# LỜI CẢM ƠN

Em xin gửi lời cảm ơn sâu sắc đến quý thầy cô Trường Đại học Cần Thơ, đặc biệt là các thầy cô Khoa Công nghệ Thông tin, đã tận tình giảng dạy, truyền đạt kiến thức và hướng dẫn em trong suốt quá trình học tập cũng như thực hiện đồ án này. Nhờ sự chỉ bảo, động viên và hỗ trợ của thầy cô, em đã có cơ hội tiếp cận các công nghệ mới, rèn luyện kỹ năng thực tế và hoàn thành tốt đề tài "Website Đăng Ký Mượn Sách Online".

Em cũng xin cảm ơn thầy/cô hướng dẫn đã dành thời gian góp ý, chỉnh sửa và định hướng giúp em vượt qua những khó khăn trong quá trình làm đồ án. Sự quan tâm và tận tâm của thầy cô là nguồn động lực lớn để em cố gắng học tập, nghiên cứu và phát triển bản thân.

Em xin chân thành cảm ơn quý thầy cô và nhà trường đã tạo điều kiện thuận lợi để em hoàn thành đồ án này. Em kính chúc quý thầy cô sức khỏe, hạnh phúc và thành công trong sự nghiệp giáo dục!

# MỤC LỤC

[LỜI CẢM ƠN 2](#_Toc216002629)

[MỤC LỤC 3](#_Toc216002630)

[DANH MỤC HÌNH 4](#_Toc216002631)

[DANH MỤC BẢNG 5](#_Toc216002632)

[CHƯƠNG 1: GIỚI THIỆU 6](#_Toc216002633)

[1.1. Đặt vấn đề 6](#_Toc216002634)

[1.2. Mục tiêu đề tài 6](#_Toc216002635)

[1.3. Đối tượng và phạm vi nghiên cứu 8](#_Toc216002636)

[1.3.1. Đối tượng sử dụng 8](#_Toc216002637)

[1.3.2. Phạm vi của đề tài 9](#_Toc216002638)

[1.4. Phương pháp nghiên cứu 9](#_Toc216002639)

[1.4.1. Phương pháp thu thập dữ liệu 9](#_Toc216002640)

[1.4.2. Phương pháp phát triển 10](#_Toc216002641)

[1.5. Các chức năng chính 11](#_Toc216002642)

[1.5.1. Chức năng dành cho độc giả 11](#_Toc216002643)

[1.5.2. Chức năng dành cho nhân viên / Admin 12](#_Toc216002644)

[1.5.3. Các yêu cầu phi chức năng 13](#_Toc216002645)

[CHƯƠNG 2: ĐẶC TẢ YÊU CẦU 15](#_Toc216002646)

[2.1. Mô tả bài toán 15](#_Toc216002647)

[2.2. Yêu Cầu Bài Toán 15](#_Toc216002648)

[2.2.1. Yêu cầu chức năng 15](#_Toc216002649)

[2.2.2. Yêu cầu phi chức năng 17](#_Toc216002650)

[2.3. Ngôn ngữ lập trình và các công cụ liên quan 18](#_Toc216002651)

[2.3.1. Vue.js 3 18](#_Toc216002652)

[2.3.2. Node.js 19](#_Toc216002653)

[2.3.3. Visual Studio Code (VS Code) 21](#_Toc216002654)

[2.3.4. Bootstrap 5 22](#_Toc216002655)

[CHƯƠNG 3: CÁC SƠ ĐỒ CỦA WEBSITE 24](#_Toc216002656)

[3.1. Phân tích và thiết kế 24](#_Toc216002657)

[3.1.1. Sơ đồ use case tổng quát 24](#_Toc216002658)

[3.1.2. Đặc tả chức năng sơ đồ use case tổng quát 25](#_Toc216002659)

[3.2. Sơ đồ use case đăng ký và đăng nhập 26](#_Toc216002660)

[3.2.1. Đặc tả use case đăng ký 26](#_Toc216002661)

[3.2.2. Đặc Tả Use Case Đăng Nhập 28](#_Toc216002662)

[3.3. Sơ đồ use case tìm kiếm và mượn sách 30](#_Toc216002663)

[3.3.1. Đặc tả chức năng tìm kiếm 30](#_Toc216002664)

[3.3.2. Đặc tả chức năng xem chi tiết sách 32](#_Toc216002665)

[3.3.3. Đặc tả chức năng đăng ký mượn sách 33](#_Toc216002666)

[3.4. Sơ đồ use case quản lý sách (cho nhân viên) 36](#_Toc216002667)

[3.4.1. Đặc tả sơ đồ quản lý sách 36](#_Toc216002668)

[3.5. Sơ đồ use case xử lý mượn/trả sách 40](#_Toc216002669)

[3.5.1. Đặc tả use case xử lý mượn/trả sách 40](#_Toc216002670)

[3.6. Mô hình cơ sở dữ liệu 45](#_Toc216002671)

[3.6.1. Mô tả các lớp (Entities) 45](#_Toc216002672)

[3.6.2. Mô tả User (Người dùng) 45](#_Toc216002673)

[3.6.3. Mô tả book (sách) 48](#_Toc216002674)

[3.6.4. Mô tả borrow record (phiếu mượn) 49](#_Toc216002675)

[CHƯƠNG 4: THIẾT KẾ GIAO DIỆN 52](#_Toc216002676)

[4.1. Giao diện Quản trị viên 52](#_Toc216002677)

[4.1.1. Giao diện đăng nhập admin 52](#_Toc216002678)

[4.1.2. Giao diện trang chủ 52](#_Toc216002679)

[4.1.3. Giao diện quản lý sách 53](#_Toc216002680)

[4.1.4. Giao diện quản lý nhà xuất bản 55](#_Toc216002681)

[4.1.5. Giao diện quản lý độc giả 56](#_Toc216002682)

[4.1.6. Giao diện quản lý nhân viên 56](#_Toc216002683)

[4.1.7. Giao diện quản lý mượn sách 56](#_Toc216002684)

[c 57](#_Toc216002685)

[4.1.9. Giao diện trả sách 58](#_Toc216002686)

[4.1.10. Giao diện trả sách 58](#_Toc216002687)

[4.1.11. Giao diện đổi mật khẩu 59](#_Toc216002688)

[4.2. Giao diện người dùng 59](#_Toc216002689)

[4.2.1. Giao diện đăng ký tài khoản 59](#_Toc216002690)

[4.2.2. Giao diện đăng nhập 60](#_Toc216002691)

[4.2.3. Giao diện danh sách sách 60](#_Toc216002692)

[4.2.4. Giao diện chỉnh sửa thông tin cá nhân 61](#_Toc216002693)

[4.2.6. Giao diện chi tiết sách 61](#_Toc216002694)

[4.2.7. Giao diện lịch sử mượn sách 62](#_Toc216002695)

[4.2.8. Giao diện đổi mật khẩu 62](#_Toc216002696)

[4.2.9. Giao diện chi tiết mượn sách 63](#_Toc216002697)

[4.2.10. Giao diện đăng ký mượn sách 63](#_Toc216002698)

[CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 64](#_Toc216002699)

[5.1. Kết quả đạt được 64](#_Toc216002700)

[5.1.1. Về mặt chức năng 64](#_Toc216002701)

[5.1.2. Về mặt kỹ thuật 64](#_Toc216002702)

[5.1.3. Về mặt bảo mật 65](#_Toc216002703)

[TÀI LIỆU THAM KHẢO 69](#_Toc216002704)

# DANH MỤC HÌNH

[Hình 4.1: Giao diện đăng nhập admin 52](#_Toc216002599)

[Hình 4.2: Giao diện trang chủ 52](#_Toc216002600)

[Hình 4.3: Giao diện liệt kê danh sách sách 53](#_Toc216002601)

[Hình 4.4: Giao diện thêm sách 53](#_Toc216002602)

[Hình 4.5: Giao diện chỉnh sửa sách 54](#_Toc216002603)

[Hình 4.6: Giao diện chi tiết sách 54](#_Toc216002604)

[Hình 4.7: Giao diện xóa sách 55](#_Toc216002605)

[Hình 4.8: Giao diện quản lý nhà xuất bản 55](#_Toc216002606)

[Hình 4.9: Giao diện quản lý độc giả 56](#_Toc216002607)

[Hình 4.10: Giao diện quản lý mượn sách 56](#_Toc216002608)

[Hình 4.11: Giao diện quản lý mượn sách 57](#_Toc216002609)

[Hình 4.12: Giao diện chi tiết mượn sách 57](#_Toc216002610)

[Hình 4.13: Giao diện trả sách 58](#_Toc216002611)

[Hình 4.14: Giao diện thông tin cá nhân 58](#_Toc216002612)

[Hình 4.15: Giao diện đổi mật khẩu 59](#_Toc216002613)

[Hình 4.16: Giao diện đăng ký tài khoản 59](#_Toc216002614)

[Hình 4.17: Giao diện đăng nhập 60](#_Toc216002615)

[Hình 4.18: Giao diện danh sách sách 60](#_Toc216002616)

[Hình 4.19: Giao diện chỉnh sửa thông cá nhân 61](#_Toc216002617)

[Hình 4.20: Giao diện chi tiết sách 61](#_Toc216002618)

[Hình 4.21: Giao diện lịch sử mượn sách 62](#_Toc216002619)

[Hình 4.22: Giao diện đổi mật khẩu 62](#_Toc216002620)

[Hình 4.23: Giao diện chi tiết mượn sách 63](#_Toc216002621)

[Hình 4.24: Giao diện đăng ký mượn sách 63](#_Toc216002622)

# DANH MỤC BẢNG

[Bảng 1.1: Chức năng dành cho độc giả 11](#_Toc216002539)

[Bảng 1.2: Chức năng dành cho nhân viên 12](#_Toc216002540)

[Bảng 3.1: Đặc tả use case đăng ký tài khoản 27](#_Toc216002552)

[Bảng 3.2: Đặc tả use case đăng nhập 28](#_Toc216002553)

[Bảng 3.3: Đặc tả use case tìm kiếm 31](#_Toc216002554)

[Bảng 3.4: Đặc tả use case xem chi tiết sách 32](#_Toc216002555)

[Bảng 3.5: Đặc tả use case đăng ký mượn sách 34](#_Toc216002556)

[Bảng 3.6: Đặc tả use case quản lý sách 36](#_Toc216002557)

[Bảng 3.7: Đặc tả use case thêm sách mới 36](#_Toc216002558)

[Bảng 3.8: Đặc tả use case sửa sách 38](#_Toc216002559)

[Bảng 3.9: Đặc tả use case xóa sách 39](#_Toc216002560)

[Bảng 3.10: Đặc tả use case tìm kiếm và lọc sách 40](#_Toc216002561)

[Bảng 3.11: Đặc tả use case xem yêu cầu mượn sách 40](#_Toc216002562)

[Bảng 3.12: Đặc tả use case duyệt yêu cầu mượn 41](#_Toc216002563)

[Bảng 3.13: Đặc tả use case từ chối yêu cầu mượn 42](#_Toc216002564)

[Bảng 3.14: Đặc tả use case ghi nhận sách trả 43](#_Toc216002565)

[Bảng 3.15: Mô tả bảng Độc giả 46](#_Toc216002566)

[Bảng 3.16: Mô tả bảng Nhân viên 47](#_Toc216002567)

[Bảng 3.17: Mô tả bảng Sách 48](#_Toc216002568)

[Bảng 3.18: Mô tả bảng Nhà xuất bản 49](#_Toc216002569)

[Bảng 3.19: Mô tả bảng theo dõi mượn sách 49](#_Toc216002570)

[Bảng 3.20: Mô tả các quan hệ giữa các bảng 50](#_Toc216002571)

# CHƯƠNG 1: GIỚI THIỆU

## 1.1. Đặt vấn đề

Trong thời đại số hóa hiện nay, các thư viện truyền thống đang phải đối mặt với những thách thức lớn trong việc quản lý và cung cấp dịch vụ cho độc giả. Các quy trình mượn sách thủ công tốn nhiều thời gian, công sức và dễ xảy ra lỗi. Độc giả phải đến trực tiếp thư viện để mượn sách, không tiện lợi và không linh hoạt. Quá trình quản lý sách, độc giả, nhân viên cũng rất phức tạp và khó kiểm soát.

Hệ thống thông tin thư viện truyền thống không thể đáp ứng được nhu cầu ngày càng tăng của người dùng:

* Độc giả không thể tìm kiếm và mượn sách trực tuyến
* Nhân viên thư viện phải quản lý thủ công toàn bộ dữ liệu
* Không có cơ chế lưu trữ và theo dõi lịch sử mượn sách hiệu quả
* Khó kiểm soát số quyển sách còn lại
* Không có thống kê và báo cáo tự động

Để giải quyết những vấn đề này, cần phải xây dựng một **Website Quản Lý Mượn Sách Online** - một hệ thống thông tin tích hợp giúp:

* Độc giả có thể tìm kiếm, xem chi tiết và đăng ký mượn sách trực tuyến
* Nhân viên thư viện có thể quản lý sách, độc giả, yêu cầu mượn một cách hiệu quả
* Tự động hóa quy trình duyệt mượn, trả sách và tính phí phạt quá hạn
* Lưu trữ lịch sử mượn sách chi tiết và dễ dàng truy vấn

## 1.2. Mục tiêu đề tài

**Mục tiêu chính:**

Xây dựng một **hệ thống website quản lý mượn sách online** hoàn chỉnh, có khả năng:

1. **Cung cấp dịch vụ cho độc giả:**
   * Tạo tài khoản và quản lý thông tin cá nhân
   * Tìm kiếm sách theo tên, tác giả, ISBN, nhà xuất bản
   * Xem danh sách sách có sẵn và chi tiết thông tin từng cuốn sách
   * Đăng ký mượn sách trực tuyến
   * Xem lịch sử mượn sách và trạng thái các yêu cầu
   * Quản lý hồ sơ cá nhân và đổi mật khẩu
2. **Cung cấp công cụ quản lý cho nhân viên/Admin:**
   * Quản lý danh sách sách (thêm, sửa, xóa, tìm kiếm)
   * Quản lý danh sách độc giả (thêm, sửa, khóa/mở khóa tài khoản)
   * Quản lý nhà xuất bản
   * Duyệt/từ chối yêu cầu mượn sách
   * Ghi nhận sách trả và tính tiền phạt quá hạn
   * Theo dõi trạng thái mượn sách của từng độc giả
   * Quản lý tài khoản nhân viên (dành cho Admin)
   * Xem báo cáo và thống kê
3. **Đảm bảo tính an toàn và bảo mật:**
   * Xác thực người dùng bằng JWT (JSON Web Tokens)
   * Mã hóa mật khẩu bằng bcryptjs
   * Phân quyền dựa trên vai trò (Role-Based Access Control)
   * Kiểm tra quyền truy cập trên từng trang/API
   * Validate dữ liệu đầu vào
4. **Cải thiện hiệu suất và trải nghiệm người dùng:**
   * Giao diện thân thiện, dễ sử dụng (UI/UX tốt)
   * Tốc độ tải trang nhanh
   * Đáp ứng trên các thiết bị khác nhau (Responsive Design)
   * Thông báo lỗi rõ ràng (Toast notifications)
   * Tự động lưu token và duy trì session

**Mục tiêu phụ:**

* Xây dựng kiến trúc ứng dụng sạch, dễ bảo trì và mở rộng
* Sử dụng công nghệ MEVN stack hiện đại (MongoDB, Express, Vue.js 3, Node.js)
* Áp dụng các best practices trong phát triển web
* Tạo tài liệu chi tiết về cấu trúc, cài đặt và sử dụng

## 1.3. Đối tượng và phạm vi nghiên cứu

### 1.3.1. Đối tượng sử dụng

Hệ thống phục vụ ba loại người dùng chính:

**1. Độc giả (Reader)**

* Sinh viên, giáo viên, cộng đồng cần mượn sách
* Có thể tự đăng ký tài khoản
* Quản lý hồ sơ cá nhân, đăng ký mượn sách, xem lịch sử

**2. Thủ thư / nhân viên thư viện (Librarian)**

* Quản lý sách, độc giả
* Duyệt/từ chối yêu cầu mượn
* Ghi nhận sách trả
* Tính tiền phạt quá hạn

**3. Quản trị viên (Admin)**

* Quản lý toàn bộ hệ thống
* Quản lý tài khoản nhân viên
* Xem báo cáo và thống kê
* Cấu hình các tham số hệ thống

### 1.3.2. Phạm vi của đề tài

**Bao gồm:**

* Phát triển giao diện web (Frontend) với Vue.js 3 + Vite
* Phát triển API backend với Node.js + Express
* Thiết kế cơ sở dữ liệu MongoDB
* Xác thực và phân quyền người dùng
* Quản lý sách và độc giả
* Xử lý yêu cầu mươn/trả sách
* Tính tiền phạt quá hạn
* Lưu trữ lịch sử mượn sách

**Không bao gồm:**

* Thanh toán trực tuyến (payment gateway)
* Gửi email/SMS thông báo
* Mobile app (chỉ web)
* Hệ thống RFID hoặc barcode
* Tích hợp với các thư viện khác
* Phân tích dữ liệu nâng cao (Business Intelligence)

## 1.4. Phương pháp nghiên cứu

### 1.4.1. Phương pháp thu thập dữ liệu

1. **Nghiên cứu tài liệu:**
   * Tìm hiểu các hệ thống quản lý thư viện hiện tại
   * Học tập từ các best practices trong phát triển web
   * Nghiên cứu MEVN stack và các công nghệ liên quan
2. **Phỏng vấn và khảo sát:**
   * Tìm hiểu nhu cầu của độc giả và nhân viên thư viện
   * Xác định các chức năng cần thiết
   * Thu thập feedback từ người dùng
3. **Quan sát thực tế:**
   * Theo dõi quy trình mượn/trả sách thủ công
   * Xác định các điểm yếu cần cải thiện
   * Hiểu rõ quy trình làm việc của nhân viên

### 1.4.2. Phương pháp phát triển

**Mô hình Agile:**

* Phát triển theo từng tính năng (feature)
* Kiểm thử thường xuyên
* Cập nhật liên tục dựa trên feedback

**Công nghệ sử dụng:**

* **Backend**: Node.js + Express.js (JavaScript)
* **Frontend**: Vue.js 3 + Vite + Bootstrap 5
* **Database**: MongoDB
* **Authentication**: JWT (JSON Web Tokens)
* **Password Encryption**: bcryptjs
* **HTTP Client**: Axios
* **State Management**: Pinia
* **Version Control**: Git

**Quy trình phát triển:**

1. Thiết kế cơ sở dữ liệu
2. Xây dựng API backend
3. Phát triển giao diện frontend
4. Kiểm thử tính năng
5. Triển khai và bảo trì

## 1.5. Các chức năng chính

### 1.5.1. Chức năng dành cho độc giả

Bảng 1.1: Chức năng dành cho độc giả

| **STT** | **Chức Năng** | **Mô tả** | **Trạng Thái** |
| --- | --- | --- | --- |
| 1 | Đăng ký tài khoản | Tạo tài khoản mới với email, mật khẩu, thông tin cá nhân | Hoàn thành |
| 2 | Đăng nhập | Xác thực người dùng bằng email/mật khẩu | Hoàn thành |
| 3 | Xem thông tin cá nhân | Hiển thị hồ sơ cá nhân của độc giả | Hoàn thành |
| 4 | Cập nhật thông tin cá nhân | Sửa đổi thông tin cá nhân (họ tên, địa chỉ, điện thoại) | Hoàn thành |
| 5 | Đổi mật khẩu | Thay đổi mật khẩu đăng nhập | Hoàn thành |
| 6 | Xem danh sách sách | Hiển thị danh sách toàn bộ sách có sẵn | Hoàn thành |
| 7 | Tìm kiếm sách | Tìm sách theo tên, tác giả, ISBN, nhà xuất bản | Hoàn thành |
| 8 | Xem chi tiết sách | Hiển thị thông tin chi tiết về một cuốn sách | Hoàn thành |
| 9 | Đăng ký mượn sách | Gửi yêu cầu mượn sách (sách phải còn quyển) | Hoàn thành |
| 10 | Xem lịch sử mượn | Hiển thị danh sách sách đã mượn và trạng thái | Hoàn thành |
| 11 | Xem chi tiết phiếu mượn | Hiển thị thông tin chi tiết về một phiếu mượn | Hoàn thành |
| 12 | Đăng xuất | Kết thúc phiên làm việc | Hoàn thành |

### 1.5.2. Chức năng dành cho nhân viên / Admin

Bảng 1.2: Chức năng dành cho nhân viên

| **STT** | **Chức Năng** | **Mô tả** | **Trạng Thái** |
| --- | --- | --- | --- |
| 1 | Dashboard | Xem tổng quan (số độc giả, sách, yêu cầu chờ duyệt) | Hoàn thành |
| 2 | Quản lý sách | Thêm, sửa, xóa, tìm kiếm sách | Hoàn thành |
| 3 | Quản lý nhà xuất bản | Thêm, sửa, xóa nhà xuất bản | Hoàn thành |
| 4 | Quản lý độc giả | Xem danh sách, tìm kiếm, sửa thông tin độc giả | Hoàn thành |
| 5 | Khóa/mở khóa độc giả | Khoá tài khoản độc giả để cấm mượn sách | Hoàn thành |
| 6 | Xem yêu cầu mượn | Hiển thị danh sách yêu cầu mượn sách chờ duyệt | Hoàn thành |
| 7 | Duyệt yêu cầu mượn | Chấp nhận yêu cầu mượn và ghi nhận ngày mượn | Hoàn thành |
| 8 | Từ chối yêu cầu mượn | Từ chối yêu cầu mượn với lý do | Hoàn thành |
| 9 | Ghi nhận trả sách | Ghi nhận ngày trả, tình trạng sách | Hoàn thành |
| 10 | Tính tiền phạt | Tính phí phạt nếu trả sách quá hạn | Hoàn thành |
| 11 | Xem lịch sử mượn | Hiển thị lịch sử mượn sách của tất cả độc giả | Hoàn thành |
| 12 | Lọc/tìm kiếm | Lọc mượn sách theo trạng thái, độc giả, ngày | Hoàn thành |
| 13 | Quản lý nhân viên (Admin) | Thêm, sửa, xóa tài khoản nhân viên | Hoàn thành |
| 14 | Phân quyền nhân viên | Gán chức vụ (Admin/Librarian) | Hoàn thành |

### 1.5.3. Các yêu cầu phi chức năng

Bảng 1.: Các yêu cầu phi chức năng

| **Tính Năng** | **Mô tả** | **Trạng Thái** |
| --- | --- | --- |
| Bảo mật | JWT authentication, password hashing, CORS protection | Hoàn thành |
| Phân quyền | Role-based access control (Admin, Librarian, Reader) | Hoàn thành |
| Responsive Design | Giao diện tương thích với mobile, tablet, desktop | Hoàn thành |
| Performance | Tối ưu tốc độ tải, caching, lazy loading | Hoàn thành |
| User Experience | Toast notifications, loading states, error handling | Hoàn thành |
| Data Persistence | Lưu token vào cookie, duy trì session | Hoàn thành |
| Input Validation | Kiểm tra dữ liệu đầu vào từ client và server | Hoàn thành |

# CHƯƠNG 2: ĐẶC TẢ YÊU CẦU

## 2.1. Mô tả bài toán

Bài toán đặt ra là xây dựng một hệ thống website quản lý mượn sách online cho thư viện, nhằm số hóa và tự động hóa quy trình mượn sách truyền thống. Hiện nay, các thư viện đang gặp khó khăn trong việc quản lý sách và độc giả theo cách thủ công, dẫn đến tốn nhiều thời gian, công sức và dễ xảy ra sai sót.

Hệ thống cần phải cung cấp một nền tảng tích hợp nơi độc giả có thể tìm kiếm, xem thông tin chi tiết về các cuốn sách, đăng ký mượn sách trực tuyến mà không cần phải đến trực tiếp thư viện. Đồng thời, nhân viên thư viện có thể quản lý danh mục sách, xử lý các yêu cầu mượn sách từ độc giả, ghi nhận sách trả, và tính tiền phạt quá hạn tự động.

Bài toán cũng yêu cầu hệ thống phải có khả năng phân quyền rõ ràng giữa các vai trò khác nhau: Quản trị viên (có quyền quản lý toàn bộ hệ thống), thủ thư hoặc nhân viên thư viện (có quyền quản lý sách, độc giả, mượn/trả sách), và độc giả (chỉ có quyền xem sách, đăng ký mượn, xem lịch sử mượn của mình). Hệ thống phải đảm bảo bảo mật cao, với xác thực người dùng bằng JWT tokens, mã hóa mật khẩu, và kiểm tra quyền truy cập nghiêm ngặt.

Ngoài ra, hệ thống cần phải có giao diện thân thiện, dễ sử dụng, tương thích với các thiết bị khác nhau (desktop, tablet, mobile), và hoạt động nhanh chóng mà không bị lag. Các tính năng cơ bản bao gồm quản lý sách (thêm, sửa, xóa, tìm kiếm), quản lý độc giả, quản lý nhà xuất bản, theo dõi và xử lý mượn/trả sách, tính tiền phạt quá hạn, và cung cấp lịch sử mượn sách chi tiết.

## 2.2. Yêu Cầu Bài Toán

### 2.2.1. Yêu cầu chức năng

Yêu cầu chức năng của hệ thống được chia thành ba nhóm chính tương ứng với ba vai trò người dùng:

**Yêu cầu chức năng cho độc giả (READER):**

Độc giả là người dùng cuối của hệ thống. Trước tiên, họ cần có khả năng đăng ký tài khoản mới bằng cách cung cấp thông tin cá nhân như email, mật khẩu, họ tên, địa chỉ, và số điện thoại. Sau khi đăng ký, họ có thể đăng nhập vào hệ thống bằng email và mật khẩu.

Một chức năng quan trọng là khả năng duyệt danh sách sách. Độc giả cần có thể xem tất cả những cuốn sách có sẵn trong thư viện, với các thông tin cơ bản như tên sách, tác giả, nhà xuất bản, giá tiền, số quyển còn lại. Hệ thống phải hỗ trợ tìm kiếm sách theo nhiều tiêu chí: tìm theo tên sách, tên tác giả, mã ISBN, hoặc nhà xuất bản, để độc giả có thể dễ dàng tìm thấy sách mình cần.

Khi tìm thấy sách mình muốn mượn, độc giả cần có thể xem chi tiết thông tin về cuốn sách đó, bao gồm Mô tả, số trang, năm xuất bản, và các thông tin liên quan khác. Sau đó, họ có thể gửi yêu cầu mượn sách. Khi gửi yêu cầu, hệ thống cần kiểm tra xem sách còn quyển không, độc giả có được phép mượn (tài khoản không bị khóa) không, và số lượng sách hiện tại đang mượn của độc giả có vượt quá giới hạn không. Yêu cầu mượn sách sẽ được gửi đến nhân viên để duyệt.

Độc giả cần có thể xem lịch sử mượn sách của mình, bao gồm danh sách các sách đã mượn, ngày mượn, hạn trả, ngày trả thực tế, và trạng thái của từng phiếu mượn (chờ duyệt, đang mượn, đã trả, quá hạn). Hệ thống cũng phải cho phép độc giả xem chi tiết về một phiếu mượn cụ thể, bao gồm các thông tin liên quan và tình trạng sách khi trả.

Ngoài ra, độc giả cần có thể quản lý hồ sơ cá nhân của mình, bao gồm cập nhật thông tin cá nhân (họ tên, địa chỉ, số điện thoại) và đổi mật khẩu. Cuối cùng, họ cần có chức năng đăng xuất để kết thúc phiên làm việc.

**Yêu cầu chức năng cho nhân viên / thủ thư (Librarian):**

Nhân viên thư viện cần có một dashboard để xem tổng quan về hệ thống, bao gồm số lượng độc giả đã đăng ký, số sách trong thư viện, số yêu cầu mượn sách chờ duyệt, và số sách quá hạn cần theo dõi.

Nhân viên cần có khả năng quản lý sách hoàn toàn: Thêm sách mới, xem danh sách sách, sửa đổi thông tin sách (tên, giá, số quyển, tác giả, nhà xuất bản), xóa sách khỏi hệ thống, tìm kiếm và lọc sách theo nhiều tiêu chí. Tương tự, nhân viên cần quản lý nhà xuất bản: tạo mới nhà xuất bản, xem danh sách, sửa đổi, và xóa.

Quản lý độc giả là một chức năng quan trọng khác. Nhân viên có thể xem danh sách tất cả độc giả, tìm kiếm độc giả, xem thông tin chi tiết của từng độc giả (bao gồm lịch sử mượn sách), sửa đổi thông tin độc giả, và quan trọng nhất, khóa hoặc mở khóa tài khoản của độc giả nếu họ vi phạm các quy định.

Chức năng xử lý mượn và trả sách là cốt lõi của công việc nhân viên. Nhân viên cần xem danh sách các yêu cầu mượn sách từ độc giả, lọc theo các tiêu chí như trạng thái (chờ duyệt, đang mượn, đã trả), độc giả, hoặc ngày. Khi xem được một yêu cầu mượn, nhân viên có thể duyệt (chấp nhận) yêu cầu, từ chối yêu cầu với lý do, hoặc xem chi tiết yêu cầu. Khi duyệt, ngày mượn sẽ được ghi nhận tự động. Khi độc giả trả sách, nhân viên cần ghi nhận ngày trả, kiểm tra tình trạng sách (sách còn tốt, bình thường, hư hỏng, hay mất), và Ghi chú (nếu có). Hệ thống sẽ tự động tính tiền phạt nếu sách được trả quá hạn.

**Yêu cầu chức năng cho Quản trị viên (Admin):**

Admin có tất cả các quyền của nhân viên thư viện (xem dashboard, quản lý sách, độc giả, mượn/trả sách). Ngoài ra, Admin có quyền quản lý tài khoản nhân viên: tạo tài khoản nhân viên mới, xem danh sách nhân viên, sửa đổi thông tin nhân viên, phân quyền chức vụ (Admin hoặc Librarian), và xóa tài khoản nhân viên. Admin cũng có thể xem báo cáo và thống kê toàn hệ thống.

### 2.2.2. Yêu cầu phi chức năng

Ngoài các yêu cầu chức năng, hệ thống cũng phải đáp ứng một số yêu cầu phi chức năng quan trọng liên quan đến hiệu năng, bảo mật, khả dụng, và trải nghiệm người dùng.

**Hiệu năng:**

Hệ thống phải có khả năng xử lý nhanh chóng. Trang chính phải tải trong thời gian không quá 2 giây. API endpoints phải trả về response trong vòng dưới 200 mili giây. Hệ thống cần hỗ trợ ít nhất 1000 độc giả đồng thời và 5000 quyển sách. Frontend bundle size phải được tối ưu để không vượt quá 500KB.

**Bảo Mật:**

Bảo mật là yêu cầu cấp cao nhất. Hệ thống phải sử dụng JWT (JSON Web Tokens) cho xác thực, với token hết hạn sau 7 ngày. Mật khẩu phải được mã hóa bằng bcryptjs với ít nhất 10 rounds để đảm bảo an toàn. Hệ thống phải có phân quyền dựa trên vai trò (Role-Based Access Control), kiểm tra quyền truy cập nghiêm ngặt ở cả phía frontend và backend. Tất cả dữ liệu đầu vào phải được validate ở server-side. CORS protection phải được cấu hình để chỉ cho phép requests từ frontend domain. Ngoài ra, hệ thống nên có rate limiting để ngăn chặn brute force attacks.

**Khả dụng:**

Hệ thống nên có uptime ít nhất 99%, không tính thời gian bảo trì. Database backups nên được thực hiện định kỳ để ngăn mất dữ liệu. Monitoring và logging nên được setup để phát hiện và xử lý sự cố kịp thời.

**Trải nghiệm người dùng:**

Giao diện người dùng phải thân thiện, trực quan, dễ sử dụng. Hệ thống phải là responsive design, tương thích với mobile (375x667), tablet (768x1024), và desktop (1920x1080). Khi có lỗi, hệ thống phải hiển thị thông báo lỗi rõ ràng (toast notifications) thay vì để người dùng bối rối. Loading states phải được hiển thị khi đang chờ API response.

**Tương thích:**

Hệ thống phải chạy được trên các trình duyệt chính: Chrome, Firefox, Safari, Edge (phiên bản mới). Backend phải tương thích với Node.js 14.x trở lên, MongoDB 4.x trở lên.

## 2.3. Ngôn ngữ lập trình và các công cụ liên quan

### 2.3.1. Vue.js 3

**Khái niệm:**

Vue.js là một framework JavaScript progressive được sử dụng để xây dựng giao diện người dùng (user interfaces) tương tác. Khác với các framework khác yêu cầu phải sử dụng toàn bộ framework, Vue cho phép bạn sử dụng từng phần của nó một cách linh hoạt. Vue.js 3 là phiên bản mới nhất, được phát hành vào năm 2020, với nhiều cải tiến về hiệu năng, kích thước bundle, và developer experience.

**Đặc điểm chính:**

Reactive Data Binding là một trong những tính năng cốt lõi của Vue. Khi dữ liệu thay đổi, giao diện tự động cập nhật mà không cần phải gọi hàm update bằng tay. Vue sử dụng hệ thống reactivity dựa trên JavaScript Proxy để theo dõi các thay đổi dữ liệu.

Component-Based Architecture là một nguyên tắc thiết kế quan trọng trong Vue. Ứng dụng được chia thành các component nhỏ, tái sử dụng được. Mỗi component có template (HTML), script (logic), và style (CSS) riêng, thường được viết trong một file .vue.

Vue 3 giới thiệu Composition API, một cách mới để tổ chức logic component. Thay vì nhóm code theo tùy chọn (data, methods, computed, etc.), Composition API cho phép nhóm code theo tính năng logic, giúp code dễ đọc và dễ bảo trì hơn khi component phức tạp.

Template Syntax của Vue là sự kết hợp giữa HTML và các directive Vue (v-if, v-for, v-bind, v-on). Cú pháp này rất trực quan và dễ hiểu, ngay cả với những người mới học.

**Lý do chọn vue.js 3 cho dự án:**

Vue.js 3 được chọn cho dự án vì nhiều lý do. Thứ nhất, Vue có curving learning curve (đường cong học tập nhẹ), nghĩa là dễ học và dễ bắt đầu so với React hay Angular. Điều này rất quan trọng khi team có các developer với kỹ năng khác nhau.

Thứ hai, Vue có performance rất tốt. Bundle size nhỏ (khoảng 33KB gzipped), khiến trang load nhanh hơn. Reactivity system của Vue rất efficient, chỉ update những phần giao diện thực sự thay đổi.

Thứ ba, ecosystem của Vue rất phong phú. Có Vue Router cho routing, Pinia cho state management, Vite cho development server nhanh chóng. Tất cả các tools này được thiết kế để hoạt động tốt cùng nhau.

Thứ tư, Vue community rất sôi động và hỗ trợ tốt. Có rất nhiều thư viện, plugin, và tài liệu tiếng Việt.

**Các công cụ liên quan:**

Vue Router là library chính thức của Vue để xử lý routing (điều hướng giữa các trang). Nó cho phép định nghĩa các route, tạo navigation guards để kiểm tra quyền trước khi vào trang, và hỗ trợ lazy loading components.

Pinia là state management library hiện đại, thay thế cho Vuex. Nó cho phép lưu trữ dữ liệu toàn ứng dụng (như thông tin người dùng, danh sách sách) ở một nơi tập trung, giúp tránh truyền props xuyên suốt component tree.

Vite là build tool hiện đại, nhanh hơn Webpack rất nhiều. Vite sử dụng native ES modules trong browser, cho phép hot reload rất nhanh khi phát triển.

Axios là HTTP client library được sử dụng để gọi API backend. Nó cung cấp cú pháp sạch, hỗ trợ promises, interceptors, và request/response transformation.

### 2.3.2. Node.js

**Khái niệm:**

Node.js là một JavaScript runtime environment (môi trường chạy JavaScript) được xây dựng trên V8 engine của Chrome. Trước khi Node.js ra đời, JavaScript chỉ có thể chạy ở phía client (trên trình duyệt). Node.js cho phép chạy JavaScript ở phía server, mở ra khả năng xây dựng toàn bộ ứng dụng web chỉ bằng JavaScript.

**Đặc điểm chính:**

Event-Driven Architecture là nguyên tắc hoạt động cơ bản của Node.js. Node.js sử dụng event loop để xử lý các yêu cầu bất đồng bộ (asynchronous). Khi một yêu cầu đến, Node.js không chờ hoàn thành, mà tiếp tục xử lý các yêu cầu khác, và khi hoàn thành, nó gọi callback. Điều này khiến Node.js có thể xử lý hàng ngàn concurrent connections với một thread duy nhất.

Non-Blocking I/O (Input/Output) là cơ chế xử lý input/output không chặn. Các hoạt động I/O như đọc file, truy vấn database không chặn main thread. Điều này giúp Node.js rất efficient trong việc xử lý các hoạt động I/O nặng.

npm (Node Package Manager) là package manager của Node.js. Nó cho phép tải xuống và cài đặt hàng triệu packages từ npm registry. npm ecosystem là một trong những lý do Node.js thành công, vì có hầu như package nào bạn cần.

**Lý do chọn node.js cho dự án:**

Node.js được chọn vì cho phép sử dụng JavaScript cả ở frontend lẫn backend. Điều này mang lại lợi ích là code reusability, kiến trúc thống nhất, và team chỉ cần học một ngôn ngữ.

Node.js có performance rất tốt cho I/O-heavy applications. Hệ thống mượn sách cần xử lý nhiều database queries và API calls, Node.js rất phù hợp.

Node.js có một ecosystem rất lớn. Express.js là framework web phổ biến nhất, rất nhẹ nhưng đủ mạnh mẽ. Mongoose là library để làm việc với MongoDB rất tiện.

Node.js dễ deploy. Có nhiều hosting services (Heroku, AWS, DigitalOcean) hỗ trợ Node.js. Startup time nhanh, không cần compile như các ngôn ngữ khác.

**Các công cụ liên quan:**

Express.js là web framework nhẹ nhưng mạnh mẽ cho Node.js. Nó cung cấp các tính năng cơ bản: routing, middleware, request/response handling. Express cho phép xây dựng RESTful APIs một cách dễ dàng.

Mongoose là ODM (Object Document Mapper) cho MongoDB. Nó giúp định nghĩa schemas, validate dữ liệu, thiết lập relationships giữa collections. Mongoose chuyển đổi các operations JavaScript thành MongoDB queries.

bcryptjs là library mã hóa mật khẩu. Nó sử dụng adaptive hashing algorithm, rất an toàn chống brute force attacks. Cách bcrypt hoạt động là mỗi lần mã hóa cùng một mật khẩu, kết quả lại khác nhau, vì bcrypt sử dụng salt ngẫu nhiên.

jsonwebtoken là library để tạo và verify JWT tokens. JWT tokens được sử dụng cho xác thực trong REST APIs. Token chứa thông tin người dùng (payload), được ký bằng secret key, và có thể verify mà không cần truy vấn database.

dotenv là library để quản lý environment variables. Nó cho phép đọc các biến từ file .env vào process.env, giúp lưu trữ các thông tin nhạy cảm (API keys, database URLs) mà không hard-code vào code.

### 2.3.3. Visual Studio Code (VS Code)

**Khái niệm:**

Visual Studio Code là một code editor mã nguồn mở được phát triển bởi Microsoft. Đó là một editor văn bản mạnh mẽ, nhẹ, và miễn phí. VS Code được xây dựng dựa trên Electron framework, cho phép nó chạy trên Windows, macOS, và Linux.

**Đặc điểm chính:**

IntelliSense là tính năng code completion thông minh. VS Code phân tích code và cung cấp gợi ý hoàn thành khi bạn gõ. Nó không chỉ gợi ý tên hàm, mà còn hiển thị documentation, parameter hints, và type information.

Integrated Terminal cho phép mở terminal directly ở trong VS Code. Bạn không cần phải mở terminal riêng để chạy commands, build, hoặc run tests.

Debugging Support cho phép debug code directly ở VS Code. Bạn có thể set breakpoints, inspect variables, step through code. Rất hữu ích khi gỡ lỗi.

Git Integration tích hợp sẵn cho phép quản lý source code. Bạn có thể commit, push, pull, resolve conflicts mà không cần terminal.

Extensions Ecosystem là một trong những điểm mạnh của VS Code. Có hàng chục ngàn extensions để mở rộng tính năng: themes, language support, linters, formatters, etc.

**Lý do chọn vs code cho dự án:**

* VS Code là editor phổ biến nhất hiện nay trong cộng đồng developer. Nó miễn phí, mã nguồn mở, và được support rất tốt.
* VS Code có performance tốt. Mặc dù được xây dựng bằng Electron (thường cũng khá nặng), VS Code vẫn rất responsive và load nhanh.
* Có rất nhiều extensions hữu ích cho Vue.js, Node.js, MongoDB development. Ví dụ: Vetur cho Vue support, ES7+ snippets, Prettier cho formatting, ESLint cho linting.
* Integrated Terminal rất tiện, giúp không phải chuyển đổi giữa editor và terminal.
* Remote Development support cho phép phát triển trên remote servers hoặc containers.

**Các extensions hữu ích:**

* Vetur cung cấp support cho Vue.js, bao gồm syntax highlighting, code completion, formatting, debugging.
* ES7+ React/Redux/React-Native snippets cung cấp các code snippets hữu ích cho JavaScript development.
* Prettier là code formatter, tự động format code theo standard, giúp code dễ đọc và nhất quán.
* ESLint là linter để kiểm tra code quality, tìm ra các lỗi tiềm ẩn, enforce coding standards.
* MongoDB for VS Code cho phép connect đến MongoDB servers và quản lý databases trực tiếp ở VS Code.
* Thunder Client hoặc REST Client cho phép test API endpoints trực tiếp ở VS Code mà không cần Postman.

### 2.3.4. Bootstrap 5

**Khái niệm:**

Bootstrap là một CSS framework (bộ thư viện CSS) được phát triển bởi Twitter. Nó cung cấp một tập hợp các pre-built CSS classes, components, và JavaScript utilities để xây dựng responsive websites nhanh chóng. Bootstrap 5 là phiên bản mới nhất, không còn phụ thuộc vào jQuery như phiên bản trước.

**Đặc điểm chính:**

Responsive Grid System là cơ sở của Bootstrap. Nó cung cấp một 12-column grid system, cho phép chia trang thành các cột và tạo layout responsive. Bạn có thể chỉ định số cột khác nhau tùy theo breakpoint (mobile, tablet, desktop).

Pre-built Components là một lợi ích lớn. Bootstrap cung cấp hàng chục components: buttons, forms, navbars, cards, modals, carousels, etc. Bạn có thể sử dụng trực tiếp hoặc customize.

Utility Classes cho phép styling nhanh mà không cần viết CSS. Ví dụ: m-3 cho margin, text-center cho text alignment, bg-primary cho background color. Cách này gọi là Utility-First CSS.

Customizable via SCSS Variables. Bootstrap được viết bằng SCSS, cho phép customize colors, spacing, fonts, breakpoints bằng cách override variables.

**Lý do chọn bootstrap 5 cho dự án:**

Bootstrap 5 được chọn vì nó rất phổ biến, lợi ích lớn nhất là giảm thời gian development. Thay vì phải viết CSS từ đầu, bạn có thể sử dụng pre-built components.

Bootstrap đảm bảo design consistency. Nếu cả team sử dụng Bootstrap, giao diện sẽ nhất quán.

Bootstrap hỗ trợ responsive design sẵn. Bạn không cần phải viết media queries phức tạp, chỉ cần sử dụng responsive classes của Bootstrap (e.g., col-md-6, d-none d-md-block).

Bootstrap không có dependency, không cần jQuery (ở phiên bản 5). Nó chỉ cần CSS và một chút JavaScript cho interactive components.

Bootstrap documentation rất tốt, có rất nhiều ví dụ, dễ tìm kiếm.

**Các components chính sử dụng trong dự án:**

Navbar (Navigation Bar) được sử dụng để hiển thị menu chính ở đầu trang. Bootstrap cung cấp responsive navbar có thể collapse thành hamburger menu ở mobile.

Forms là một phần quan trọng của ứng dụng (login, register, thêm/sửa sách). Bootstrap cung cấp styled form controls: text inputs, selects, checkboxes, radios, với built-in validation styles.

Cards là component linh hoạt để hiển thị content. Trong dự án, cards được sử dụng để hiển thị thông tin sách, độc giả, phiếu mượn.

Tables được sử dụng để hiển thị danh sách (sách, độc giả, lịch sử mượn). Bootstrap tables có striped rows, hover effects, responsive scrolling.

Buttons là elements cơ bản. Bootstrap cung cấp các button styles khác nhau (primary, secondary, danger, success, etc.) và sizes.

Modals là dialog boxes dùng để xác nhận hành động hoặc nhập dữ liệu. Bootstrap modals có animations mượt, accessible.

Alerts/Toasts được sử dụng để hiển thị thông báo. Toasts thường hiển thị ở góc màn hình với auto-dismiss.

Pagination dùng để navigate giữa các trang trong danh sách lớn.

Dropdowns dùng để hiển thị menu options.

Badges được sử dụng để hiển thị status (ví dụ: status của phiếu mượn).

# CHƯƠNG 3: CÁC SƠ ĐỒ CỦA WEBSITE

## 3.1. Phân tích và thiết kế

### 3.1.1. Sơ đồ use case tổng quát

Sơ đồ use case tổng quát Mô tả tất cả các chức năng chính của hệ thống Website Quản Lý Mượn Sách Online từ góc độ người dùng. Use case được sử dụng để Mô tả các tương tác giữa người dùng (actor) và hệ thống.

Hệ thống có ba loại người dùng chính: Độc Giả (Reader), Thủ Thư/Nhân Viên (Librarian), và Quản trị viên (Admin). Mỗi người dùng có các use case khác nhau tùy theo vai trò của họ.

**Use Case cho độc giả:**

Độc giả là người dùng cuối, có thể tham gia vào các hoạt động sau:

* Đăng ký tài khoản (Register)
* Đăng nhập (Login)
* Xem danh sách sách (View Book List)
* Tìm kiếm sách (Search Books)
* Xem chi tiết sách (View Book Details)
* Đăng ký mượn sách (Request to Borrow)
* Xem lịch sử mượn sách (View Borrow History)
* Cập nhật thông tin cá nhân (Update Profile)
* Đổi mật khẩu (Change Password)
* Đăng xuất (Logout)

**Use Case cho thủ thư/nhân viên:**

Thủ thư là nhân viên quản lý thư viện, có thể thực hiện các hoạt động:

* Đăng nhập (Login)
* Xem dashboard (View Dashboard)
* Quản lý sách: Thêm, sửa, xóa, tìm kiếm sách (Manage Books)
* Quản lý nhà xuất bản: Thêm, sửa, xóa (Manage Publisher)
* Quản lý độc giả: Xem, sửa, khóa/mở khóa tài khoản (Manage Readers)
* Xem yêu cầu mượn sách (View Borrow Requests)
* Duyệt mượn sách (Approve Borrow Request)
* Từ chối mượn sách (Reject Borrow Request)
* Ghi nhận sách trả (Record Book Return)
* Tính tiền phạt quá hạn (Calculate Fine)
* Xem lịch sử mượn sách (View Borrow History)
* Đăng xuất (Logout)

**Use Case cho Quản trị viên:**

Admin có tất cả các quyền của thủ thư, cộng thêm:

* Quản lý tài khoản nhân viên: Tạo, sửa, xóa, phân quyền (Manage Staff Accounts)
* Xem báo cáo thống kê (View Reports and Statistics)

### 3.1.2. Đặc tả chức năng sơ đồ use case tổng quát

**Mô tả chung:**

Sơ đồ use case tổng quát thể hiện mối quan hệ giữa các actor (Độc Giả, Thủ Thư, Admin) và các use case (chức năng) trong hệ thống. Mỗi use case đại diện cho một tính năng hoặc chức năng mà hệ thống cung cấp.

**Các thành phần chính:**

1. **Actors (Người dùng)**
   * Reader (Độc Giả): Người mượn sách
   * Librarian (Thủ Thư): Nhân viên thư viện quản lý mượn/trả
   * Admin (Quản trị viên): Quản lý toàn bộ hệ thống
2. **Use Cases (Chức năng)**
   * Use case chung (có thể thực hiện bởi nhiều actors)
   * Use case riêng (chỉ một actor có thể thực hiện)
   * Include relationship: Một use case bao gồm use case khác
   * Extend relationship: Một use case mở rộng use case khác
3. **System Boundary (ranh giới hệ thống)**
   * Giới hạn của hệ thống được biểu diễn bằng một hộp chữ nhật
   * Tất cả use cases nằm trong ranh giới này

**Các mối quan hệ:**

* **Association**: Mối quan hệ giữa actor và use case (được biểu diễn bằng một đường thẳng)
* **Include**: Một use case bao gồm một use case khác là bắt buộc (ví dụ: Mượn Sách bao gồm Xác Thực, được biểu diễn bằng mũi tên với nhãn "<<include>>")
* **Extend**: Một use case mở rộng một use case khác trong những trường hợp đặc biệt (ví dụ: Tính Tiền Phạt là mở rộng của Ghi Nhận Trả Sách, được biểu diễn bằng mũi tên với nhãn "<<extend>>")
* **Generalization**: Một actor là tổng quát hơn actor khác (ví dụ: Admin là tổng quát của Librarian)

## 3.2. Sơ đồ use case đăng ký và đăng nhập

### 3.2.1. Đặc tả use case đăng ký

**Tên Use Case**: Register (Đăng ký tài khoản)

**Actor**: Reader (Độc giả)

Bảng 3.1: Đặc tả use case đăng ký tài khoản

| **Khía cạnh** | **Mô tả** |
| --- | --- |
| **Mô tả** | Người dùng chưa có tài khoản có thể đăng ký một tài khoản mới bằng cách cung cấp các thông tin cá nhân cần thiết. |
| **Điều kiện tiên quyết** | - Người dùng chưa có tài khoản trong hệ thống  - Người dùng có quyền truy cập vào trang đăng ký |
| **Tác nhân chính** | Độc Giả (Reader) |
| **Luồng chính** | 1. Người dùng truy cập trang đăng ký  2. Hệ thống hiển thị form đăng ký với các trường: Họ Lót, Tên, Email, Mật Khẩu, Xác Nhận Mật Khẩu, Ngày Sinh, Giới Tính, Địa Chỉ, Số Điện Thoại  3. Người dùng nhập đầy đủ các thông tin  4. Người dùng nhấn nút "Đăng Ký"  5. Hệ thống kiểm tra tính hợp lệ của các trường (email phải là định dạng email hợp lệ, mật khẩu phải có độ dài tối thiểu, hai mật khẩu phải khớp)  6. Hệ thống kiểm tra xem email có đã tồn tại trong cơ sở dữ liệu không  7. Hệ thống mã hóa mật khẩu bằng bcryptjs  8. Hệ thống tạo một document mới trong collection DocGia  9. Hệ thống trả về thông báo thành công  10. Hệ thống tự động chuyển hướng người dùng tới trang đăng nhập |
| **Luồng ngoại lệ** | **Lỗi 1: Email đã tồn tại** - Ở bước 6, nếu email đã tồn tại, hệ thống hiển thị thông báo lỗi "Email này đã được đăng ký"  **Lỗi 2: Dữ liệu không hợp lệ** - Ở bước 5, nếu dữ liệu không hợp lệ, hệ thống hiển thị thông báo lỗi chi tiết  **Lỗi 3: Lỗi server** - Ở bước 8, nếu có lỗi khi tạo tài khoản, hệ thống hiển thị thông báo lỗi "Có lỗi xảy ra, vui lòng thử lại sau" |
| **Hậu điều kiện** | - Tài khoản mới được tạo trong hệ thống  - Người dùng được chuyển hướng tới trang đăng nhập  - Email đó không thể được đăng ký lại |
| **Ghi chú** | - Mật khẩu phải có độ dài tối thiểu 6 ký tự  - Email phải là định dạng email hợp lệ (ví dụ: [user@example.com](vscode-file://vscode-app/d:/Program%20Files/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html))  - Số điện thoại phải có định dạng hợp lệ |

### 3.2.2. Đặc Tả Use Case Đăng Nhập

**Tên Use Case**: Login (Đăng Nhập)

**Actors**: Reader (Độc Giả), Librarian (Thủ Thư), Admin (Quản trị viên)

Bảng 3.2: Đặc tả use case đăng nhập

| **Khía cạnh** | **Mô tả** |
| --- | --- |
| **Mô tả** | Người dùng đã có tài khoản có thể đăng nhập vào hệ thống bằng cách cung cấp email và mật khẩu. |
| **Điều kiện tiên quyết** | - Người dùng có tài khoản trong hệ thống  - Người dùng chưa đăng nhập (chưa có token JWT hợp lệ) |
| **Tác nhân chính** | Độc Giả, Thủ Thư, Admin |
| **Luồng chính** | 1. Người dùng truy cập trang đăng nhập  2. Hệ thống hiển thị form đăng nhập với các trường: Email, Mật Khẩu  3. Người dùng nhập email và mật khẩu  4. Người dùng nhấn nút "Đăng Nhập"  5. Hệ thống gửi request tới API backend  6. Backend tìm kiếm người dùng theo email trong cơ sở dữ liệu (DocGia hoặc NhanVien)  7. Backend kiểm tra xem người dùng đó có tồn tại không  8. Backend so sánh mật khẩu nhập vào với mật khẩu được mã hóa trong cơ sở dữ liệu bằng bcrypt  9. Backend tạo JWT token chứa userId, email, và vai trò (role)  10. Backend gửi lại token, thông tin người dùng, và vai trò  11. Frontend lưu token vào localStorage  12. Frontend lưu thông tin người dùng vào Pinia store (auth store)  13. Hệ thống hiển thị thông báo thành công  14. Hệ thống chuyển hướng người dùng tới trang chủ phù hợp với vai trò (Admin/Librarian → Dashboard, Reader → Book List) |
| **Luồng ngoại lệ** | **Lỗi 1: Email không tồn tại** - Ở bước 7, nếu email không tìm thấy, backend trả về lỗi 401 Unauthorized, frontend hiển thị "Email hoặc mật khẩu không đúng"  **Lỗi 2: Mật khẩu sai** - Ở bước 8, nếu mật khẩu không khớp, backend trả về lỗi 401, frontend hiển thị thông báo lỗi  **Lỗi 3: Tài khoản bị khóa** - Ở bước 7, nếu TrangThai = "Khóa", backend trả về lỗi, frontend hiển thị "Tài khoản của bạn đã bị khóa"  **Lỗi 4: Server error** - Ở bước 6, nếu có lỗi kết nối database, frontend hiển thị "Có lỗi xảy ra, vui lòng thử lại sau" |
| **Hậu điều kiện** | - Người dùng được xác thực thành công  - JWT token được lưu và có thể sử dụng cho các request tiếp theo  - Người dùng được chuyển hướng tới trang phù hợp với vai trò của họ |
| **Ghi chú** | - Token JWT có hạn sử dụng 7 ngày. Sau đó, người dùng cần đăng nhập lại  - Mật khẩu không bao giờ được gửi về frontend (chỉ token)  - Khi token hết hạn, hệ thống có thể tự động refresh token (nếu có cơ chế refresh token) |

## 3.3. Sơ đồ use case tìm kiếm và mượn sách

### 3.3.1. Đặc tả chức năng tìm kiếm

**Tên use case**: search books (tìm kiếm sách)

**Actor**: reader (độc giả)

Bảng 3.3: Đặc tả use case tìm kiếm

| **Khía cạnh** | **Mô tả** |
| --- | --- |
| **Mô tả** | Độc giả có thể tìm kiếm sách theo nhiều tiêu chí khác nhau như tên sách, tên tác giả, mã ISBN, hoặc nhà xuất bản. |
| **Điều kiện tiên quyết** | - Người dùng đang ở trang xem danh sách sách hoặc trang tìm kiếm  - Hệ thống có ít nhất một cuốn sách trong cơ sở dữ liệu |
| **Tác nhân chính** | Độc Giả (Reader) |
| **Luồng chính** | 1. Người dùng truy cập trang danh sách sách hoặc trang tìm kiếm  2. Hệ thống hiển thị danh sách sách với 10-20 quyển trên mỗi trang, kèm thông tin cơ bản (tên sách, tác giả, nhà xuất bản, số quyển còn lại)  3. Người dùng nhập từ khóa tìm kiếm vào ô search (ví dụ: "JavaScript")  4. Hệ thống tự động gửi request tới API với từ khóa tìm kiếm  5. Backend tìm kiếm trong cơ sở dữ liệu những sách có tên, tác giả, hoặc ISBN chứa từ khóa  6. Backend trả về danh sách sách phù hợp (tối đa 20 quyển mỗi trang)  7. Frontend hiển thị kết quả tìm kiếm  8. Người dùng có thể nhấp vào một cuốn sách để xem chi tiết |
| **Luồng ngoại lệ** | **Không tìm thấy kết quả** - Ở bước 6, nếu không tìm thấy sách nào phù hợp, backend trả về danh sách rỗng, frontend hiển thị "Không tìm thấy sách nào phù hợp với tiêu chí tìm kiếm"  **Lỗi server** - Ở bước 5, nếu có lỗi kết nối database, frontend hiển thị thông báo lỗi |
| **Hậu điều kiện** | - Danh sách sách phù hợp được hiển thị  - Người dùng có thể chọn một cuốn sách để xem chi tiết hoặc mượn |
| **Ghi chú** | - Tìm kiếm là case-insensitive (không phân biệt hoa/thường)  - Có thể lọc kết quả theo nhà xuất bản hoặc năm xuất bản |

### 3.3.2. Đặc tả chức năng xem chi tiết sách

**Tên use case**: view book details (xem chi tiết sách)

**Actor**: reader (độc giả)

Bảng 3.4: Đặc tả use case xem chi tiết sách

| **Khía cạnh** | **Mô tả** |
| --- | --- |
| **Mô tả** | Độc giả có thể xem thông tin chi tiết về một cuốn sách cụ thể, bao gồm tên, tác giả, nhà xuất bản, giá tiền, số quyển còn lại. Từ trang này, độc giả có thể quyết định có mượn sách hay không. |
| **Điều kiện tiên quyết** | - Độc giả đã xem danh sách sách hoặc kết quả tìm kiếm  - Sách tồn tại trong cơ sở dữ liệu |
| **Tác nhân chính** | Độc giả (reader) |
| **Luồng chính** | 1. Người dùng nhấp vào một cuốn sách từ danh sách  2. Hệ thống gửi request tới API để lấy chi tiết sách (bao gồm cả thông tin nhà xuất bản)  3. Backend lấy thông tin sách từ cơ sở dữ liệu và populate thông tin nhà xuất bản  4. Backend trả về chi tiết sách đầy đủ  5. Frontend hiển thị trang chi tiết với thông tin: Tên Sách, Tác Giả, Nhà Xuất Bản, Năm Xuất Bản, ISBN, Giá Tiền, Số Quyển Còn Lại  6. Nếu sách còn quyển, hệ thống hiển thị nút "Mượn Sách"  7. Nếu sách hết quyển, hệ thống hiển thị thông báo "Sách hết quyển"  8. Người dùng có thể nhấp nút "Mượn Sách" để tiến hành đăng ký mượn |
| **Luồng ngoại lệ** | **Sách không tồn tại** - Ở bước 3, nếu sách không tìm thấy (bị xóa), backend trả về lỗi 404  **Lỗi server** - Ở bước 3, nếu có lỗi database, frontend hiển thị thông báo lỗi |
| **Hậu điều kiện** | - Thông tin chi tiết sách được hiển thị  - Người dùng có thể tiến hành mượn sách nếu còn quyển |
| **Ghi chú** | - Hiển thị giá tiền để độc giả tham khảo  - Số quyển còn lại giúp độc giả biết tính sẵn có |

### 3.3.3. Đặc tả chức năng đăng ký mượn sách

**Tên use case**: request to borrow book (đăng ký mượn sách)

**Actor**: reader (độc giả)

Bảng 3.5: Đặc tả use case đăng ký mượn sách

| **Khía cạnh** | **Mô tả** |
| --- | --- |
| **Mô tả** | Độc giả có thể gửi yêu cầu mượn một cuốn sách. Yêu cầu sẽ được ghi nhận trong hệ thống và chờ nhân viên thư viện duyệt. |
| **Điều kiện tiên quyết** | - Độc giả đã đăng nhập  - Độc giả đang xem chi tiết một cuốn sách  - Sách còn quyển để mượn  - Tài khoản độc giả không bị khóa  - Độc giả chưa mượn quá 5 quyển cùng lúc |
| **Tác nhân chính** | Độc Giả (Reader) |
| **Luồng chính** | 1. Độc giả nhấp nút "Mượn Sách" từ trang chi tiết sách  2. Hệ thống hiển thị dialog/form chọn hạn trả sách (mặc định 30 ngày từ ngày hôm nay)  3. Độc giả có thể chọn ngày hạn trả hoặc giữ mặc định  4. Độc giả nhấp nút "Xác Nhận Mượn"  5. Hệ thống kiểm tra điều kiện:  - Sách còn quyển không?  - Độc giả đã mượn bao nhiêu quyển?  - Tài khoản độc giả có bị khóa không?  6. Hệ thống tạo một document mới trong collection TheoDoiMuonSach với:  - MaDocGia: ID của độc giả  - MaSach: ID của sách  - NgayHanTra: Ngày hạn trả được chọn  - TrangThai: "Chờ duyệt"  7. Hệ thống trả về thông báo thành công  8. Hệ thống hiển thị thông báo "Yêu cầu mượn sách đã được gửi. Vui lòng chờ nhân viên duyệt"  9. Frontend cập nhật giao diện để hiện thị trạng thái "Đang chờ duyệt" |
| **Luồng ngoại lệ** | **Lỗi 1: Sách hết quyển** - Ở bước 5, nếu số quyển còn lại ≤ 0, backend trả về lỗi, frontend hiển thị "Sách hết quyển, không thể mượn"  **Lỗi 2: Độc giả đã mượn quá 5 quyển** - Ở bước 5, backend đếm số sách đang mượn, nếu >= 5, trả về lỗi  **Lỗi 3: Tài khoản bị khóa** - Ở bước 5, nếu TrangThai = "Khóa", backend trả về lỗi  **Lỗi 4: Độc giả chưa xác thực token** - Ở bước 4, nếu token không hợp lệ, backend trả về lỗi 401 |
| **Hậu điều kiện** | - Yêu cầu mượn sách được tạo trong hệ thống  - Yêu cầu có trạng thái "Chờ duyệt"  - Yêu cầu sẽ được hiển thị trong danh sách lịch sử mượn của độc giả  - Nhân viên sẽ thấy yêu cầu này trong danh sách "Yêu cầu mượn chờ xử lý" |
| **Ghi chú** | - Độc giả có thể chọn hạn trả từ 7 đến 60 ngày  - Mặc định hạn trả là 30 ngày  - Một độc giả không thể mượn cùng một cuốn sách quá một lần trong danh sách "Chờ duyệt" hoặc "Đang mượn" |

## 3.4. Sơ đồ use case quản lý sách (cho nhân viên)

### 3.4.1. Đặc tả sơ đồ quản lý sách

**Tên use case**: manage books (quản lý sách)

**Actor**: librarian (thủ thư), admin (Quản trị viên)

**Mô tả tổng quát**: Nhân viên thư viện có thể quản lý danh mục sách trong hệ thống, bao gồm Thêm sách mới, sửa đổi thông tin sách, xóa sách, và tìm kiếm sách.

Bảng 3.6: Đặc tả use case quản lý sách

| **Khía cạnh** | **Mô tả** |
| --- | --- |
| **Điều kiện tiên quyết** | - Người dùng đã đăng nhập với tư cách là Librarian hoặc Admin  - Người dùng có quyền truy cập vào trang quản lý sách |
| **Tác nhân chính** | Librarian (Thủ Thư), Admin (Quản trị viên) |

**Use case con 1: add book (thêm sách mới)**

Bảng 3.7: Đặc tả use case thêm sách mới

| **Khía cạnh** | ***Mô tả*** |
| --- | --- |
| **Mô tả** | Nhân viên có thể thêm một cuốn sách mới vào hệ thống với các thông tin cần thiết. |
| **Luồng chính** | 1. Nhân viên truy cập trang quản lý sách  2. Nhân viên nhấp nút "Thêm sách"  3. Hệ thống hiển thị form thêm sách với các trường: Tên sách, ISBN, giá tiền, số quyển, năm xuất bản, nhà xuất bản (dropdown), tác giả/nguồn gốc  4. Nhân viên nhập các thông tin  5. Nhân viên nhấp nút "Lưu"  6. Hệ thống kiểm tra tính hợp lệ:  - Tên sách không rỗng  - Giá tiền >= 0  - Số quyển >= 0  - ISBN (nếu nhập) phải là định dạng ISBN hợp lệ  7. Hệ thống tạo document mới trong collection Sach  8. Hệ thống trả về thông báo thành công  9. Trang quản lý sách được cập nhật, sách mới hiển thị trong danh sách |
| **Luồng ngoại lệ** | **Lỗi validation** - Nếu dữ liệu không hợp lệ, hệ thống hiển thị thông báo lỗi cụ thể  **ISBN đã tồn tại** - Nếu ISBN đã được nhập, hệ thống cảnh báo |
| **Hậu điều kiện** | - Sách mới được thêm vào hệ thống  - Số quyển của sách mới = giá trị nhập vào  - Sách có thể được độc giả tìm kiếm và mượn |

**Use case con 2: edit book (sửa sách)**

Bảng 3.8: Đặc tả use case sửa sách

| **Khía cạnh** | **Mô tả** |
| --- | --- |
| **Mô tả** | Nhân viên có thể sửa đổi thông tin của một cuốn sách đã tồn tại. |
| **Luồng chính** | 1. Nhân viên nhấp vào nút "Sửa" trên một cuốn sách trong danh sách  2. Hệ thống lấy thông tin sách từ cơ sở dữ liệu  3. Hệ thống hiển thị form sửa sách với các thông tin hiện tại  4. Nhân viên sửa đổi các thông tin cần thiết  5. Nhân viên nhấp nút "Lưu"  6. Hệ thống kiểm tra tính hợp lệ  7. Hệ thống cập nhật document trong collection Sach  8. Hệ thống trả về thông báo thành công  9. Danh sách sách được cập nhật |
| **Luồng ngoại lệ** | **Lỗi validation** - Nếu dữ liệu không hợp lệ, hệ thống hiển thị thông báo lỗi  **Sách không tồn tại** - Nếu sách bị xóa, hệ thống hiển thị lỗi 404 |
| **Hậu điều kiện** | - Thông tin sách được cập nhật  - Thay đổi hiển thị ngay trên danh sách  - Độc giả sẽ thấy thông tin mới khi xem sách |

**Use case con 3: delete book (xóa sách)**

Bảng 3.9: Đặc tả use case xóa sách

| **Khía cạnh** | **Mô tả** |
| --- | --- |
| **Mô tả** | Nhân viên có thể xóa một cuốn sách khỏi hệ thống (nhưng chỉ nếu sách không được mượn). |
| **Luồng chính** | 1. Nhân viên nhấp vào nút "Xóa" trên một cuốn sách  2. Hệ thống hiển thị hộp thoại xác nhận "Bạn có chắc chắn muốn xóa sách này?"  3. Nhân viên nhấp "Xác Nhận"  4. Hệ thống kiểm tra: Có phiếu mượn của sách này không?  5. Nếu có phiếu mượn, hệ thống hiển thị cảnh báo "Không thể xóa sách vì đang có phiếu mượn"  6. Nếu không có phiếu mượn, hệ thống xóa document khỏi collection Sach  7. Danh sách sách được cập nhật |
| **Luồng ngoại lệ** | **Không thể xóa sách đang được mượn** - Ở bước 5, nếu sách có phiếu mượn, nhân viên không thể xóa |
| **Hậu điều kiện** | - Sách bị xóa khỏi hệ thống  - Sách không còn trong danh sách  - Độc giả không thể tìm kiếm và mượn sách này nữa |

**Use case con 4: search and filter books (tìm kiếm và lọc sách)**

Bảng 3.10: Đặc tả use case tìm kiếm và lọc sách

| **Khía cạnh** | **Mô tả** |
| --- | --- |
| **Mô tả** | Nhân viên có thể tìm kiếm sách theo nhiều tiêu chí để quản lý dễ dàng hơn. |
| **Luồng chính** | 1. Nhân viên nhập từ khóa tìm kiếm vào ô search  2. Hệ thống tìm kiếm theo tên sách, tác giả, ISBN  3. Hệ thống cũng cho phép lọc theo nhà xuất bản, năm xuất bản  4. Danh sách sách được cập nhật để hiển thị kết quả |
| **Luồng ngoại lệ** | **Không tìm thấy kết quả** - Nếu không có sách nào phù hợp, hiển thị "Không tìm thấy sách nào" |
| **Hậu điều kiện** | - Danh sách sách được lọc theo tiêu chí  - Nhân viên có thể dễ dàng tìm được sách cần quản lý |

## 3.5. Sơ đồ use case xử lý mượn/trả sách

### 3.5.1. Đặc tả use case xử lý mượn/trả sách

**Tên use case**: handle book borrow/return (xử lý mượn/trả sách)

**Actor**: librarian (thủ thư), admin (quản trị viên)

**Mô tả tổng quát**: Nhân viên thư viện có thể xem danh sách yêu cầu mượn sách từ độc giả, duyệt hoặc từ chối các yêu cầu, ghi nhận sách trả, và tính tiền phạt nếu có.

**Use case con 1: view borrow requests (xem yêu cầu mượn)**

Bảng 3.11: Đặc tả use case xem yêu cầu mượn sách

| **Khía cạnh** | **Mô tả** |
| --- | --- |
| **Mô tả** | Nhân viên có thể xem danh sách tất cả yêu cầu mượn sách từ độc giả. |
| **Luồng chính** | 1. Nhân viên truy cập trang xử lý mượn/trả sách  2. Hệ thống hiển thị danh sách yêu cầu mượn với trạng thái "Chờ duyệt"  3. Danh sách hiển thị: Tên Độc Giả, Tên Sách, Ngày Yêu Cầu, Hạn Trả Được Chọn  4. Nhân viên có thể lọc theo:  - Trạng thái (Chờ duyệt, Đang mượn, Đã trả, Quá hạn)  - Độc giả  - Ngày  5. Nhân viên có thể sắp xếp theo ngày yêu cầu, ngày hạn trả |
| **Luồng ngoại lệ** | **Không có yêu cầu** - Nếu danh sách rỗng, hiển thị "Không có yêu cầu mượn" |
| **Hậu điều kiện** | - Danh sách yêu cầu được hiển thị  - Nhân viên có thể dễ dàng tìm yêu cầu cần xử lý |

**Use case con 2: approve borrow request (duyệt yêu cầu mượn)**

Bảng 3.12: Đặc tả use case duyệt yêu cầu mượn

| **Khía cạnh** | **Mô tả** |
| --- | --- |
| **Mô tả** | Nhân viên có thể duyệt một yêu cầu mượn sách, thay đổi trạng thái từ "Chờ duyệt" sang "Đang mượn" và ghi nhận ngày mượn. |
| **Luồng chính** | 1. Nhân viên nhấp vào một yêu cầu mượn  2. Hệ thống hiển thị chi tiết yêu cầu  3. Nhân viên nhấp nút "Duyệt"  4. Hệ thống cập nhật:  - TrangThai: "Đang mượn"  - NgayMuon: Ngày hôm nay  5. Hệ thống trả về thông báo thành công  6. Yêu cầu được chuyển sang trạng thái "Đang mượn" |
| **Luồng ngoại lệ** | **Sách không còn quyển** - Nếu trong lúc chờ, sách hết quyển, nhân viên không thể duyệt. Hệ thống hiển thị cảnh báo  **Độc giả bị khóa** - Nếu độc giả bị khóa, không thể duyệt |
| **Hậu điều kiện** | - Phiếu mượn chuyển sang trạng thái "Đang mươn"  - Số quyển sách giảm đi 1  - Độc giả có thể xem phiếu mượn ở trang lịch sử |

**Use case con 3: reject borrow request (từ chối yêu cầu mượn)**

Bảng 3.13: Đặc tả use case từ chối yêu cầu mượn

| **Khía cạnh** | **Mô tả** |
| --- | --- |
| **Mô tả** | Nhân viên có thể từ chối một yêu cầu mượn sách với lý do cụ thể. |
| **Luồng chính** | 1. Nhân viên nhấp vào một yêu cầu mượn  2. Nhân viên nhấp nút "Từ chối"  3. Hệ thống hiển thị form nhập lý do từ chối  4. Nhân viên nhập lý do (ví dụ: "Sách bị hư hỏng")  5. Nhân viên nhấp nút "Xác nhận"  6. Hệ thống cập nhật:  - TrangThai: "Từ chối"  - GhiChu: Lý do từ chối  7. Yêu cầu được chuyển sang trạng thái "Từ chối" |
| **Luồng ngoại lệ** | **Không nhập lý do** - Nếu nhân viên không nhập lý do, hệ thống yêu cầu nhập |
| **Hậu điều kiện** | - Phiếu mượn có trạng thái "Từ chối"  - Độc giả có thể xem lý do từ chối ở lịch sử  - Độc giả có thể thử mượn sách khác |

**Use case con 4: record book return (ghi nhận sách trả)**

Bảng 3.14: Đặc tả use case ghi nhận sách trả

| **Khía cạnh** | **Mô tả** |
| --- | --- |
| **Mô tả** | Nhân viên ghi nhận sách được trả, kiểm tra tình trạng sách, và tính tiền phạt nếu trả quá hạn. |
| **Luồng chính** | 1. Độc giả mang sách đến trả  2. Nhân viên tìm kiếm phiếu mượn tương ứng  3. Nhân viên nhấp vào phiếu mượn có trạng thái "Đang mượn"  4. Nhân viên nhấp nút "Ghi Nhận Trả"  5. Hệ thống hiển thị form ghi nhận trả với các trường:  - NgayTra: Ngày hôm nay (có thể sửa)  - TinhTrangSach: Chọn từ (Tốt, Bình thường, Hư hỏng, Mất)  - GhiChu: Ghi chú thêm (nếu cần)  6. Nhân viên chọn tình trạng sách  7. Nhân viên nhấp nút "Lưu"  8. Hệ thống kiểm tra:  - NgayTra >= NgayMuon?  9. Hệ thống tính tiền phạt (nếu có):  - Nếu NgayTra > NgayHanTra:  - SoNgayQuaHan = NgayTra - NgayHanTra  - TienPhat = SoNgayQuaHan \* 10,000 VND  - Nếu NgayTra <= NgayHanTra: TienPhat = 0  10. Hệ thống cập nhật document TheoDoiMuonSach:  - NgayTra: Ngày trả  - TrangThai: "Đã trả" (hoặc "Quá hạn" nếu quá hạn)  - TinhTrangSach: Tình trạng được chọn  - TienPhat: Số tiền phạt  11. Hệ thống trả về thông báo thành công (kèm thông báo tiền phạt nếu có)  12. Giao diện cập nhật, phiếu mượn không còn trong danh sách "Đang mượn" |
| **Luồng ngoại lệ** | **Lỗi ngày trả** - Nếu ngày trả < ngày mượn, hệ thống hiển thị cảnh báo  **Lỗi server** - Nếu có lỗi khi lưu, hiển thị thông báo lỗi |
| **Hậu điều kiện** | - Phiếu mượn được cập nhật với thông tin trả sách  - Tiền phạt được tính và ghi nhận (nếu có)  - Độc giả có thể thấy phiếu này trong lịch sử mượn của mình  - Số quyển sách tăng lên 1 |

## 3.6. Mô hình cơ sở dữ liệu

### 3.6.1. Mô tả các lớp (Entities)

Hệ thống được xây dựng dựa trên 5 entities (lớp dữ liệu) chính trong MongoDB:

* **Entity DocGia (Độc giả)**: Đại diện cho người mượn sách. Mỗi độc giả có thông tin cá nhân, tài khoản đăng nhập, trạng thái tài khoản.
* **Entity sach (Sách)**: Đại diện cho một cuốn sách (hay một loại sách). Chứa thông tin về tên, giá, số quyển, tác giả, và liên kết với nhà xuất bản.
* **Entity NhaXuatBan (Nhà xuất bản)**: Đại diện cho nhà xuất bản. Một cuốn sách thuộc về một nhà xuất bản (có thể một nhà xuất bản xuất bản nhiều quyển sách).
* **Entity TheoDoiMuonSach (Theo dõi mượn sách)**: Đại diện cho một phiếu mượn. Ghi nhận thông tin về độc giả nào mượn sách nào, vào ngày nào, trả vào ngày nào, trạng thái mượn.
* **Entity NhanVien (Nhân viên)**: Đại diện cho nhân viên thư viện. Chứa thông tin cá nhân, tài khoản đăng nhập, chức vụ.

### 3.6.2. Mô tả User (Người dùng)

**Collection: DocGia**

DocGia (Độc giả) là collection lưu trữ thông tin của người mượn sách. Mỗi document trong collection này đại diện cho một độc giả.

Bảng 3.15: Mô tả bảng Độc giả

| **Tên trường** | **Kiểu dữ liệu** | **Mô tả** | **Ràng buộc** |
| --- | --- | --- | --- |
| \_id | ObjectId | Mã duy nhất của độc giả | Primary Key |
| HoLot | String | Họ lót độc giả | Required, Max 50 |
| Ten | String | Tên độc giả | Required, Max 50 |
| Email | String | Email dùng để đăng nhập | Required, Unique, Email format |
| Password | String | Mật khẩu được mã hóa bằng bcrypt | Required, Bcrypt 10 rounds |
| NgaySinh | Date | Ngày sinh | Optional |
| Phai | String | Giới tính (Nam/Nữ) | Optional |
| DiaChi | String | Địa chỉ cư trú | Required, Max 200 |
| DienThoai | String | Số điện thoại | Required, Unique, Phone format |
| TrangThai | String | Trạng thái tài khoản (Hoạt động/Khóa) | Default: "Hoạt động" |
| NgayDangKy | Date | Ngày đăng ký tài khoản | Default: now() |
| createdAt | Date | Thời gian tạo document | Auto |
| updatedAt | Date | Thời gian cập nhật lần cuối | Auto |

**Collection: NhanVien**

NhanVien (Nhân Viên) là collection lưu trữ thông tin của nhân viên thư viện (gồm Thủ Thư và Admin).

Bảng 3.16: Mô tả bảng Nhân viên

| **Tên trường** | **Kiểu dữ liệu** | **Mô tả** | **Ràng buộc** |
| --- | --- | --- | --- |
| \_id | ObjectId | Mã duy nhất của nhân viên | Primary Key |
| HoTenNV | String | Họ tên nhân viên | Required, Max 100 |
| Email | String | Email đăng nhập | Required, Unique, Email format |
| Password | String | Mật khẩu được mã hóa | Required, Bcrypt 10 rounds |
| Chucvu | String | Chức vụ (Admin/Librarian) | Required |
| DiaChi | String | Địa chỉ | Optional, Max 200 |
| SoDienThoai | String | Số điện thoại | Optional, Unique |
| NgayThamGia | Date | Ngày tham gia | Default: now() |
| createdAt | Date | Thời gian tạo document | Auto |
| updatedAt | Date | Thời gian cập nhật | Auto |

### 3.6.3. Mô tả book (sách)

**Collection: Sach**

Sach (Sách) là collection lưu trữ thông tin về các cuốn sách trong thư viện.

Bảng 3.17: Mô tả bảng Sách

| **Tên trường** | **Kiểu dữ liệu** | **Mô tả** | **Ràng buộc** |
| --- | --- | --- | --- |
| \_id | ObjectId | Mã duy nhất của sách | Primary Key |
| TenSach | String | Tên cuốn sách | Required, Max 200 |
| ISBN | String | Mã ISBN | Unique, Optional |
| DonGia | Number | Giá tiền (VND) | Required, > 0 |
| SoQuyen | Number | Số quyển hiện có | Required, >= 0 |
| NamXuatBan | Number | Năm xuất bản | Optional, 4 digits |
| MaNXB | ObjectId | ID nhà xuất bản | Foreign Key → NhaXuatBan |
| NguonGoc\_TacGia | String | Tác giả hoặc nguồn gốc sách | Optional, Max 200 |
| NgayThem | Date | Ngày Thêm sách vào thư viện | Default: now() |
| createdAt | Date | Thời gian tạo | Auto |
| updatedAt | Date | Thời gian cập nhật | Auto |

**Collection: NhaXuatBan**

NhaXuatBan (Nhà Xuất Bản) là collection lưu trữ thông tin về các nhà xuất bản.

Bảng 3.18: Mô tả bảng Nhà xuất bản

| **Tên trường** | **Kiểu dữ liệu** | **Mô tả** | **Ràng buộc** |
| --- | --- | --- | --- |
| \_id | ObjectId | Mã duy nhất của nhà xuất bản | Primary Key |
| TenNXB | String | Tên nhà xuất bản | Required, Unique, Max 200 |
| DiaChi | String | Địa chỉ nhà xuất bản | Required, Max 300 |
| createdAt | Date | Thời gian tạo | Auto |
| updatedAt | Date | Thời gian cập nhật | Auto |

### 3.6.4. Mô tả borrow record (phiếu mượn)

**Collection: TheoDoiMuonSach**

TheoDoiMuonSach (theo dõi mượn sách) là collection lưu trữ thông tin về các phiếu mượn sách.

Bảng 3.19: Mô tả bảng theo dõi mượn sách

| **Tên trường** | **Kiểu dữ liệu** | **Mô tả** | **Ràng buộc** |
| --- | --- | --- | --- |
| \_id | ObjectId | Mã duy nhất của phiếu mượn | Primary Key |
| MaDocGia | ObjectId | ID độc giả | Foreign Key → DocGia.\_id, Required |
| MaSach | ObjectId | ID sách | Foreign Key → Sach.\_id, Required |
| NgayMuon | Date | Ngày mượn | Optional (ghi nhận khi duyệt) |
| NgayHanTra | Date | Ngày hạn trả (độc giả chọn) | Optional, Required khi đăng ký |
| NgayTra | Date | Ngày thực tế trả | Optional |
| TrangThai | String | Trạng thái phiếu mượn | Default: "Chờ duyệt", Enum: [Chờ duyệt, Đang mượn, Đã trả, Quá hạn, Từ chối] |
| TinhTrangSach | String | Tình trạng sách khi trả | Enum: [Tốt, Bình thường, Hư hỏng, Mất], Optional |
| TienPhat | Number | Tiền phạt quá hạn (VND) | Default: 0, >= 0 |
| GhiChu | String | Ghi chú thêm | Optional, Max 500 |
| createdAt | Date | Ngày tạo phiếu mượn | Auto |
| updatedAt | Date | Ngày cập nhật lần cuối | Auto |

**Các quan hệ (Relationships):**

Bảng 3.20: Mô tả các quan hệ giữa các bảng

| **Quan hệ** | **Mô tả** |
| --- | --- |
| **1 NhaXuatBan : N Sach** (One-to-Many) | Một nhà xuất bản có thể xuất bản nhiều cuốn sách. MaSach tham chiếu đến \_id của NhaXuatBan |
| **1 DocGia : N TheoDoiMuonSach** (One-to-Many) | Một độc giả có thể mượn nhiều phiếu sách. MaDocGia tham chiếu đến \_id của DocGia |
| **1 Sach : N TheoDoiMuonSach** (One-to-Many) | Một cuốn sách có thể được mượn trong nhiều phiếu. MaSach tham chiếu đến \_id của Sach |

# CHƯƠNG 4: THIẾT KẾ GIAO DIỆN

## 4.1. Giao diện Quản trị viên

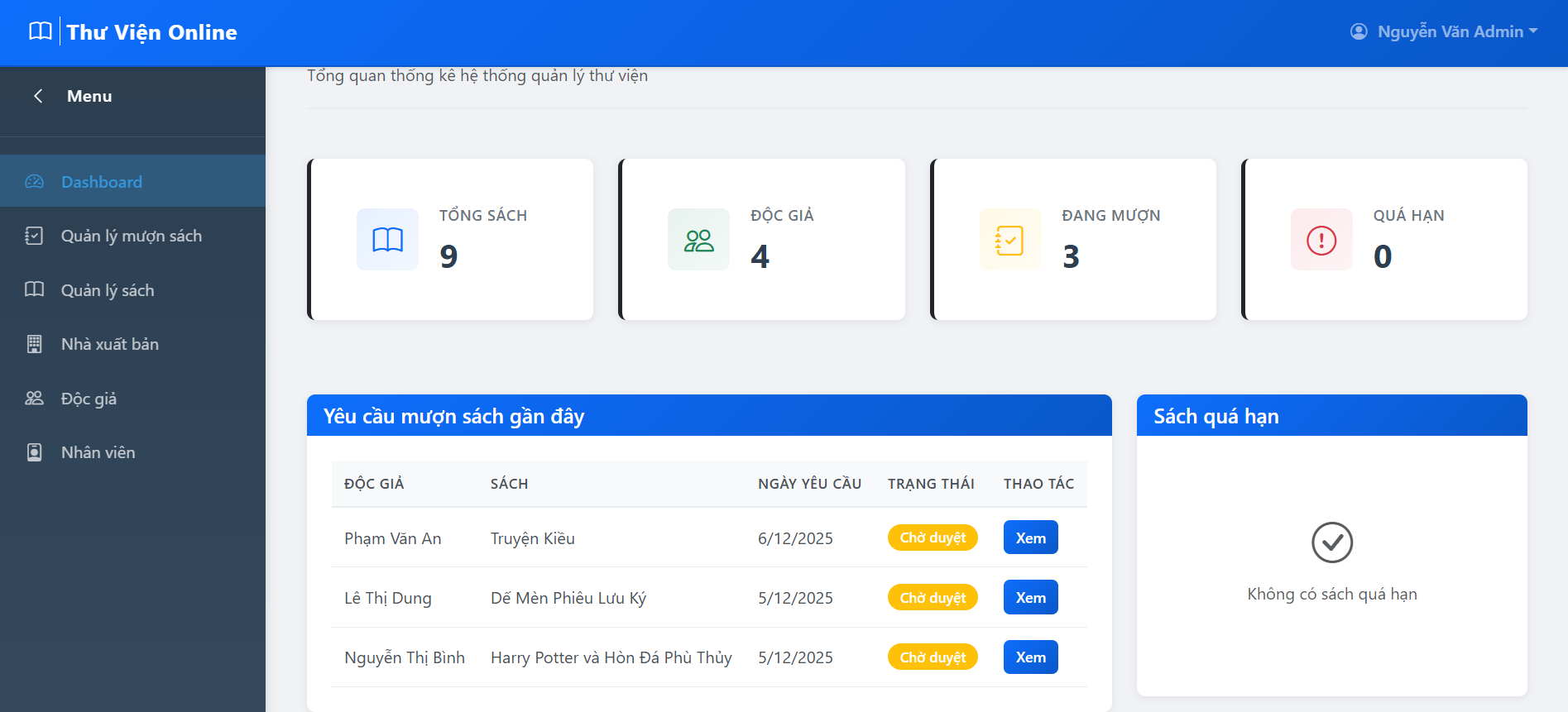
### 4.1.1. Giao diện đăng nhập admin

A screenshot of a login form

AI-generated content may be incorrect.

Hình 4.1: Giao diện đăng nhập admin

### 4.1.2. Giao diện trang chủ



Hình 4.2: Giao diện trang chủ

### 4.1.3. Giao diện quản lý sách

#### 4.1.3.1. Liệt kê danh sách sách

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.3: Giao diện liệt kê danh sách sách

#### 4.1.3.1. Thêm sách

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.4: Giao diện thêm sách

#### 4.1.3.2. Chỉnh sửa sách

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.5: Giao diện chỉnh sửa sách

#### 4.1.3.3. Chi tiết sách

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.6: Giao diện chi tiết sách

#### 4.1.3.3. Xóa sách

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.7: Giao diện xóa sách

### 4.1.4. Giao diện quản lý nhà xuất bản

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.8: Giao diện quản lý nhà xuất bản

### 4.1.5. Giao diện quản lý độc giả

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.9: Giao diện quản lý độc giả

### 4.1.6. Giao diện quản lý nhân viên

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.10: Giao diện quản lý mượn sách

### 4.1.7. Giao diện quản lý mượn sách

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.11: Giao diện quản lý mượn sách

### c

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.12: Giao diện chi tiết mượn sách

### 4.1.9. Giao diện trả sách

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.13: Giao diện trả sách

### 4.1.10. Giao diện trả sách

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.14: Giao diện thông tin cá nhân

### 4.1.11. Giao diện đổi mật khẩu

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.15: Giao diện đổi mật khẩu

## 4.2. Giao diện người dùng

### 4.2.1. Giao diện đăng ký tài khoản

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.16: Giao diện đăng ký tài khoản

### 4.2.2. Giao diện đăng nhập

A screenshot of a computer screen

AI-generated content may be incorrect.

Hình 4.17: Giao diện đăng nhập

### 4.2.3. Giao diện danh sách sách

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.18: Giao diện danh sách sách

### 4.2.4. Giao diện chỉnh sửa thông tin cá nhân

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.19: Giao diện chỉnh sửa thông cá nhân

### 4.2.6. Giao diện chi tiết sách

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.20: Giao diện chi tiết sách

### 4.2.7. Giao diện lịch sử mượn sách

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.21: Giao diện lịch sử mượn sách

### 4.2.8. Giao diện đổi mật khẩu

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.22: Giao diện đổi mật khẩu

### 4.2.9. Giao diện chi tiết mượn sách

**A screenshot of a computer

AI-generated content may be incorrect.**

Hình 4.23: Giao diện chi tiết mượn sách

### 4.2.10. Giao diện đăng ký mượn sách

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.24: Giao diện đăng ký mượn sách

# CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 5.1. Kết quả đạt được

### 5.1.1. Về mặt chức năng

Hệ thống Website Quản Lý Mượn Sách Online đã được phát triển thành công với đầy đủ các chức năng cốt lõi theo yêu cầu đặt ra. Đối với độc giả, hệ thống cung cấp đầy đủ các tính năng từ đăng ký tài khoản, đăng nhập, xem danh sách sách, tìm kiếm sách theo nhiều tiêu chí, xem chi tiết sách, đăng ký mượn sách trực tuyến, theo dõi lịch sử mượn sách cá nhân, và quản lý thông tin hồ sơ. Hệ thống phân quyền rõ ràng giữa ba vai trò: Admin có toàn quyền quản lý hệ thống bao gồm quản lý nhân viên, Librarian có quyền quản lý sách và xử lý mượn/trả sách, và Reader chỉ có quyền xem sách và quản lý hoạt động mượn sách của bản thân.

Các chức năng quản lý cho nhân viên thư viện cũng được triển khai hoàn thiện, bao gồm quản lý toàn bộ danh mục sách (thêm, sửa, xóa), quản lý nhà xuất bản, quản lý độc giả (tạo tài khoản, khóa/mở khóa), xử lý yêu cầu mượn sách (duyệt/từ chối), ghi nhận trả sách và tính tiền phạt tự động khi quá hạn. Hệ thống cũng có dashboard tổng quan giúp nhân viên nắm bắt tình hình hoạt động của thư viện.

### 5.1.2. Về mặt kỹ thuật

Về mặt kiến trúc, hệ thống được xây dựng theo mô hình MEVN stack hiện đại với MongoDB làm cơ sở dữ liệu NoSQL, Express.js làm framework backend, Vue.js 3 với Composition API làm frontend framework, và Node.js làm runtime environment. Kiến trúc này đảm bảo hiệu năng cao, khả năng mở rộng tốt và dễ bảo trì.

Frontend được phát triển bằng Vue.js 3 kết hợp với Vite build tool, mang lại trải nghiệm phát triển nhanh chóng với hot reload. Giao diện sử dụng Bootstrap 5 đảm bảo responsive design, tương thích trên nhiều thiết bị từ desktop đến mobile. Pinia được sử dụng để quản lý state toàn ứng dụng một cách hiệu quả.

Backend triển khai RESTful APIs với Express.js, sử dụng JWT để xác thực và phân quyền. Mật khẩu được mã hóa bằng bcryptjs với 10 rounds để đảm bảo bảo mật. Mongoose được sử dụng làm ODM để tương tác với MongoDB, cung cấp schema validation và relationship management.

### 5.1.3. Về mặt bảo mật

Hệ thống đã triển khai nhiều lớp bảo mật quan trọng. JWT authentication đảm bảo chỉ những người dùng đã xác thực mới có thể truy cập các chức năng nhạy cảm. Role-based access control (RBAC) đảm bảo mỗi vai trò chỉ truy cập được những tính năng phù hợp với quyền hạn của mình. Input validation được thực hiện ở cả frontend và backend để ngăn chặn các cuộc tấn công injection. CORS protection được cấu hình để chỉ cho phép requests từ domain được phép.

**5.2. Hạn chế**

**5.2.1. Hạn chế về chức năng**

Hệ thống hiện tại chưa có tính năng thông báo realtime cho độc giả khi yêu cầu mượn sách được duyệt hoặc bị từ chối. Độc giả phải chủ động vào hệ thống để kiểm tra trạng thái. Chưa có tính năng gửi email nhắc nhở trước hạn trả sách hoặc thông báo quá hạn.

Hệ thống chưa hỗ trợ đặt trước sách khi sách hết quyển. Độc giả chỉ có thể mượn sách khi có sẵn, không thể đăng ký vào danh sách chờ. Chưa có báo cáo thống kê chi tiết về hoạt động mượn sách theo thời gian, độc giả active nhất, sách được mượn nhiều nhất.

Tính năng tìm kiếm sách hiện tại chỉ hỗ trợ tìm kiếm cơ bản theo tên, tác giả, ISBN. Chưa có tìm kiếm nâng cao theo nhiều tiêu chí kết hợp, sắp xếp theo độ phổ biến, hoặc gợi ý sách liên quan.

**5.2.2. Hạn chế về kỹ thuật**

Hệ thống chưa được tối ưu hóa cho hiệu năng khi có lượng dữ liệu lớn. MongoDB queries chưa được index đầy đủ, có thể gây chậm khi database có hàng chục ngàn records. Chưa có caching mechanism để giảm tải cho database.

Frontend chưa implement service worker cho offline capabilities. Khi mất kết nối internet, người dùng không thể sử dụng bất kỳ tính năng nào. Chưa có lazy loading cho danh sách dài, có thể ảnh hưởng hiệu năng khi load nhiều sách cùng lúc.

Hệ thống chưa có logging và monitoring chi tiết để theo dõi performance và debug issues trong production. Error handling chưa được chuẩn hóa hoàn toàn, một số lỗi edge cases có thể chưa được xử lý tốt.

**5.2.3. Hạn chế về triển khai**

Hiện tại hệ thống được thiết kế để chạy trên single server, chưa có kiến trúc microservices hoặc load balancing cho high availability. Database chưa có replication và backup strategy cụ thể cho production environment.

Chưa có CI/CD pipeline tự động cho việc testing và deployment. Việc deploy lên production vẫn phải thực hiện thủ công, tăng nguy cơ lỗi và downtime.

**5.3. Hướng phát triển**

**5.3.1. Cải tiến chức năng**

**Hệ thống thông báo realtime**: Triển khai WebSocket hoặc Server-Sent Events để gửi thông báo realtime cho độc giả khi trạng thái mượn sách thay đổi. Tích hợp email service (như SendGrid, Nodemailer) để gửi email nhắc nhở trước hạn trả và thông báo quá hạn.

**Tính năng đặt trước sách**: Cho phép độc giả đăng ký vào waiting list khi sách hết quyển. Khi có sách trả về, hệ thống tự động thông báo cho người đầu tiên trong danh sách chờ.

**Báo cáo và thống kê nâng cao**: Phát triển dashboard với charts và graphs hiển thị thống kê mượn sách theo thời gian, top sách được mượn nhiều nhất, độc giả active nhất, revenue từ tiền phạt. Sử dụng libraries như Chart.js hoặc D3.js.

**Tìm kiếm và đề xuất thông minh**: Triển khai Elasticsearch cho full-text search mạnh mẽ. Phát triển recommendation system dựa trên lịch sử mượn sách để đề xuất sách phù hợp cho từng độc giả.

**5.3.2. Tối ưu kỹ thuật**

**Tối ưu hiệu năng database**: Tạo indexes cho các trường thường xuyên query (email, tên sách, ISBN). Implement MongoDB aggregation pipelines cho các queries phức tạp. Triển khai Redis cache cho dữ liệu được truy cập thường xuyên như danh sách sách, thông tin nhà xuất bản.

**Progressive web app (pwa)**: Chuyển đổi website thành PWA với service worker, cho phép hoạt động offline limited và cải thiện performance. Implement app-like experience với push notifications.

**Microservices architecture**: Tách hệ thống thành các microservices riêng biệt: User Service, Book Service, Borrowing Service, Notification Service. Sử dụng Docker containerization và Kubernetes orchestration.

**API optimization**: Implement GraphQL thay vì REST để giảm over-fetching data. Thêm API versioning và rate limiting để bảo vệ server khỏi abuse.

**5.3.3. Tăng cường bảo mật**

**Multi-factor authentication (MFA)**: Thêm xác thực hai bước qua SMS hoặc authenticator app cho tài khoản nhân viên. Triển khai OAuth2 để tích hợp đăng nhập bằng Google, Facebook.

**Advanced security measures**: Implement CSRF protection, Content Security Policy (CSP), và SQL injection prevention. Thêm audit logging để track tất cả actions của nhân viên. Encryption cho sensitive data trong database.

**Session management**: Triển khai refresh token mechanism để tăng cường bảo mật JWT. Implement session timeout và automatic logout khi inactive.

**5.3.4. Mở rộng tính năng**

**Mobile application**: Phát triển mobile app bằng React Native hoặc Flutter để độc giả có thể mượn sách dễ dàng hơn từ điện thoại.

**QR code integration**: Tạo QR codes cho mỗi cuốn sách để nhân viên có thể scan nhanh chóng khi ghi nhận mượn/trả. Integrate với barcode scanner cho ISBN lookup.

**Payment integration**: Tích hợp payment gateway (VNPay, ZaloPay) để độc giả có thể thanh toán tiền phạt online thay vì phải đến thư viện.

**Multi-library support**: Mở rộng hệ thống để hỗ trợ nhiều thư viện khác nhau trong cùng một network, cho phép độc giả mượn sách từ các chi nhánh.

**5.3.5. DevOps và triển khai**

**CI/CD pipeline**: Triển khai GitHub Actions hoặc GitLab CI để tự động run tests, build, và deploy khi có code changes. Implement automated testing với unit tests, integration tests, và end-to-end tests.

**Monitoring và logging**: Triển khai ELK stack (Elasticsearch, Logstash, Kibana) hoặc modern alternatives như Grafana + Prometheus để monitor application performance và infrastructure health.

**Cloud deployment**: Deploy lên AWS, Google Cloud, hoặc Azure với auto-scaling, load balancing. Sử dụng CDN cho static assets và database clustering cho high availability.

# TÀI LIỆU THAM KHẢO

**Sách và Giáo Trình**

[1] Nguyễn Văn Hiếu, "Lập Trình Web với Node.js và Express", Nhà Xuất Bản Đại Học Quốc Gia, 2023.

[2] Trần Minh Tuấn, "Vue.js 3 - Hướng Dẫn Toàn Diện", Nhà Xuất Bản Thông Tin và Truyền Thông, 2023.

[3] Lê Hồng Phong, "MongoDB - Cơ Sở Dữ Liệu NoSQL", Nhà Xuất Bản Khoa Học và Kỹ Thuật, 2022.

[4] Phạm Văn Nam, "Thiết Kế và Phát Triển Ứng Dụng Web", Nhà Xuất Bản Giáo Dục Việt Nam, 2022.

[5] Hoàng Minh Đức, "Bảo Mật Ứng Dụng Web Hiện Đại", Nhà Xuất Bản Công Nghệ, 2023.

**Tài Liệu Trực Tuyến**

[6] Vue.js Official Documentation, "Vue.js 3 Guide", [https://vuejs.org/guide/](vscode-file://vscode-app/d:/Program%20Files/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html), truy cập ngày 15/11/2024.

[7] Node.js Official Documentation, "Node.js v18.x Documentation", [https://nodejs.org/docs/latest-v18.x/api/](vscode-file://vscode-app/d:/Program%20Files/Microsoft%20VS%20Code/resources/app/out/vs/code/electron-browser/workbench/workbench.html), truy cập ngày 12/11/2024.