TRƯỜNG ĐẠI HỌC TRÀ VINH
TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ
KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN KẾT THÚC HỌC PHẦN

CÔNG NGHỆ PHẦN MỀM

(MSHP: 220055)

TÊN ĐỀ TÀI

XÂY DỰNG ỨNG DỤNG WEB QUẢN LÝ THƯ VIỆN

Sinh viên thực hiện:

110122227	Phan Đăng Khoa	DA22TTB
110122142	Trần Trung Phúc	DA22TTB
110122107	Hồ Hoàng Long	DA22TTB

Giáo viên hướng dẫn: TS. Nguyễn Bảo Ân

Trà Vinh, tháng 07 năm 2025

TRƯỜNG ĐẠI HỌC TRÀ VINH

TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ

KHOA CÔNG NGHỆ THÔNG TIN

ĐỒ ÁN KẾT THÚC HỌC PHẦN

CÔNG NGHỆ PHẦN MỀM

(MSHP: 220055)

TÊN ĐỀ TÀI

XÂY DỰNG ỨNG DỤNG WEB QUẢN LÝ THƯ VIỆN

Sinh viên thực hiện:

110122227	Phan Đăng Khoa	DA22TTB
110122142	Trần Trung Phúc	DA22TTB
110122107	Hồ Hoàng Long	DA22TTB

Giáo viên hướng dẫn: TS. Nguyễn Bảo Ân

Trà Vinh, tháng 07 năm 2025

LỜI CẢM ƠN

Để thực hiện và hoàn thành tốt báo cáo bài tập lớn này, chúng em đã nhận được sự giúp đỡ và hướng dẫn rất tận tình của Thầy Nguyễn Bảo Ân, Thầy đã cung cấp cho chúng em các thông tin, kiến thức vô cùng quý báu và cần thiết trong suốt thời gian qua để chúng em có thể thực hiện và hoàn thành báo cáo của mình.

Chúng em xin chân thành cảm ơn các bạn trong ngành công nghệ thông tin đã ủng hộ, giúp đỡ, chia sẻ kiến thức, kinh nghiệm và tài liệu có được giúp chúng em trong quá trình nghiên cứu và thực hiện báo cáo.

Mặc dù đã cố gắng hết sức nhưng do kiến thức và kinh nghiệm còn hạn chế, do đó bài báo cáo này khó tránh khỏi những thiếu sót, kính mong nhận được sự chỉ dẫn của Thầy để chúng em củng cố, hoàn thiện kiến thức của mình hơn.

Cuối cùng, chúng em xin chúc Thầy luôn thành công và dồi dào sức khỏe để có thể tiếp tục sự nghiệp truyền đạt kiến thức cho thế hệ mai sau.

Chúng em xin chân thành cảm ơn !

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

MỤC LỤC

DANH MỤC HÌNH ẢNH

CHƯƠNG 1: TỔNG QUAN

1.1 Lý do chọn đề tài

Trong thời đại công nghệ số phát triển mạnh mẽ, việc ứng dụng công nghệ thông tin vào quản lý dữ liệu và quy trình làm việc tại các cơ quan, đơn vị hành chính, trường học là xu hướng tất yếu nhằm nâng cao hiệu suất, tiết kiệm thời gian và nguồn lực. Một trong những lĩnh vực vẫn còn tồn tại nhiều bất cập trong quy trình quản lý truyền thống chính là hệ thống thư viện. Hầu hết các thư viện hiện nay, đặc biệt tại các trường học hoặc cơ sở giáo dục quy mô nhỏ, vẫn đang vận hành dựa trên phương pháp thủ công như ghi chép tay, sử dụng Excel đơn lẻ hoặc chưa có hệ thống quản lý dữ liệu tập trung. Điều này dẫn đến nhiều khó khăn trong việc kiểm kê sách, tra cứu thông tin, thống kê số liệu, quản lý người mượn, theo dõi tình trạng mượn/trả sách, v.v. Nhận thấy được nhu cầu thực tiễn và tính ứng dụng cao của một hệ thống quản lý thư viện số, nhóm quyết định lựa chọn đề tài "Xây dựng ứng dụng web quản lý thư viện" nhằm mang đến giải pháp thay thế hiệu quả cho phương pháp quản lý truyền thống.

Đồng thời, đề tài cũng cho phép nhóm áp dụng toàn bộ các công nghệ theo yêu cầu của đồ án như kiến trúc client/server, giao tiếp qua API, tích hợp CI/CD, quản lý dự án qua Jira, kiểm thử bằng Postman, triển khai Docker,... tạo điều kiện thuận lợi để nhóm tương tác thực với quy trình phát triển phần mềm như trong môi trường doanh nghiệp thực tế.

1.2 Mục tiêu

Mục tiêu của đề tài "Xây dựng ứng dụng web quản lý thư viện" là thiết kế và phát triển một hệ thống ứng dụng web hoàn chỉnh, hỗ trợ các chức năng quản lý sách, quản lý người dùng, tra cứu sách, theo dõi tình trạng mượn và trả sách một cách trực tuyến, tiện lợi và chính xác. Ứng dụng phải được xây dựng dựa trên kiến trúc hiện đại, tách biệt frontend và backend theo mô hình client/server rõ ràng, đảm bảo khả năng mở rộng, bảo trì lâu dài và dễ tích hợp về sau. Giao diện người dùng được thiết kế bằng Figma với mục tiêu mang lại trải nghiệm trực quan, thân thiện và dễ sử dụng cho cả người quản trị lẫn người dùng cuối. Phần frontend sử dụng framework Vue.js để phát triển, tận dụng tính linh hoạt và khả năng tái sử dụng thành phần UI. Backend được xây dựng để cung cấp các API RESTful, phục vụ

giao tiếp với frontend và tương tác cơ sở dữ liệu. Hệ thống còn được kiểm thử API bằng Postman, tích hợp CI/CD qua GitHub Actions giúp tự động hoá quá trình triển khai và kiểm thử mã nguồn. Bên cạnh đó, Docker được sử dụng để đóng gói ứng dụng, đảm bảo tính nhất quán trong môi trường chạy thực tế. Toàn bộ quy trình phát triển được tổ chức chặt chẽ qua công cụ Jira nhằm chia nhỏ công việc, theo dõi tiến độ, và hỗ trợ làm việc nhóm hiệu quả. Thông qua việc thực hiện đề tài, nhóm không chỉ hoàn thành sản phẩm có giá trị ứng dụng thực tế, mà còn rèn luyện được kỹ năng lập trình fullstack, kỹ năng làm việc nhóm, sử dụng công cụ chuyên nghiệp, cũng như tư duy giải quyết vấn đề trong quá trình triển khai một dự án phần mềm thực tế.

CHƯƠNG 2: KHẢO SÁT HIỆN TRẠNG VÀ XÁC ĐỊNH YÊU CẦU

2.1 Các yêu cầu chức năng của hệ thống

2.1.1 Người dùng (Độc giả)

2.1.1.1 Đăng ký tài khoản mới

- Người dùng truy cập trang đăng ký, nhập các thông tin: tên đăng nhập, mật khẩu, xác nhận mật khẩu, họ tên, email, địa chỉ.

- Hệ thống kiểm tra hợp lệ dữ liệu: không để trống, mật khẩu nhập lại phải khớp, email đúng định dạng, tên đăng nhập/email không bị trùng với tài khoản đã có.

- Khi hợp lệ, gửi yêu cầu đăng ký lên server. Server kiểm tra lại, lưu thông tin vào cơ sở dữ liệu, gán quyền mặc định là "độc giả".

- Nếu đăng ký thành công, thông báo cho người dùng và chuyển hướng sang trang đăng nhập. Nếu thất bại (trùng username/email, lỗi server), hiển thị thông báo lỗi cụ thể.

2.1.1.2 Đăng nhập hệ thống

- Người dùng nhập tên đăng nhập và mật khẩu trên trang đăng nhập.

- Hệ thống gửi thông tin lên server để xác thực.

- Nếu đúng, lưu thông tin người dùng vào localStorage, chuyển đến giao diện người dùng (UserHome). Nếu sai, hiển thị thông báo lỗi (sai mật khẩu, tài khoản không tồn tại).

2.1.1.3 Tìm kiếm sách

- Người dùng truy cập trang tìm kiếm sách, nhập từ khóa (tên sách, tác giả), chọn thể loại, trạng thái (có sẵn/đang mượn).

- Hệ thống gửi yêu cầu lên server, nhận về danh sách sách phù hợp.

- Hiển thị danh sách sách với các thông tin: mã sách, tên sách, tác giả, năm xuất bản, ngôn ngữ, thể loại, trạng thái, số lượng còn lại.

- Người dùng có thể nhấn vào tên sách để xem chi tiết.

2.1.1.4 Xem thông tin chi tiết sách

- Khi chọn một sách, hệ thống hiển thị đầy đủ thông tin: mã sách, tên sách, tác giả, năm xuất bản, ngôn ngữ, thể loại, trạng thái.

- Có thể xem trạng thái mượn/trả, lịch sử mượn sách.

2.1.1.5 Gửi yêu cầu hỗ trợ

- Người dùng truy cập trang hỗ trợ, nhập tiêu đề và nội dung yêu cầu hỗ trợ (ví dụ: lỗi đăng nhập, thắc mắc về sách, yêu cầu mượn sách).

- Hệ thống gửi yêu cầu lên server, lưu lại yêu cầu và phản hồi cho người dùng biết đã gửi thành công hay thất bại.

2.1.1.6 Quản lý thông tin cá nhân

- Người dùng có thể xem và cập nhật thông tin cá nhân: họ tên, email, địa chỉ.

- Khi cập nhật, hệ thống kiểm tra hợp lệ dữ liệu, gửi yêu cầu lên server để cập nhật vào cơ sở dữ liệu.

- Nếu thành công, thông báo cho người dùng; nếu thất bại, hiển thị lỗi.

2.1.1.7 Đăng xuất

- Người dùng nhấn nút đăng xuất, hệ thống xóa thông tin đăng nhập khỏi localStorage.

- Chuyển về trang đăng nhập, đảm bảo bảo mật thông tin cá nhân.

2.1.2 Quản trị viên (Admin)

2.1.2.1 Đăng nhập hệ thống

- Quản trị viên nhập tên đăng nhập, mật khẩu trên trang đăng nhập.

- Hệ thống xác thực thông tin, kiểm tra quyền (loại user = 1).

- Nếu đúng, chuyển đến giao diện quản trị (AdminHome), hiển thị các chức năng quản lý.

2.1.2.2 Quản lý người dùng

- Quản trị viên truy cập trang quản lý người dùng, xem danh sách tất cả người dùng (trừ mật khẩu).

- Có thể tìm kiếm, lọc theo loại người dùng (độc giả/quản trị).

- Thực hiện cập nhật thông tin người dùng: họ tên, email, địa chỉ. Gửi yêu cầu lên server, cập nhật vào cơ sở dữ liệu.

- Xóa người dùng: chọn người dùng cần xóa, xác nhận thao tác, hệ thống xóa khỏi cơ sở dữ liệu.

2.1.2.3 Xem danh sách độc giả

- Quản trị viên có thể lọc và xem danh sách người dùng.

- Hiển thị thông tin chi tiết từng độc giả, hỗ trợ tìm kiếm, lọc theo tên, email, địa chỉ.

2.1.2.4 Thống kê tổng quan

- Quản trị viên xem số lượng tổng quan: tổng số người dùng, tổng số sách, tổng số lượt mượn, tổng số danh mục sách.

- Dữ liệu được hiển thị dạng bảng hoặc biểu đồ trực quan (sử dụng chart.js).

2.1.2.5 Thống kê đăng ký theo tháng

- Hiển thị số lượng người dùng đăng ký mới từng tháng trong năm.

- Dữ liệu giúp quản trị viên đánh giá mức độ sử dụng hệ thống, xu hướng đăng ký.

2.1.2.6 Thống kê mượn sách theo tháng

- Hiển thị số lượng lượt mượn sách từng tháng trong năm.

- Quản trị viên có thể phân tích nhu cầu mượn sách, thời điểm cao điểm.

2.1.2.7 Thống kê sách theo danh mục

- Hiển thị phân bố số lượng sách theo từng danh mục (ví dụ: Khoa học, Văn học, Công nghệ…).

- Dữ liệu giúp quản trị viên cân đối, bổ sung sách cho các danh mục thiếu.

2.1.2.8 Quản lý sách

- Thêm mới sách: nhập thông tin sách (tên, tác giả, năm xuất bản, thể loại, trạng thái, số lượng), gửi lên server để lưu vào cơ sở dữ liệu.

- Sửa thông tin sách: cập nhật các trường thông tin, trạng thái, số lượng.

- Xóa sách: chọn sách cần xóa, xác nhận thao tác, hệ thống xóa khỏi cơ sở dữ liệu.

- Quản lý trạng thái sách: cập nhật trạng thái (có sẵn, đang mượn, hết sách).

2.1.2.9 Quản lý yêu cầu hỗ trợ

- Xem danh sách các yêu cầu hỗ trợ do người dùng gửi lên.

- Xem chi tiết từng yêu cầu, phản hồi hoặc xử lý.

- Lưu lại lịch sử xử lý yêu cầu để theo dõi.

2.2 Các yêu cầu phi chức năng của hệ thống

2.2.1 Hiệu năng

- Hệ thống phải đảm bảo thời gian phản hồi cho các thao tác cơ bản (đăng nhập, đăng ký, tìm kiếm sách, xem thông tin sách, gửi yêu cầu hỗ trợ) không vượt quá 4-5 giây trong điều kiện tải bình thường.

- Khi có nhiều người dùng truy cập đồng thời, hệ thống vẫn phải duy trì tốc độ phản hồi ổn định, hạn chế bị treo hoặc chậm đáng kể.

- Các truy vấn dữ liệu lớn phải được tối ưu hóa, sử dụng phân trang (pagination) để giảm tải cho server và client.

- Ứng dụng frontend phải tải nhanh, tối ưu hóa tài nguyên tĩnh (ảnh, JS, CSS), sử dụng cache trình duyệt và CDN nếu triển khai thực tế.

2.2.2 Bảo mật

- Phân quyền rõ ràng: chỉ quản trị viên mới truy cập được các chức năng quản lý, thống kê, xóa/sửa người dùng, sách; người dùng thường không thể truy cập các API này.

- Các API phải kiểm tra xác thực và phân quyền trước khi xử lý yêu cầu.

- Dữ liệu cá nhân (email, địa chỉ, lịch sử mượn sách) phải được bảo vệ, không tiết lộ cho bên thứ ba.

- Ghi log các thao tác quan trọng (đăng nhập, xóa dữ liệu, thay đổi thông tin) để phục vụ kiểm tra bảo mật.

2.2.3 Khả năng mở rộng

- Hệ thống được thiết kế theo mô hình client-server, tách biệt frontend và backend, dễ dàng mở rộng số lượng người dùng, dữ liệu sách, chức năng mới.

- Có thể bổ sung thêm các module mới (quản lý tài liệu số, tích hợp thanh toán, gửi email tự động) mà không ảnh hưởng đến các chức năng hiện tại.

- Cơ sở dữ liệu có thể nâng cấp lên các hệ quản trị mạnh hơn hoặc chuyển sang mô hình phân tán khi số lượng dữ liệu lớn.

2.2.4 Khả năng tương thích

- Giao diện người dùng hoạt động tốt trên các trình duyệt phổ biến: Chrome, Firefox, Edge, Safari.

- Không sử dụng các tính năng trình duyệt đặc thù hoặc các API chưa được chuẩn hóa.

2.2.5 Khả năng sử dụng

- Giao diện thân thiện, dễ sử dụng cho cả người dùng và quản trị viên, các thao tác chính (đăng nhập, tìm kiếm, mượn sách, gửi hỗ trợ) được bố trí rõ ràng.

- Có hướng dẫn sử dụng, tooltip, thông báo trạng thái (thành công/thất bại) rõ ràng.

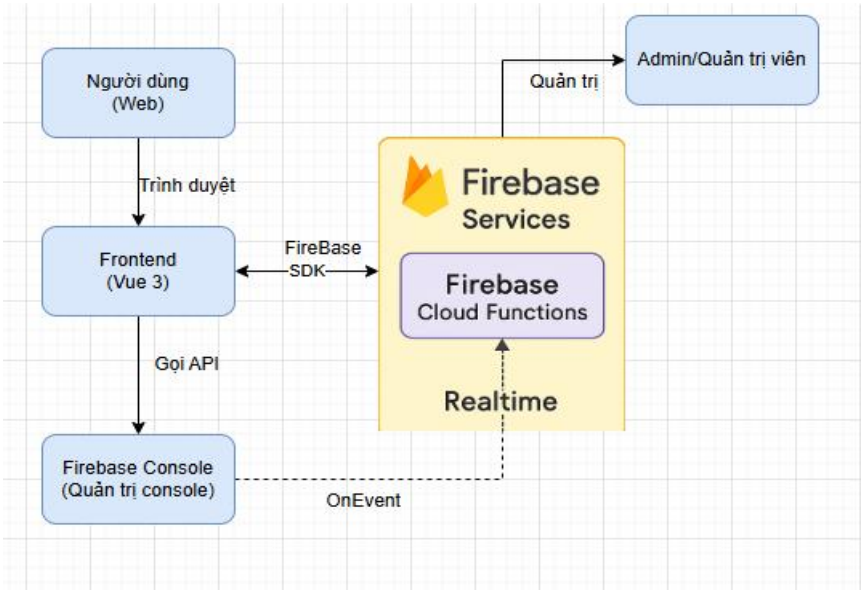
- Các lỗi nhập liệu được kiểm tra và thông báo ngay cho người dùng (ví dụ: sai định dạng email, mật khẩu không khớp).

- Hỗ trợ truy cập nhanh các chức năng thường dùng, giảm số bước thao tác.

CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

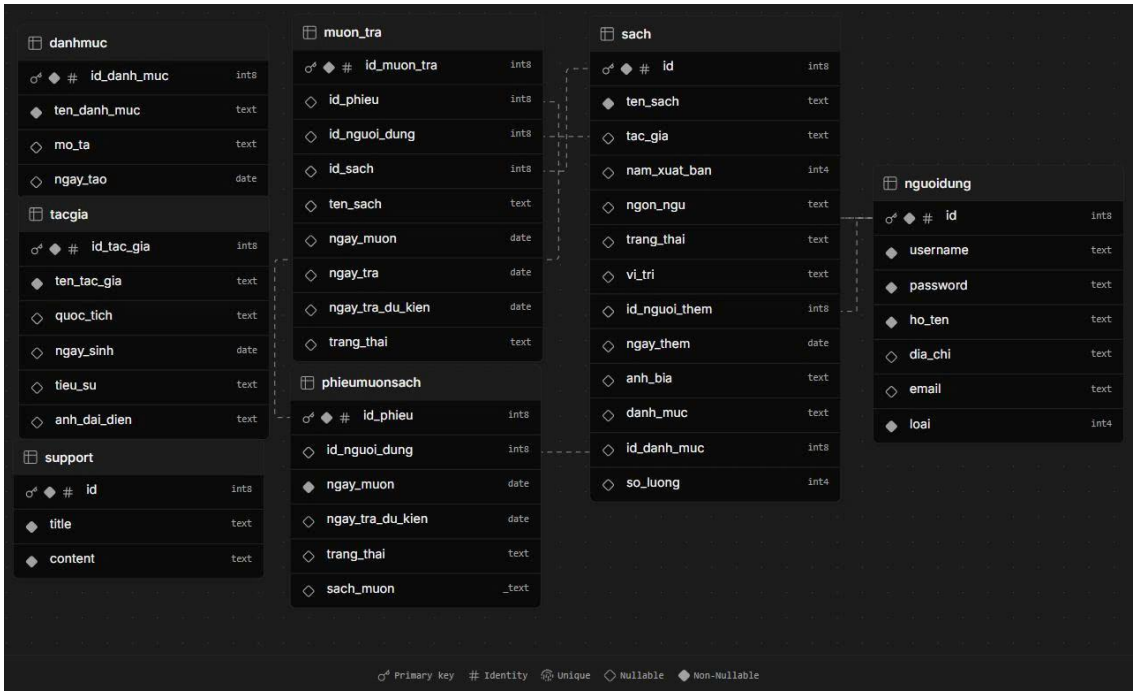
3.1 Thiết kế hệ thống

3.1.1 Kiến trúc tổng thể



Hình 3.1 Sơ đồ kiến trúc hệ thống

3.1.2 Thiết kế cơ sở dữ liệu



Hình 3.2 Sơ đồ dữ liệu quan hệ

3.1.3 Thiết kế API

Một số các endpoint chính:

3.1.3.1 POST /api/login

Mô tả: Xác thực người dùng và trả về token

Request:

```
{
  "username": "dangkhoa",
  "password": "123456"
}
```

Response:

```
{
  "message": "Đăng nhập thành công!",
  "loai": 2,
  "user": {
    "ID": "U002",
    "username": "dangkhoa",
    "ho_ten": "Đăng Khoa",
    "email": "dangkhoa@example.com",
    "dia_chi": "456 Trần Hưng Đạo, Q5, TP.HCM",
    "loai": 2
  }
}
```

3.1.3.2 POST /api/books

Mô tả: Thêm một cuốn sách mới vào hệ thống

Request:

```
{
  "ID": "999",
  "ten_sach": "999 đóa hoa hồng",
  "tac_gia": "Long",
  "nam_xuat_ban": 2025,
  "ngon_ngu": "Tiếng Việt",
  "vi_tri": "Việt Nam",
  "id_nguoi_them": "U001",
  "ngay_them": "22-07-2025",
  "anh_bia":
"https://res.cloudinary.com/ddrdabpn3/image/upload/v1752378127/ID139_xfam9c.jpg",
  "danh_muc": "TV",
  "ID_danh_muc": "DM01",
  "so_luong": 10
}
```


Response:


```
{
  "ID": "999",
  "ten_sach": "999 đóa hoa hồng",
  "tac_gia": "Long",
  "nam_xuat_ban": 2025,
  "ngon_ngu": "Tiếng Việt",
  "trang_thai": "Có sẵn",
  "vi_tri": "Việt Nam",
```

```
  "id_nguoi_them": "U001",
  "ngay_them": "22-07-2025",
  "anh_bia":
"https://res.cloudinary.com/ddrdabpn3/image/upload/v
1752378127/ID139_xfam9c.jpg",
  "danh_muc": "TV",
  "ID_danh_muc": "DM01",
  "so_luong": 10,
  "_id": "687f0eb412cc2352beebb171",
  "__v": 0
}
```


3.1.3.3 GET /api/books

Mô tả: Trả về danh sách tất cả sách trong hệ thống

Response:


```
[
  {
    "_id": "687dfa315f15c6f40f3acfae",
    "ID": "034",
    "ten_sach": "The Industries of the Future (Công
nghiệp tương lai)",
    "tac_gia": "Alec Ross",
    "nam_xuat_ban": 2008,
    "ngon_ngu": "Tiếng Anh",
    "trang_thai": "Có sẵn",
    "vi_tri": "Giá: 810.285 NGX 2014",
    "id_nguoi_them": "ID00123",
```

```
    "ngay_them": "2025-03-02",
    "anh_bia":
"https://res.cloudinary.com/ddrdabpn3/image/upload/v
1751339081/ID034_spejmu.jpg",
    "danh_muc": "Công nghệ thông tin",
    "ID_danh_muc": "DM001",
    "so_luong": 0
  },
  {
    "_id": "687dfa315f15c6f40f3acfaf",
    "ID": "097",
    "ten_sach": "Lập trình và Cuộc sống",
    "tac_gia": "Jeff Atwood",
    "nam_xuat_ban": 2013,
    "ngon_ngu": "Tiếng Việt",
    "trang_thai": "Đang mượn",
    "vi_tri": "Giá: 810.264 NGX 2013",
    "id_nguoi_them": "ID00123",
    "ngay_them": "2025-03-02",
    "anh_bia":
"https://res.cloudinary.com/ddrdabpn3/image/upload/v
1751339082/ID097_azu2zg.jpg",
    "danh_muc": "Công nghệ thông tin",
    "ID_danh_muc": "DM001",
    "so_luong": 1
  }, ...
```

3.1.3.4 GET /api/books/{id}

Mô tả: Lấy thông tin chi tiết của một cuốn sách theo Id

Request:

Response:


```
{
  "_id": "687dfa315f15c6f40f3acfae",
  "ID": "034",
  "ten_sach": "The Industries of the Future (Công nghiệp tương lai)",
  "tac_gia": "Alec Ross",
  "nam_xuat_ban": 2008,
  "ngon_ngu": "Tiếng Anh",
  "trang_thai": "Có sẵn",
  "vi_tri": "Giá: 810.285 NGX 2014",
  "id_nguoi_them": "ID00123",
  "ngay_them": "2025-03-02",
  "anh_bia": "https://res.cloudinary.com/ddrdabpn3/image/upload/v1751339081/ID034_spejmu.jpg",
  "danh_muc": "Công nghệ thông tin",
  "ID_danh_muc": "DM001",
```

```
  "so_luong": 0
}
```

3.1.3.5 POST /api/author

Mô tả: Tạo mới một tác giả để liên kết với sách

Request:

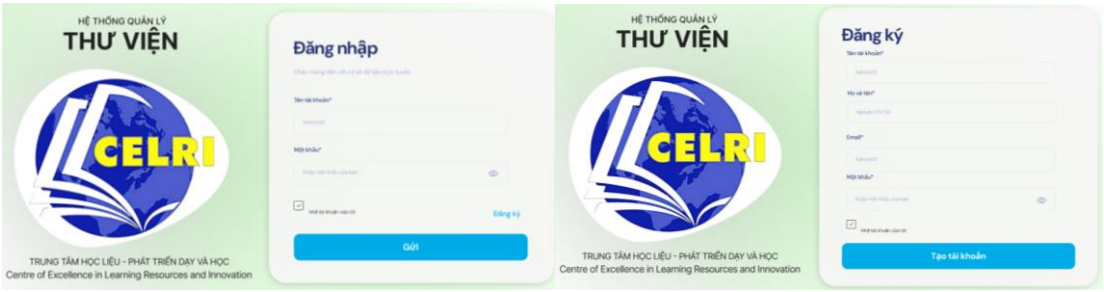
```
{
  "ID_tac_gia": "TG999",
  "ten_tac_gia": "Long999",
  "ngay_sinh": "12-12-2003",
  "quoc_gia": "VN",
  "mo_ta": "0"
}
```

Response:

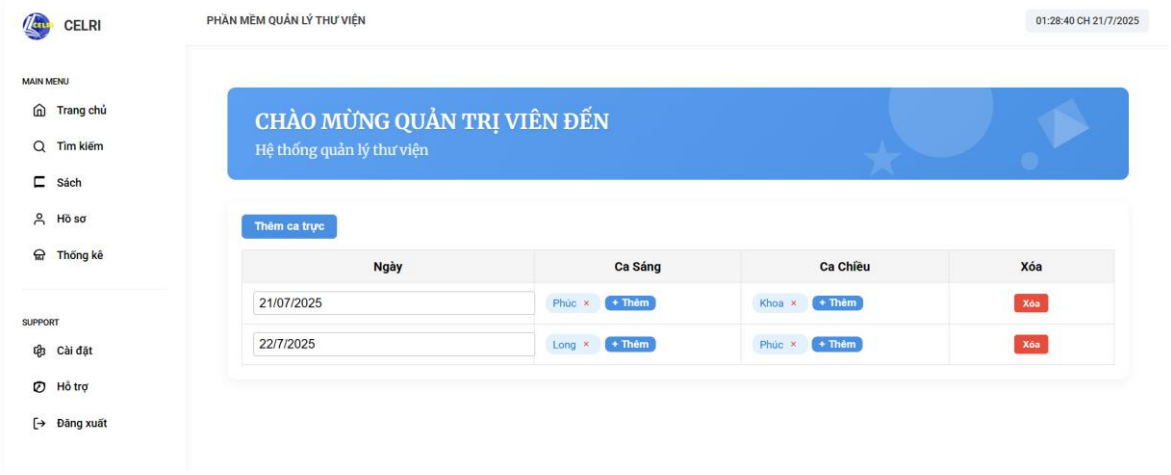
```
{
  "ID_tac_gia": "TG999",
  "ten_tac_gia": "Long999",
  "ngay_sinh": "12-12-2003",
  "_id": "687f15783d886dd94d7b74c6",
  "__v": 0
}
```

3.1.4 Thiết kế giao diện

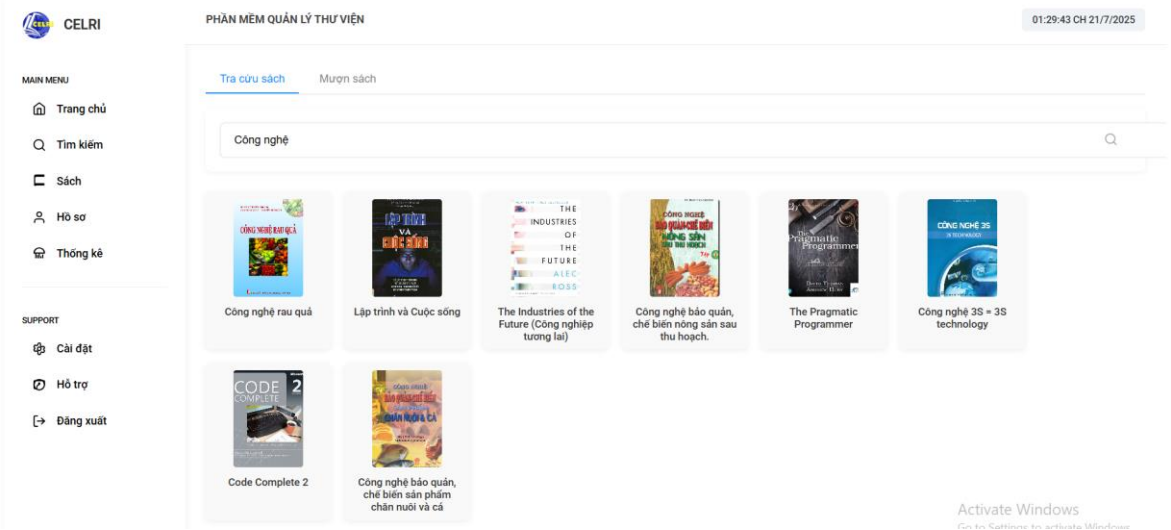
3.1.4.1 Giao diện Người quản trị



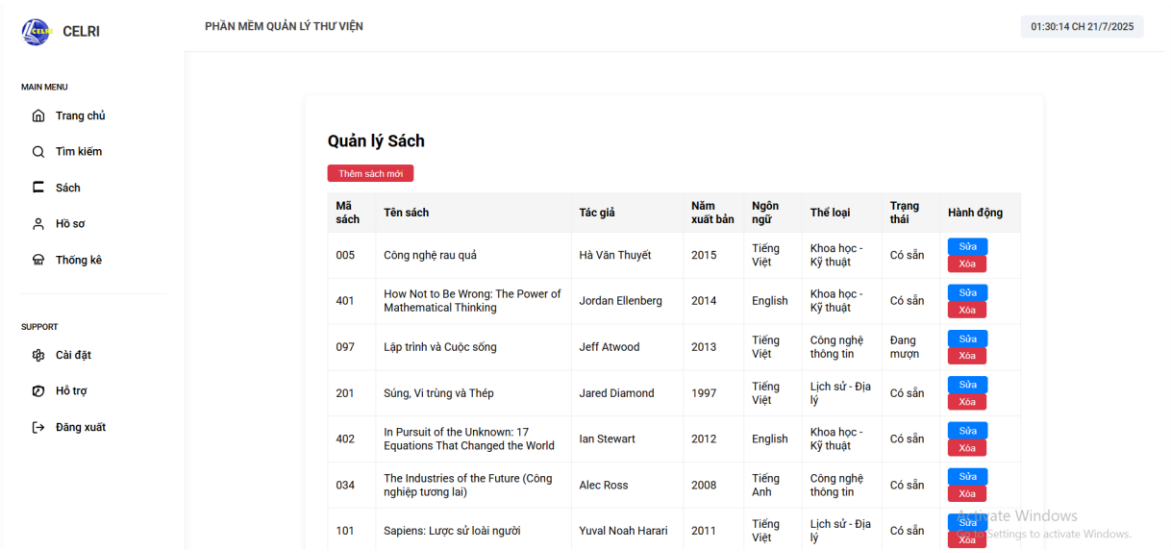
Hình 3.3 Giao diện Đăng nhập và Đăng ký



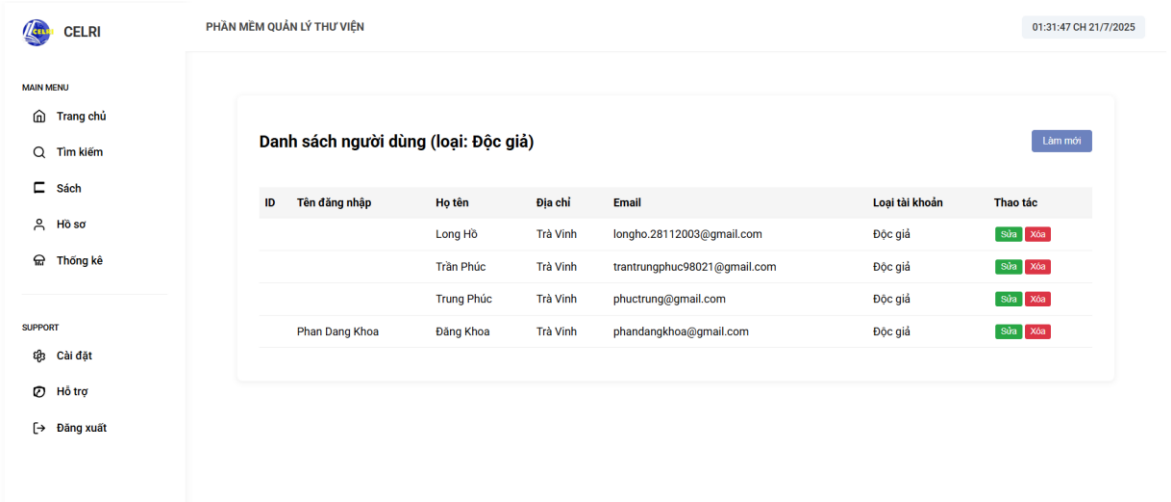
Hình 3.4 Giao diện Trang chủ - quản trị



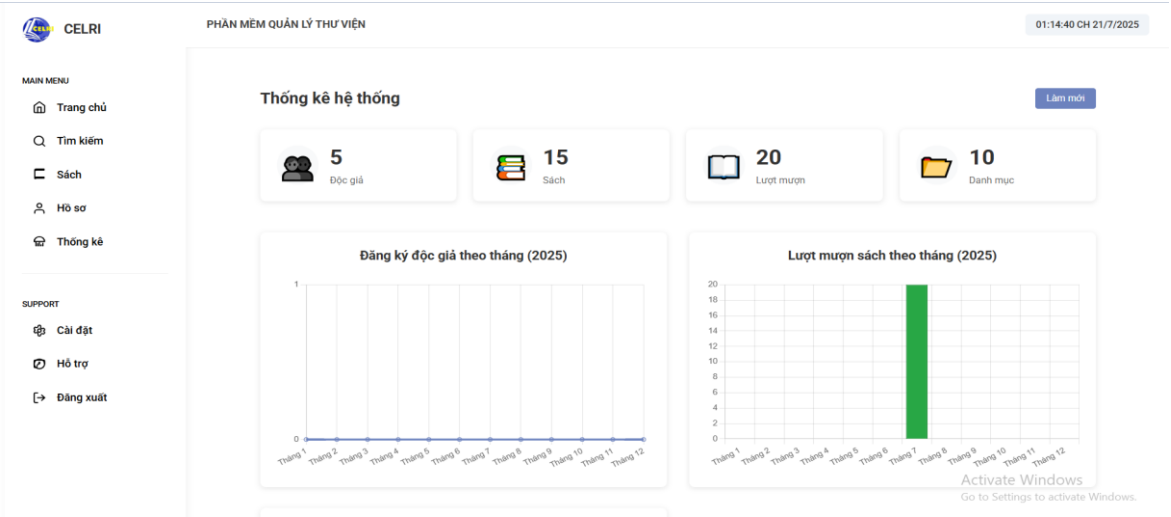
Hình 3.5 Giao diện trang Tìm kiếm - quản trị



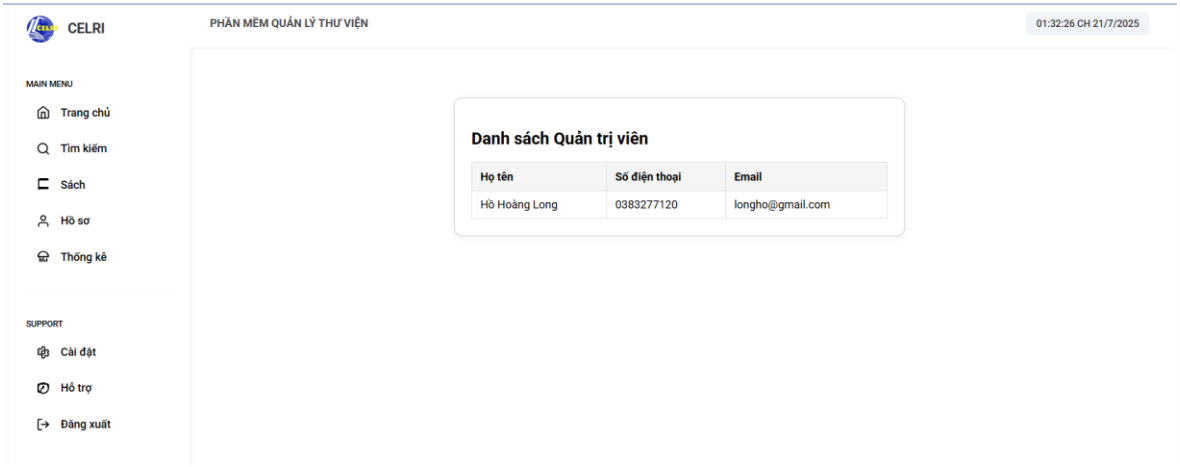
Hình 3.6 Giao diện trang Sách - quản trị



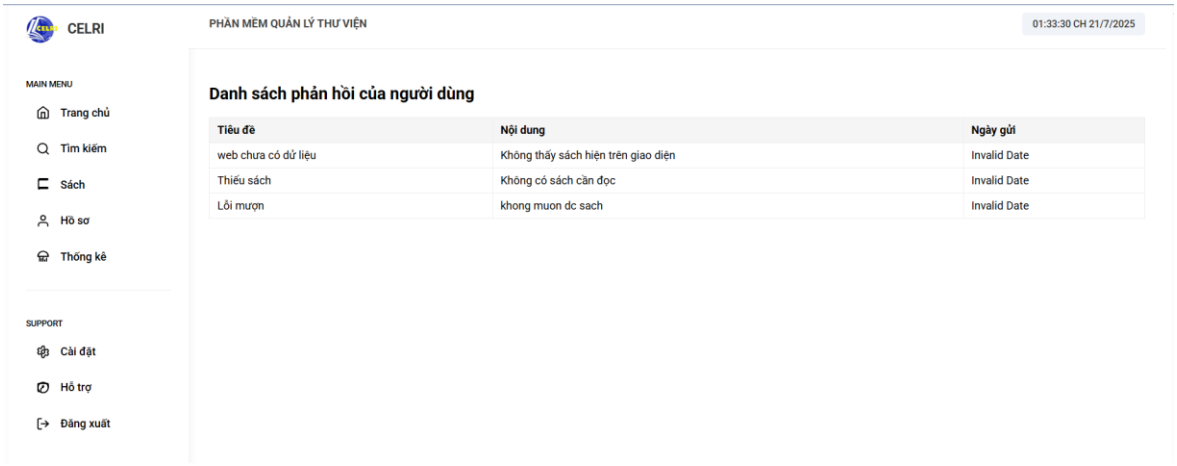
Hình 3.7 Giao diện trang Hồ sơ - quản trị



Hình 3.8 Giao diện trang Thống kê - quản trị

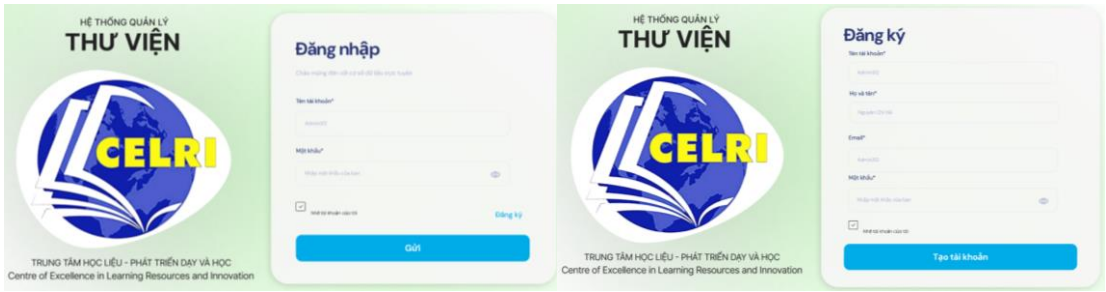


Hình 3.9 Giao diện trang Cài đặt - quản trị



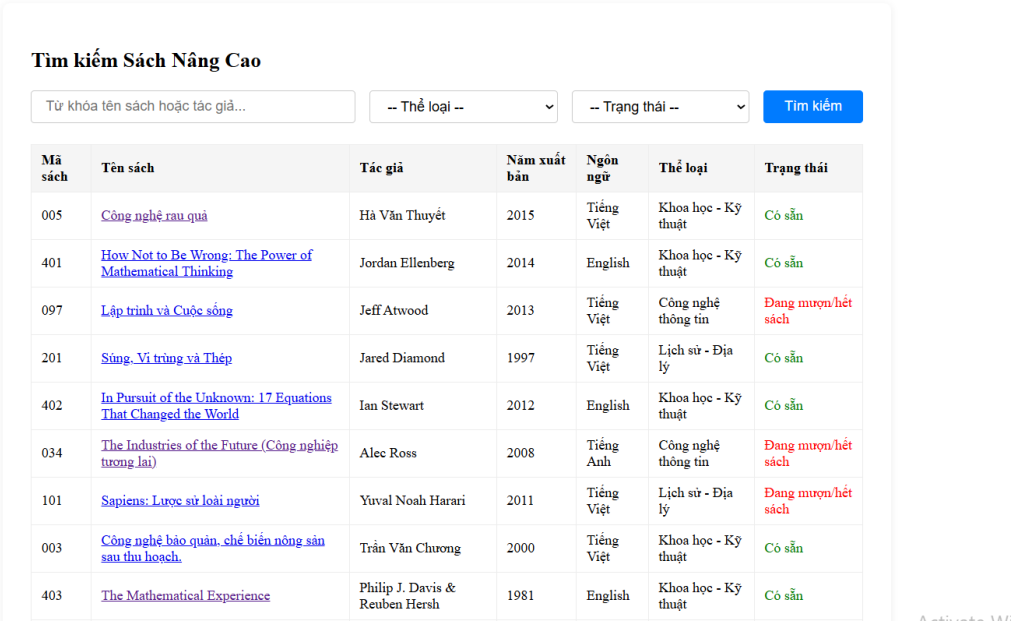
Hình 3.10 Giao diện trang Hỗ trợ - quản trị

3.1.4.2 Giao diện Người dùng



Hình 3.11 Giao diện Đăng nhập và Đăng ký

Hình 3.12 Giao diện Trang chủ - người dùng

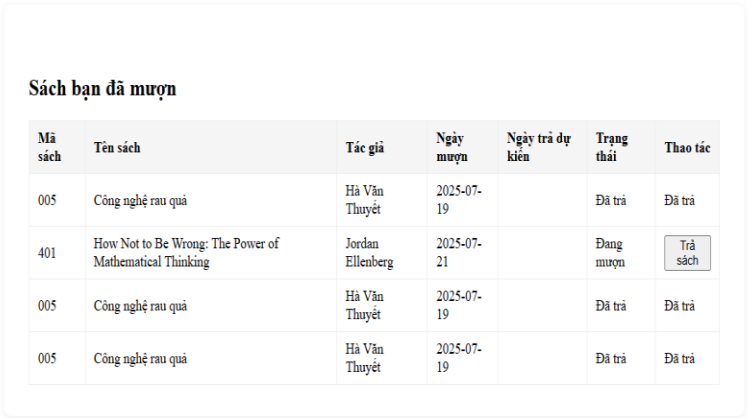


Tìm kiếm Sách Nâng Cao

Từ khóa tên sách hoặc tác giả... | -- Thể loại -- | -- Trạng thái -- | Tìm kiếm

Mã sách	Tên sách	Tác giả	Năm xuất bản	Ngôn ngữ	Thể loại	Trạng thái
005	Công nghệ rau quả	Hà Văn Thuyết	2015	Tiếng Việt	Khoa học - Kỹ thuật	Có sẵn
401	How Not to Be Wrong: The Power of Mathematical Thinking	Jordan Ellenberg	2014	English	Khoa học - Kỹ thuật	Có sẵn
097	Lập trình và Cuộc sống	Jeff Atwood	2013	Tiếng Việt	Công nghệ thông tin	Đang mượn/hết sách
201	Súng, Vi trùng và Thép	Jared Diamond	1997	Tiếng Việt	Lịch sử - Địa lý	Có sẵn
402	In Pursuit of the Unknown: 17 Equations That Changed the World	Ian Stewart	2012	English	Khoa học - Kỹ thuật	Có sẵn
034	The Industries of the Future (Công nghiệp tương lai)	Alec Ross	2008	Tiếng Anh	Công nghệ thông tin	Đang mượn/hết sách
101	Sapiens: Lược sử loài người	Yuval Noah Harari	2011	Tiếng Việt	Lịch sử - Địa lý	Đang mượn/hết sách
003	Công nghệ bảo quản, chế biến nông sản sau thu hoạch.	Trần Văn Chương	2000	Tiếng Việt	Khoa học - Kỹ thuật	Có sẵn
403	The Mathematical Experience	Philip J. Davis & Reuben Hersh	1981	English	Khoa học - Kỹ thuật	Có sẵn

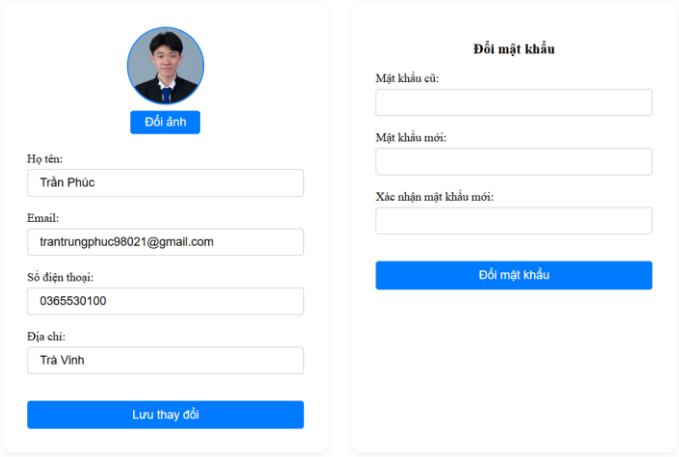
Hình 3.13 Giao diện trang Tìm kiếm – người dùng



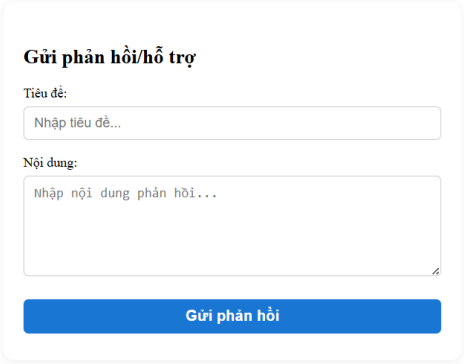
Sách bạn đã mượn

Mã sách	Tên sách	Tác giả	Ngày mượn	Ngày trả dự kiến	Trạng thái	Thao tác
005	Công nghệ rau quả	Hà Văn Thuyết	2025-07-19		Đã trả	Đã trả
401	How Not to Be Wrong: The Power of Mathematical Thinking	Jordan Ellenberg	2025-07-21		Đang mượn	Trả sách
005	Công nghệ rau quả	Hà Văn Thuyết	2025-07-19		Đã trả	Đã trả
005	Công nghệ rau quả	Hà Văn Thuyết	2025-07-19		Đã trả	Đã trả

Hình 3.14 Giao diện trang Sách – người dùng



Hình 3.15 Giao diện trang Cài đặt – người dùng



Hình 3.16 Giao diện trang Hỗ trợ - người dùng

Link Figma:
https://www.figma.com/design/s7YJRHCETzRKtuuHXI5VLt/D%E1%BB%B0_%C3%81N_QU%E1%BA%A2N_L%C3%9D_TH%C6%AF_VI%E1%BB%86N?node-id=0-1&p=f&t=HyVFp4uKUNizTWFb-0

3.2 Công nghệ sử dụng và triển khai

3.2.1 Công nghệ sử dụng

3.2.1.1 Ngôn ngữ lập trình

JavaScript:

- Được sử dụng cho cả frontend (Vue.js) và backend (Node.js/Express).

- Là ngôn ngữ chính để xây dựng giao diện người dùng, xử lý logic phía client và server.

HTML/CSS:

- Sử dụng để xây dựng cấu trúc và định dạng giao diện người dùng trên frontend.

- CSS được tổ chức thành các file riêng biệt (base.css, main.css) và sử dụng scoped style trong các file .vue.

3.2.1.2 Framework và Thư viện Frontend

Vue.js:

- Framework chính để xây dựng giao diện người dùng dạng SPA (Single Page Application).

- Sử dụng cú pháp Composition API (<script setup>) hiện đại, giúp code ngắn gọn, dễ bảo trì.

- Các thành phần giao diện được tổ chức thành các file .vue (component-based).

Vite:

- Công cụ build và phát triển hiện đại cho frontend, thay thế Webpack.

- Hỗ trợ hot-reload, build nhanh, tối ưu hóa tài nguyên tĩnh.

- Cấu hình qua file vite.config.js.

Vue Router:

- Thư viện định tuyến cho Vue.js, cho phép chuyển trang không cần tải lại trang, quản lý các route như: login, register, user-home, v.v.

Pinia:

- Thư viện quản lý trạng thái state - trạng thái toàn cục hiện đại thay thế Vuex.

- Dùng để lưu trữ thông tin đăng nhập, trạng thái người dùng.

Axios:

- Thư viện gửi HTTP request từ frontend đến backend (REST API).

- Được sử dụng để gọi các API như đăng nhập, đăng ký, tìm kiếm sách, gửi yêu cầu hỗ trợ.

Chart.js & Vue-Chartjs:

- Chart.js là thư viện vẽ biểu đồ phổ biến.

- Vue-Chartjs là wrapper cho Chart.js, giúp tích hợp biểu đồ vào component Vue dễ dàng.

- Sử dụng cho các chức năng thống kê, trực quan hóa dữ liệu.

3.2.1.3 Framework và Thư viện Backend

Express.js:

- Framework web cho Node.js, giúp xây dựng RESTful API nhanh chóng, dễ mở rộng.

- Quản lý các route như /api/auth/login, /api/books/search, /api/users, v.v.

Swagger-jsdoc & Swagger-ui-express:

- Tạo và hiển thị tài liệu API tự động theo chuẩn Swagger/OpenAPI.

- Hỗ trợ kiểm thử API trực tiếp trên trình duyệt.

Mongoose:

- Thư viện ORM (Object Relational Mapping) giúp tương tác với MongoDB một cách dễ dàng. Cho phép định nghĩa schema và thực hiện truy vấn hiệu quả.

CORS:

- Thư viện dùng để xử lý CORS (Cross-Origin Resource Sharing), cho phép frontend truy cập API từ domain khác.

3.2.2 Quy trình CI/CD

Workflow CI/CD sẽ được kích hoạt mỗi khi có thao tác push hoặc pull request lên nhánh main. Các bước chính trong pipeline như sau:

Bước 1: Checkout mã nguồn

- Lấy mã từ GitHub repo về runner (máy chủ CI/CD tạm thời).

Bước 2: Thiết lập môi trường Node.js

- Sử dụng phiên bản Node.js 18 cho cả frontend và backend.

Bước 3: Cài đặt và kiểm thử Backend

- Cài dependencies cho backend

- Chạy test backend

Bước 4: Cài đặt và build Frontend

- Cài dependencies cho frontend

- Build ra static files để triển khai

Bước 5: Deploy lên server thật

- Kết nối đến server qua SSH

- Pull code mới nhất từ GitHub

- Cài đặt lại thư viện

- Build lại ứng dụng nếu cần

- Restart bằng PM2 để cập nhật thay đổi

3.2.3 Cấu hình Docker và quy trình triển khai

Dự án sử dụng Docker và Docker Compose để container hóa toàn bộ hệ thống gồm:

- Frontend (React + Vite).

- Backend (Node.js + Express).

- MongoDB Atlas (dịch vụ cloud, không container hóa).

3.2.3.1 Cấu trúc thư mục liên quan đến Docker:

```
CNPM_2025_DA22TT/
├── backend/
│   └── Dockerfile
├── frontend/
│   └── Dockerfile
├── docker-compose.yml
```

3.2.3.2 Dockerfile cho Backend:

```
dockerfile
FROM node:18

WORKDIR /app

COPY package*.json ./
RUN npm install

COPY . .

EXPOSE 5000
CMD ["node", "index.js"]
```

3.2.3.3 Dockerfile cho Frontend:

```
dockerfile
FROM node:18

WORKDIR /app

COPY package*.json ./
RUN npm install

COPY . .

EXPOSE 5173
CMD ["npm", "run", "dev", "--", "--host"]
```

3.2.3.4 Docker-compose.yml

```
yaml
services:
  backend:
    build:
```

```
      context: ./backend
      dockerfile: Dockerfile
    ports:
      - "5000:5000"
    environment:
      -
MONGO_URI=mongodb+srv://<username>:<password>@qltv.
tl6vupn.mongodb.net/QLTV
    networks:
      - app-network

  frontend:
    build:
      context: ./frontend
      dockerfile: Dockerfile
    ports:
      - "5173:5173"
    networks:
      - app-network

networks:
  app-network:
    driver: bridge
```

3.2.3.5 Quy trình triển khai:

1. Cài đặt Docker :

* Docker Desktop: https://www.docker.com/products/docker-desktop

2. Mở terminal tại thư mục dự án:

```
docker-compose down --volumes --remove-orphans
     docker-compose up --build
```

3. Truy cập ứng dụng:

Dịch vụ	***URL***
Frontend	http://localhost:5173
Backend	http://localhost:5000
Swagger	http://localhost:5000/api-docs

4. Triển khai thực tế :

Link Firebase: https://studio-bsit4.web.app

Link Render: https://thuviencelri.onrender.com

3.3 Quản lý dự án

3.3.1 Sử dụng Jira để lập kế hoạch và theo dõi tiến độ

Trong dự án "Xây dựng ứng dụng quản lý thư viện", nhóm đã sử dụng Jira để quản lý công việc, theo dõi tiến độ và kiểm soát việc phân công giữa các thành viên.

1. Tạo không gian dự án trên Jira

Bước đầu tiên là tạo một dự án trên Jira với tên gọi CNPM_HK2_2425. Các bước cụ thể như sau:

- Đăng nhập vào tài khoản Jira theo đường dẫn của hệ thống nhóm sử.
- Tại màn hình chính, chọn Projects > Create Project.
- Chọn mẫu dự án là Scrum software project, vì nhóm sử dụng phương pháp Scrum để chia giai đoạn thành các Sprint.
- Nhập tên dự án là CNPM_HK2_2425 và hoàn tất tạo mới.

Kết quả là nhóm có được một không gian làm việc riêng, có thể tạo Sprint, thêm thành viên và nhập nhiệm vụ cụ thể.

2. Thêm thành viên vào dự án

Sau khi tạo xong dự án, nhóm thêm các thành viên vào để có thể phân công công việc. Các bước:

- Vào mục Project settings > People.

- Nhấn nút Add people, nhập email hoặc tên người dùng Jira của các thành viên.
- Phân quyền là Member hoặc Developer tùy vai trò.
- Trong dự án này, các thành viên gồm: Khoa Phan, Long Hoàng và Trần Phúc đều được thêm với vai trò là người thực hiện chính.

3. Tạo các Sprint theo giai đoạn phát triển

Nhóm chia toàn bộ tiến độ thực hiện thành bốn Sprint, mỗi Sprint tương ứng với một giai đoạn phát triển. Các bước tạo Sprint:

- Vào tab Backlog.
- Nhấn nút Create Sprint để tạo Sprint mới.
- Nhập tên Sprint: CH2 Sprint 1, CH2 Sprint 2, CH2 Sprint 3, CH2 Sprint 4.
- Các Sprint này được dùng để gom nhóm các nhiệm vụ theo từng giai đoạn: thiết kế, lập trình, kiểm thử, trình bày...

Khi đã tạo đủ 4 Sprint, nhóm chuyển sang bước tiếp theo là thêm các nhiệm vụ cụ thể.

4. Tạo các nhiệm vụ - Issue trong từng Sprint

Để phản ánh các công việc cụ thể cần làm, nhóm tiến hành tạo các Issue.
Các bước:

Trong mục Backlog, nhấn + Create Issue ở phần Sprint tương ứng.

Điền đầy đủ:

- Summary (tiêu đề nhiệm vụ): ví dụ “Khảo sát nghiệp vụ và xác định yêu cầu”.
- Assignee (người thực hiện): chọn đúng thành viên nhóm.
- Sprint: chọn Sprint mà công việc thuộc về.
- Issue type: thường để mặc định là Task.
- Labels: để phân loại frontend, backend, tài liệu...

5. Theo dõi tiến độ thực hiện công việc

Sau khi nhiệm vụ được phân công, các thành viên vào Jira và bắt đầu cập nhật tiến độ theo các trạng thái sau:

- To Do: công việc chưa bắt đầu.
- In Progress: đang thực hiện.
- Done: hoàn thành.

Trong suốt quá trình phát triển, mỗi người khi bắt đầu làm sẽ chuyển nhiệm vụ sang "In Progress" và khi làm xong sẽ đánh dấu là "Done".

6. Kết thúc Sprint và tổng kết tiến độ

Khi đến thời hạn kết thúc mỗi Sprint, nhóm vào Jira và chọn Complete Sprint. Hệ thống sẽ ghi nhận:

Các nhiệm vụ đã hoàn thành (hiện là "DONE").

3.3.2 Phân công nhiệm vụ

Khoa Phan						
	CH2-32	Khảo sát nghiệp vụ và xác định yêu cầu	DONE	Add comment	CH2 Sprint 1	Khoa Phan
	CH2-35	Khởi tạo repo Github, cấu trúc MVC	DONE	Add comment	CH2 Sprint 1	Khoa Phan
	CH2-39	Tìm kiếm sách theo tên/tác giả/ thể loại	DONE	Add comment	CH2 Sprint 2	Khoa Phan
	CH2-40	Chi tiết sách và hiển thị ảnh bìa	DONE	Add comment	CH2 Sprint 2	Khoa Phan
	CH2-44	Tích hợp Github Action CI/CD	DONE	Add comment	CH2 Sprint 2	Khoa Phan
	CH2-43	Tạo Swagger mô tả API	DONE	Add comment	CH2 Sprint 2	Khoa Phan
	CH2-48	Kiểm thử toàn bộ API bằng Postman	DONE	Add comment	CH2 Sprint 4	Khoa Phan
	CH2-50	Viết phụ lục: hướng dẫn cài đặt, tài liệu Swagger	DONE	Add comment	CH2 Sprint 4	Khoa Phan
Long Hoàng						
	CH2-33	Vẽ sơ đồ Use Case, DFD, ERD	DONE	Add comment	CH2 Sprint 1	Long Hoàng
	CH2-36	Thiết kế CSDL MongoDB	DONE	Add comment	CH2 Sprint 1	Long Hoàng
	CH2-38	Đăng ký, đăng nhập	DONE	Add comment	CH2 Sprint 2	Long Hoàng
	CH2-42	Mượn và trả sách	DONE	Add comment	CH2 Sprint 2	Long Hoàng
	CH2-46	Gửi thông báo trả sách (email hoặc hiển thị trong app)	DONE	Add comment	CH2 Sprint 3	Long Hoàng
	CH2-51	Chuẩn bị slide, trình bày và demo	DONE	Add comment	CH2 Sprint 4	Long Hoàng
Trần Phúc						
	CH2-34	Thiết kế UI/UX bằng Figma	DONE	Add comment	CH2 Sprint 1	Trần Phúc
	CH2-37	Xây dựng giao diện người dùng với VueJS	DONE	Add comment	CH2 Sprint 2	Trần Phúc
	CH2-41	CRUD sách (dành cho thủ thư)	DONE	Add comment	CH2 Sprint 2	Trần Phúc
	CH2-45	Lịch sử mượn/trả	DONE	Add comment	CH2 Sprint 3	Trần Phúc
	CH2-47	Phân quyền người dùng	DONE	Add comment	CH2 Sprint 3	Trần Phúc
	CH2-49	Viết báo cáo	DONE	Add comment	CH2 Sprint 4	Trần Phúc

Hình 3.11 Bảng phân công nhiệm vụ trong Jira

CHƯƠNG 4: KẾT QUẢ THỰC NGHIỆM

4.1 Chiến lược kiểm thử

Kiểm thử đơn vị (Unit Test tự viết):

Các script kiểm thử như `test-register.js`, `test-admin-functions.js`, `test-stats.js` được viết bằng tay, sử dụng thư viện `axios` để gửi yêu cầu HTTP đến các API endpoint.

Mỗi script có nhiệm vụ kiểm tra một chức năng cụ thể như đăng ký người dùng, thống kê hệ thống hoặc quản lý người dùng.

Kiểm thử tích hợp (Integration Test):

Hệ thống được chạy trong môi trường Docker thông qua `docker-compose`. Tại đây, các chức năng được kiểm thử khi backend kết nối thực tế với cơ sở dữ liệu MongoDB, đảm bảo tính ổn định của toàn hệ thống.

Kiểm thử API thủ công bằng Postman:

Sử dụng Postman để gửi các request GET, POST, DELETE đến backend nhằm kiểm tra các phản hồi HTTP, cấu trúc dữ liệu trả về, và xác minh logic nghiệp vụ trong các tình huống khác nhau.

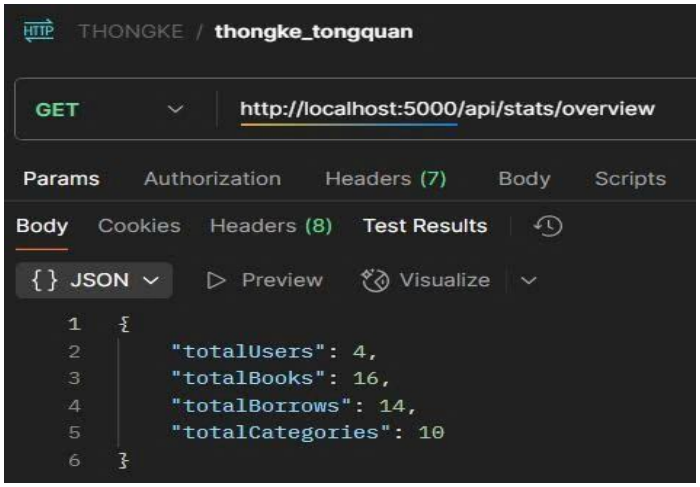
4.2 Công cụ sử dụng

Công cụ	Mô tả
Axios + Script Node.js	Sử dụng trong các file kiểm thử tự viết. Gửi các request đến API và in ra kết quả phản hồi. Cung cấp khả năng kiểm thử linh hoạt và có thể dễ dàng mở rộng.
Postman	Dùng để kiểm thử thủ công từng API endpoint, hỗ trợ kiểm tra phản hồi khi thay đổi dữ liệu đầu vào và mô phỏng hành vi người dùng.
Docker Compose	Tạo môi trường triển khai đồng nhất giữa các thành viên trong nhóm. Cho phép chạy backend, database trong môi trường cô lập để kiểm thử tích hợp và triển khai thực tế.

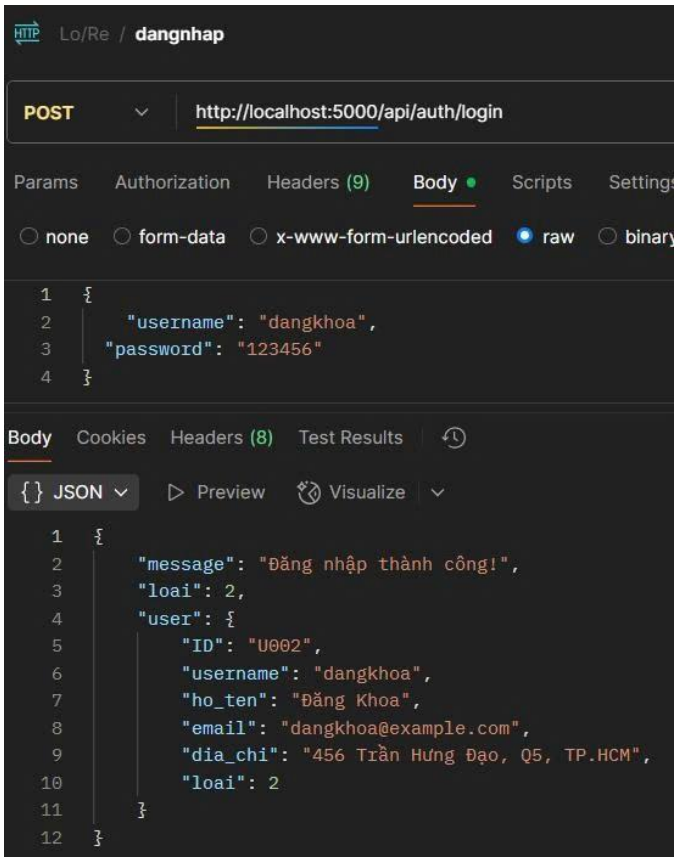
GitHub	Dự án được quản lý bằng Git, giúp theo dõi lịch sử phát triển, phối hợp làm việc nhóm. Có thể mở rộng kiểm thử tự động bằng cách tích hợp GitHub Actions để chạy test khi có cập nhật mã nguồn.

4.3 Kết quả kiểm thử API

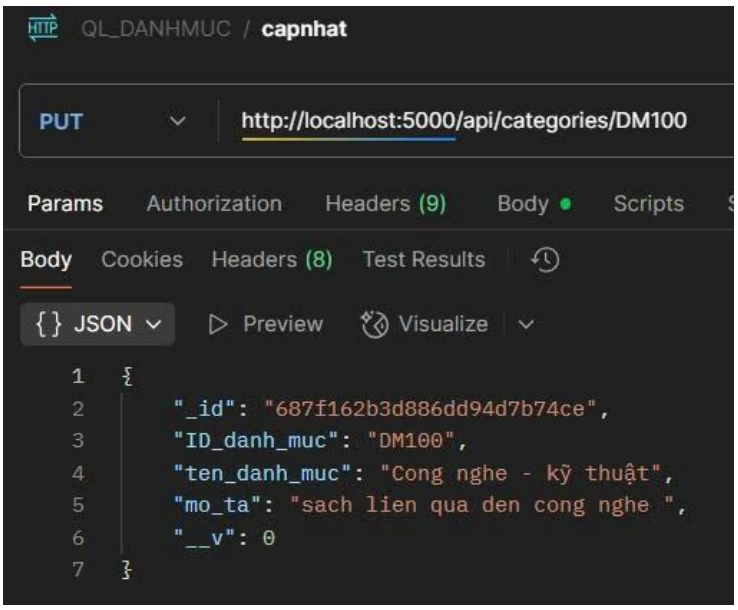
Một số kết quả kiểm thử API bằng Postman:



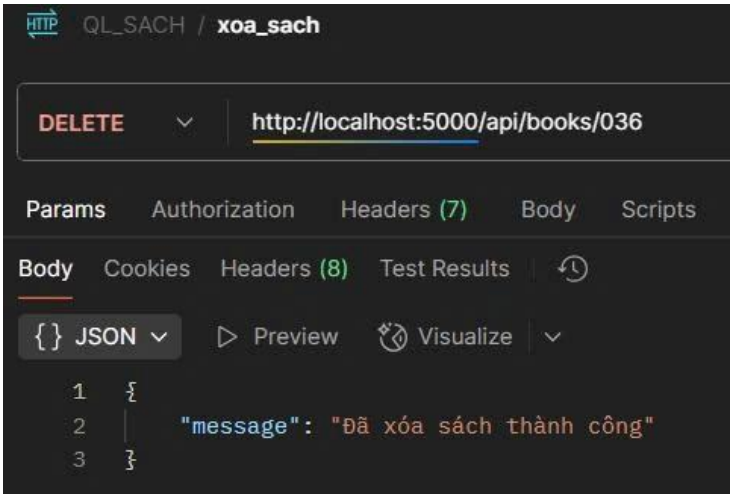
Hình 4.1 Tổng quan thống kê qua phương thức GET bằng Postman



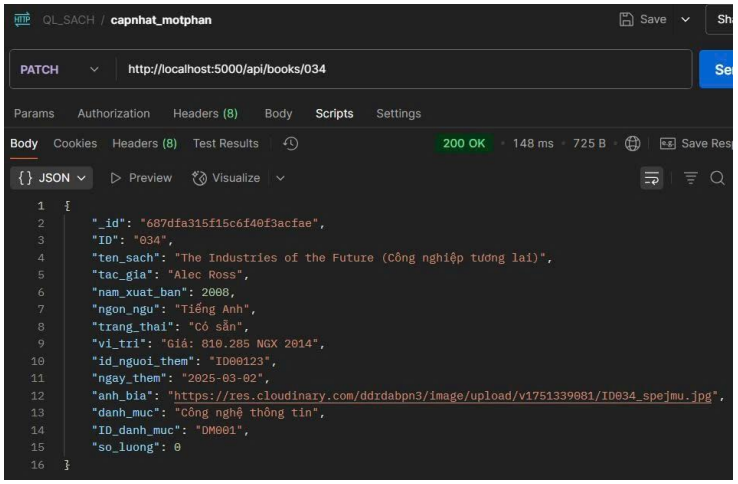
Hình 4.2 Đăng nhập thông qua phương thức POST bằng Postman



Hình 4.3 Cập nhật danh mục sách qua phương thức PUT bằng Postman



Hình 4.4 Xóa sách ra khỏi hệ thống qua phương thức DELETE bằng Postman



Hình 4.5 Cập nhật một phần thông tin sách qua phương thức PATCH bằng Postman

CHƯƠNG 5: ƯU NHƯỢC ĐIỂM VÀ HƯỚNG PHÁT TRIỂN

5.1 Ưu điểm

Được thiết kế và phát triển dựa trên kiến trúc phần mềm hiện đại theo mô hình client-server, với frontend và backend được tách biệt rõ ràng, giúp hệ thống đạt được tính module hóa cao, dễ dàng bảo trì và mở rộng về sau. Ở phía frontend, nhóm sử dụng Vue.js kết hợp với Vite, Vue Router và Pinia để xây dựng giao diện người dùng theo hướng SPA (Single Page Application), đảm bảo trải nghiệm liền mạch, phản hồi nhanh và giao diện trực quan, dễ sử dụng. Phía backend được phát triển trên nền Node.js với Express.js, cung cấp các RESTful API được tổ chức khoa học, dễ tái sử dụng, đồng thời hỗ trợ Swagger để tự động tạo tài liệu API, tăng khả năng kiểm thử và tích hợp. Dữ liệu được lưu trữ và truy vấn thông qua MongoDB, sử dụng Mongoose ORM giúp đơn giản hóa quá trình thao tác với cơ sở dữ liệu phi quan hệ. Hệ thống còn áp dụng quy trình CI/CD thông qua GitHub Actions, hỗ trợ kiểm thử và triển khai tự động, tăng tốc độ phát hành sản phẩm. Ngoài ra, toàn bộ hệ thống còn được đóng gói bằng Docker và quản lý thông qua Docker Compose, giúp tăng tính nhất quán, đảm bảo hoạt động ổn định trong môi trường triển khai thực tế.

5.2 Nhược điểm

Mặc dù hệ thống đã hoàn thiện về mặt chức năng và đáp ứng được nhiều yêu cầu thực tế, nhưng vẫn còn tồn tại một số nhược điểm đáng lưu ý. Thứ nhất, về mặt giao diện người dùng, hệ thống hiện chưa tối ưu cho các thiết bị di động có kích thước màn hình nhỏ. Thứ hai, quá trình bảo mật API vẫn còn đơn giản, chưa áp dụng các cơ chế mã hóa mạnh mẽ như JWT (JSON Web Token) có thời hạn, mã hóa dữ liệu đầu vào điều này làm tăng nguy cơ lỗ hổng bảo mật trong môi trường vận hành thực tế. Thứ ba, khả năng phân tích dữ liệu thống kê còn hạn chế khi các biểu đồ được tích hợp vẫn mang tính mô tả, chưa hỗ trợ các tính năng nâng cao như lọc theo thời gian, lọc theo danh mục, phân tích chuyên sâu theo hành vi người dùng. Cuối cùng, hệ thống vẫn chưa triển khai các tính năng mở rộng quan trọng như gửi email xác thực, reset mật khẩu qua email, tích hợp thanh toán điện tử (nếu phát sinh phí phạt khi trả sách trễ), dẫn đến việc khó mở rộng quy mô sử dụng trong môi trường thư viện thực tế với lượng người dùng lớn.

5.3 Hướng phát triển

Để nâng cao chất lượng và phạm vi ứng dụng trong tương lai, hệ thống được định hướng mở rộng hệ thống theo nhiều khía cạnh, cả về chức năng và công nghệ. Trước hết, nhóm sẽ triển khai responsive UI sử dụng framework UI hiện đại nhằm tối ưu giao diện trên nhiều loại thiết bị (desktop, tablet, mobile). Tiếp đến, nhóm sẽ tích hợp các chức năng xác thực người dùng nâng cao như OTP/email verification, reset mật khẩu tự động, và áp dụng OAuth 2.0 để hỗ trợ đăng nhập qua tài khoản Google hoặc Facebook. Về phía backend, nhóm hướng đến sử dụng các công nghệ như Redis caching để lưu tạm dữ liệu truy vấn để tăng tốc độ phản hồi, rate limiting để giới hạn số lần yêu cầu từ người dùng để chống spam, và mở rộng sang mô hình microservices để tách biệt từng chức năng như xử lý thống kê, gửi email, xử lý ảnh,... Ngoài ra, hệ thống sẽ tích hợp AI đơn giản để gợi ý sách phù hợp theo lịch sử mượn/trả, kết hợp machine learning nhằm phân tích xu hướng đọc sách theo thời gian.

DANH MỤC TÀI LIỆU THAM KHẢO

[1] “https://fptshop.com.vn” 2018. [Trực tuyến]. Available: https://fptshop.com.vn/vue-js-cach-chia-se-du-lieu-giua-cac-thanh-phan-#-168413. [Đã truy cập 29 6 2025].

[2] “howkteam.vn” 2021. [Trực tuyến]. Available: https://howkteam.vn/course/phuong-phap-ket-noi-giua-frontend-va-backend. [Đã truy cập 5 7 2025].

[3] “www.studocu.vn” 2019. [Trực tuyến]. Available: https://www.studocu.vn/vn/document/truong-dai-hoc-kinh-te-thanh-pho-ho-chi-minh/quy-trinh-trien-khai-docker-va-huong-dan-su-dung. [Đã truy cập 10 7 2025].

[4] “www.slideshare.net” 2020. [Trực tuyến]. Available: https://www.slideshare.net/slideshow/postman-cach-su-dung-co-ban-va-nang-cao-120411053120phpapp01. [Đã truy cập 19 7 2025].

PHỤ LỤC

Hướng dẫn cài đặt và chạy ứng dụng Quản Lý Thư Viện

Bước 1: Cài đặt các gói cần thiết

- Mở terminal, chuyển vào thư mục backend, chạy lệnh npm install để cài đặt backend.
- Sau đó chuyển sang thư mục frontend, tiếp tục chạy npm install để cài đặt frontend.

Bước 2: Khởi động MongoDB

- Đảm bảo MongoDB đã được cài đặt và đang chạy tại địa chỉ localhost:27017.

Bước 3: Chạy backend

- Vào thư mục backend, chạy lệnh npm start. Backend sẽ chạy tại địa chỉ http://localhost:5000.

Bước 4: Chạy frontend

- Vào thư mục frontend, chạy lệnh npm run dev. Giao diện người dùng sẽ hoạt động tại http://localhost:5173.

Bước 5: Kiểm thử hệ thống

- Có thể truy cập http://localhost:5173/register để đăng ký tài khoản mới.
- Sau đó kiểm tra danh sách người dùng tại http://localhost:5000/api/users để xác nhận người dùng đã được thêm thành công.

Liên kết Github Repository: https://github.com/Silnix/CNPM_2025_DA22TT

Liên kết Demo: https://studio-bsit4.web.app/login

https://thuviencelri.onrender.com