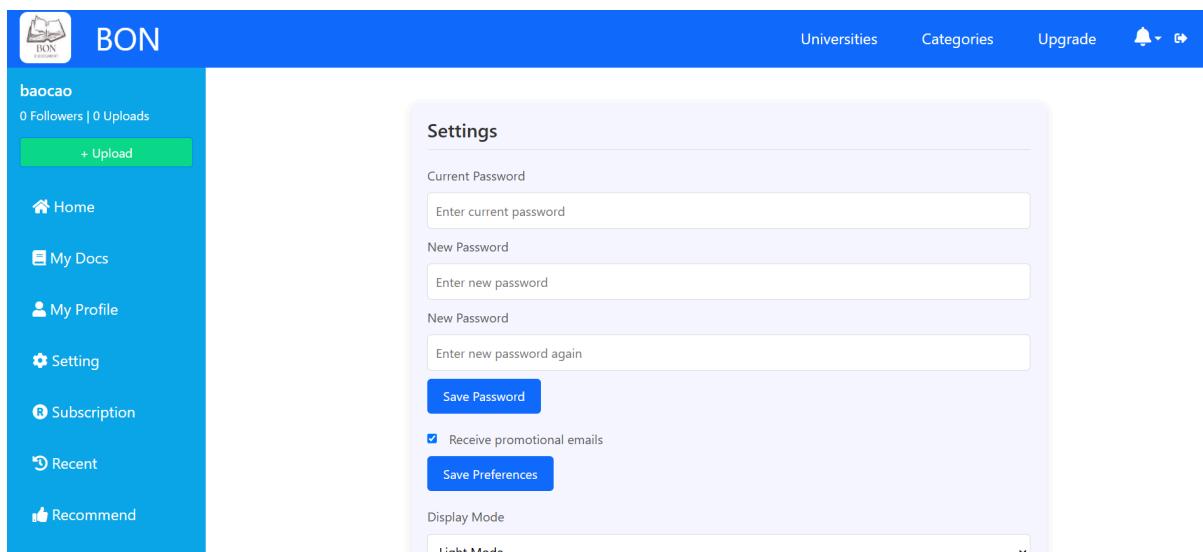
Hiển thị các thông tin về người dùng

The screenshot shows the BON platform interface. On the left, there is a sidebar with a blue header "BON". Below it, the user's profile information is displayed: "baocao", "0 Followers | 0 Uploads", and a green "+ Upload" button. The sidebar also contains links for Home, My Docs, My Profile, Setting, Subscription, Recent, and Recommend. The main content area features a circular profile picture of four people. Below the picture, the user's name "baocao" and a welcome message "Xin chào thế giới!" are displayed. Information such as Followers: 0, Uploaded Documents: 0, and Contact: baocao@baocao.com is shown. There is a "Edit Profile" button. At the bottom of the page, there are links for Company (About Us), Contact & Help (F.A.Q., Contact), Legal (Terms, Privacy Policy), and Apps & Reviews (Excellent 4.5 stars, 7,562 reviews).

Hình 57: Trang my profile

2.3. Trang setting

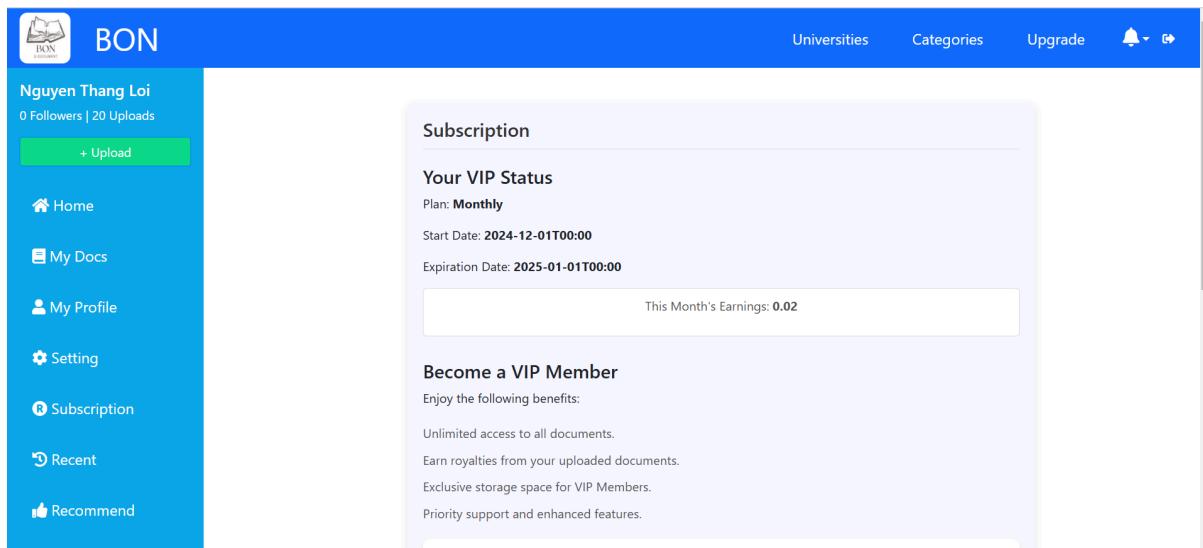
Nơi thiết lập các cấu hình tài khoản và website

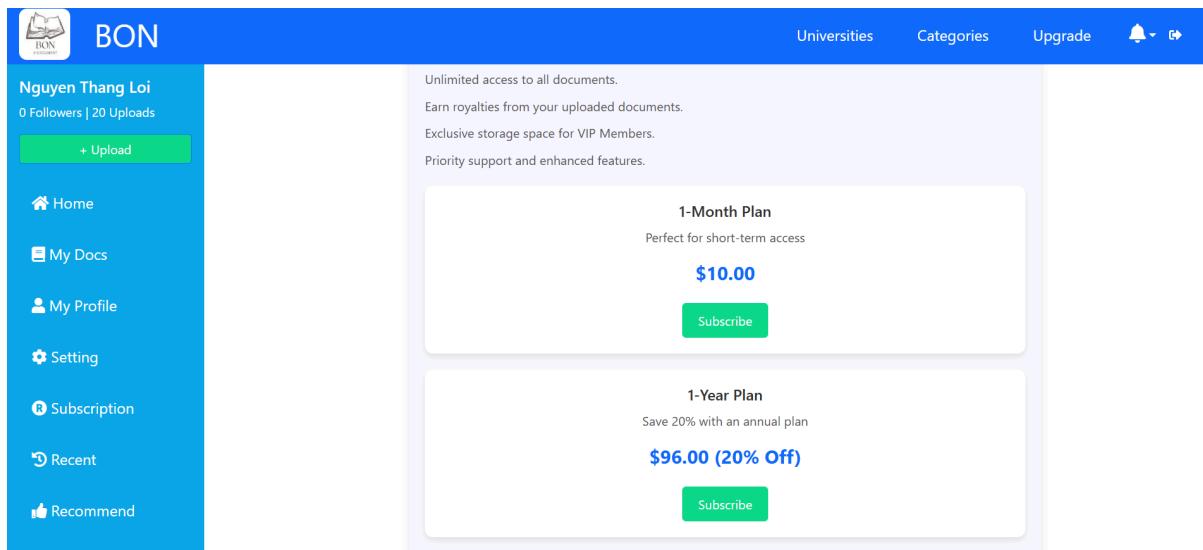


Hình 58: Trang cài đặt

2.2. Trang vip member

Hiển thị trạng thái thành viên: non-vip, vip 1M, vip 1Y

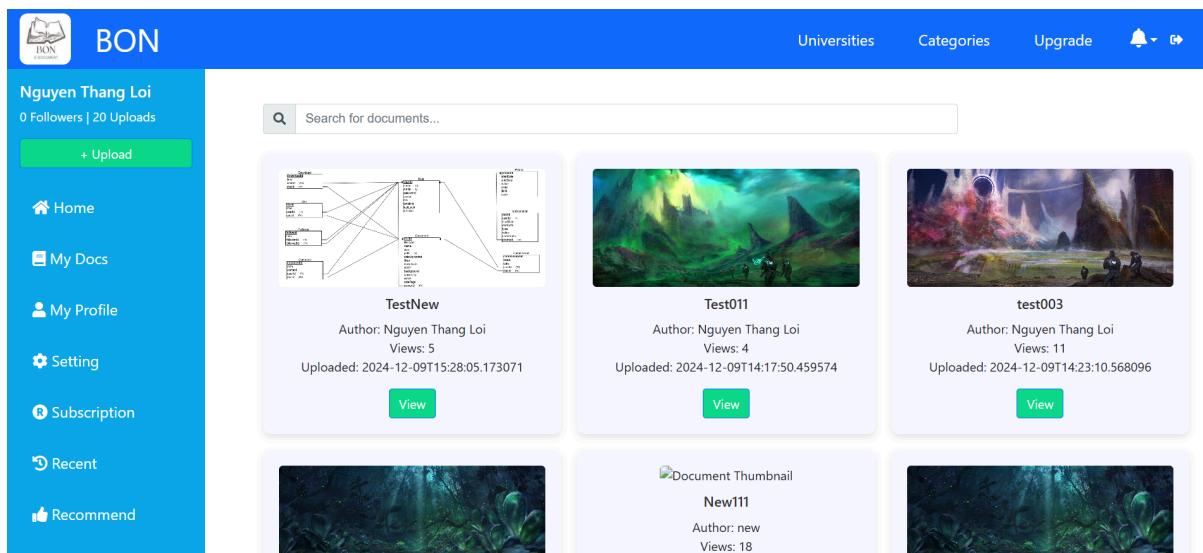




Hình 59: Trang vip member

2.6. Trang recent

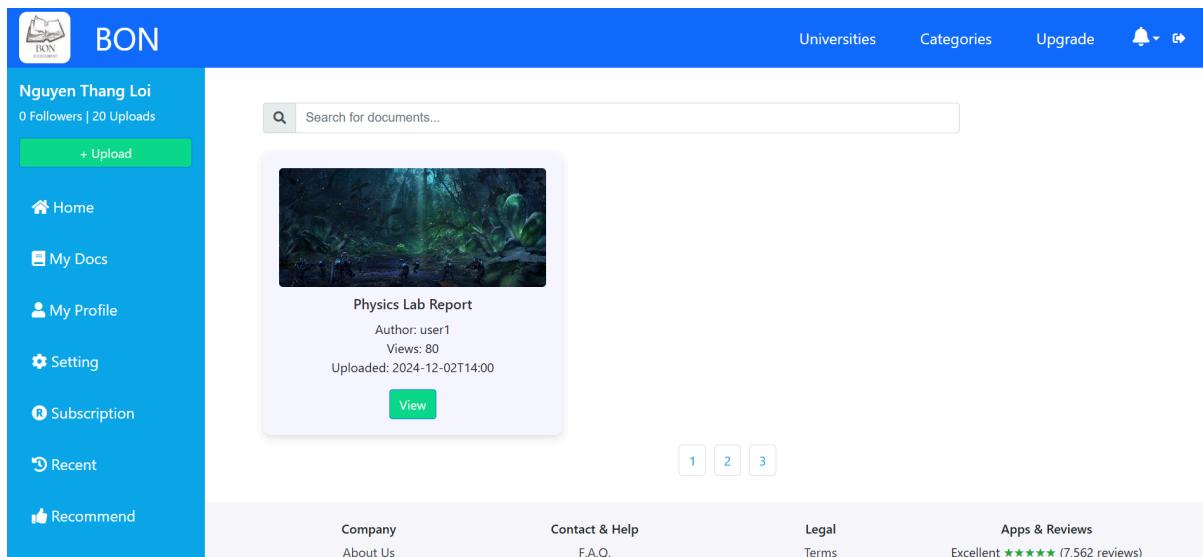
Hiển thị các tài liệu đã xem gần đây theo thời gian



Hình 60: Trang tài liệu gần đây

2.7. Trang recommend

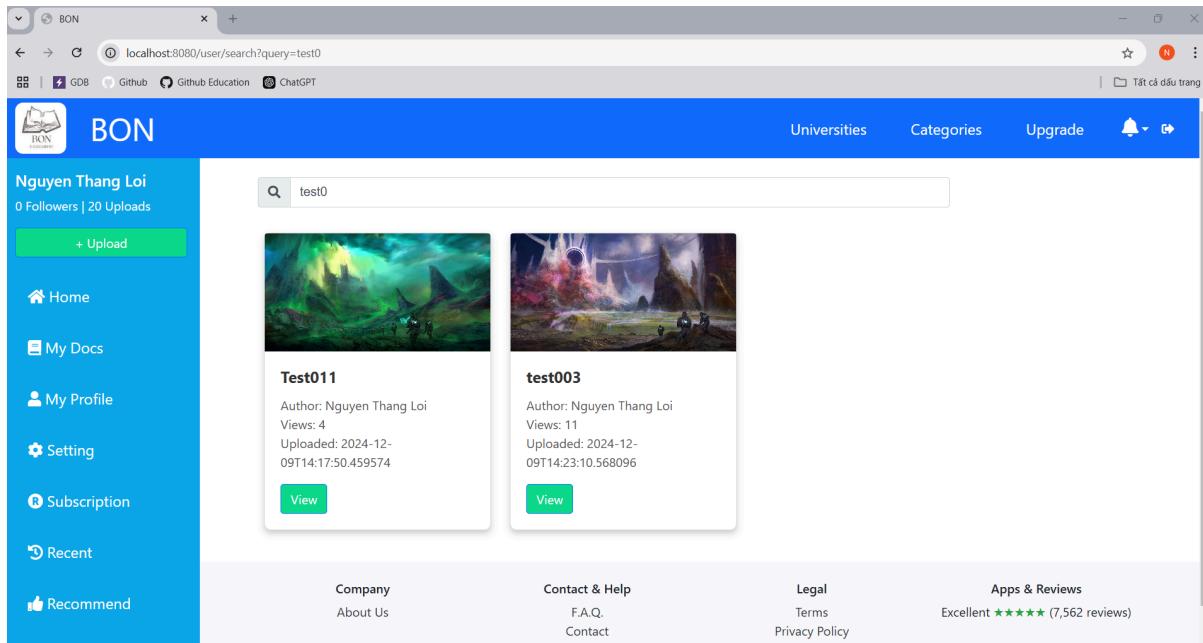
Hiển thị các tài liệu được đề xuất cho người dùng



Hình 61: Trang tài liệu được đề xuất

2.8. Trang tìm kiếm

Hiển thị kết quả tìm kiếm



Hình 62: Trang tìm kiếm

2.9. Các trang danh mục

Tìm kiếm và hiển thị các tài liệu theo trường

The screenshot shows the BON platform's user interface. On the left, there is a sidebar with a profile picture for 'Nguyen Thang Loi' (0 Followers | 20 Uploads) and a green '+ Upload' button. Below this are links for Home, My Docs, My Profile, Setting, Subscription, Recent, and Recommend. At the top right, there are buttons for Universities, Categories, Upgrade, and a notification icon. In the center, there is a search bar with the placeholder 'Enter university...' and three blue circular buttons labeled 'HCMUTE', 'HUST', and 'UEH'. Below the search bar, there are links for Company (About Us), Contact & Help (F.A.Q., Contact), Legal (Terms, Privacy Policy), and Apps & Reviews (Excellent ★★★★ (7,562 reviews)).

This screenshot shows the same BON platform interface, but the search results are now displayed. The search bar now contains 'HUST'. Below it, the results section is titled 'Documents from:' and shows two document thumbnails. The first document is titled 'test003' and was uploaded by 'Nguyen Thang Loi' on 09/14/23 at 10:56:09. The second document is titled 'VipDoc' and was uploaded by 'Nguyen Thang Loi' on 09/14/23 at 23:37:56. Both documents have a 'View' button below them.

Hình 63: Trang danh mục theo trường đại học

Tìm kiếm và hiển thị theo hashtag thẻ loại

The screenshot shows a user profile for 'Nguyen Thang Loi' with 0 Followers and 20 Uploads. The sidebar includes links for Home, My Docs, My Profile, Setting, Subscription, Recent, and Recommend. A search bar at the top right contains the hashtag '#p'. Below it, two document cards are displayed:

- Physics Lab Report** by user1 (Views: 80, Uploaded: 2024-12-02T14:00) with a 'View' button.
- New11** by new (Views: 18, Uploaded: 2024-12-09T00:41:14.516307) with a 'View' button.

Hình 64: Trang danh mục theo hash tag

2.10. Trang upload

Nơi đăng tải các tài liệu

The screenshot shows the 'Upload Document' form. The sidebar is identical to the previous screenshot. The main form fields are:

- Select Document (PDF/DOCX only): Chọn tệp | Không có tệp nào được chọn
- Upload Thumbnail: Chọn tệp | Không có tệp nào được chọn
- Document Title: Enter document title
- Author Name: Nguyen Thang Loi
- Categories (Hashtags): Add a tag

Hình 65: Trang đăng tải

2.11. Trang xem tài liệu

Hiển thị nội dung tài liệu trong frame, like, download và comment về tài liệu

The screenshot displays a web application interface with a blue header bar. The header includes a logo, the text "BON", and navigation links for "Universities", "Categories", "Upgrade", and a notification bell. Below the header, a sidebar on the left shows a user profile for "guyen Thang Loi" with 20 uploads, and a list of menu items: Home, My Docs, My Profile, Setting, Subscription, Recent, and Recommend. The main content area shows a document titled "BỘ GIÁO DỤC VÀ ĐÀO TẠO" from the "HIỆP HỘI AN TOÀN THÔNG TIN VIỆT NAM" and the "CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM". The document is dated "Hà Nội, ngày 06 tháng 12 năm 2024" and discusses the ASEAN Cyber Shield competition. Below the document, there is a list of requirements:

2. Any updates or announcements regarding the education will be posted on the official website and communicated via email during the competition.

At the bottom of the document area, there is a note: "Documents are submitted to participate in the competition. Rules and Limitations apply throughout".

Below the document preview, there is a section for the document "test003" by "guyen Thang Loi". It shows the author's name, upload date (2024-12-09T14:23:10.568096), and university/institute (HUST). The document has 13 views, 1 like, and 0 downloads. There are buttons for "Like" and "Download", and tags "#Science", "#Technology", and "#Math". A green button labeled "Edit Document" is also present.

The next section is titled "Comments" and contains a comment from "guyen Thang Loi" dated 2024-12-10T14:21:00.931343, which simply says "hello".

Hình 66: Trang xem chi tiết

V. Kết luận và hướng phát triển

1. Kết quả đạt được

Đề tài "Xây dựng website tài liệu điện tử" đã đạt được nhiều kết quả đáng ghi nhận. Hệ thống cơ sở dữ liệu MySQL được thiết kế đầy đủ và tối ưu, hỗ trợ quản lý người dùng, tài liệu, trạng thái VIP, cũng như các thông tin về lượt tải xuống, lượt xem, và bình luận. Phần backend được xây dựng trên nền tảng Spring Boot với API RESTful, đảm bảo bảo mật và tính năng xác thực thông qua Spring Security.

Giao diện người dùng được phát triển bằng Thymeleaf và Bootstrap, mang lại trải nghiệm hiện đại, thân thiện và tương thích tốt trên nhiều thiết bị. Các trang chính như Home, Recent Documents, Settings, Subscription, và Upload Document được hoàn thiện với thiết kế trực quan và dễ sử dụng. Hệ thống hỗ trợ đầy đủ chức năng quản lý tài liệu, từ upload, chỉnh sửa, tìm kiếm, đến phân loại theo hashtag và trường đại học, giúp người dùng dễ dàng tiếp cận nội dung phù hợp.

Một điểm nổi bật là tính năng tích hợp thanh toán, cho phép người dùng đăng ký gói VIP thông qua QR Code hoặc thẻ ngân hàng. Gói VIP mang lại các quyền lợi như không giới hạn xem và tải tài liệu, kiếm tiền từ tài liệu tải lên, và không gian lưu trữ lớn hơn. Ngoài ra, trang Admin cung cấp công cụ quản lý người dùng, tài liệu, và thống kê, trong khi trang Vendor hỗ trợ quản lý shop, sản phẩm, đơn hàng và khuyến mãi.

Hệ thống ứng dụng công nghệ hiện đại như Spring Boot, JPA và Bootstrap để tối ưu hiệu năng và nâng cao trải nghiệm. Kết quả đạt được là một website ổn định, trực quan, đáp ứng tốt nhu cầu chia sẻ kiến thức, học tập, và kiếm thu nhập từ tài liệu, đồng thời mở ra tiềm năng mở rộng với các tính năng và nền tảng mới trong tương lai.

2. Hướng phát triển

Hướng phát triển của website tài liệu điện tử này tập trung vào việc nâng cao trải nghiệm người dùng, mở rộng tính năng và áp dụng các công nghệ tiên tiến. Một trong những mục tiêu chính là phát triển ứng dụng di động trên cả iOS và Android, giúp người dùng dễ dàng truy cập tài liệu mọi lúc, mọi nơi. Đồng thời, hệ thống sẽ tích hợp trí tuệ nhân tạo (AI) để cải thiện tính năng đề xuất tài liệu dựa trên sở thích và hành vi của người dùng.

Hệ thống cũng có thể mở rộng sang hỗ trợ các loại tài liệu đa dạng hơn, như video bài giảng hoặc sách điện tử, và phát triển tính năng học nhóm trực tuyến. Để thúc đẩy cộng đồng học tập, website có thể bổ sung hệ thống gamification, trao huy hiệu và điểm thưởng cho các hoạt động tích cực như chia sẻ tài liệu hoặc đóng góp bình luận.

Về mặt kỹ thuật, việc triển khai cơ sở hạ tầng trên nền tảng cloud sẽ đảm bảo khả năng mở rộng và hiệu suất ổn định khi số lượng người dùng tăng lên. Hệ thống thanh toán cũng có thể tích hợp thêm các cổng thanh toán quốc tế như PayPal hoặc ví điện tử, phục vụ người dùng ở nhiều quốc gia. Cuối cùng, để hỗ trợ tốt hơn cho người bán (Vendor), website có thể phát triển các công cụ phân tích doanh thu, hiệu quả bán hàng và thiết lập chiến dịch marketing tự động. Với các định hướng này, hệ thống sẽ trở thành một nền tảng toàn diện, đáp ứng nhu cầu ngày càng đa dạng của người dùng.