

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH



NGUYỄN VĂN KIM HẢI

Chuyên đề: **PHÁT TRIỂN HỆ THỐNG FULLSTACK
QUẢN LÝ NHÀ SÁCH**

KHÓA LUẬN TỐT NGHIỆP
NGÀNH KHOA HỌC MÁY TÍNH

TP. HỒ CHÍ MINH, 2020
BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH



NGUYỄN VĂN KIM HẢI

**PHÁT TRIỂN HỆ THỐNG FULLSTACK QUẢN LÝ
NHÀ SÁCH**

Mã số sinh viên: 1751012015

**KHÓA LUẬN TỐT NGHIỆP
NGÀNH KHOA HỌC MÁY TÍNH**

Giảng viên hướng dẫn: Thầy Dương Hữu Thành

TP. HỒ CHÍ MINH, 2020

LỜI CẢM ƠN

Chúng em xin trân trọng cảm ơn Ban Giám Hiệu và quý thầy cô giảng viên của trường Đại học Mở Thành phố Hồ Chí Minh vì đã luôn tạo điều kiện và trực tiếp truyền dạy cho chúng em những kiến thức bổ ích giúp em hiểu rõ hơn về công nghệ, thực tiễn của một đề tài nghiên cứu.

Qua những kiến thức đã học được từ quý thầy cô giảng dạy, chúng em tiếp tục nghiên cứu, tìm hiểu về nghiệp vụ của đề tài. Tuy nhiên trong quá trình nghiên cứu đã gặp không ít khó khăn và sự bỡ ngỡ, dẫu là như thế, chúng em nhận được sự hướng dẫn tận tình và nhiệt huyết của thầy Dương Hữu Thành, nhờ sự truyền lửa và tâm huyết của thầy, chúng em đã có thể hiểu ra và áp dụng tốt vào đề tài đồ án này. Bên cạnh đó, bằng cách nhìn khách quan từ một sinh viên ở trường, em nhận thấy rằng việc tích hợp kinh doanh vào thư viện của trường là một ý tưởng góp phần phát triển toàn diện những chức năng tương tác tốt với người dùng ở thời đại công nghệ 4.0. Không chỉ thế, sự thuận tiện ở mọi thao tác trong việc sử dụng phần mềm là một trong những nguyên do cốt lõi khiến chúng em chọn đề tài:

“Phát triển hệ thống fullstack quản lý nhà sách”.

Tuy nhiên trong quá trình nghiên cứu đề tài còn nhiều thiếu sót, chúng em mong quý thầy cô hướng dẫn có thể nhận xét và góp ý để chúng em có thể hoàn thành tốt hơn. Dù kết quả có ra sao, thì đây cũng chính là bước đầu tiên để chúng em có thể xác định rõ định hướng con đường đi trong công việc ở tương lai bằng sản phẩm tràn đầy tâm huyết của chúng em.

Thành phố Hồ Chí Minh, Ngày Tháng Năm 2020

Sinh viên thực hiện báo cáo:

Nguyễn Văn Kim Hải

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

MỤC LỤC

MỤC LỤC	5
DANH MỤC HÌNH.....	9
DANH MỤC BẢNG	13
CHƯƠNG 1. TỔNG QUAN.....	15
1.1 Giới thiệu đề tài	15
1.2 Giới thiệu công nghệ	15
1.2.1. Công nghệ Angular 10 (Front-end site)	15
1.2.2. Công nghệ Python Flask (Back-end site)	19
1.2 Quy trình khảo sát thư viện	21
1.2.1. Quá trình khảo sát hiện trạng	21
1.2.2. Kết quả khảo sát	21
CHƯƠNG 2. PHÂN TÍCH, THIẾT KẾ CÁC CHỨC NĂNG HỆ THỐNG	25
2.1 Sơ đồ phân rã chức năng	25
2.2 Các chức năng của hệ thống	26
2.2.1. Quản lý sách	26
2.2.2. Quản lý tài khoản	26
2.2.3. Quản lý nhân viên.....	27
2.2.4. Quản lý nhà cung cấp	27
2.2.5. Quản lý bán sách	27
2.2.6. Quản lý mượn trả.....	27
2.2.7. Liên hệ, chăm sóc khách hàng.....	28

2.2.8.	Thanh toán bằng ví điện tử Momo	28
2.2.9.	Thanh toán bằng Paypal	35
2.2.10.	Tích hợp JWT (Json Web Token)	41
2.2.11.	Tích hợp chức năng đăng nhập bằng Google	46
2.2.12.	Tích hợp Telegram Bot	53
2.2.13.	Lưu trữ ảnh trên Cloundinary.....	60
2.3	Biểu đồ Usecase.....	62
2.3.1.	Tổng quát.....	62
2.3.2.	Quản lý sách	63
2.3.3.	Quản lý nhân viên.....	63
2.3.4.	Quản lý đọc giả và khách hàng	64
2.3.5.	Quản lý tác giả.....	64
2.3.6.	Quản lý nhà cung cấp	65
2.3.7.	Quản lý mượn trả.....	65
2.3.8.	Quản lý mua bán sách	66
2.3.9.	Quản lý phiếu nhập xuất kho.....	66
2.3.10.	Báo cáo, thống kê	67
2.4	Cơ sở dữ liệu hệ thống.....	68
2.5	Biểu đồ lớp	69
2.5.1.	Lớp sách	69
2.5.2.	Lớp tài khoản.....	70
2.5.3.	Lớp nhân viên và tài khoản nhân viên	70

2.5.4.	Lớp khách hàng và tài khoản khách hàng	71
2.5.5.	Lớp phiếu mượn	71
2.5.6.	Lớp hóa đơn.....	72
2.5.7.	Lớp phiếu xuất, nhập kho	72
2.5.8.	Lớp tác giả	73
2.5.9.	Lớp nhà cung cấp	73
CHƯƠNG 3. PHÂN TÍCH, THIẾT KẾ GIAO DIỆN	74
3.1	Các giao diện chung	74
3.1.1.	Trang đăng nhập cho nhân viên	74
3.1.2.	Trang đăng ký tài khoản dành cho khách hàng và tạo KH	75
3.1.3.	Trang yêu cầu reset mật khẩu.....	77
3.2	Các giao diện dành cho quản trị viên (Admin).....	79
3.2.1.	Trang thống kê.....	79
3.2.2.	Trang chăm sóc khách hàng	80
3.2.3.	Trang thiết lập thông tin tài khoản	81
3.2.4.	Trang danh sách hóa đơn.....	84
3.2.5.	Trang tạo mới, thanh toán hóa đơn (POS)	85
3.2.6.	Trang tính toán, quét mã bằng ví điện tử Momo môi trường Test	87
3.2.7.	Trang chi tiết hóa đơn.....	88
3.2.8.	Trang tạo sách mới	88
3.2.9.	Trang chi tiết sách	91
3.2.10.	Trang cập nhật sách	91

3.2.11.	Trang tạo mới phiếu mượn	93
3.2.12.	Trang chi tiết phiếu mượn	95
3.2.13.	Trang danh sách tài khoản hệ thống	97
3.3	Các giao diện về phía người dùng (User)	98
3.3.1.	Khung chat yêu cầu hỗ trợ, tư vấn của KH	98
3.3.2.	Trang chủ nhà sách	99
3.3.3.	Trang tìm kiếm sản phẩm	100
3.3.4.	Trang thanh toán	101
3.3.5.	Trang chi tiết sản phẩm	102
CHƯƠNG 4. TỔNG KẾT VÀ HƯỚNG PHÁT TRIỂN		106
4.1	Kết quả đạt được	106
4.2	Nhược điểm	106
CHƯƠNG 5. TÀI LIỆU THAM KHẢO		108

DANH MỤC HÌNH

Hình 1.1. Cài đặt Angular	17
Hình 1.2. Khởi tạo thư mục.....	17
Hình 1.4. Chạy thành công project.....	17
Hình 1.5. Cấu trúc cơ bản của thư mục.....	18
Hình 1.6. Cấu trúc của project	20
Hình 1.7. Tập tin Flask đầu tiên.....	21
Hình 2.1. Sơ đồ phân rã chức năng	25
Hình 2.2. Cấu hình request Momo	30
Hình 2.3. Request mẫu Momo	30
Hình 2.4. Signature Momo.....	31
Hình 2.5. Sơ đồ xử lý Momo	31
Hình 2.6. Request HTTP Momo mẫu	32
Hình 2.7. Chú thích params trong request Momo	33
Hình 2.8. Response Momo mẫu	34
Hình 2.9. Chú thích params trong response Momo	34
Hình 2.10. Request mẫu Paypal	40
Hình 2.11. Response mẫu Paypal.....	41
Hình 2.12. Các thành phần chính của JWT.....	42
Hình 2.13. Cấu trúc Header JWT	42
Hình 2.14. Cấu trúc Payload JWT	43
Hình 2.15. Quy trình tạo Signature JWT	43
Hình 2.16. Quy trình xác thực quyền người dùng trong JWT	44
Hình 2.17. Quy trình tạo token JWT.....	44
Hình 2.18. Quy trình xác thực người dùng dựa trên token JWT	45
Hình 2.19. Quy định token trong JWT.....	45
Hình 2.20. Tạo project Google Dashboard	47

Hình 2.21. Chứng thực trên Google Dashboard	47
Hình 2.22. Chứng thực trên Google Dashboard	48
Hình 2.23. Chứng thực trên Google Dashboard	48
Hình 2.24. Cấu hình Google trong Flask	49
Hình 2.25. Hàm login vào Google	50
Hình 2.26. Google provider config	50
Hình 2.27. Request_uri trong Flask	50
Hình 2.28. Thư viện cần thiết trong Flask để sử dụng Google	51
Hình 2.29. Tìm kiếm BotFather	54
Hình 2.30. Khởi động BotFather.....	55
Hình 2.31. Tạo một Bot mới	55
Hình 2.32. Đặt tên cho Bot.....	56
Hình 2.33. Chuỗi token được tạo ra trong Bot.....	56
Hình 2.34. Trạng thái của Bot.....	57
Hình 2.35. Lấy ID của Bot.....	57
Hình 2.36. Hàm sendMessage Telegram	58
Hình 2.37. Hàm testMessage Telegram	59
Hình 2.38. Kết quả của quá trình tích hợp	59
Hình 2.39. Chi tiết hàm UploadBookImage	61
Hình 2.40. Usecase tổng quát.....	62
Hình 2.41. Usecase quản lý sách.....	63
Hình 2.42. Usecase quản lý nhân viên	63
Hình 2.43. Usecase quản lý đọc giả và khách hàng	64
Hình 2.44. Usecase quản lý tác giả	64
Hình 2.45. Usecase quản lý nhà cung cấp.....	65
Hình 2.46. Usecase quản lý mượn trả	65
Hình 2.47. Usecase quản lý mua bán sách	66

Hình 2.48. Usecase quản lý phiếu xuất nhập kho	66
Hình 2.49. Usecase báo cáo thống kê	67
Hình 2.50. Biểu đồ cơ sở dữ liệu của hệ thống	68
Hình 2.51. Biểu đồ lớp sách.....	69
Hình 2.52. Biểu đồ lớp tài khoản	70
Hình 2.53. Biểu đồ lớp nhân viên và tài khoản nhân viên	70
Hình 2.54. Biểu đồ lớp khách hàng và tài khoản khách hang.....	71
Hình 2.55. Biểu đồ lớp phiếu mượn.....	71
Hình 2.56. Biểu đồ lớp hóa đơn	72
Hình 2.57. Biểu đồ lớp phiếu xuất, nhập kho	72
Hình 2.58. Biểu đồ lớp tác giả	73
Hình 2.59. Biểu đồ lớp nhà cung cấp	73
Hình 3.1. Trang đăng nhập cho nhân viên	74
Hình 3.2. Trang đăng ký tài khoản dành cho khách hàng và tạo KH	75
Hình 3.3. Trang quên mật khẩu.....	77
Hình 3.4. Trang đặt lại mật khẩu cho khách hàng, nhân viên.....	78
Hình 3.5. Trang thống kê	79
Hình 3.6. Trang chăm sóc KH	80
Hình 3.7. Trang thiết lập thông tin tài khoản	82
Hình 3.8. Trang thiết lập thông tin người dùng	82
Hình 3.9. Popup thay đổi mật khẩu tài khoản	83
Hình 3.10. Popup thay đổi thông tin nhân viên	83
Hình 3.11. Trang danh sách hóa đơn	85
Hình 3.12. Trang thanh toán hóa đơn	86
Hình 3.13. Trang thanh toán bằng Momo	87
Hình 3.14. Trang chi tiết hóa đơn	88
Hình 3.15. Trang thanh toán bằng Momo	88

Hình 3.16. Trang tạo sách mới	89
Hình 3.17. Popup tạo mới nhà cung cấp	89
Hình 3.18. Trang chi tiết sách	91
Hình 3.19. Trang cập nhật thông tin sách	92
Hình 3.20. Trang tạo mới phiếu mượn.....	94
Hình 3.21. Trang chi tiết phiếu mượn.....	95
Hình 3.22. Popup gửi email nhắc nhở khách hàng	95
Hình 3.23. Trang danh sách tài khoản hệ thống.....	97
Hình 3.24. Khung chat khách hàng.....	98
Hình 3.25. Trang chủ nhà sách.....	100
Hình 3.26. Trang tìm kiếm sản phẩm.....	100
Hình 3.27. Trang thanh toán đơn hàng	101
Hình 3.28. Trang chi tiết sách	103
Hình 3.29. Danh sách bình luận của sách	104
Hình 3.30. Modal so sánh sản phẩm với sản phẩm tương tự tại các trang khác	105

DANH MỤC BẢNG

Bảng 2.1. Chú thích params trong request, response Paypal	39
Bảng 2.2. Request mẫu Paypal.....	Error! Bookmark not defined.

MỞ ĐẦU

Lý do chọn đề tài:

Trong các quá trình quản lý trong công tác giáo dục hiện nay , quá trình quản lý thư viện cũng như thư quán cũng là một việc rất quan trọng, không thể thiếu. Vì thế, ta cần có phải xây dựng một hệ thống nhằm giúp quản lý thư viện, thư quán mới hiện đại, tiện dụng trong việc quản lý

Tại các trường đại học hiện nay, việc quản lý thư viện cũng như thư quán một cách hiệu quả, dễ dàng là một việc rất quan trọng, không thể thiếu. Nhưng sẽ rất là khó khăn nếu chúng ta vẫn còn giữ các hình thức quản lý thô sơ, nhiều công đoạn như trước..., vì thế ta cần có một hệ thống quản lý thư viện, thư quán mới nhằm giải quyết những bài toán trên, mang lại một phương pháp quản lý dễ dàng kiểm soát, minh bạch, rõ ràng hơn, và hơn hết tất cả là có thể tiện lợi, nhanh chóng hơn trong việc phục vụ sinh viên, khách hàng. Và đó cũng là lý do chúng em chọn đề tài “Phát triển hệ thống fullstack quản lý nhà sách” cho đồ án tốt nghiệp của chúng em.

Nhiệm vụ nghiên cứu:

Tìm hiểu quy trình nghiệp vụ website quản lý thư viện tích hợp quản lý mua bán sách trực tuyến và tại chỗ của trường Đại học Mở Thành phố Hồ Chí Minh (quản lý admin, quản lý đọc giả, quản lý mua bán sách, quản lý mượn trả, quản lý kho...)

Tìm hiểu công cụ xây dựng website (Python Flask, Angular, Microsoft SQL Server 2014, ngôn ngữ lập trình Java, ngôn ngữ lập trình Python...).

CHƯƠNG 1. TỔNG QUAN

Chương này giới thiệu về nội dung, ý nghĩa của đề tài cũng như là tìm hiểu về công nghệ Angular 10 – công nghệ đang được vận dụng vào phần giao diện (Front-end) của đồ án. Ngoài ra, chương còn cung cấp cho ta quy trình khảo sát thư viện nhằm giúp ta nắm rõ hơn về các yêu cầu của hệ thống.

1.1 Giới thiệu đề tài

Tại các trường đại học hiện nay, việc quản lý thư viện cũng như thư quán một cách hiệu quả, dễ dàng là một việc rất quan trọng, không thể thiếu. Nhưng sẽ rất là khó khăn nếu chúng ta vẫn còn giữ các hình thức quản lý thô sơ, nhiều công đoạn như trước..., vì thế ta cần có một hệ thống quản lý thư viện, thư quán mới nhằm giải quyết những bài toán trên, mang lại một phương pháp quản lý dễ dàng kiểm soát, minh bạch, rõ ràng hơn, và hơn hết tất cả là có thể tiện lợi, nhanh chóng hơn trong việc phục vụ sinh viên, khách hàng.

Hệ thống cung cấp các chức năng quản lý nhằm giúp nhân viên dễ dàng quản lý các đối tượng thuộc thư viện cũng như thư viện một cách dễ dàng như quản lý sách, quản lý nhân viên, quản lý khách hàng, quản lý tài khoản thành viên.... cũng như dịch vụ chăm sóc khách hàng thông qua chức năng chat application giữa người dùng và nhân viên. Ngoài các chức năng quản lý, hệ thống còn cung cấp các dịch vụ mượn trả sách, dịch vụ bán sách và dịch vụ thanh toán hóa đơn thông qua dịch vụ thanh toán trực tuyến như ví điện tử MOMO.

Bên cạnh các chức năng, hệ thống cũng cung cấp đến cho người dùng một giao diện thông dụng, dễ dùng, tiện nghi để sử dụng, trải nghiệm một cách thoái moái, tiện lợi.

1.2 Giới thiệu công nghệ

1.2.1. Công nghệ Angular 10 (Front-end site)

Trang chủ: <https://angular.io/>

Sơ lược về Angular 10:

- Là một framework cho phía giao diện, front-end của hệ thống, được xây dựng và hỗ trợ bởi Google, được viết bằng Javascript, lần đầu ra mắt vào 2009, nay đã có đến phiên bản Angular 10 và là phiên bản hiện tại của hệ thống và cứ 6 tháng, theo google, sẽ cập nhật một phiên bản mới.
- Mang lại trải nghiệm nhanh chóng, tiện dụng cho người dùng nhờ việc cung cấp các công cụ giúp xây dựng một chương trình SPA (Single Page App) một cách dễ dàng.
- Angular mang lại hiệu năng cao hơn, dễ dàng sử dụng và nâng cấp, bảo trì, phù hợp cho các dự án lớn.
- Một component trong công nghệ angular gồm 3 phần:

HTML file: dùng để render layout view (giao diện) cho người dùng, được tạo dựa trên các thẻ HTML thông thường kết hợp với các Angular Binding và Directives.

CSS (SCSS) file: Chứa các thuộc tính CSS được dùng để xây dựng giao diện.

Một file .ts: Đây là 1 phần không thể thiếu trong component, là 1 Class để liên kết 3 phần lại với nhau, được tạo ra bằng Typescript. Class được dùng để xử lý các chức năng logic của component thông qua các thuộc tính (property) và phương thức (function) do người lập trình viên tự khai báo và định nghĩa.

TS file: là nơi chứa các thuộc tính, phương thức của component. Đồng thời cũng giúp liên kết 3 phần với nhau và thuộc một module cụ thể.

♦ **Cài đặt project Angular và cấu trúc thư mục cơ bản:**

Các môi trường cần có: Node.js, NPM (Node package manager), IDE.

Các bước cài đặt:

Mở IDE hoặc Terminal và thực hiện các lệnh sau.

Cài đặt angular CLI:

```
npm install -g @angular/cli
```

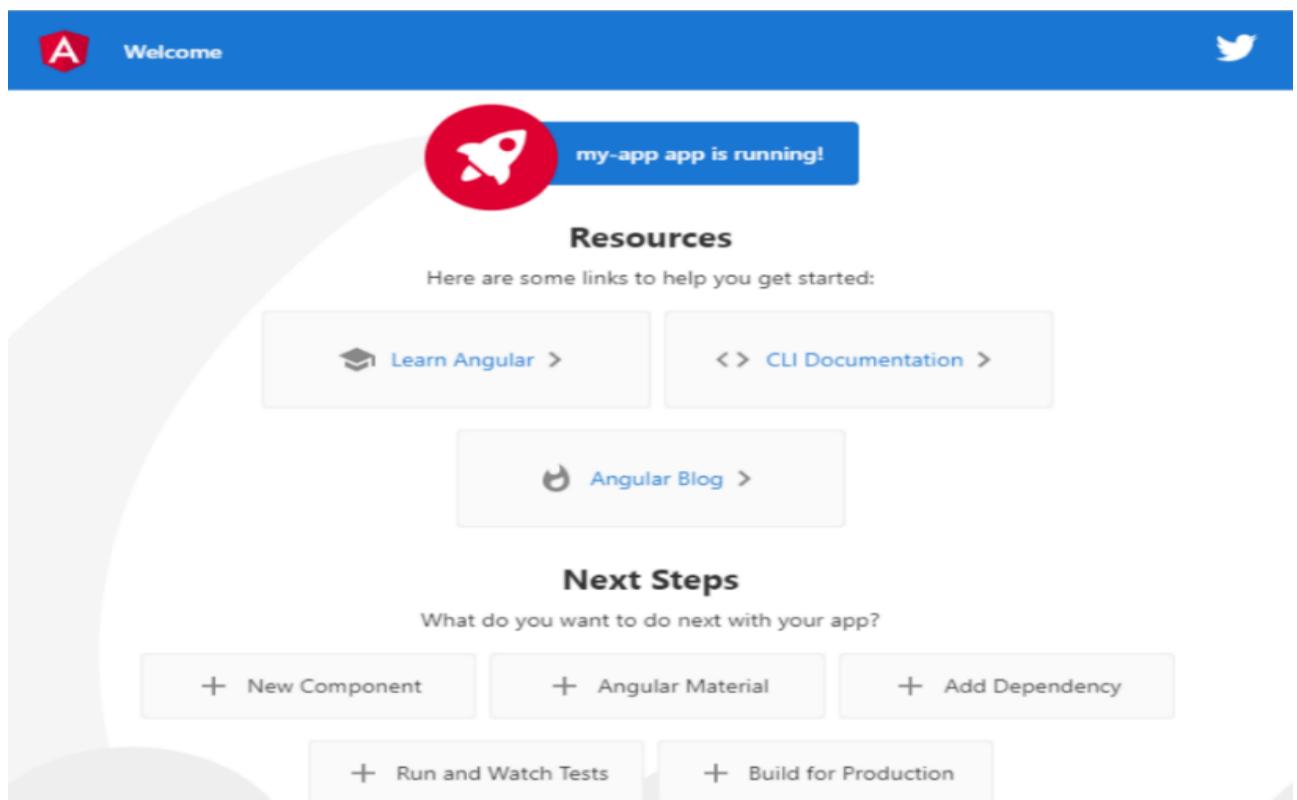
Hình 1.1. Cài đặt Angular

Khởi tạo thư mục chứa dự án:

```
cd my-app
ng serve --open
```

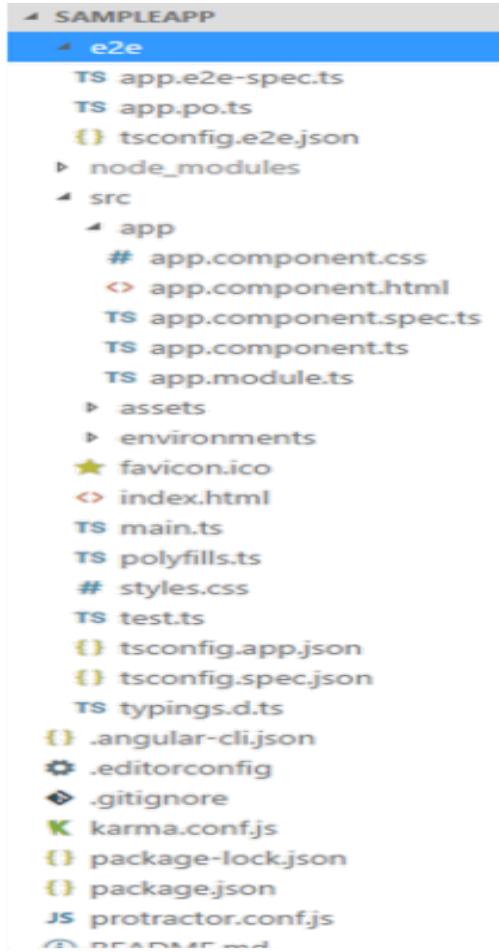
Hình 1.2. Khởi tạo thư mục

Khởi tạo và chạy thành công project:



Hình 1.3. Chạy thành công project

Cấu trúc thư mục cơ bản ban đầu:



Hình 1.4. Cấu trúc cơ bản của thư mục

Trong đó:

Src: Là 1 thư mục, gồm các component, chứa các mã code của chương trình.

Angular-cli.json: là 1 tập tin, chứa thông tin cấu hình của chương trình , có dạng dữ liệu json.

Package.json: là 1 tập tin có dạng dữ liệu json, chứa danh sách các thư viện được cài đặt trong chương trình.

Tại sao nên chọn Angular ?

Cơ chế Two-way Data Binding: Là một cơ chế cực kỳ tiện lợi của Angular, giúp ta có thể đồng bộ dữ liệu một cách tự động, ngay lập tức giữa giao diện và component class.

Hỗ trợ Routing đầy đủ: Giúp cho chương trình tải các trang một cách bất đồng bộ, rất tiện lợi khi dùng để xây dựng 1 chương trình SPA (Single Page Application).

Hỗ trợ các Directive HTML: Giúp cho chương trình có thể sử dụng các cấu trúc lập trình như if else, for, while, switch casenhằm giúp hiển thị, render các template.

Module hóa chương trình: Chương trình được thiết kế module hóa nhằm giúp chương trình có tổ chức tốt hơn

Hỗ trợ làm việc với hệ thống Backend: Angular được xây dựng hỗ trợ làm việc với backend server và thực thi bất cứ logic nào và nhận dữ liệu về.

Hỗ trợ kết nối với server đơn giản: với gói package module “HttpClient”.

- ◆ Tài liệu: <https://angular.io/guide/setup-local>

1.2.2. Công nghệ Python Flask (Back-end site)

- ◆ Trang chủ:

<https://www.python.org/>

<https://flask.palletsprojects.com/en/1.1.x/>

- ◆ Sơ lược về Python Flask:

Python là một ngôn ngữ kịch bản cấp cao, rất dễ học, dễ đọc, dễ bảo trì, thân thiện rất thích hợp cho người mới bắt đầu (Beginner). Python đang một ngày một phát triển và sẽ còn được sử dụng rộng rãi hơn trong các lĩnh vực Ứng dụng web, Machine Learning, Deep Learning....

Flask là web framework được xây dựng bằng ngôn ngữ lập trình Python, dùng để xây dựng các website một cách dễ dàng, nhanh chóng. Flask framework sử dụng bộ công cụ WSGI và bộ Ninja2 Template. Flask

framework hỗ trợ các phần tiện ích mở rộng cho ứng dụng như template, email, RESTful...

- ◆ Cài đặt dự án:

Các môi trường cần có:

Python 3 (<https://www.python.org/downloads/>)

PIP (<https://pip.pypa.io/en/stable/installing/>)

IDE (Pycharm)

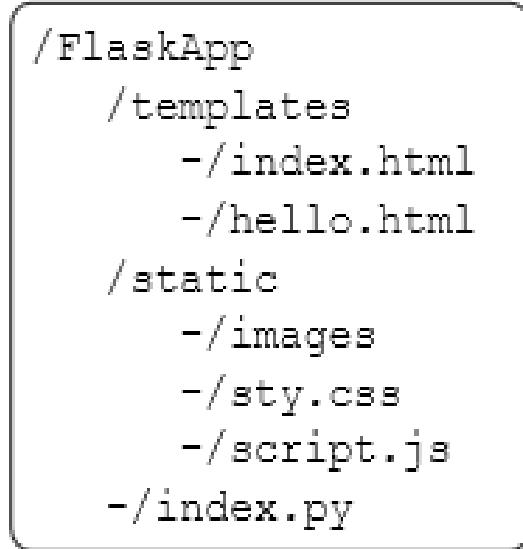
Các bước cài đặt:

Bước 1: Tạo thư mục chứa project.

Bước 2: Tạo Flask project bằng lệnh: pip install Flask

Bước 3: Tạo 1 tập tin .py trong folder và viết code.

Cấu trúc của 1 project Flask:



Hình 1.5. Cấu trúc của project

- Tạo tập tin Flask đầu tiên:

```

from flask import Flask
app = Flask(__name__)

@app.route("/")
def hello():
    return "Hello World!"

if __name__ == "__main__":
    app.run(debug=True)

```

Hình 1.6. Tập tin Flask đầu tiên

Để chạy chương trình, ta dùng lệnh: “python hello.py”.

Chương trình sẽ được chạy trên port mặc định: <http://127.0.0.1/5000/>

Ta có thể thay đổi port và địa chỉ “app.run (host, port, debug, options)”

1.2 Quy trình khảo sát thư viện

1.2.1. Quá trình khảo sát hiện trạng

Tất cả quy trình khảo sát và nghiệp vụ được thực hiện tại thư viện trường Đại học Mở Thành phố Hồ Chí Minh (cơ sở Nguyễn Kiệm). Khảo sát về quy trình nghiệp vụ của thư viện trường và một số trang web bán sách online như: <https://tiki.vn>, <https://shopee.vn>, <https://www.amazon.com> ...

1.2.2. Kết quả khảo sát

Về cơ sở vật chất kĩ thuật:

Các loại sách được phân theo từng khoa, từng ngành học và được lưu trữ, để vào từng giá sách tương ứng. Các giá sách được để trong phòng riêng của thư viện, mỗi giá sách có hai mặt, mặt trước và mặt sau, có kèm theo tên từng giá sách để người dùng tìm được sách mình muốn. Ngoài ra, còn có hai dàn máy tính để sinh viên có thể sử dụng, phòng được trang bị máy lạnh và các loại bàn lớn nhỏ, cho việc học nhóm.

Thực trạng về quản lý thư viện của trường Đại học Mở Thành phố Hồ Chí Minh:

Các dữ liệu về sách, lịch sử mượn sách, lịch sử sinh viên đã mượn sách, thông tin sinh viên...đều được lưu trữ ở máy tính và số sách tương ứng. Tài liệu chủ yếu là sách và dữ liệu sinh viên.

Quy trình nhập sách:

Sau khi nhập sách từ kho về, thủ thư phải nhập lại toàn bộ thông tin các đầu sách vào sổ và trên máy tính, cất giữ các chứng từ liên quan.Thủ thư đánh dấu mã cho từng quyển sách và cất chúng vào giá sách tương ứng, mỗi quyển sách được để lên kệ sẽ đánh vào loại sách cho mượn, ngoài ra mỗi mã sách đều có mã riêng có thể dùng để scan để kiểm tra mất mát trong quá trình nhập sách.

Quy trình trả sách:

Sau khi đã hoàn thành xong hạn mượn, đọc giả phải trả đúng sách đã mượn, thủ thư sẽ yêu cầu đọc giả cung cấp mã số sinh viên, ngày mượn, số phiếu mượn (nếu đối tượng là sinh viên) và mã giáo viên, ngày mượn (nếu đối tượng là giáo viên). Sau khi cung cấp xong, thủ thư sẽ đánh dấu vào mục đã trả sách và kiểm tra đã trả đúng hạn hay chưa, nếu đã vượt hạn sẽ nhắc nhở hoặc có hình phạt tương ứng nếu quá số lần nhắc nhở.

Mọi lịch sử mượn trả sách sẽ được lưu trữ lên cơ sở dữ liệu của hệ thống quản lý thư viện.

Quy trình mượn sách.

Với sinh viên, giáo viên, có thể mượn sách đọc tại chỗ hoặc mượn về.

Khi mượn sách, sinh viên hoặc giáo viên đem thẻ sinh viên hay thẻ giáo viên, sau đó thủ thư sẽ kiểm tra thông tin. Sau khi hợp lệ sẽ báo cho sinh viên và giáo viên có thể vào thư viện để chọn sách cần mượn

Sau khi có sách đã cần mượn, thì sinh viên hay giáo viên sẽ đem ra quầy để cung cấp chi tiết thông tin để có thể xuất phiếu mượn bao gồm: họ và tên, mã số sinh viên (giáo viên), số lượng sách mượn, tên sách mượn, thời gian mượn, thời gian trả, chữ ký sinh viên (giáo viên)

Sau khi đã cung cấp đầy đủ thông tin, thủ thư sẽ trả lại thẻ sinh viên hay thẻ giáo viên.

Mọi thông tin sẽ được lưu trữ ở hệ thống của thư viện.

Thông kê, báo cáo:

Thư viện thực hiện thống kê theo cuối học kì, thống kê theo từng khoa và ngành học và theo tiêu chí như sau:

Thống kê sách đã nhập mới.

Thống kê sách đang mượn.

Thống kê sách còn lại trong thư viện.

Thống kê sách đã thanh lý.

Xử lý sách thanh lý:

Các loại sách tồn kho quá lâu hoặc hư hỏng nặng nề, không sử dụng đến sẽ tiến hành thanh lý, đồng thời trừ số lượng trong cơ sở dữ liệu của hệ thống.

Ưu và nhược điểm của hệ thống hiện tại:

– Ưu điểm:

Hiện đại, dễ dàng quản lý mọi thông tin của các quy trình của thư viện.

Nhân viên thư viện không cần biết về tin học vẫn có thể dễ dàng sử dụng hệ thống.

Dễ dùng, dễ tiếp cận người dùng.

– Nhược điểm:

Do là hệ thống chứa quá nhiều thông tin và dữ liệu nên đôi khi cần nâng cấp và tích hợp việc bán sách vào rất bất tiện vì có thể gây mất dữ liệu.

Yêu cầu người dùng từ hệ thống mới:

Từ những khó khăn trong việc quản lý thư viện, nhóm đã đề xuất một website quản lý tích hợp việc kinh doanh sách trực tuyến và tại chỗ, giúp cho hệ thống đổi mới và mở rộng ra lĩnh vực kinh doanh dễ dàng hơn.

Hệ thống mới phải thỏa các yêu cầu:

- Website có giao diện dễ sử dụng.
- Cho phép lưu các dữ liệu về sách, có thể mở rộng và nâng cấp.
- Quản lý mượn, trả, mua, bán một cách dễ dàng, thuận tiện cho người dùng.
 - Tìm kiếm thông tin thủ thư đọc giả một cách dễ dàng.
 - Cho phép thống kê theo doanh thu tháng, năm, top 3 sách mượn nhiều, top 3 sách bán nhiều.
 - Cho phép thanh toán trực tiếp qua: ví điện tử Momo, cổng thanh toán Paypal.

Yêu cầu cần đạt được từ hệ thống mới:

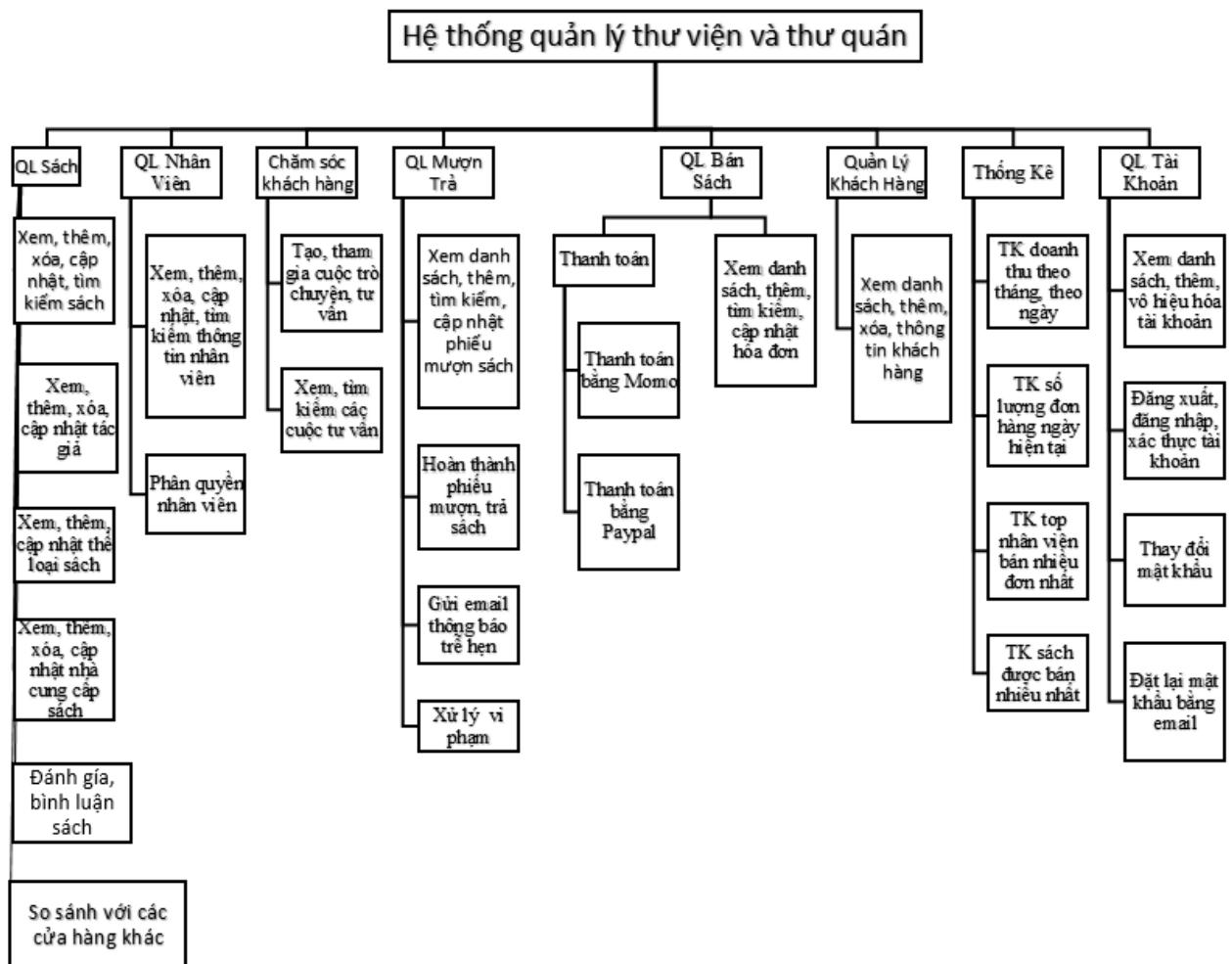
Qua quá trình khảo sát thực tế từ thư viện của trường, tuy hệ thống hiện đại dễ dùng, nhưng để có thể tích hợp việc kinh doanh sách vào hệ thống là một điều khó khăn. Do đó nhằm khắc phục nhược điểm này, phần mềm quản lý mới phải gồm các chức năng sau:

- Quản lý mượn trả, quản lý sách (loại sách, số lượng, tên sách...).
- Quản lý báo cáo thống kê, quản lý khách hàng.
- Quản lý nhân viên (thủ thư), quản lý bán sách (tại chỗ và trực tuyến).
- Quản lý vận chuyển. Chăm sóc khách hàng. Quản lý tài khoản.

CHƯƠNG 2. PHÂN TÍCH, THIẾT KẾ CÁC CHỨC NĂNG HỆ THỐNG

Chương này giới thiệu về các chức năng của hệ thống mà mình đang xây dựng bằng các công cụ cụ thể như sơ đồ phân rã chức năng, biểu đồ, đặc tả usecase, biểu đồ quan hệ cơ sở dữ liệu, nhằm giúp ta hiểu rõ hơn về các chức năng mà ta đang xây dựng trong hệ thống.

2.1 Sơ đồ phân rã chức năng



Hình 2.1. Sơ đồ phân rã chức năng

2.2 Các chức năng của hệ thống

2.2.1. Quản lý sách

- Xem danh sách sách: Nhân viên, nhà quản lý được quyền xem danh sách sách.
- Thêm, xóa, sửa tạo sách: Nhân viên, nhà quản lý được quyền thêm, xóa, cập nhật thông tin sách.
- Tìm kiếm sách: Nhân viên tìm kiếm sách theo các trường như mã sách, tên sách, nhà cung cấp, thể loại...
- Xem chi tiết sách: Nhân viên được quyền xem chi tiết thông tin của sách

2.2.2. Quản lý tài khoản

- Đăng nhập, xác thực: Người dùng dùng để đăng nhập tài khoản, để thực hiện các chức năng yêu cầu quyền truy cập.
- Đăng xuất: Đăng xuất tài khoản khỏi hệ thống.
- Xem danh sách tài khoản: Nhà quản lý được quyền xem danh sách tài khoản của người dùng.
- Tạo mới tài khoản nhân viên, khách hàng: Nhà quản lý có thể tạo mới các tài khoản cho nhân viên, khách hàng với các thông tin được cung cấp.
- Tìm kiếm tài khoản: Nhà quản lý được quyền tìm kiếm tài khoản theo các trường như tên đăng nhập, mã tài khoản, số điện thoại...
- Vô hiệu hóa tài khoản: Mỗi khi có người dùng vi phạm tiêu chuẩn do thư viện đề ra thì nhà quản lý có quyền hạn chế quyền hạn bằng cách vô hiệu hóa tài khoản người dùng trong khoảng thời gian theo quy định.
- Thay đổi thông tin tài khoản (mật khẩu, email): dùng để thay đổi thông tin của tài khoản người dùng sau khi có yêu cầu từ người dùng.
- Reset mật khẩu thông qua email: Hệ thống sẽ gửi email của tài khoản một đường dẫn kèm với token nhằm để đặt lại mật khẩu của người dùng.

2.2.3. Quản lý nhân viên

- Xem danh sách nhân viên: Người quản lý được xem danh sách thông tin nhân viên.
- Thêm, xóa, sửa tạo nhân viên: Người quản lý có thể thêm, cập nhật, xóa nhân viên khỏi hệ thống.
- Tìm kiếm nhân viên: Người quản lý có thể tìm kiếm nhân viên theo các trường như họ tên, mã nhân viên, sđt...
- Xem chi tiết nhân viên: Người quản lý có thể xem chi tiết các thông tin của nhân viên.

2.2.4. Quản lý nhà cung cấp

- Thêm, xóa, sửa nhà cung cấp: Người quản lý, nhân viên có thể thêm, cập nhật, xóa thông tin nhà cung cấp sách.
- Xem chi tiết thông tin nhà cung cấp: Người dùng có thể xem chi tiết thông tin các nhà cung cấp sách.

2.2.5. Quản lý bán sách

- Xem danh sách hóa đơn: Người quản lý, nhân viên có thể xem danh sách thông tin các hóa đơn, đơn hàng của hệ thống.
- Tìm kiếm, xóa hóa đơn: Người quản lý, nhân viên có thể xóa, tìm kiếm thông tin hóa đơn.
- Thanh toán trực tiếp: Nhân viên tạo hóa đơn, đơn hàng.
- Thanh toán bằng thẻ Paypal, ví điện tử MOMO: Khách hàng có thể tự động tạo đơn hàng và thanh toán bằng các thẻ tính dụng như Paypal, Momo.

2.2.6. Quản lý mượn trả

- Xem danh sách phiếu mượn: Nhân viên có thể xem danh sách phiếu mượn.
- Xem chi tiết phiếu mượn: Nhân viên có thể xem chi tiết các thông tin của phiếu mượn.

- Xóa, sửa, tìm kiếm, thêm mới phiếu mượn: Nhân viên được thêm, xóa, cập nhật, tìm kiếm các phiếu mượn sách theo ngày, tên khách hàng....
- Hoàn thành phiếu mượn đã trả sách: Nhân viên cập nhật lại ngày trả của phiếu mượn khi khách hàng trả sách.
- Xử lý vi phạm: Nhân viên được quyền xử lý vi phạm bằng cách vô hiệu hóa hoặc cảnh cáo nếu khách hàng vi phạm quy định khi mượn sách.
- Gửi email nhắc nhở trễ hẹn: Nhân viên gửi email nhắc nhở khách hàng nếu họ trả sách không đúng theo quy định.

2.2.7. Liên hệ, chăm sóc khách hàng

- Tìm kiếm cuộc trò chuyện, tư vấn: Nhân viên được quyền tìm kiếm cuộc trò chuyện giữa khách hàng và nhân viên của hệ thống.
- Tạo mới, tham gia cuộc trò chuyện, tư vấn: Khi khách hàng đăng nhập thì có thể tham gia và nhắn tin với nhân viên hệ thống nếu muốn được tư vấn mua hàng.
- Xem danh sách cuộc trò chuyện, tư vấn: Nhân viên có thể xem danh sách các cuộc hội thoại để tìm kiếm thông tin khi có yêu cầu.

2.2.8. Thanh toán bằng ví điện tử Momo

♦ Giới thiệu sơ lược về nền tảng thanh toán Momo

Trang chủ: <https://developers.momo.vn/#/>

Momo Payment Platform API là giải pháp thanh toán qua tài khoản ví Momo trên các ứng dụng, website dành cho các doanh nghiệp.

Hiện nay người ta ứng dụng việc tích hợp api Momo để thanh toán như là một phần phải có trong các ứng dụng doanh nghiệp bởi sự tương thích cao và dễ tích hợp, đặc biệt là Momo đang được phổ biến rộng rãi, một số doanh nghiệp đang sử dụng dịch vụ Momo để thanh toán bao gồm: CGV, Nhaccutui, Lazada, Shopee...

♦ Về phương thức thanh toán

Momo bao gồm các phương thức thanh toán tích hợp như:

Cổng thanh toán MoMo (All In One): Áp dụng đối tác có thanh toán trên nền tảng Website, Mobile, Smart TV,...

Thanh toán App-In-App: Áp dụng cho đối tác có ứng dụng di động (android/ios) muốn mở trực tiếp ứng dụng MoMo để thanh toán.

Thanh toán POS: Áp dụng cho đối tác có hệ thống bán hàng bằng máy POS. Thu ngân dùng máy scan để quét "MÃ THANH TOÁN" trên app MoMo để thanh toán

Thanh toán QR Code: Đối tác tạo QR code theo định dạng MoMo cung cấp, khách hàng chỉ cần dùng app MoMo để quét và thanh toán.

Với một lưu ý: khi sử dụng các API phương thức thanh toán này đó là không sử dụng chung, mỗi API đều là riêng biệt, chỉ được chọn 1 trong các phương thức thanh toán để áp dụng.

♦ **Quy trình tích hợp:**

Các bước cơ bản để tiến hành tích hợp cho API Momo:

Bước 1: Đăng ký tài khoản doanh nghiệp: <https://business.momo.vn>.

Bước 2: Lựa chọn phương thức thanh toán hợp lý (ở đây chúng em chọn phương thức thanh toán All In One)

Bước 3: Tiến hành tích hợp.

Bước 4: Quy trình kiểm thử phần mềm, bao gồm các testcase của Momo cung cấp để kiểm tra các lỗi phổ biến trong quá trình thanh toán.

Bước 5: Tuy nhiên toàn bộ quá trình tích hợp chỉ là ở môi trường test, bên cạnh đó Momo sẽ xác thực môi trường test trước khi lên môi trường production (thực tế).

Bước 6: Sau khi đã được xác thực đã lên môi trường production bên tài khoản doanh nghiệp, chúng ta sẽ thay đổi API phù hợp với production.

Bước 7: Triển khai dịch vụ thanh toán cho khách hàng.

♦ **Thông tin tích hợp**

Thông tin cấu hình kết nối với API Momo:

Partner Code: Thông tin để định danh tài khoản doanh nghiệp.

Access Key: Cấp quyền truy cập vào hệ thống MoMo.

Secret Key: Dùng để tạo chữ ký điện tử signature.

Signature: chữ ký điện tử được tạo bằng thuật toán HMAC_SHA256 bao gồm dữ liệu đầu vào là secretkey và data, data được tạo theo định dạng:

key1=value1&key2=value2...

Cấu hình request:

Key	Value
Content-Type	application/json; charset=UTF-8
Method	POST
Domain	Production: https://payment.momo.vn Sandbox: https://test-payment.momo.vn

Hình 2.2. Cấu hình request Momo

Request mẫu:

```
{
  "partnerCode": "MOMO",
  "partnerName": "Test",
  "storeId": "MomoTestStore",
  "requestType": "captureWallet",
  "ipnUrl": "https://momo.vn",
  "redirectUrl": "https://momo.vn",
  "orderId": "MM1540456472575",
  "amount": 150000,
  "lang": "vi",
  "orderInfo": "SDK team.",
  "requestId": "MM1540456472575",
  "extraData": "eyJ1c2VybmFtZSI6ICJtb21vIn0=",
  "signature": "fd37abbee777e13eaa0d0690d184e4d7e2fb43977281ab0e20701721f07a0e07"
}
```

Hình 2.3. Request mẫu Momo

Cấu trúc dữ liệu theo định dạng:

```
partnerCode=$partnerCode&accessKey=$accessKey&requestId=$requestId&amount=$amount&orderId=$orderId&orderInfo=$orderInfo&returnUrl=$returnUrl&notifyUrl=$notifyUrl&extraData=$extraData
```

Dữ liệu được tạo ra từ request mẫu trên:

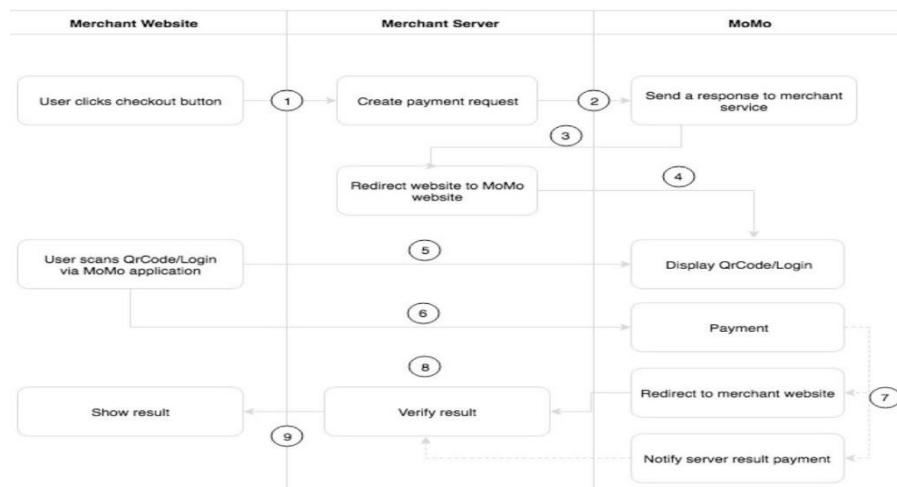
```
"partnerCode=MOMO&accessKey=F8BBA842ECF85&requestId=MM1540456472575&amount=150000&orderId=MM1540456472575&orderInfo=SDKteam.&returnUrl=https://momo.vn&notifyUrl=https://momo.vn&extraData=email=abc@gmail.com"
```

Chữ ký được tạo ra: bằng thuật toán HMAC_SHA256 chuyển về hex

```
h = hmac.new(serecetkey.encode('utf-8'), rawSignature.encode('utf-8'), hashlib.sha256)
signature = h.hexdigest()
```

Hình 2.4. Signature Momo

♦ Sơ đồ xử lý:



Hình 2.5. Sơ đồ xử lý Momo

Bước 1: Khách hàng kiểm tra đơn hàng và chọn MoMo là phương thức thanh toán

Bước 2: Server của bạn tạo session thanh toán và gửi yêu cầu thanh toán qua MoMo

Bước 3: Chuyển trang mua hàng sang trang thanh toán của MoMo.

Bước 4: Khách hàng sử dụng ứng dụng MoMo để quét mã QR hoặc đăng nhập để thanh toán

Bước 5: Sau khi thanh toán MoMo sẽ chuyển khách hàng về trang mua hàng

Bước 6: Server của bạn xác thực giao dịch và cập nhật dịch vụ cho khách hàng

◆ HTTP Request

Request mẫu:

```
{
    "partnerCode": "MOMO",
    "partnerName" : "Test",
    "storeId" : "MomoTestStore",
    "requestType": "captureWallet",
    "ipnUrl": "https://momo.vn",
    "redirectUrl": "https://momo.vn",
    "orderId": "MM1540456472575",
    "amount": 150000,
    "lang": "vi",
    "orderInfo": "SDK team.",
    "requestId": "MM1540456472575",
    "extraData": "eyJ1c2VybmcFtZSI6ICJtb21vIn0=",
    "signature": "fd37abbee777e13eaa0d0690d184e4d7e2fb43977281ab0e20701721f07a0e07"
}
```

Hình 2.6. Request HTTP Momo mẫu

Trong đó:

Attribute	Type	Required	Description
partnerCode	String	✓	Thông tin tích hợp
partnerName	String		Tên đối tác
storeId	String		Thông tin cửa hàng
requestId	String	✓	Định danh mỗi yêu cầu
amount	Long	✓	Số tiền cần thanh toán Min: 1.000 VND Max: 20.000.000 VND Tiền tệ: VND Kiểu dữ liệu: Long
orderId	String	✓	Mã đơn hàng thanh toán của đối tác
orderInfo	String	✓	Thông tin đơn hàng
redirectUrl	String	✓	Một URL của đối tác. URL này được sử dụng để chuyển trang (redirect) từ MoMo về trang mua hàng của đối tác sau khi khách hàng thanh toán. Hỗ trợ: AppLink và WebLink
ipnUrl	String	✓	API của đối tác. Được MoMo sử dụng để gửi kết quả thanh toán theo phương thức IPN (server-to-server)
requestType	String	✓	captureWallet
extraData	String	✓	Giá trị mặc định là trống "" Encode base64 theo định dạng Json: <code>{"key": "value"}</code> Ví dụ với dữ liệu: <code>{"username": "momo"}</code> thì data của extraData là <code>eyJ1c2VybmFtZSI6ICJtb21vIn0=</code>
lang	String	✓	Ngôn ngữ của message được trả về (vi hoặc en)

Hình 2.7. Chú thích params trong request Momo

◆ **HTTP Response**

Response mẫu:

```
{
    "partnerCode": "MOMO",
    "requestId": "MM1540456472575",
    "orderId": "MM1540456472575",
    "amount": 150000,
    "responseTime": 1619422543360,
    "message": "Thành công",
    "resultCode": 0,
    "payUrl": "https://test-payment.momo.vn/v2/gateway/pay?t=TU9NT1NKSTEyMDE5MDgy",
    "deeplink": "momo://?a...appScheme=",
    "qrCodeUrl": "https://test-payment.momo.vn/gw_payment/s/zoVKZd",
    "deeplinkWebInApp": "http://mo...23fb3d1469"
}
```

Hình 2.8. Response Momo mẫu

Trong đó:

Attribute	Type	Required	Description
partnerCode	String	✓	<u>Thông tin tích hợp</u>
requestId	String	✓	Giống với yêu cầu ban đầu
orderId	String	✓	Mã đơn hàng của đối tác
amount	Long	✓	Giống với số tiền yêu cầu ban đầu
responseTime	Long	✓	Thời gian trả kết quả thanh toán về đối tác Định dạng: <u>timestamp</u>
message	String	✓	Mô tả lỗi dựa trên <u>lang</u>
resultCode	int	✓	<u>Mã lỗi</u>
payUrl	String	✓	URL để chuyển từ trang mua hàng của đối tác sang trang thanh toán của MoMo
deeplink	String		URL để mở ứng dụng trực tiếp MoMo (Khách hàng phải cài đặt ứng dụng MoMo trước) và trang xác nhận thanh toán.

Hình 2.9. Chú thích params trong response Momo

♦ Kiểm tra tính toàn vẹn dữ liệu

Dữ liệu trả về được kiểm tra qua resultCode và Signature:

Nếu *resultCode = 0 => giao dịch thành công.*

Nếu `resultCode <> 0 =>` giao dịch **thất bại**.

Việc kiểm tra **resultCode** và cả **Signature** để đảm bảo chắc rằng thông tin không bị thay đổi mỗi khi người dùng đổi `resultCode` ở **returnUrl** mặc dù không thanh toán nhưng dịch vụ vẫn trừ tiền dẫn tới sai lệch, do đó để hoàn thiện tính đúng đắn thì hệ thống phải tạo ra một chữ ký **signature** từ data response so với signature của response trả về.

2.2.9. Thanh toán bằng Paypal

- ◆ **Sơ lược về Paypal**

Trang chủ: <https://api-m.sandbox.paypal.com>

Paypal là một trong những phương tiện phổ biến cho việc thanh toán trực tuyến, hay đơn giản là một ví điện tử được tích hợp trên các trang web để có thể hỗ trợ cho việc mua hàng trở nên nhanh gọn và tiện ích hơn nhất có thể.

Hiện nay Paypal trở nên phổ biến hầu hết ở mọi quốc gia, trong đó có Việt Nam bởi cách thức tích hợp đơn giản và hiệu quả, bên cạnh đó việc sở hữu tài khoản cá nhân và tài khoản doanh nghiệp của Paypal không quá phức tạp để người dùng có thể thực hiện giao dịch bằng những cú click chuột nhẹ nhàng.

- ◆ **Về phương thức hoạt động**

Paypal gồm hai môi trường để tích hợp API là sandbox và live:

Sandbox: <https://api-m.sandbox.paypal.com>

Live: <https://api-m.paypal.com>

Các bước tiến hành để:

Bước 1: Vào <https://www.paypal.com/vn/webapps/mpp/account-selection> có thể đăng ký tài khoản cá nhân hay tài khoản doanh nghiệp, tiếp theo đăng nhập vào tài khoản vừa tạo.

Bước 2: Tiến hành lấy các dữ liệu liên quan trong tài khoản như: secret key và clientId.

Bước 3: Phân rã token bằng Base64 theo cấu trúc *Authorization: Basic <clientid:secret>* ở header.

Bước 4: Gửi request gồm dữ liệu đã cung cấp với các method: GET, POST, PUT, PATCH, DELETE để thực hiện cập nhật, xóa, lấy, hay submit dữ liệu ở header, gửi kèm token đã đính kèm ở header.

Bước 5: Nhận về response và tiến hành thanh toán thông qua link url redirect của response.

Lưu ý: mọi dữ liệu được gửi và nhận ở dạng JSON.

◆ **API request, response**

Các trường dữ liệu của request, response

Parameter	Required	Type	Description
<i>Id</i>	✓	String	Id đã được generate bởi paypal
<i>Name</i>	✓	String	Tên hóa đơn
<i>Description</i>	✓	String	Mô tả hóa đơn

<i>Start_date</i>	✓	String	<p>Thời gian gửi request, thời gian bắt đầu gửi đi không được nhỏ hơn thời gian hiện tại cũng như đơn hàng sẽ mất 24 tiếng để kích hoạt.</p> <p><u>Lưu ý:</u> khi gửi request thì ngày giờ sẽ được lấy và chuyển sang ngày giờ của người bán Ví dụ: ngày giờ bắt đầu gửi là 2017-01-02T14:36:21Z sẽ được chuyển đổi thành ngày giờ của người bán ở Berlin bởi timezone (UTC+1) 2017-01-02T00:00:00.</p>
<i>Payer</i>	✓	Object	Các chi tiết cho khách hàng thanh toán tiền
<i>Shipping_address</i>		Object	Chi tiết ship hàng, phải được cung cấp nếu như nó khác với trong tài khoản mặc định

<i>Override_merchant_preferences</i>		Object	Thực hiện quy định các thông tin trong đơn hàng, bao gồm chi phí đơn hàng, url mà khách hàng có thể phê duyệt hoặc hủy, số lần thanh toán không thành công tối đa được phép, liệu PayPal có tự động lập hóa đơn cho số dư chưa thanh toán trong chu kỳ thanh toán tiếp theo hay không và hành động nếu khoản thanh toán ban đầu của khách hàng không thành công. Nếu bỏ qua params này thì đơn hàng sẽ sử dụng params mặc định.
<i>Override_charge_models</i>		Array	Ghi đè các model tính phí vận chuyển và thông tin thuế.
<i>Plan</i>	✓	Object	Id của đơn hàng được tạo
<i>Links</i>		Array	Mảng bao gồm các đường link của request theo HATEOAS

<i>State</i>	Enum	<p>Trạng thái của đơn hàng:</p> <p>Pending: đang chờ tiến trình thanh toán</p> <p>Active : hóa đơn đã được chấp nhận và các thanh toán đã sẵn sàng.</p> <p>Suspended: các thanh toán bị đình chỉ và sẽ không được tiến hành cho đến khi được kích hoạt lần nữa</p> <p>Cancelled: các đơn hàng bị hủy bỏ và thanh toán sẽ không tiến hành</p> <p>Expired: các hóa đơn hết hạn và thanh toán sẽ không được tiến hành</p>
--------------	------	---

Bảng 2Error! No text of specified style in document.1. Chú thích params trong request, response Paypal

Request mẫu:

```

curl -v -X POST https://api-m.sandbox.paypal.com/v1/payments/payment \
-H "Content-Type: application/json" \
-H "Authorization: Bearer Access-Token" \
-d '{
    "intent": "sale",
    "payer": {
        "payment_method": "paypal"
    },
    "note_to_payer": "Contact us for any questions on your order.",
    "redirect_urls": {
        "return_url": "https://example.com/return",
        "cancel_url": "https://example.com/cancel"
    }
    "transactions": [
        {
            "amount": {3 keys},
            "description": "The payment transaction description.",
            "custom": "EBAY_EMS_90048630024435",
            "invoice_number": "48787589673",
            "payment_options": {
                "allowed_payment_method": "INSTANT_FUNDING_SOURCE"
            },
            "soft_descriptor": "ECHI5786786",
            "item_list": {2 keys}
        ],
    }
}'

```

Hình 2.10. Request mẫu Paypal

```
{  
    "id": "PAY-1B56960729604235TKQQIYVY",  
    "create_time": "2017-09-22T20:53:43Z",  
    "update_time": "2017-09-22T20:53:44Z",  
    "state": "CREATED",  
    "intent": "sale",  
    "payer": {  
        "payment_method": "paypal"  
    },  
    "transactions": [  
        {  
            "amount": {3 keys},  
            "description": "The payment transaction description.",  
            "custom": "EBAY_EMS_90048630024435",  
            "invoice_number": "48787589673",  
            "item_list": {2 keys}  
        },  
        {"links": [  
            - {3 keys},  
            - {3 keys},  
            - {3 keys}  
        ]}  
    ]  
}
```

Hình 2.11. Response mẫy Paypal

2.2.10.Tích hợp JWT (Json Web Token)

♦ Giới thiệu sơ lược về JWT

Trang chủ: <https://jwt.io/>

JWT (*JSON WEB TOKEN*) là một phương tiện giao tiếp chuyên giao giữa client và server được định dạng bằng JSON. Trong đó các token sẽ được chia làm 3 phần: **Header, Payload, Signature**.

Lưu ý: các phần được ngăn cách bằng dấu “.”



Hình 2.12. Các thành phần chính của JWT

♦ Chi tiết cấu trúc của JWT

Header: thường sẽ chứa kiểu dữ liệu và thuật toán dùng để tạo JWT.

```
{
  "typ": "JWT",
  "alg": "HS256"
}
```

Hình 2.13. Cấu trúc Header JWT

Trong đó typ chỉ ra đối tượng là JWT, còn alg là chỉ thuật toán dùng để mã hóa là HS256.

Payload: sẽ chứa những thông tin cần đặt vào trong chuỗi JWT như userId, username, note...

```
{
  "user_name": "admin",
  "user_id": "1513717410",
  "authorities": "ADMIN_USER",
  "jti": "474cb37f-2c9c-44e4-8f5c-1ea5e4cc4d18"
}
```

Hình 2.14. Cấu trúc Payload JWT

Signature: là chuỗi mã hóa được truyền vào payload và secret key, trong đó secret key được người dùng quy định:

```
secret_key = app.config['SECRET_KEY']
payload = {
    'account_id': account['account_id'],
    'iat': datetime.utcnow(),
    'exp': datetime.utcnow() + timedelta(minutes=300000)
}
access_token = jwt.encode(payload, secret_key)
result = {
    'access_token': access_token,
    'account': account,
    'user_info': user,
}
```

Hình 2.15. Quy trình tạo Signature JWT

- ◆ **Sơ đồ xử lý**

Bước 1: người dùng đăng nhập vào tài khoản.

Bước 2: sau khi người dùng đăng nhập hệ thống sẽ xác thực người dùng thông qua phân quyền, nếu hợp lệ sẽ tìm kiếm accountId đồng thời truyền vào **payload** kèm dateTIme và expireDate.

```
account = AccountRep.Authenticate(acc)
if (account['role']['role_id'] == 3): # customer
    user = (models.Customers.query.filter(models.Customers.account_id == account['account_id'],
                                           models.Customers.account_id != None).first().serialize())
if (account['role']['role_id'] == 1 or account['role']['role_id'] == 2): # admin, manager
    user = (models.Employees.query.filter(models.Employees.account_id == account['account_id'],
                                           models.Employees.account_id != None).first().serialize())
```

Hình 2.16. Quy trình xác thực quyền người dùng trong JWT

Bước 3: encode ra chuỗi token gồm secret key và payload.

```
secect_key = app.config['SECRET_KEY']
payload = {
    'account_id': account['account_id'],
    'iat': datetime.utcnow(),
    'exp': datetime.utcnow() + timedelta(minutes=300000)
}
access_token = jwt.encode(payload, secect_key)
result = {
    'access_token': access_token,
    'account': account,
    'user_info': user,
}
```

Hình 2.17. Quy trình tạo token JWT

Bước 4: mỗi lần thực hiện hành động cần xác thực như thay đổi mật khẩu hoặc vào hệ thống thêm nhân viên...Hệ thống sẽ lấy chuỗi token của người dùng tiến hành *decode* với **secret key** để lấy **accountId** cũng như phân quyền của người dùng để do đó có thể xử lý hợp lệ và rõ ràng bảo mật tốt hơn.

```

def extractToken(token):
    payload = jwt.decode(token, app.config['SECRET_KEY'])
    accountId = payload["account_id"]

    account = AccountRep.getAccountById(accountId)
    return account

```

Hình 2.18. Quy trình xác thực người dùng dựa trên token JWT

Lưu ý: với những trường hợp mã token đã được tạo ra thì sẽ có thời gian hết hạn được quy định bởi người dùng ở payload để kiểm soát hoạt động của người dùng qua đó ngăn chặn việc xâm nhập vào hệ thống mà không qua đăng nhập.

```

def sessionInfo(token):
    invalid_msg = {
        'message': 'Token không hợp lệ.',
        'authenticated': False
    }
    expired_msg = {
        'message': 'Token hết hạn sử dụng.',
        'authenticated': False
    }
    try:
        payload = jwt.decode(token, app.config['SECRET_KEY'])
        accountId = payload["accountId"]
        account = AccountRep.getAccountById(accountId)
        result = {
            'accessToken': token,
            'account': account,
        }
        return result

    except jwt.ExpiredSignatureError:
        return jsonify(expired_msg), 401 # 401 is Unauthorized HTTP status code
    except (jwt.InvalidTokenError) as e:
        return jsonify(invalid_msg), 401

```

Hình 2.19. Quy định token trong JWT

◆ **Vai trò cần thiết của JWT**

Hầu hết các hoạt động tương tác với hệ thống cần xác thực đều áp dụng token để việc bảo mật trở nên chặt chẽ và cần thiết để hệ thống phát triển lâu dài hơn.

Ở mỗi nền tảng ngôn ngữ lập trình khác nhau thì JWT sẽ được tích hợp trong những cách khác nhau và đa dạng hơn tùy nhu cầu của người dùng nhưng chúng đều đưa chúng ta đến một mục đích cuối cùng là xác thực và bảo mật bởi vì đi với những rủi ro trong ngành công nghệ là việc truy cập trái phép, xâm nhập hệ thống sử dụng tài nguyên một cách không hợp lệ là điều thiết yếu cần được quan tâm hơn bao giờ hết.

2.2.11.Tích hợp chức năng đăng nhập bằng Google

◆ **Sơ lược về Google**

Google là một trong những công ty công nghệ đa quốc gia của Mỹ. Không ai trong chúng ta là không biết về Google. Ngay từ việc tìm kiếm thông tin thì Google đã là công cụ tuyệt vời, lớn nhất thế giới bởi sở hữu lượng truy cập nhiều và lượng người dùng rộng rãi.

Với sự lớn mạnh như thế nên Google ngày càng phát triển cùng với những tiện ích dịch vụ hỗ trợ người dùng một cách tuyệt đối. Trong đó có dịch vụ giúp người dùng đăng nhập vào tài khoản Google (Gmail).

◆ **Các bước tiến hành cấu hình Google Console**

Bước 1: truy cập vào Google Console. Thực hiện đăng kí và đăng nhập thông tin.

Bước 2: tạo một project mới và đặt tên,

The screenshot shows the Google Developers Console dashboard. At the top left is the 'Google Developers Console' logo. To its right is a dropdown menu labeled 'Select a project'. Below the header is a search bar with the placeholder 'Filter by label, name or id' and a 'Columns' dropdown. A 'Labels' icon is also present. A prominent blue button labeled 'Create Project' is highlighted with a red rectangular box. To the right of the button is a table with columns: 'Project Name', 'Project ID', 'Requests', 'Errors', and 'Charges'. The table lists several projects, each with edit and delete icons.

Hình 2.20. Tạo project Google Dashboard

Sau đó click vào trang chi tiết của dự án theo đường dẫn APIs & auth => Credentials => OAuth. Thực hiện cung cấp các thông tin cần thiết cho dự án theo Google.

The screenshot shows the 'Credentials' page within the Google Developers Console for the project 'test-project-123xyz'. On the left sidebar, under 'APIs & auth', the 'Credentials' tab is highlighted with a red box. The main content area is titled 'OAuth consent screen'. It contains fields for 'Email address' (brijeshb42@gmail.com), 'Product name' (Product name), 'Homepage URL (Optional)', and 'Product logo (Optional)'. A note above the fields states: 'The consent screen will be shown to users who grant your app permission to access their private data using your client ID'. A note below the fields states: 'Note: This screen will be shown for all application client IDs'.

Hình 2.21. Chứng thực trên Google Dashboard

Bước 3: sau đó click vào tab Credentials => Add credentials => OAuth 2.0
 Client ID => Tick vào Web Application => Cung cấp đường dẫn
<http://localhost:port> => URIs <http://localhost:port/gCallback>

Name
Web client 1

Authorized JavaScript origins
Enter JavaScript origins here or redirect URIs below (or both) ⓘ
Cannot contain a wildcard (http://*.example.com) or a path (<http://example.com/subdir>).

<http://localhost:5000> x
<http://www.example.com>

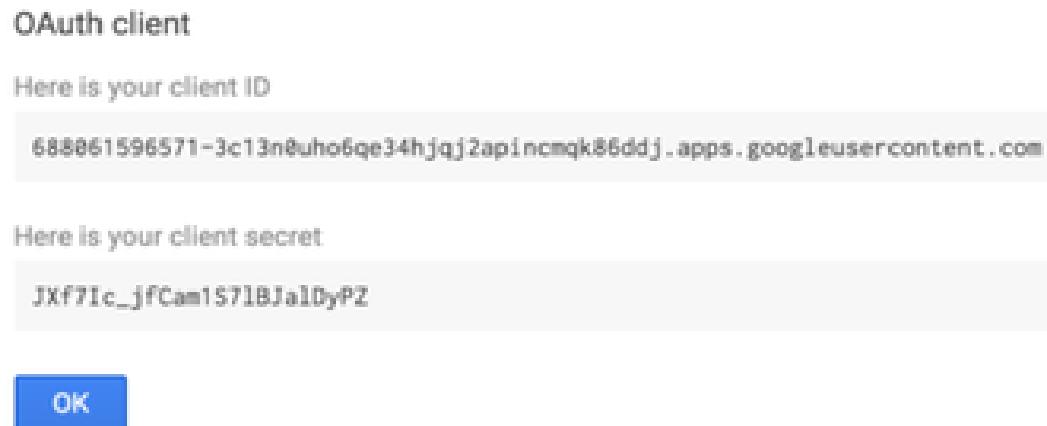
Authorized redirect URIs
Must have a protocol. Cannot contain URL fragments or relative paths. Cannot be a public IP address.

<http://localhost:5000/gCallback>

Create Cancel

Hình 2.22. Chứng thực trên Google Dashboard

Bước 4: sao chép cả 2 chuỗi là ClientID và clientSecret lưu trong văn bản



Hình 2.23. Chứng thực trên Google Dashboard

◆ Tiết hành tích hợp

Sau khi đã thực hiện các bước trên và đã có bảng User dưới cơ sở dữ liệu.

Chúng ta tiến hành cấu hình cho ứng dụng từ những thông tin tích hợp mà Google cung cấp.

```
class Auth:
    CLIENT_ID = ('258914989074-2rkptjvoc5mv1biv91pg4hhqd1igc9fs.apps.googleusercontent.com')
    CLIENT_SECRET = '1DtIDHTRYgVUQtdiGB_FNdXh'
    REDIRECT_URI = 'http://127.0.0.1:5000/gLogin/gCallback'
    AUTH_URI = 'https://accounts.google.com/o/oauth2/auth'
    TOKEN_URI = 'https://accounts.google.com/o/oauth2/token'
    USER_INFO = 'https://www.googleapis.com/userinfo/v2/me'
```

Hình 2.24. Cấu hình Google trong Flask

Trong đó:

ClientID và **ClientSecret** là do Google cung cấp đã được sao chép vào tệp ở bước trên.

REDIRECT_URI: là đường dẫn google sẽ chuyển trang đồng thời là API của người dùng xử lý kết quả trả về của Google sau khi đăng nhập.

AUTH_URI: là đường dẫn Google sẽ chuyển đến trang đăng nhập.

TOKEN_URI: là token tạm thời được Google sử dụng để trao đổi thông báo cho access_token

USER_INFO: các thông tin của người dùng được lấy thông qua URL này.

SCOPE: là loại thông tin người dùng được mã hóa sau khi truy cập vào AUTH_URI để xác thực

Sau khi config xong tiến hành đăng nhập vào Google thông qua hàm **glogin()**:

```

@app.route("/gLogin", methods=['POST', 'GET'])
def gLogin():
    google_provider_cfg = get_google_provider_cfg()
    authorization_endpoint = google_provider_cfg["authorization_endpoint"]
    request_uri = client.prepare_request_uri(
        authorization_endpoint,
        redirect_uri=request.base_url + "/gCallback",
        scope=["openid", "email", "profile"],
    )
    return jsonify({
        "url": request_uri
    })

```

Hình 2.25. Hàm login vào Google

Với hàm này sẽ lấy được toàn bộ config của google thông qua việc gọi `get_google_provider_cfg()`:

```

def get_google_provider_cfg():
    return requests.get(GOOGLE_DISCOVERY_URL).json()

```

Hình 2.26. Google provider config

Với **GOOGLE_DISCOVERY_URL** được lấy từ thông tin của Google Console trong tài khoản.

Sau đó lấy link xác thực thông qua `authorization_endpoint`, tiến hành tạo request thông qua `request_uri`:

```

request_uri = client.prepare_request_uri(
    authorization_endpoint,
    redirect_uri=request.base_url + "/gCallback",
    scope=["openid", "email", "profile"],
)

```

Hình 2.27. Request_uri trong Flask

Với client được lấy từ thư viện của Google:

```
from config import GOOGLE_DISCOVERY_URL, GOOGLE_CLIENT_ID, GOOGLE_CLIENT_SECRET
client = WebApplicationClient(GOOGLE_CLIENT_ID)
def get_google_provider_cfg():
    return requests.get(GOOGLE_DISCOVERY_URL).json()
def get_google_auth(state=None, token=None):
    if token:
        return OAuth2Session(Auth.CLIENT_ID, token=token)
    if state:
        return OAuth2Session(Auth.CLIENT_ID, state=state, redirect_uri=Auth.REDIRECT_URI)
    oauth = OAuth2Session(Auth.CLIENT_ID, redirect_uri=Auth.REDIRECT_URI, scope=Auth.SCOPES)
    return oauth
```

Hình 2.28. Thư viện cần thiết trong Flask để sử dụng Google

Sau khi đã tạo được request tiến hành trả về dưới dạng json.

Tiếp theo là phần hàm **gCallback()**:

```
@app.route("/gLogin/gCallback", methods=['POST', 'GET'])
def gCallback():
    # Get authorization code Google sent back to you
    code = request.args.get("code")
    google_provider_cfg = get_google_provider_cfg()
    token_endpoint = google_provider_cfg["token_endpoint"]

    # Prepare and send a request to get tokens! Yay tokens!
    token_url, headers, body = client.prepare_token_request(
        token_endpoint,
        authorization_response=request.url,
        redirect_url=request.base_url,
        code=code
    )

    token_response = requests.post(
        token_url,
        headers=headers,
        data=body,
        auth=(GOOGLE_CLIENT_ID, GOOGLE_CLIENT_SECRET),
    )

    # Parse the tokens!
    client.parse_request_body_response(json.dumps(token_response.json()))

    userinfo_endpoint = google_provider_cfg["userinfo_endpoint"]
```

```

uri, headers, body = client.add_token(userinfo_endpoint)
userinfo_response = requests.get(uri, headers=headers, data=body)
print("userinfo_response: ", userinfo_response.json())
if userinfo_response.json().get("email_verified"):
    unique_id = userinfo_response.json()["sub"]
    users_email = userinfo_response.json()["email"]
    picture = userinfo_response.json()["picture"]
    last_name = userinfo_response.json()["family_name"]
    first_name = userinfo_response.json()["given_name"]
else:
    return "User email not available or not verified by Google.", 400

customer = getCustomerByEmail(users_email)
if customer != None:
    account = getAccountByID(customer.account_id)
else:
    account = CreateAccount(CreateAccountReq(
        {
            "role_id": 3,
            "account_name": ''.join(random.choices(string.ascii_uppercase + string.digits, k=10)),
            "account_password": ''.join(random.choices(string.ascii_uppercase + string.digits, k=10))
        }
    ))
    create_customer = models.Customers(last_name=last_name,
                                         first_name=first_name,
                                         email=users_email,
                                         account_id=account['account_id'],
                                         )
    db.session.add(create_customer)
    db.session.commit()

if (account['role']['role_id'] == 3): # customer
    user = (models.Customers.query.filter(models.Customers.account_id
== account['account_id'],
                                         models.Customers.account_id
!= None).first()).serialize()
    secect_key = app.config['SECRET_KEY']
    payload = {
        'account_id': account['account_id'],
        'iat': datetime.utcnow(),
        'exp': datetime.utcnow() + timedelta(minutes=300000)
    }
    access_token = jwt.encode(payload, secect_key)
    result = {
        'access_token': access_token,
        'account': account,
        'user_info': user,
    }
    res = LoginRsp(result).serialize()

```

```

print(res)
socketio.emit('login-google', {"data": res, "success": True}, room=0)
return render_template("hello.html")

```

Thông qua hàm này, sẽ xác thực được thông tin google và tạo ra token, qua đó Nếu có đã có email trùng với gmail dưới cơ sở dữ liệu hệ thống sẽ tiến hành đăng nhập bằng chính account có email đó.Còn nếu không có sẽ thành lập account với username và password là duy nhất và mang tính che giấu dữ liệu đối với người dùng đăng nhập bằng cách này.

Mọi người dùng đăng nhập thông qua Google với tài khoản chưa có trên ứng dụng web sẽ không biết được mật khẩu của tài khoản trên ứng dụng web là gì và họ chỉ có thể đổi mật khẩu trên Gmail thay vì đổi trên ứng dụng so với những người dùng đã đăng ký trực tiếp.Việc sử dụng socket nhằm chuyển port giữa backend và frontend, để truyền tải dữ liệu. Khi front end nhận được sự kiện **login-google** sẽ nhận được dữ liệu và message, thông qua đó dữ liệu được xử lý để render trực tiếp cho người dùng gồm những thông tin cần thiết.

2.2.12.Tích hợp Telegram Bot

- ◆ **Giới thiệu sơ lược về Telegram**

Telegram là ứng dụng dùng để nhắn tin, liên lạc trao đổi giữa các người dùng. Sở dĩ Telegram trở nên phổ biến rộng rãi là nhờ sự kết hợp của tốc độ và tính bảo mật cao.

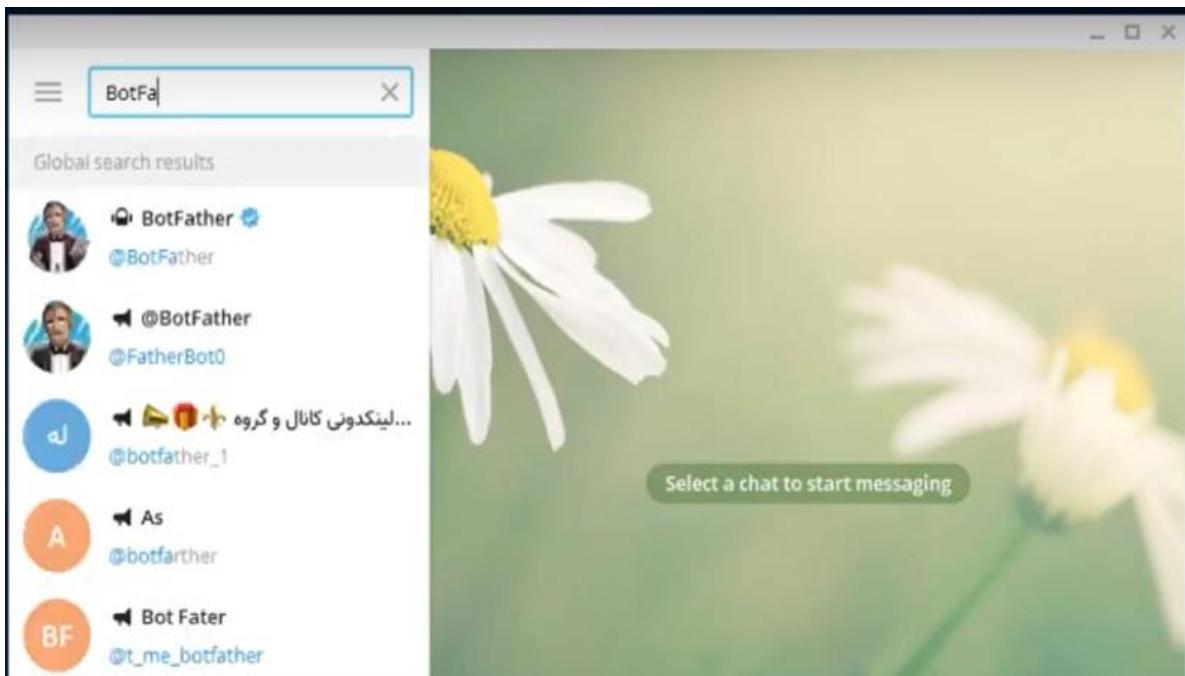
Các tin nhắn của Telegram có tính bảo mật rất cao thậm chí đến cả các công ty an ninh cũng rất khó để tiếp cận. Được ra đời từ nước Nga, Telegram được phát triển như một ứng dụng mã nguồn mở, do đó ứng dụng này không ngừng phát triển bởi các developer từ bên ngoài. Ngoài ra Telegram là một ứng dụng miễn phí, mọi thứ về Telegram đều là phi lợi nhuận.

Với quan niệm hướng về người dùng, Telegram luôn lắng nghe nhu cầu của người dùng và phát triển chúng theo thị hiếu của họ. Bởi vì những yếu tố trên mà Telegram trở thành ứng dụng mà nhiều người biết đến. Bên cạnh đó việc tích hợp Telegram vào sản phẩm là điều hết sức bình thường bởi quy trình đơn giản, người dùng cao, dễ sử dụng và tiện lợi. Một trong số đó là Telegram Bot.

◆ Quy trình tiến hành cấu hình Telegram Bot

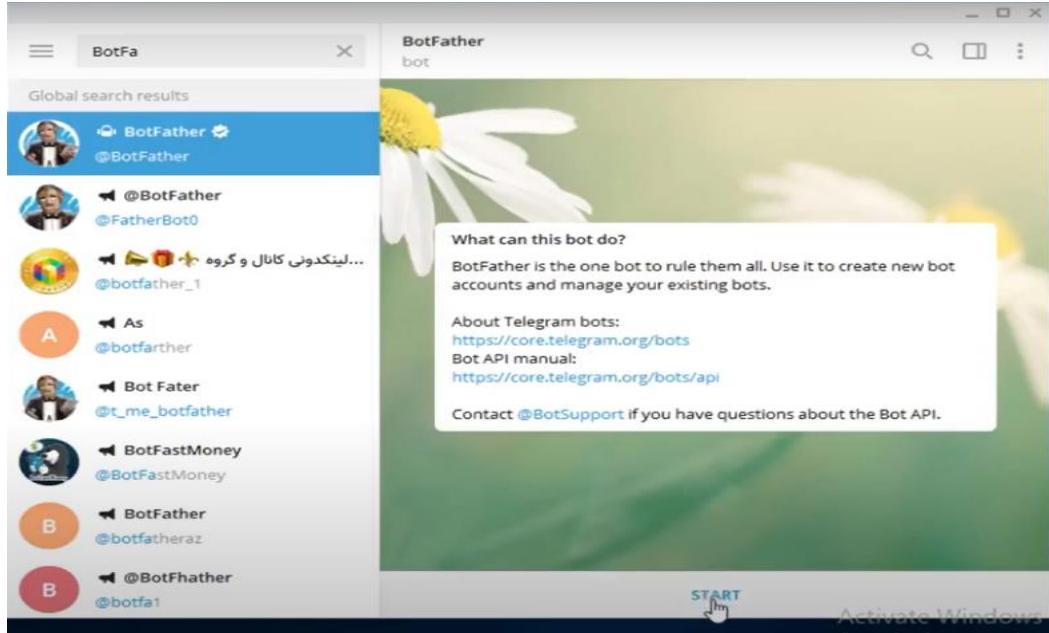
Telegram Bot là một robot dùng để thực hiện cái nhiệm vụ được định giờ sẵn và tự động trên Telegram, việc sở hữu con Bot này giúp việc làm được tự động hóa và tiết kiệm thời gian. Cùng với sự đo lường khủng về số lượng người dùng nên việc tích hợp con Bot này ngày càng được ưa chuộng. Nếu thế thì làm sao để cấu hình và tích hợp Telegram Bot vào ứng dụng:

- Mở ứng dụng Telegram đã cài đặt
- Tìm kiếm tên người dùng BotFather



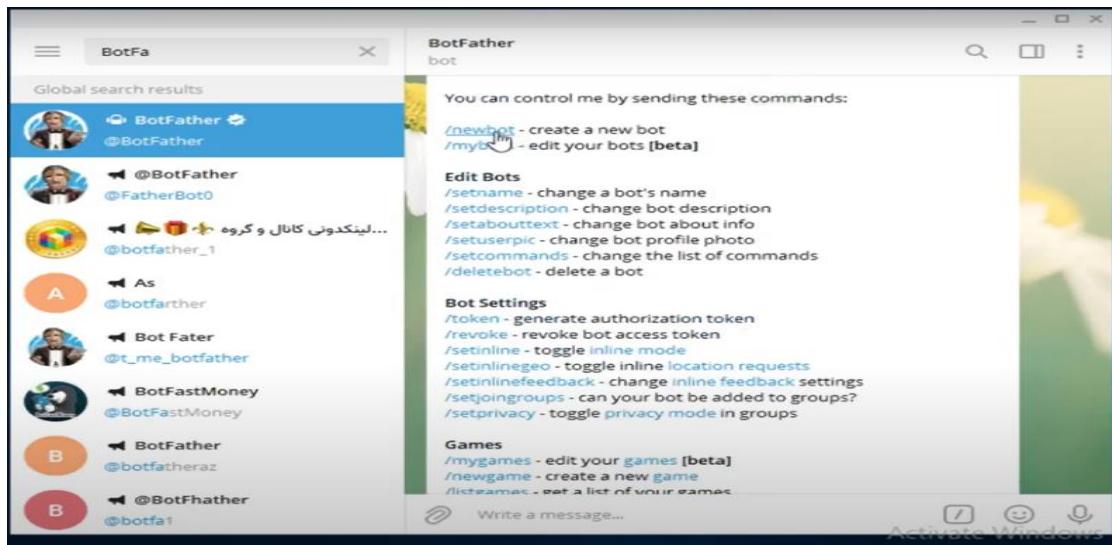
Hình 2.29. Tìm kiếm BotFather

- Click vào BotFather và click vào start để bắt đầu sử dụng Bot



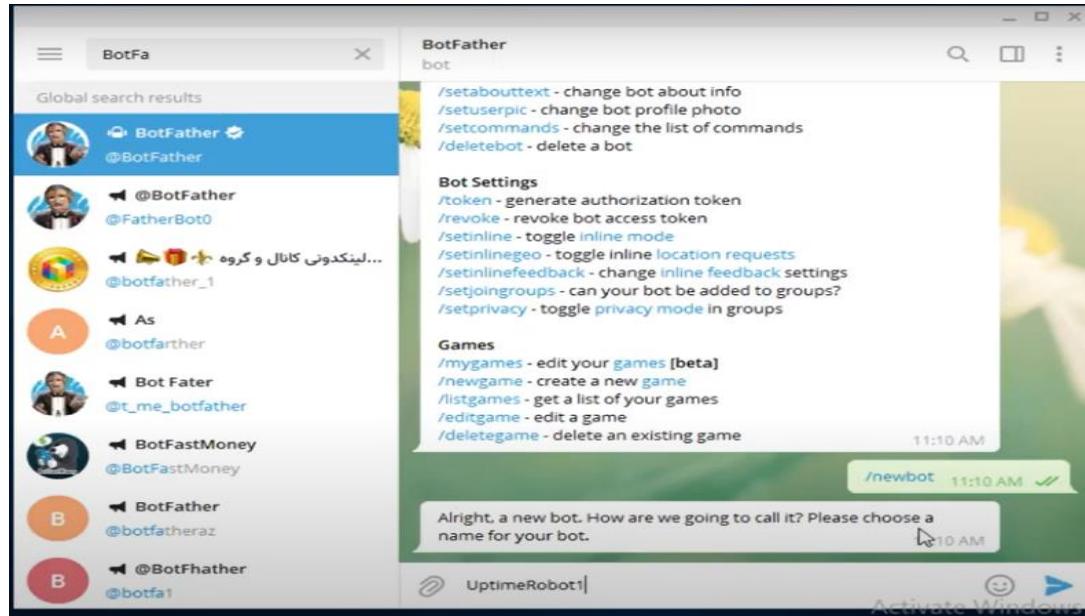
Hình 2.30. Khởi động BotFather

- Sau khi start, màn hình sẽ xuất hiện các lựa chọn, click vào newbot để tạo Bot mới:



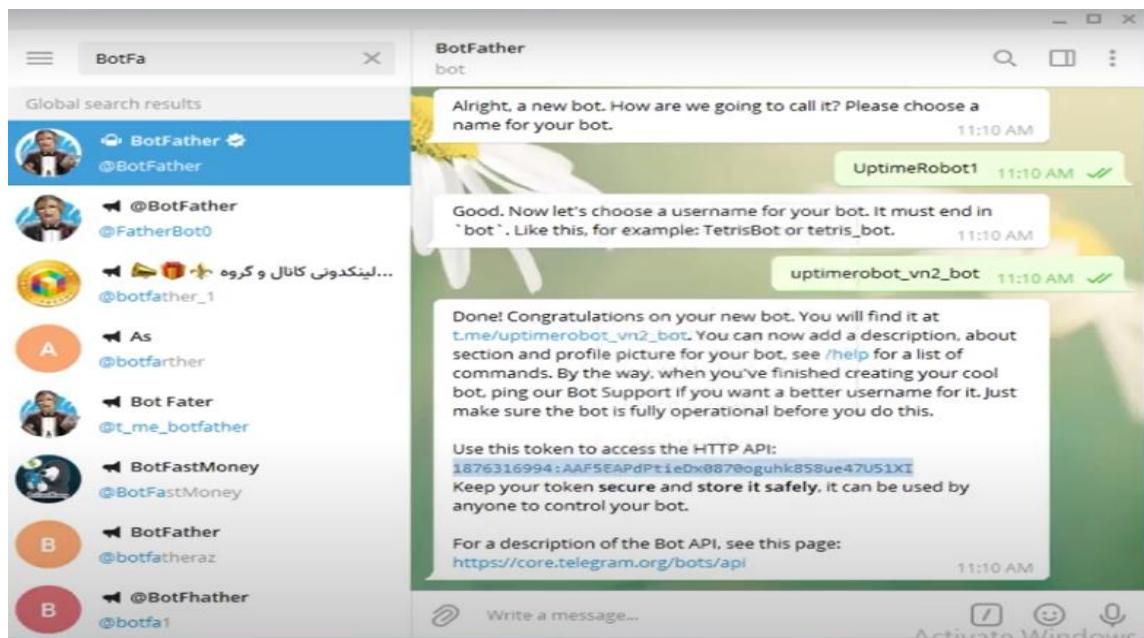
Hình 2.31. Tạo một Bot mới

- Sau đó đặt tên cho Bot:



Hình 2.32. Đặt tên cho Bot

- Sau khi đặt tên cho Bot, bạn được một chuỗi token:



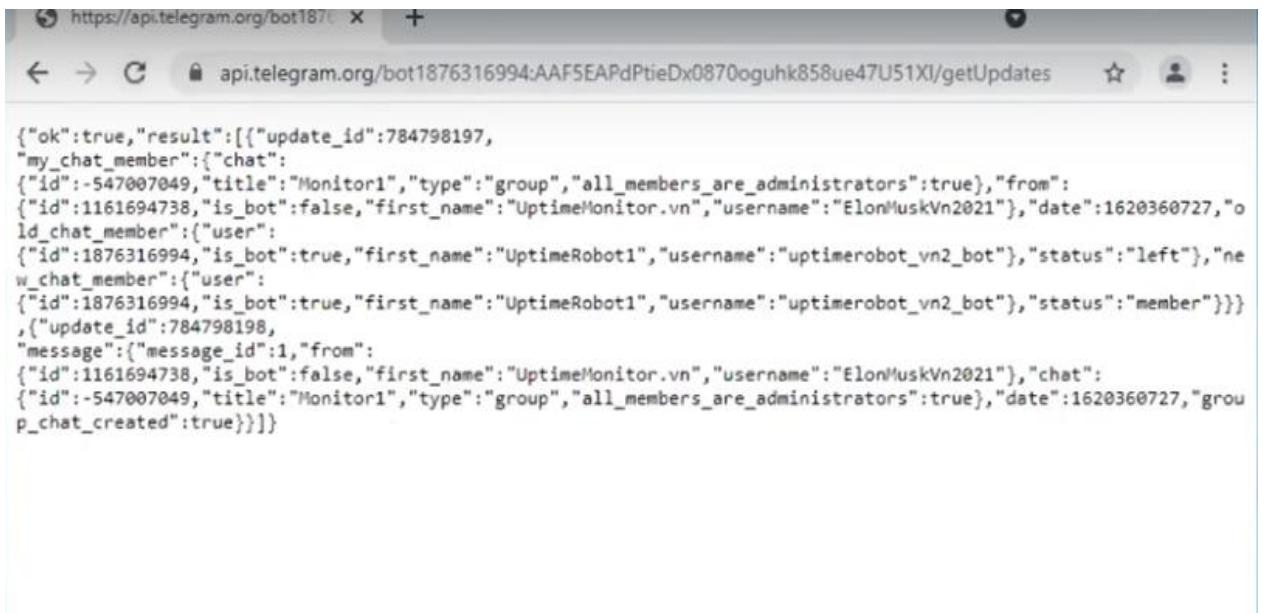
Hình 2.33. Chuỗi token được tạo ra trong Bot

- Bạn dùng chuỗi này copy vào đường dẫn:
<https://api.telegram.org/bot{token}/getUpdates>



Hình 2.34. Trạng thái của Bot

- Sau đó tạo nhóm người dùng và bot trong Telegram, thực hiện nhắn tin từ phía người dùng.
- Sau đó reload lại trang api.telegram như trên:



Hình 2.35. Lấy ID của Bot

- Copy chuỗi token khi nãy và id có dấu “-” phía trước lưu vào tệp.

◆ Tiết hành tích hợp

Sau khi có thông tin ID và token của Bot, tiến hành gửi api về phía front-end:

```
@Injectable({
  providedIn: 'root'
})
export class TelegramService {
  groupId = '-436944285';

  constructor(
    private http: HttpClient,
  ) {
  }

  async sendMessage(doAn0uGroupId, msg, enableHTML = true) {
    const doAn0uUrl =
      'https://api.telegram.org/bot1704225056:AAFXTHV_JiyGYkHBBLI24ByE_oRcH_SSaqg/sendMessage';
    const doAn0uData: any = {
      chat_id: doAn0uGroupId,
      text: StringHandler.parseHtmlSpecialCharacters(msg)
    };
    if (enableHTML) {
      doAn0uData.parse_mode = 'HTML';
    }

    return this.http.post(doAn0uUrl, doAn0uData).toPromise();
  }
}
```

Hình 2.36. Hàm sendMessage Telegram

Gửi đến đường dẫn:

<https://api.telegram.org/bot{token}/sendMessage>

để tiến hành truyền dữ liệu theo key:value với key lần lượt là chat_id, text và value lần lượt là ID đã lấy ở trên và message do người dùng truyền vào.

enableHTML dùng để truyền chuỗi chứa HTML vào.

Sau khi thực hiện hàm sendMessage xong, sẽ đến lúc gọi hàm testMessage để gửi tin nhắn mẫu thông qua Bot:

```
async sendCreateCustomerAccount(account, customer) {
  const msg = `
<strong>ĐĂNG KÝ TÀI KHOẢN KHÁCH HÀNG!</strong>
-----
Họ tên: <strong>${customer.last_name} ${customer.first_name}</strong>
Số điện thoại: <strong>${customer.phone}</strong>
Email: ${customer.email}
Tên tài khoản: ${account.account_name}
`;
  try {
    await this.sendMessage(this.groupId, msg);
  } catch (e) {
    debug.log('ERROR in send testMessage: ', e);
  }
}
```

Hình 2.37. Hàm testMessage Telegram

Thực hiện try catch để bắt ngoại lệ nếu không thành công

Sau khi thực hiện hàm testMessage thì ở Bot Telegram sẽ hiển thị như thế này:



Hình 2.38. Kết quả của quá trình tích hợp

2.2.13. Lưu trữ ảnh trên Clouudinary

- ◆ **Sơ lược về Clouudinary**

Cloudinary là một dịch vụ đám mây dùng để quản lý ảnh và video, với tốc độ load ảnh nhanh và kho lưu trữ lớn, Cloudinary ngày càng được sử dụng rộng rãi trong các công nghệ đặc biệt là những trang web có rất nhiều hình ảnh cần hiển thị nhưng không biết lưu trữ nó ở đâu để không bị tốn hay bị ngắn các tài nguyên.

Một trong những yếu tố giúp Cloudinary trở nên phổ biến là do tính hướng đến người dùng bằng cách tối ưu hóa việc lưu trữ, chỉnh sửa ảnh một cách miễn phí, bên cạnh đó vẫn có gói trả phí tuy nhiên với gói miễn phí thì người dùng đã có thể sử dụng hầu hết và tốt các dịch vụ của Cloudinary.

- ◆ **Quy trình cấu hình và chuẩn bị môi trường cho Cloudinary trong Flask**

Bước 1: truy cập vào <https://cloudinary.com/> để đăng ký tài khoản sử dụng, bao gồm các gói trả phí và miễn phí. Ở đây đăng ký gói miễn phí vì nó đã quá đủ tiện ích để dùng cho ứng dụng.

Bước 2: sau khi đã có tài khoản và đăng nhập thành công, tiến hành sao chép các thông tin cần thiết như: Cloud name, API key, Secret key và chép chúng vào một tệp để lưu trữ

Bước 3: cài đặt thư viện Cloudinary thông qua lệnh: pip install cloudinary vào ứng dụng python Flask

- ◆ **Quy trình tích hợp Cloudinary vào Flask**

Bước 1: sao chép các mục như Cloud nam, API key, Secret key vào file .config, lần lượt import các thư viện và file cần thiết như: request, cloudinary.uploader từ thư viện Cloudinary, config...

Bước 2: tiến hành viết API dùng để gọi hàm UploadBookImage():

```
@app.route('/admin/book-management/upload-book-image', methods=['POST'])
def UploadBookImage():
    import cloudinary.uploader
    file = request.files['image']
    tail_image = file.filename.split('.')[1]
    filename = secure_filename(str(randint(1,1000000000000))+ '.' +tail_image)
    # file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename)) // luu file anhr vào thư mục
    cloudinary.config(
        cloud_name = app.config['CLOUD_NAME'],
        api_key = app.config['API_KEY'],
        api_secret = app.config['API_SECRET']
    )

    res = cloudinary.uploader.upload(file, public_id=filename.split('.')[0], width=600, height=900, crop="lfill")
    cloudinary.utils.cloudinary_url("sample_remote.jpg")
    return jsonify({'image': res['url']})
```

Hình 2.39. Chi tiết hàm UploadBookImage

Trong đó:

Request sẽ nhận một form data có chứa image từ phía client.

Tiến hành random unique chuỗi tên của file dùng để lưu trên Cloudinary

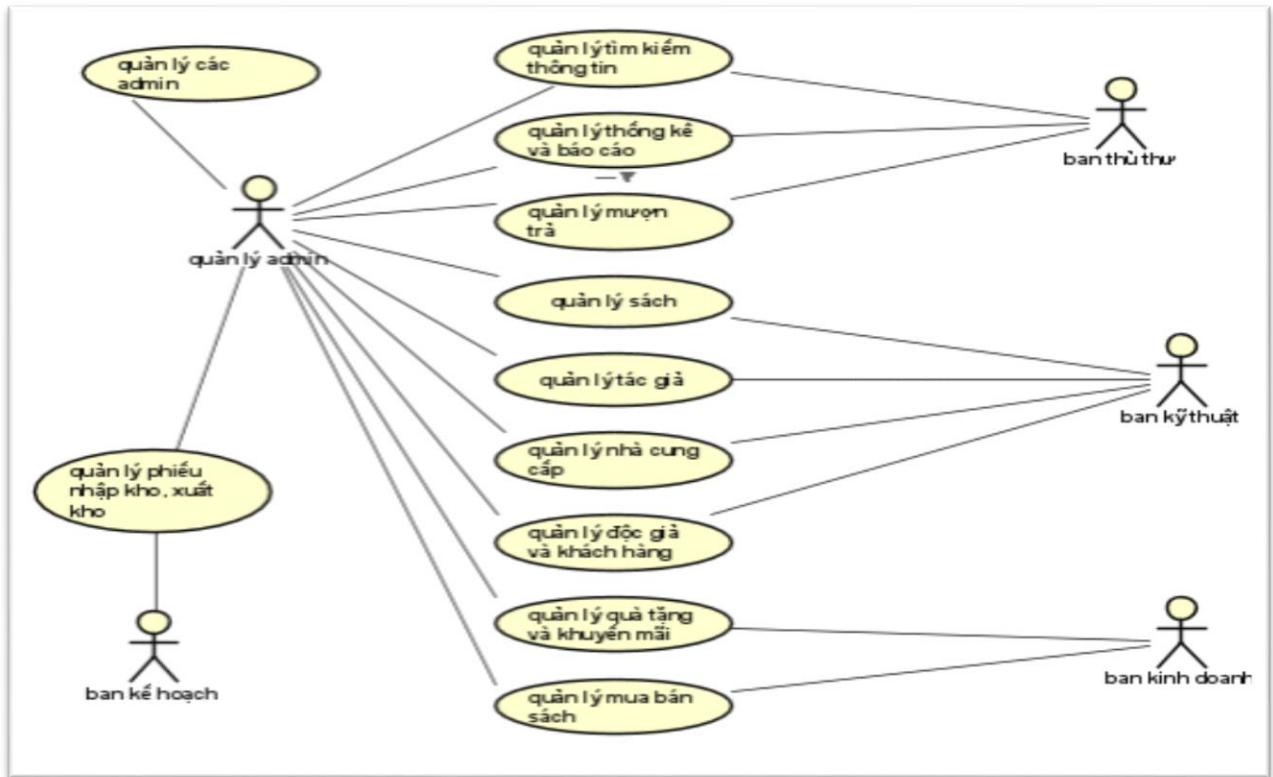
Tiếp theo sử dụng phương thức config có từ thư viện của Cloudinary để upload các thông tin cần thiết ở file .config.

Thực hiện upload file với publicId, width, height (kích thước của ảnh) và crop (kéo dãn hình ảnh). Upload thành công lên cloudinary sẽ trả về một object gồm lần lượt các thuộc tính.

Cuối cùng thực hiện trả về thuộc tích url cần thiết lấy từ object trên để xử lý logic cho việc lưu xuống database trong các API tiếp theo do người dùng quyết định.

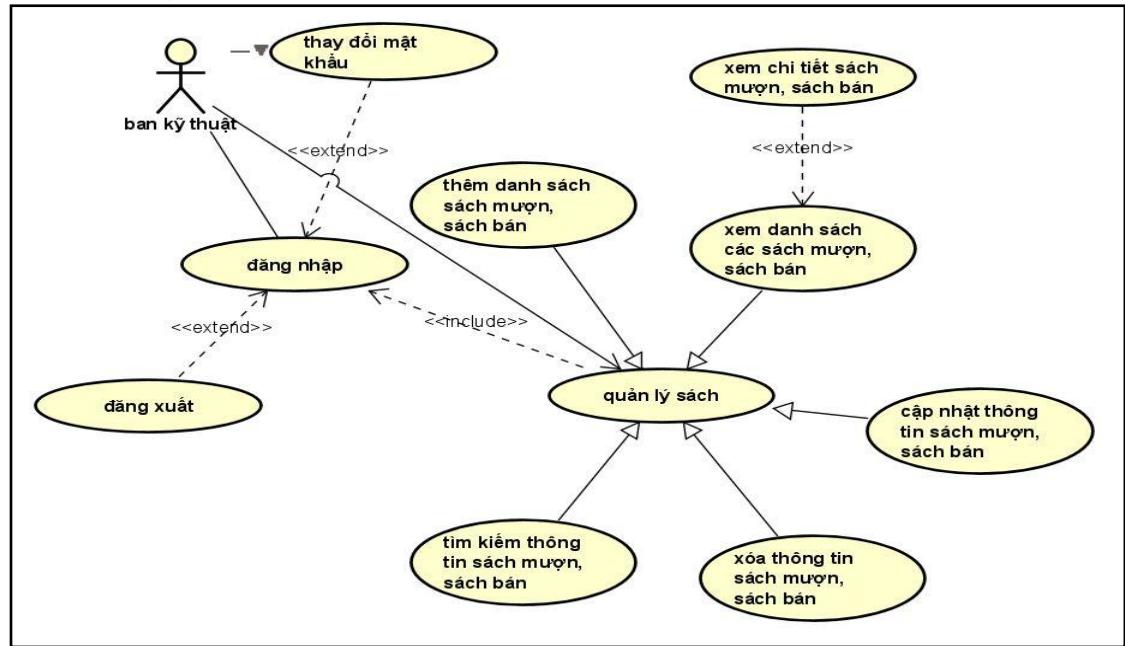
2.3 Biểu đồ Usecase

2.3.1. Tổng quát



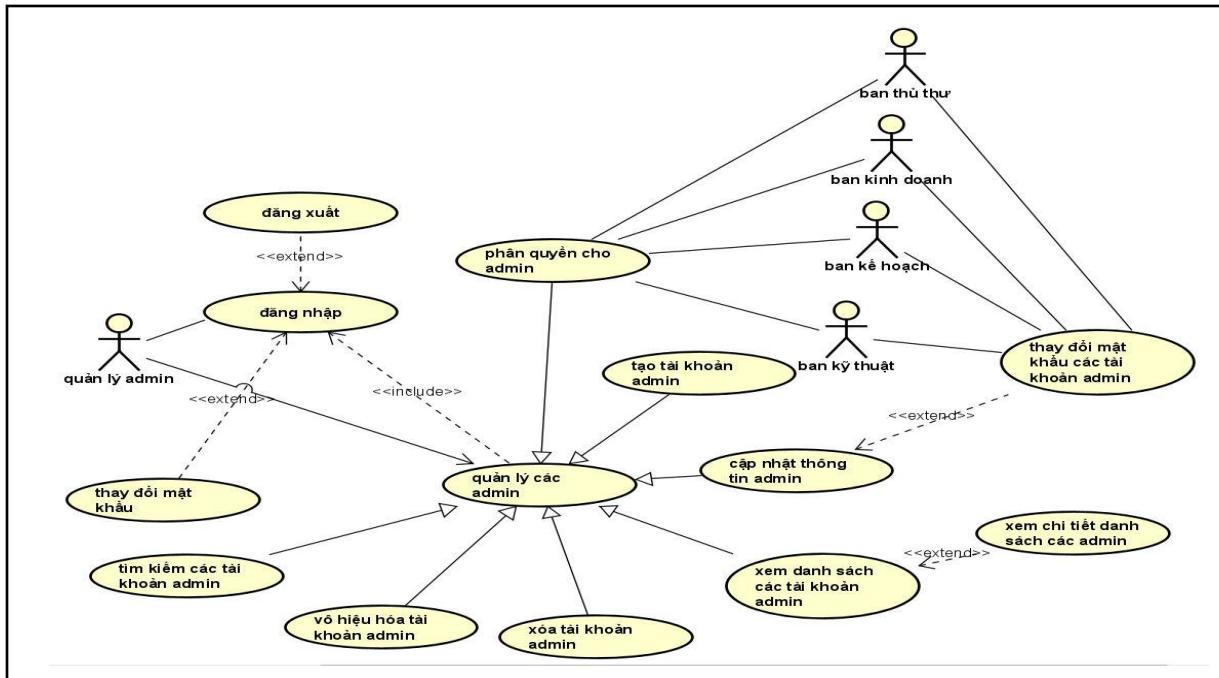
Hình 2.40. Usecase tổng quát

2.3.2. Quản lý sách



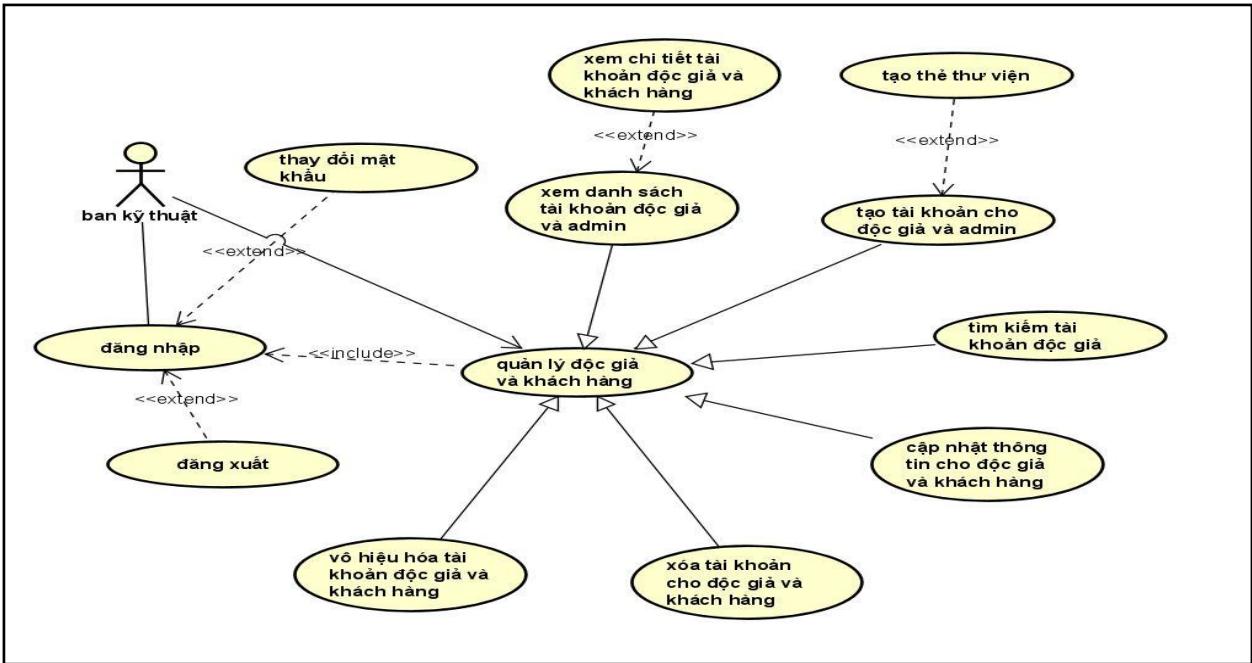
Hình 2.41. Usecase quản lý sách

2.3.3. Quản lý nhân viên



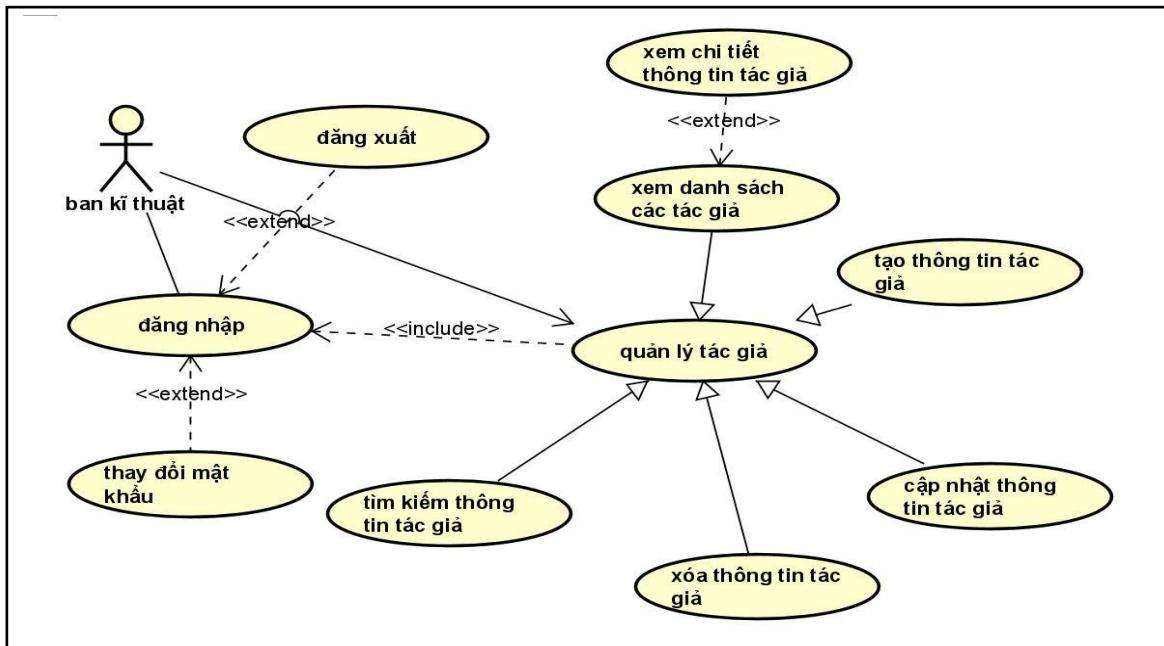
Hình 2.42. Usecase quản lý nhân viên

2.3.4. Quản lý đọc giả và khách hàng



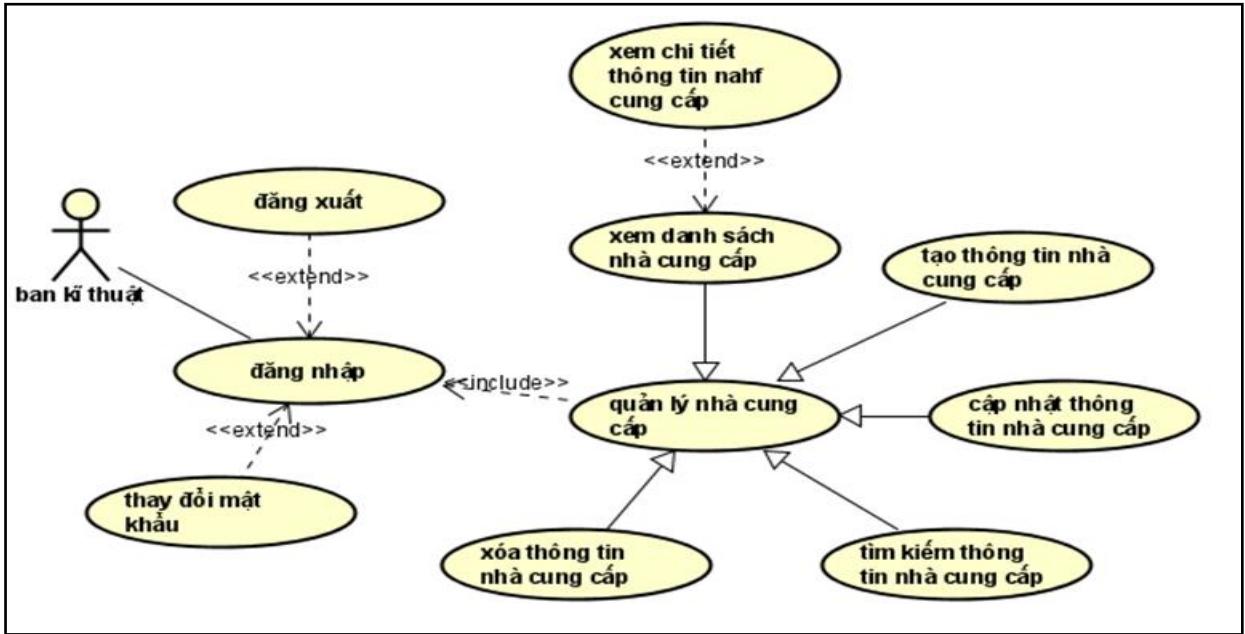
Hình 2.43. Usecase quản lý đọc giả và khách hàng

2.3.5. Quản lý tác giả



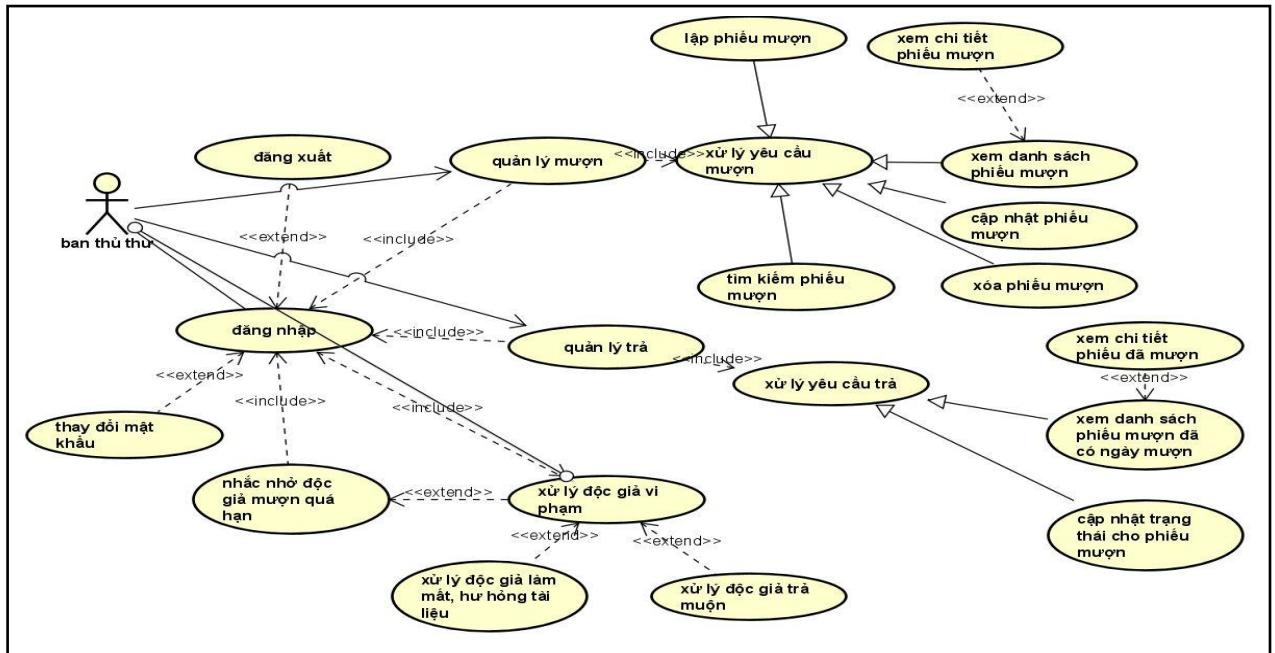
Hình 2.44. Usecase quản lý tác giả

2.3.6. Quản lý nhà cung cấp



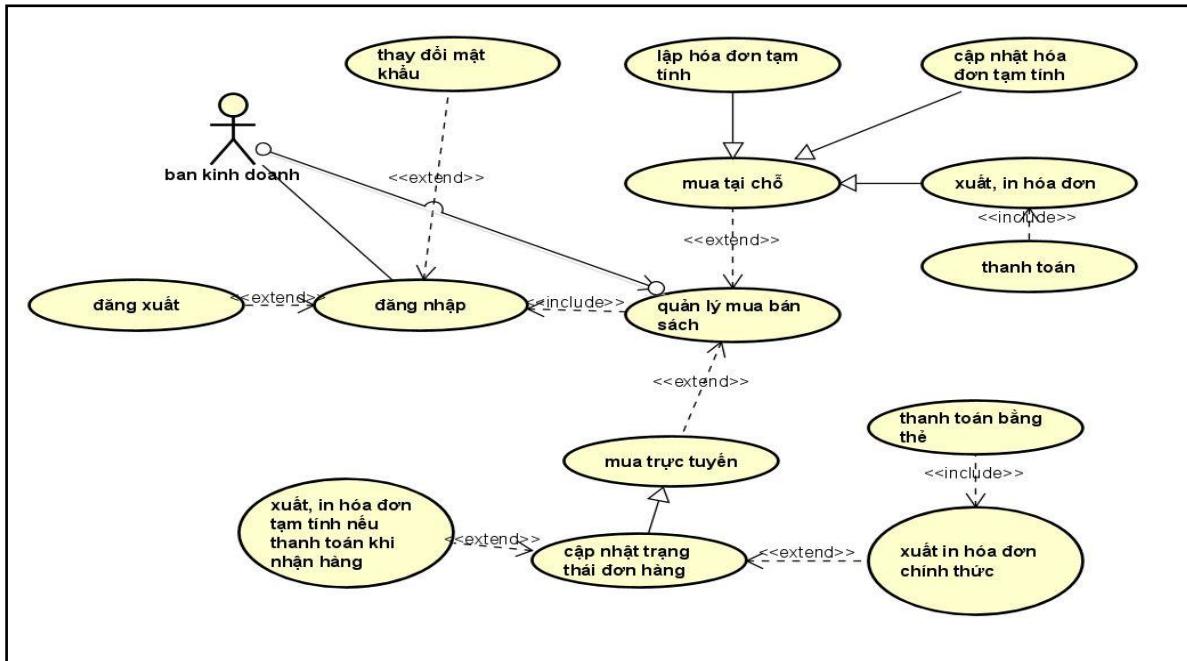
Hình 2.45. Usecase quản lý nhà cung cấp

2.3.7. Quản lý mượn trả



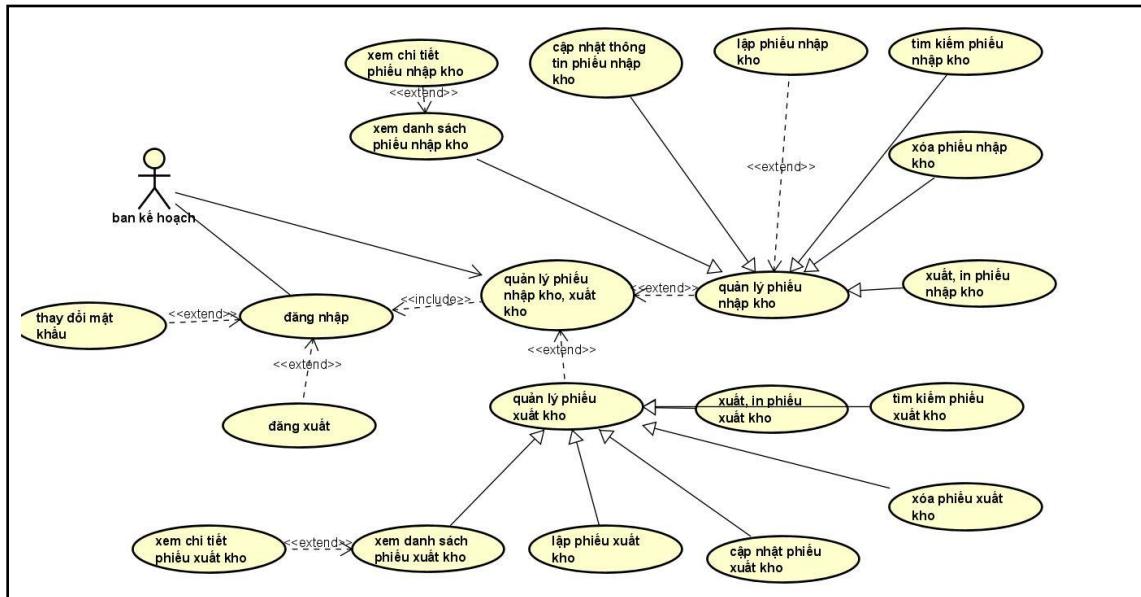
Hình 2.46. Usecase quản lý mượn trả

2.3.8. Quản lý mua bán sách



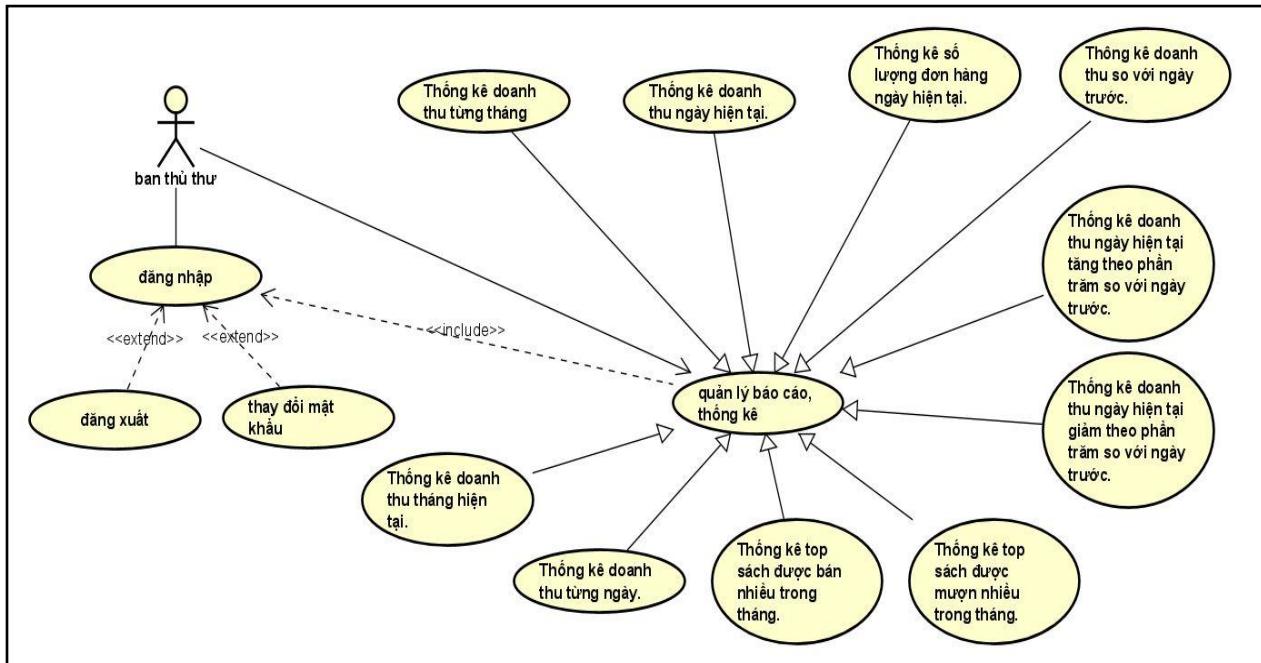
Hình 2.47. Use case quản lý mua bán sách

2.3.9. Quản lý phiếu nhập xuất kho



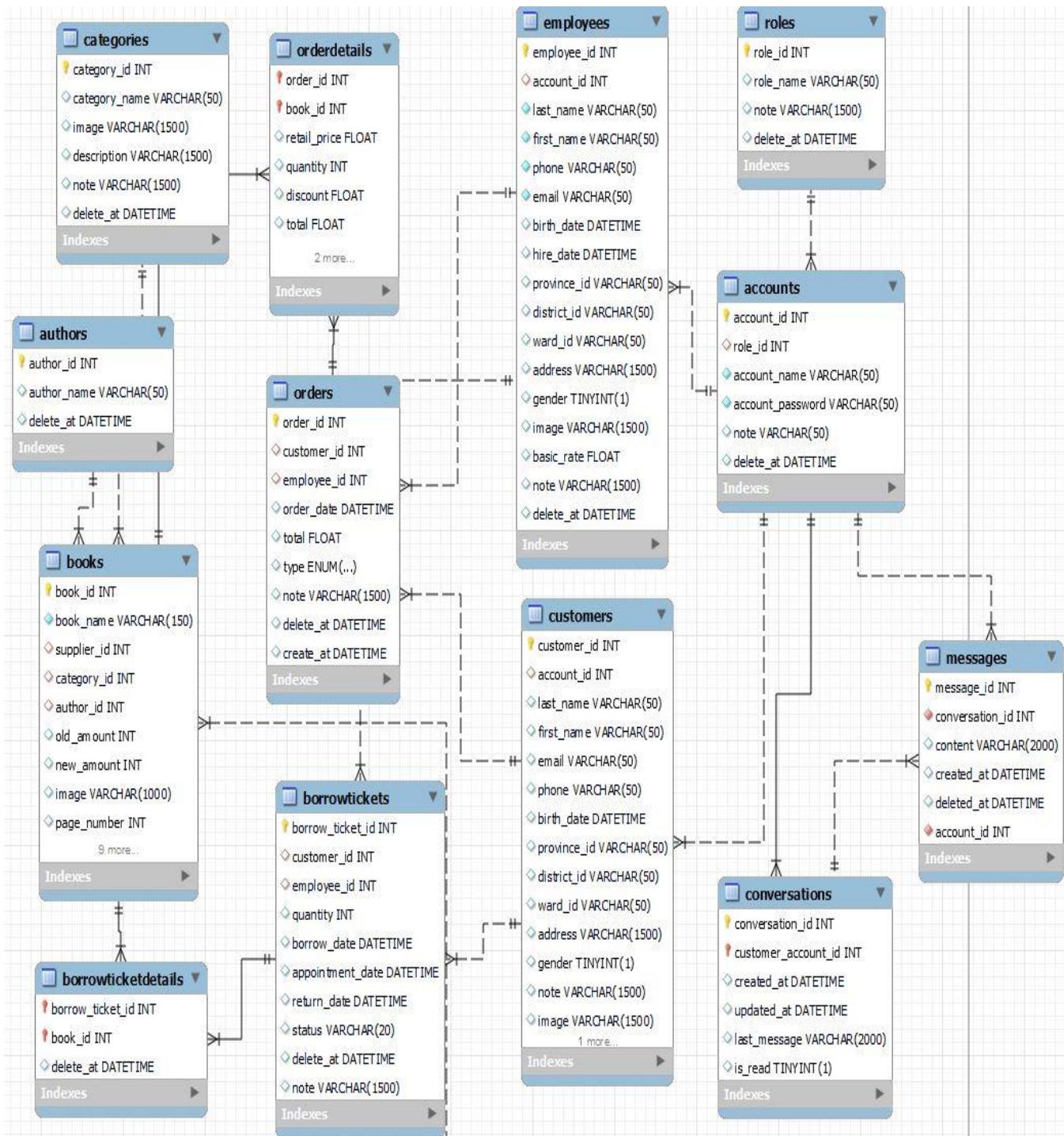
Hình 2.48. Use case quản lý phiếu xuất nhập kho

2.3.10. Báo cáo, thống kê



Hình 2.49. Usecase báo cáo thống kê

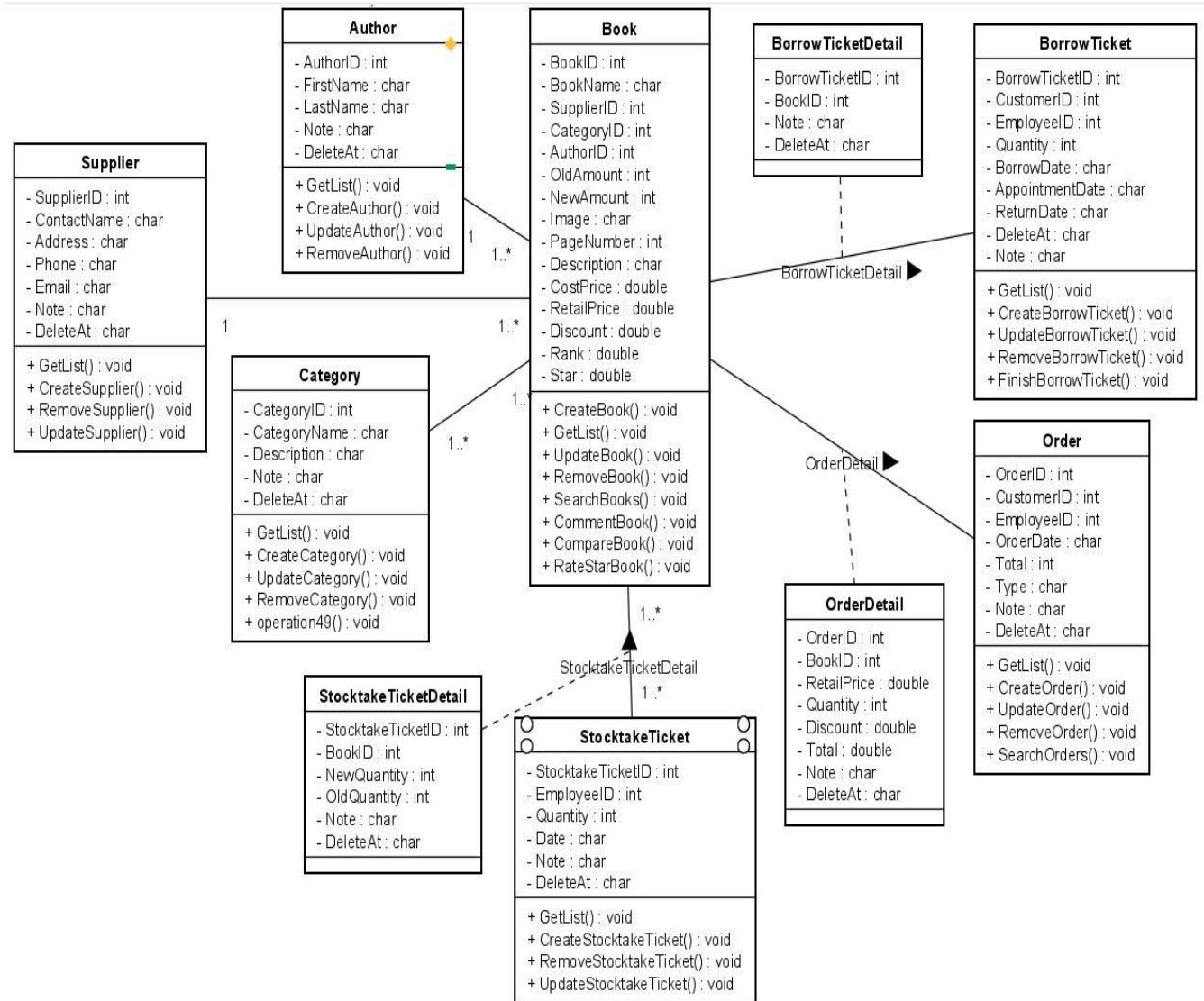
2.4 Cơ sở dữ liệu hệ thống



Hình 2.50. Biểu đồ cơ sở dữ liệu của hệ thống

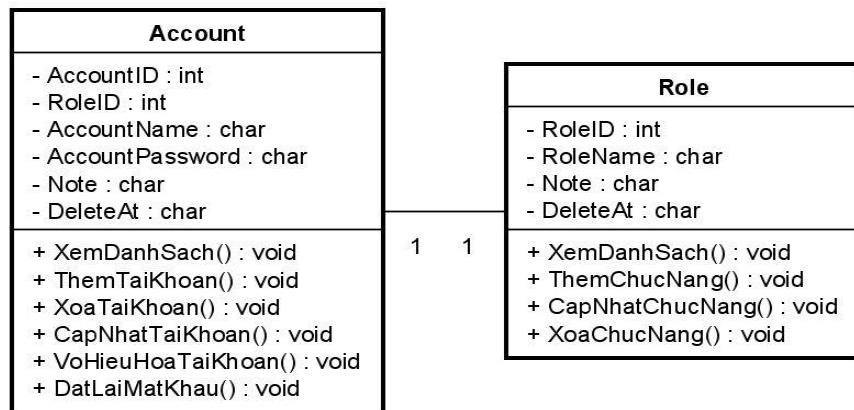
2.5 Biểu đồ lớp

2.5.1. Lớp sách



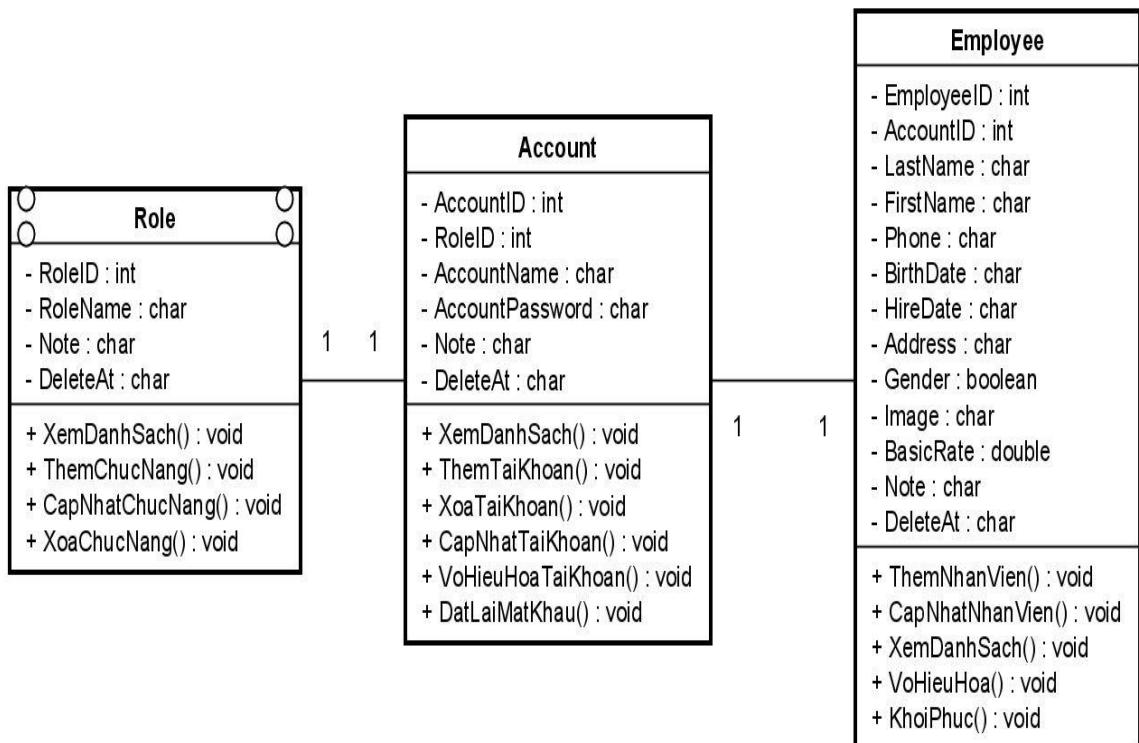
Hình 2.51. Biểu đồ lớp sách

2.5.2. Lớp tài khoản



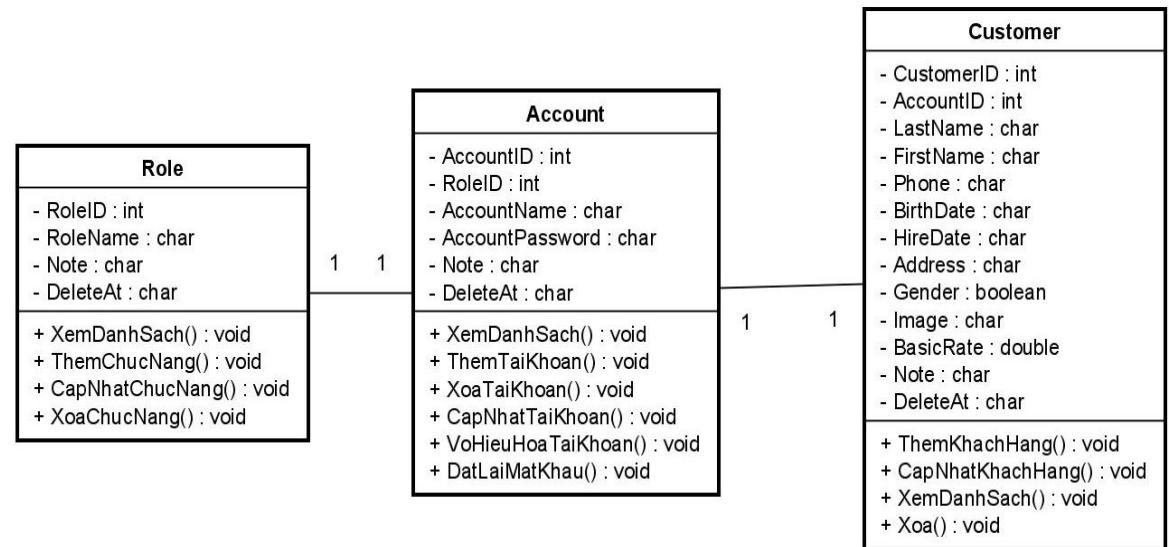
Hình 2.52. Biểu đồ lớp tài khoản

2.5.3. Lớp nhân viên và tài khoản nhân viên



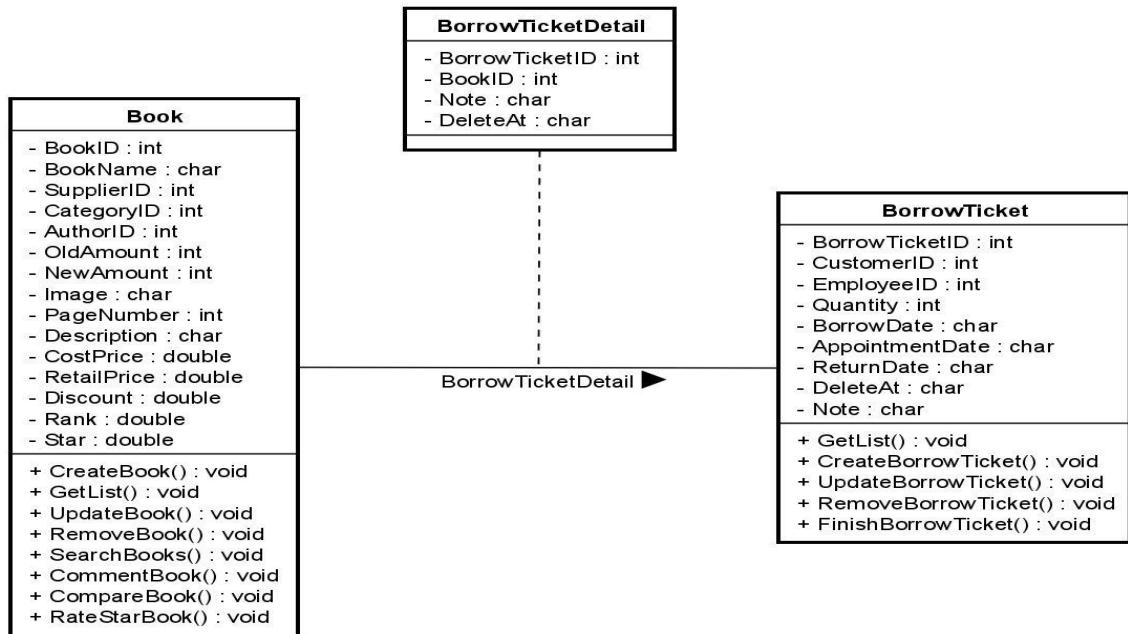
Hình 2.53. Biểu đồ lớp nhân viên và tài khoản nhân viên

2.5.4. Lớp khách hàng và tài khoản khách hàng



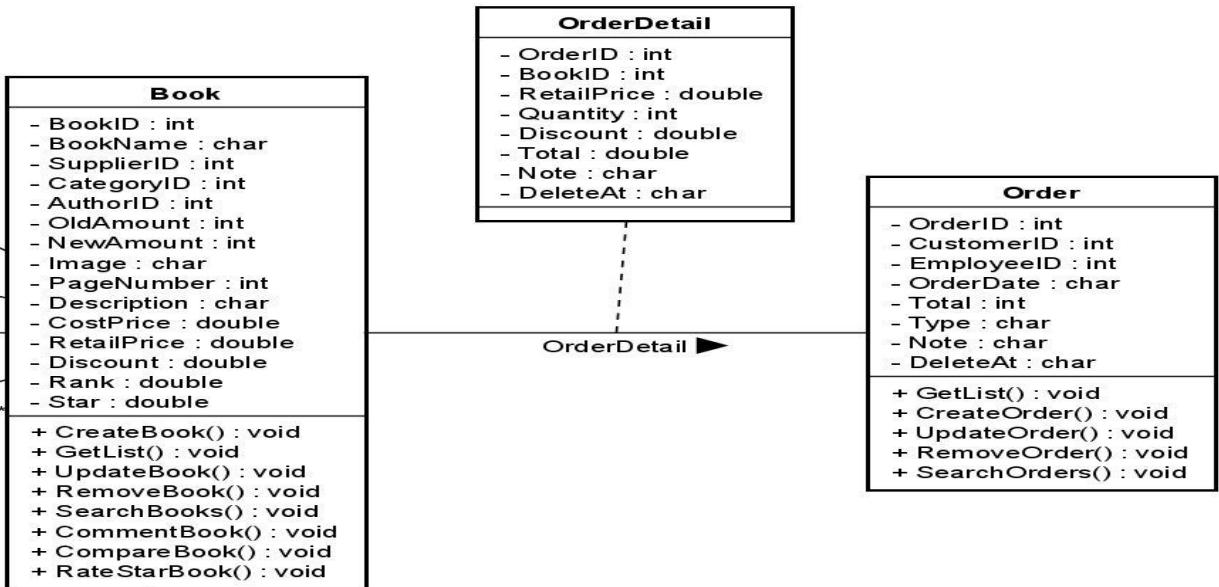
Hình 2.54. Biểu đồ lớp khách hàng và tài khoản khách hàng

2.5.5. Lớp phiếu mượn



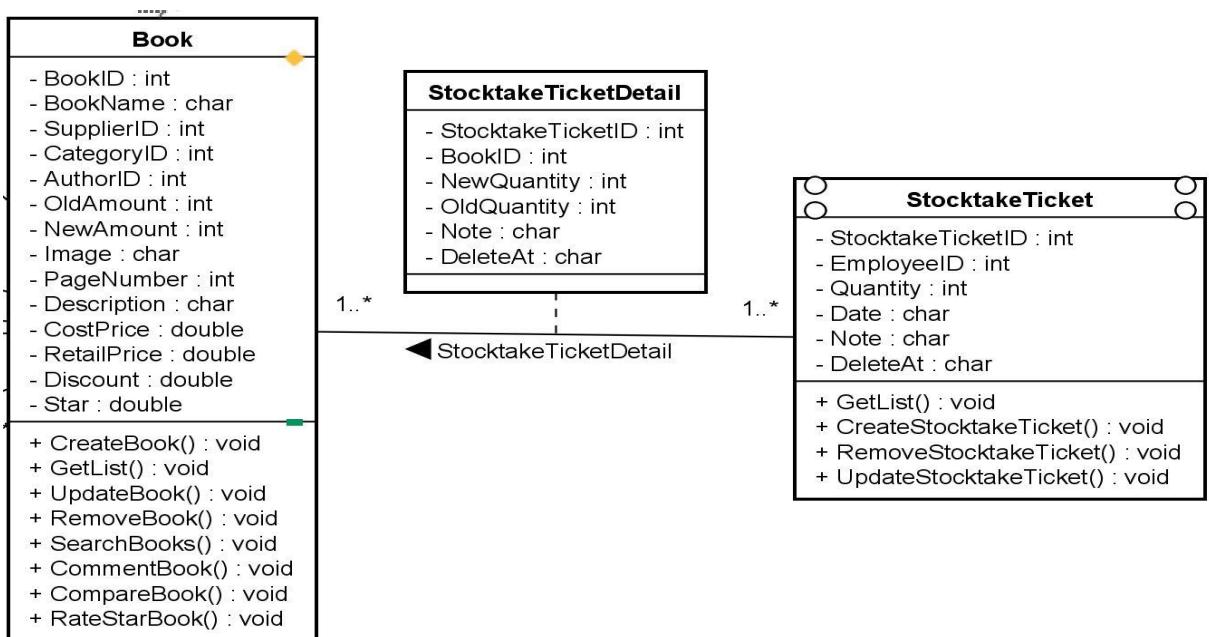
Hình 2.55. Biểu đồ lớp phiếu mượn

2.5.6. Lớp hóa đơn



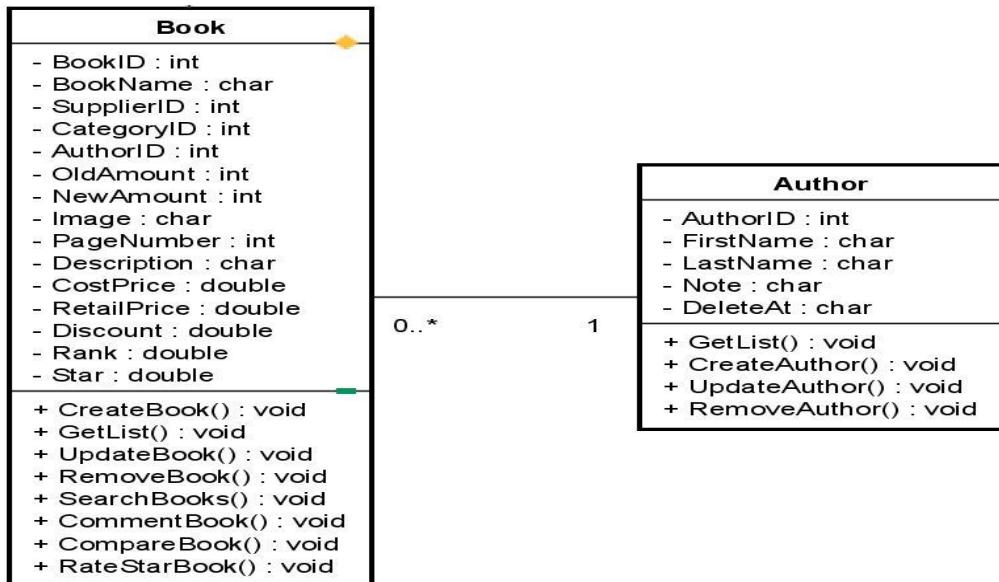
Hình 2.56. Biểu đồ lớp hóa đơn

2.5.7. Lớp phiếu xuất, nhập kho



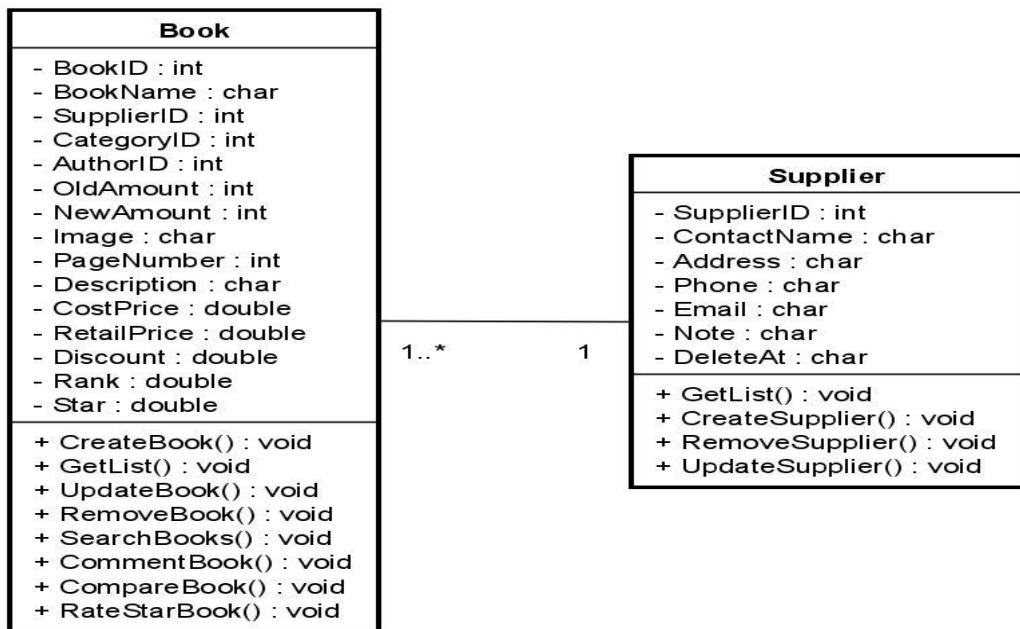
Hình 2.57. Biểu đồ lớp phiếu xuất, nhập kho

2.5.8. Lớp tác giả



Hình 2.58. Biểu đồ lớp tác giả

2.5.9. Lớp nhà cung cấp



Hình 2.59. Biểu đồ lớp nhà cung cấp

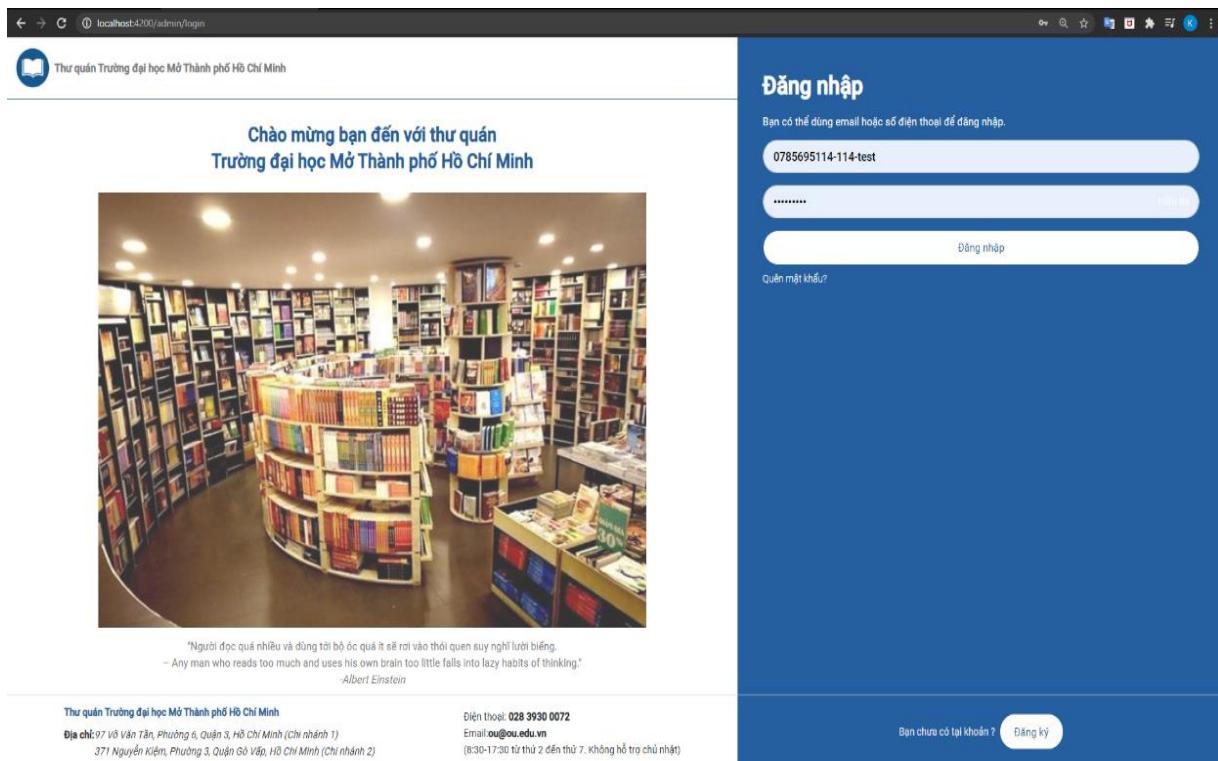
CHƯƠNG 3. PHÂN TÍCH, THIẾT KẾ GIAO DIỆN

Chương này giới thiệu về thiết kế giao diện của hệ thống dựa trên công nghệ Angular cũng như liệt kê nội dung, thông tin các API liên quan tại các trang do server (BE, python-flask) cung cấp.

3.1 Các giao diện chung

3.1.1. Trang đăng nhập cho nhân viên

- Thực hiện chức năng đăng nhập, xác thực, xác minh tài khoản của nhân viên.



Hình 3.1. Trang đăng nhập cho nhân viên

Các API sử dụng:

Endpoint: /admin/account-management/login

Method: POST

Request data:

`user_name: string`

`password: string`

Response data:

`access_token: string`

`user_info: object`

`current_account: object`

3.1.2. Trang đăng ký tài khoản dành cho khách hàng và tạo khách hàng

- Đăng ký tài khoản và tạo thông tin khách hàng.

The image shows two side-by-side screenshots. On the left is a screenshot of a library website. It features a logo of an open book, the text 'Thư quán Trường đại học Mở Thành phố Hồ Chí Minh', and a banner that reads 'Chào mừng bạn đến với thư quán Trường đại học Mở Thành phố Hồ Chí Minh'. Below the banner is a photograph of a well-lit library interior with many bookshelves and circular display units. A quote by Albert Einstein is displayed at the bottom: "'Người đọc quá nhiều và dùng túi bộ óc quá ít sẽ rơi vào thói quen suy nghĩ lười biếng.' - Any man who reads too much and uses his own brain too little falls into lazy habits of thinking." -Albert Einstein". On the right is a screenshot of a registration form titled 'Đăng ký'. The form asks for 'Vui lòng điền đầy đủ thông tin tài khoản.' and includes fields for 'testadmin@gmail.com' (email), 'Nhập họ' (last name), 'Nhập tên' (first name), 'Tỉnh thành (*)' (Province/City) set to 'Hồ Chí Minh', 'Quận huyện (*)' (District/Sub-district) set to 'Huyện Cần Giờ', 'Phường xã (*)' (Ward/Village) set to 'Xã An Thới Đông', 'Nhập địa chỉ' (Address), 'Nhập địa chỉ email(*)' (Email address), 'Nhập số điện thoại' (Phone number), and a date field 'mm/dd/yyyy'. A 'Đăng ký' (Register) button is at the bottom right.

Hình 3.2. Trang đăng ký tài khoản dành cho khách hàng và tạo KH

Các API sử dụng:

- ◆ **Endpoint:** `/admin/account-management/create-account`

Method: POST

Request data:

role_id: *string*
 account_name: *string*
 account_password: *string*
 confirm_account_password: *string*

Response data:

is_success: *boolean*
 message: *string*

Vai trò, mục đích: đăng ký tài khoản khách hàng.

- ◆ **Endpoint:** `/admin/account-management/create-customer`

Method: POST

Request data:

account_id: *string*
 last_name: *string*
 first_name: *string*
 email: *string*
 phone: *string*
 birth_date: *datetime*
 address: *string*
 gender: *boolean*

Response data:

is_success: *boolean*
 message: *string*

Vai trò, mục đích: đăng ký thành viên khách hàng.

3.1.3. Trang yêu cầu reset mật khẩu

- Khách hàng có thể thực hiện chức năng đặt lại mật khẩu thông qua email. Hệ thống sẽ gửi link đặt lại mật khẩu đến email kèm token dùng để đặt lại mật khẩu.

Quên mật khẩu ×

Chọn phương thức xác thực:

Xác thực bằng email

Vui lòng cung cấp địa chỉ **email** mà bạn đã dùng khi đăng ký tài khoản. Chúng tôi sẽ gửi bạn email cho phép bạn đặt lại mật khẩu.

Xác thực bằng số điện thoại

Hủy
Đặt lại mật khẩu

Hình 3.3. Trang quên mật khẩu

Các API sử dụng:

Endpoint: `/send-reset-password-email-customer`

Method: POST

Request data:

`customer_email: string`

Response data:

`is_success: Boolean`

`message: string`

Vai trò, mục đích: Yêu cầu đặt lại mật khẩu, hệ thống sẽ gửi email reset mật khẩu đến email đã nhập.

3.1.4. Trang đặt lại mật khẩu

- Khách hàng có thể đặt lại mật khẩu của mình, kèm theo đó là token được hệ thống gửi qua email. Từ token ấy, hệ thống có thể xác thực được tài khoản khách hàng cần đặt lại mật khẩu.



Hình 3.4. Trang đặt lại mật khẩu cho khách hàng, nhân viên

Các API sử dụng:

Endpoint: /send-reset-password-email-customer

Method: POST

Request data:

reset_token: string

new_password: string

Response data:

is_success: boolean

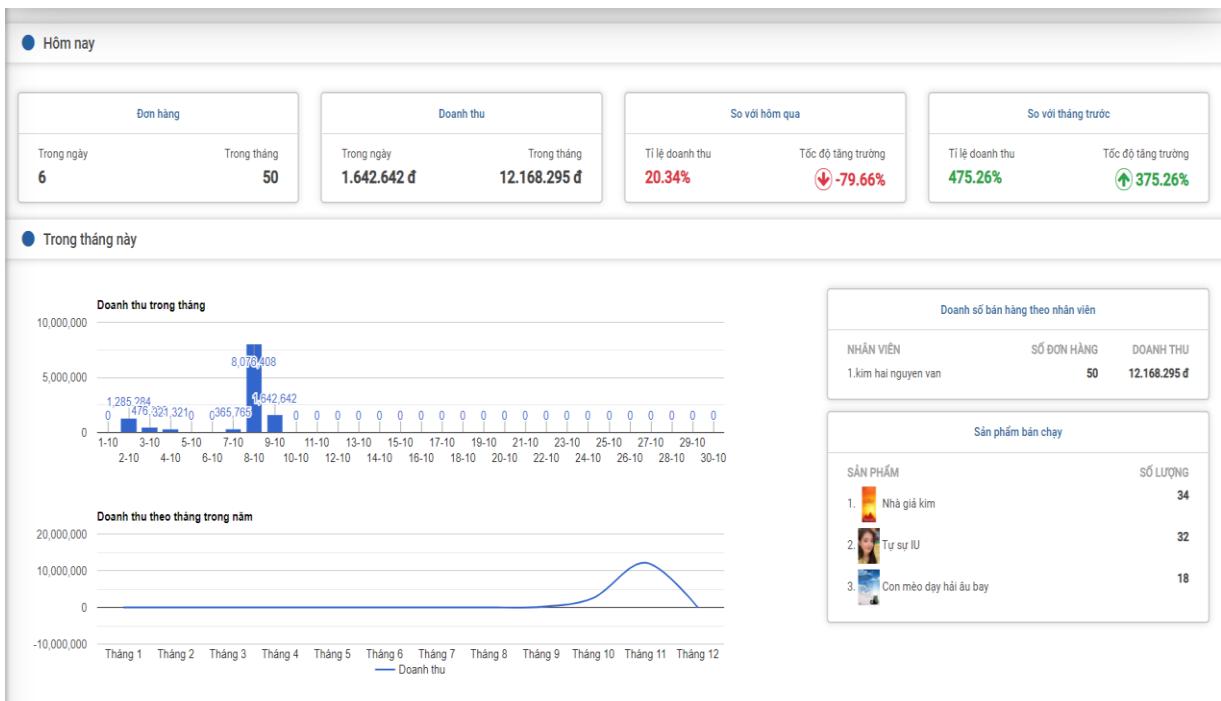
message: string

Vai trò, mục đích: được truy cập từ email do hệ thống gửi, có đính kèm token, dùng để đặt lại mật khẩu mới.

3.2 Các giao diện dành cho quản trị viên (Admin)

3.2.1. Trang thống kê

- Thống kê tình hình kinh doanh, doanh thu, số lượng đơn hàng ... của cửa hàng sách, hệ thống.



Hình 3.5. Trang thống kê

Các API sử dụng:

Endpoint: /admin/revenue-management

Method: GET

Request data:

Response data:

today_order_count: object

month_order_count: object

revenue_today: object

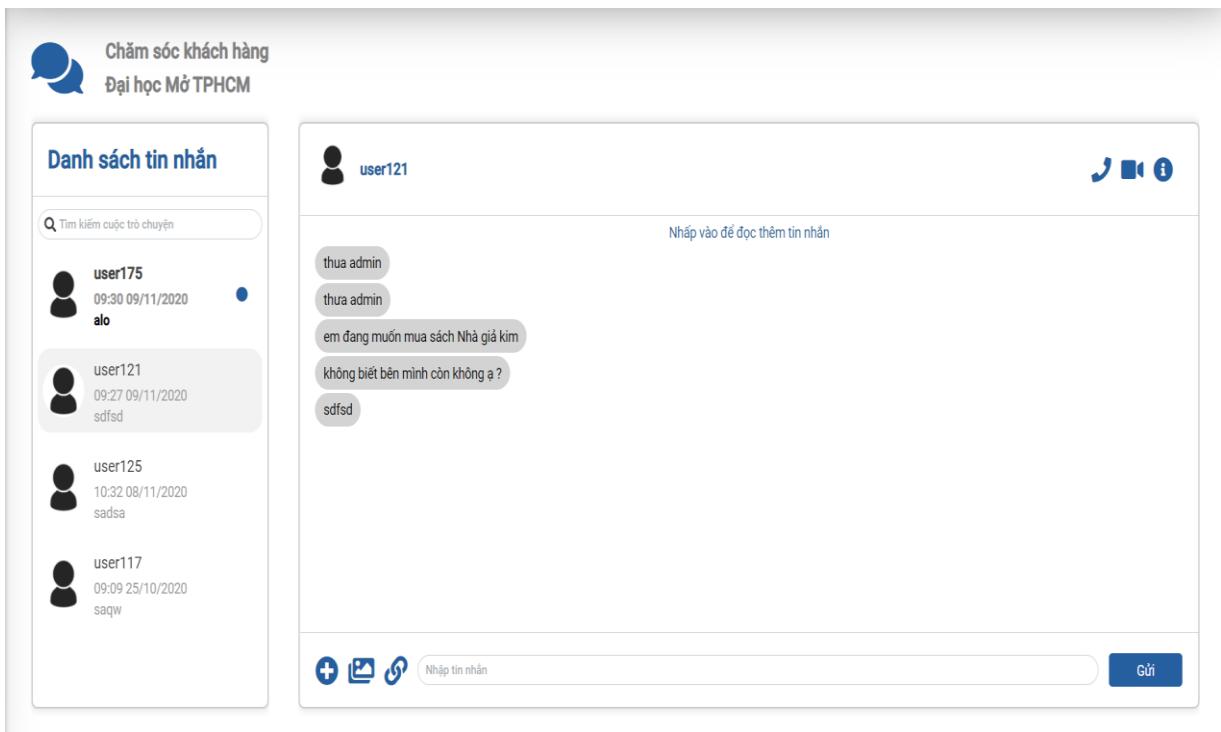
percentage_today_revenue_to_prev_day: object

percentage_month_revenue_to_prev_month: *object*
 grow_percentage_to_prev_day: *object*
 grow_percentage_to_prev_month: *object*
 revenues_each_day_in_month: *object*
 revenue_of_each_month_in_year: *object*
 month_total_revenue: *object*
 best_sellers_in_month: *object*
 most_favorite_books: *object*
 borrow_tickets_count: *object*

Vai trò, mục đích: thống kê doanh thu, số lượng, đơn hàng của hệ thống trong ngày, trong tháng.

3.2.2. Trang chăm sóc khách hàng

- Danh sách các cuộc tư vấn giữa nhân viên và khách hàng của hệ thống.



Hình 3.6. Trang chăm sóc KH

Các API sử dụng:

- ◆ **Endpoint:** `/message/get-all-conversations`

Method: GET

Request data: None

Response data:

all_conversations: *array*

Vai trò, mục đích: lấy về tất cả các đoạn hội thoại giữa hệ thống với tất cả tài khoản KH.

- ◆ **Endpoint:** `@socketio.on('incoming-msg')`

Request data:

conversation_id: *string*

content: *string*

account_id: *string*

Response data:

conversation_id: *string*

content: *string*

account_id: *string*

message_id: *string*

is_read: *boolean*

created_at: *datetime*

Vai trò, mục đích: dùng để gửi tin nhắn đến KH.

3.2.3. Trang thiết lập thông tin tài khoản

- Người dùng có thể thay đổi, cập nhật thông tin tài khoản, thông tin người dùng như ảnh đại diện, họ tên, ngày sinh....



**Thư viện
Đại học Mở TPHCM**

Thông tin cá nhân

Thông tin tài khoản

Thông tin người dùng

Thông tin tài khoản

Mã tài khoản:	2	
Tên tài khoản:	admin	
Email:	shinichi@gmail.com	Thay đổi
Số điện thoại:	0123456789	Thay đổi

Thay đổi mật khẩu

Hình 3.7. Trang thiết lập thông tin tài khoản



**Thư viện
Đại học Mở TPHCM**

Thông tin cá nhân

Thông tin tài khoản

Thông tin người dùng

Thông tin người dùng

Ảnh đại diện		
Mã nhân viên	#000001	
Tên người dùng	kim hai	nguyen van
Email:	shinichi@gmail.com	
Chứng minh thư:	0129321321	
Số điện thoại:	0123456789	
Địa chỉ:	12 abcc	
Ngày sinh:	Tue, 29 Sep 2020 00:00:00 GMT	

Cập nhật thông tin

Hình 3.8. Trang thiết lập thông tin người dùng

Thay đổi mật khẩu

Hủy
Đặt lại mật khẩu

Hình 3.9. Popup thay đổi mật khẩu tài khoản

Cập nhật thông tin cá nhân

Ảnh đại diện:

Upload ảnh

Họ:

Tên:

Ngày sinh:

Giới tính:

Địa chỉ:

Ghi chú:

Đóng
Cập nhật

Hình 3.10. Popup thay đổi thông tin nhân viên

Các API sử dụng:

- ◆ **Endpoint:** `/change-password`

Method: POST

Request data:

`current_password: string`

`new_password: string`

`account_id: string`

Response data:

`Is_success: boolean`

`message: string`

Vai trò, mục đích: thay đổi mật khẩu

- ◆ **Endpoint:** `/admin/employee-management/update-employee`

Method: POST

Request data:

`employee_id: string`

`last_name: string`

`first_name: string`

`phone: string`

`image_url: string`

`address: string`

Response data:

`Is_success: boolean`

`message: string`

Vai trò, mục đích: cập nhật thông tin nhân viên

3.2.4. Trang danh sách hóa đơn

- Quản lý hóa đơn với các chức năng như xem danh sách, tạo mới, xóa, cập nhật, tìm kiếm theo nhiều thông tin...

MÃ HÓA ĐƠN	MÃ ĐỌC GIẢ	TỔNG TIỀN	HÌNH THỨC	TỔNG SL	GHI CHÚ
#000018	#000001 ✎ Khách lè	150.000 ₫	Tại chỗ	1	-
#000017	#000002 ✎ Nguyễn Quang Trường	290.000 ₫	Trực tuyến	2	-
#000016	#000002 ✎ Nguyễn Quang Trường	422.000 ₫	Trực tuyến	2	-

Hình 3.11. Trang danh sách hóa đơn

Các API sử dụng:**Endpoint:** /admin/order-management/get-orders**Method:** GET**Request data:**

page: int

per_page: int

Response data:

has_next: boolean

has_prev: boolean

items: array

Vai trò, mục đích: lấy danh sách đơn hàng.

3.2.5. Trang tạo mới, thanh toán hóa đơn (POS)

- Nhân viên có thể thanh toán hóa đơn bằng tiền mặt hoặc bằng ứng dụng Momo cho khách hàng khi mua sách tại chỗ.

The screenshot displays a user interface for a point-of-sale (POS) system. On the left, a 'Giỏ hàng' (Shopping Cart) section shows two items: 'Thuật Đọc Tâm' (150,000đ) and 'Tư Duy Sâu' (66,000đ). In the center, a 'Sản phẩm' (Products) section lists various books with their titles, authors, prices, and descriptions. On the right, a 'Khách hàng' (Customer) section shows details for '2 - Nguyễn Quang Trường' (Mã khách hàng: 2, Số điện thoại: 0987456123, Họ và tên: Nguyễn Quang Trường). Below it, a 'Hóa đơn' (Invoice) section provides a breakdown of the total amount (216,000đ), quantity (2), discounts (0), taxes (0), and payment methods (Thanh toán hóa đơn bằng MOMO, activate Windows).

Sản Phẩm	Tác giả	Giá
LAGOM - Biết đủ mới là Thành công	Đỗ Anh	102.000đ
DÂN BÃ SAO KIM	Đỗ Anh	145.000đ
BÍ KÍP TƯ DUY SÂU	Tư Duy Sâu	66.000đ
THUẬT ĐỌC	John Richard Price	150.000đ
Dừng Chạy Theo Sô Động	John Richard Price	99.000đ
KHỞI NGHIỆP BẢN LẺ	Jack Ma	150.000đ
JACK MÃ	Jack Ma	60.000đ
OKRs		

Hình 3.12. Trang thanh toán hóa đơn

Các API sử dụng:**Endpoint:** /admin/order-management/create-order**Method:** POST**Request data:**

```

customer_id: string
employee_id: string
order_date: datetime
type: string
total: float
order_detail_list: array
  
```

Response data:

```

is_success: boolean
message: string
  
```

Vai trò, mục đích: tạo mới, lập đơn hàng.

3.2.6. Trang thanh toán, quét mã bằng ví điện tử Momo môi trường Test

- Trang scan dùng để thanh toán đơn hàng được trả về từ server Momo.



Hình 3.13. Trang thanh toán bằng Momo

Các API sử dụng:

Endpoint: /admin/order-management/create-order

Method: POST

Request data:

```
customer_id: string
order_date: datetime
type: string
total: float
order_detail_list: array
```

Response data:

```
url_link: string
Is_success: boolean
```

Vai trò, mục đích: thanh toán, tạo mới đơn hàng bằng Momo sau khi quét mã QR code.

3.2.7. Trang chi tiết hóa đơn

- Hiển thị thông tin chi tiết hóa đơn, đơn hàng của hệ thống.

Thông tin chi tiết hóa đơn						
	Mã hóa đơn: #000006					
Khách hàng:	#000003 - asdsda dsadsa					
Nhân viên:	#000001 - kim hai nguyen van					
Ngày mua:	16/10/2020					
Lô:	Nhập					
Tổng số lượng:	2					
Trạng thái:	Hoàn thành					
Ghi chú:	-					
Quay lại	Hủy bỏ					
Danh sách sách mượn						
STT	MÃ SÁCH	TÊN SÁCH	SỐ LƯỢNG	ĐƠN GIÁ	GIẢM GIÁ	THÀNH TIỀN
1	#000001	Nhà già kim	2	22.222 đ	0 đ	44.444 đ

Hình 3.14. Trang chi tiết hóa đơn

Hình 3.15. Trang thanh toán bằng Momo

Các API sử dụng:

Endpoint: /admin/order-management/get-order

Method: GET

Request data:

order_id: string

Response data:

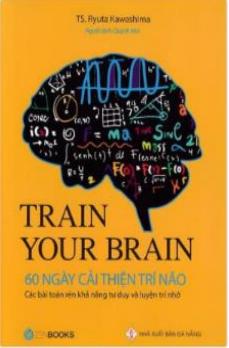
order: object

Vai trò, mục đích: lấy đơn hàng theo Mã đơn hàng

3.2.8. Trang tạo sách mới

- Thêm mới, tạo mới sách cũng như nhà cung cấp, tác giả cho hệ thống, cửa hàng.

Đăng ký sách mới



Tải ảnh bìa

Tên sách:	Train your brain
Tác giả:	Tổ Hữu
Nhà cung cấp:	Nhà sách Phương nam - 0932147831 - aaaa@gmail.com
Thể loại:	Tự truyện
Số trang:	121
Giá vốn:	100000
Giá bán:	250000
Số lượng sách bán:	150
Số lượng sách mượn:	100
Giảm giá (%):	Nhập giá trị giảm giá
Vui lòng đầy đủ, dương như là bao..	
Không thể nhớ được tên của ai đó.	
Không thể nhớ được bạn để vắt dụng ở đâu.	
Không thể tìm được từ thích hợp để diễn tả một ý nào đó.	
Nếu đây đúng là vấn đề mà bạn gặp phải khi làm bài kiểm tra quyền sách của Bác sĩ Ryuta Kawashima sẽ hoàn toàn phù hợp với bạn. Mỗi ngày chỉ cần đơn giản hoàn thành một bài tập toán để tăng và bạn sẽ từng bước trả hòa bộ não của mình. Sứ kết hợp giữa các bài kiểm tra hàng tuần và nhật ký cá nhân cho phép bạn theo dõi sự tiến bộ của mình trong quá trình rèn luyện. Hãy dành cho chúng tôi 60 ngày thời gian để chúng tôi sẽ trả lại cho bạn một bộ não mạnh khỏe hơn và làm việc tốt hơn!	
Ghi chú:	
Nhập ghi chú	

[← Quay lại](#)
Hủy
Tạo mới

Hình 3.16. Trang tạo sách mới

Thêm nhà cung cấp

Tên nhà cung cấp:	Nhập tên nhà cung cấp
Địa chỉ:	Nhập địa chỉ
Số điện thoại:	Nhập số điện thoại
Email:	Nhập email
Ghi chú:	Nhập ghi chú

Đóng
Xác nhận

Hình 3.17. Popup tạo mới nhà cung cấp

Các API sử dụng:

- ◆ **Endpoint:** /admin/book-management/create-book

Method: POST

Request data:

book_name: *string*
 supplier_id: *string*
 category_id: *string*
 author_id: *string*
 old_amount: *int*
 new_amount: *int*
 image_url: *string*
 page_number: *int*
 description: *string*
 cost_price: *float*
 retail_price: *float*
 discount: *float*
 ranking: *float*

Response data:

Is_success: *boolean*
 message: *string*

Vai trò, mục đích: tạo mới, thêm mới sách.

- ◆ **Endpoint:** /admin/book-management/upload-book-image

Method: POST

Request data:

Image_file: file

Response data:

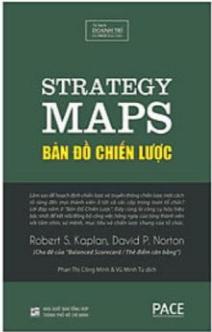
Image_url: *string*

Vai trò, mục đích: upload ảnh lên server Cloudinary để lưu trữ hình ảnh, trả về đường link url của ảnh

3.2.9. Trang chi tiết sách

- Hiển thị thông tin chi tiết sách của hệ thống

Thông tin chi tiết sách Vòng Lặp Ooda - Phương Pháp Ra Quyết Định Và Hành Động Tối Ưu Trong Công Việc

 Mã sách: #000012	Tên sách: Vòng Lặp Ooda - Phương Pháp Ra Quyết Định Và Hành Động Tối Ưu Trong Công Việc
Tác giả: Lione Ronaodal	Nhà cung cấp: NXB Nhà Bè
Thể loại: Sách tư duy - Kỹ năng sống	Số trang: 250
Giá vốn: 250.000 ₫	Giá bán: 280000
Giảm giá: 0	Tồn kho sách bán : 50
Tồn kho sách mượn: 50	Tóm tắt: Bạn chậm trễ trong xử lý sự cố máy chiếu và phá hỏng buổi thuyết trình. Bạn chần chờ khi giải quyết khiếu nại nên bị mất đi một khách hàng. Bạn loay hoay lựa chọn và không kịp gọi món ăn ngon nhất trong nhà hàng. Và còn rất nhiều tình huống thực tế khác khi chúng ta thất bại trong phản ứng và ra quyết định nhanh chóng. Ở kỷ nguyên công nghệ và thời đại mà thông tin và tốc độ đang nắm quyền quyết định. Trong hầu hết trường hợp thì nhanh hơn không đồng nghĩa
 Mã sách: #000012	

Hình 3.18. Trang chi tiết sách

Các API sử dụng:

Endpoint: /admin/book-management/get-book

Method: GET

Request data:

book_id: string

Response data:

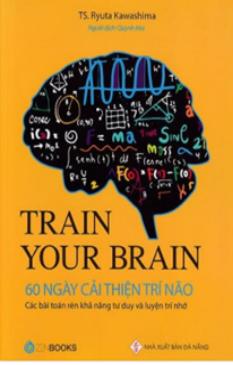
book: object

Vai trò, mục đích: lấy thông tin theo Mã sách

3.2.10.Trang cập nhật sách

- Nhân viên có thể cập nhật các thông tin của sách như tên sách, tác giả, giá bán....

Thông tin chi tiết sách Train your brain

 Hình ảnh thực tế từ khách hàng     Mã sách:	Tên sách: Train your brain Tác giả: Tố Hữu Nhà cung cấp: Nhà sách Phương nam - 0932147831 - aaaa@gmail.com Thể loại: Tự truyện Số trang: 121 Giá vốn: 100000 Giá bán: 250000 Giảm giá (%): 0 Tóm tắt: <small>Vừa mới đây, dường như là bạn... Không thể nhớ được tên của ai đó; Không thể nhớ được bạn để vật dụng ở đâu; Không thể tìm được từ thích hợp để diễn tả một ý nào đó. Nếu đây đúng là vấn đề mà bạn gặp phải khi lớn tuổi hơn thì quyển sách của Bác sĩ Ryuta Kawashima sẽ hoàn toàn phù hợp với bạn. Mỗi ngày, chỉ cần</small> Ghi chú: <input type="text" value="Nhập ghi chú"/>
← Quay lại	Hủy Xác nhận

Hình 3.19. Trang cập nhật thông tin sách

Các API sử dụng:

Endpoint: /admin/book-management/update-book

Method: POST

Request data:

book_name: string

supplier_id: string

category_id: string

author_id: string

old_amount: int

new_amount: *int*
image: *string*
page_number: *int*
description: *string*
cost_price: *float*
retail_price: *float*
discount: *float*
ranking: *float*

Response data:

Is_success: *boolean*
message: *string*
book: *object*

Vai trò, mục đích: cập nhật thông tin sách

3.2.11.Trang tạo mới phiếu mượn

- Nhân viên tạo mới phiếu mượn sách cho khách hàng khi khách hàng có nhu cầu mượn sách tại nhà sách.

Phiếu mượn sách

Mã đọc giả:	17
	Mã khách hàng: #000017 Họ và tên: asdas <input type="button" value="SACKZ"/> Số điện thoại: 0321458796 Email: asdas@gmail.com Chứng minh thư: 0258963144
	<input type="button" value="Xóa đọc giả"/>
Mã sách	4

Thông tin chi tiết sách

#000004		Train your brain	Tự truyện	Tố Hữu	<input type="button" value="Thêm sách"/>
---------	---	------------------	-----------	--------	--

Danh sách sách mượn

1		Nhà giả kim	sadsa	dsadsa	123	<input type="button" value="X"/>
---	---	-------------	-------	--------	-----	----------------------------------

Số lượng sách mượn: 1

Hình 3.20. Trang tạo mới phiếu mượn

Các API sử dụng:

Endpoint: /admin/borrow-ticket-management/create-borrow-ticket

Method: POST

Request data:

```
customer_id: string
employee_id: string
borrow_book_ids: array
```

Response data:

```
Is_success: boolean
message: string
borrow_ticket: object
```

Vai trò, mục đích: dùng để tạo mới phiếu mượn sách cho khách hàng.

3.2.12.Trang chi tiết phiếu mượn

- Hiển thị thông tin chi tiết phiếu mượn sách cũng như các chức năng đi kèm pop-up Gửi email nhắc nhở khách hàng của hệ thống.

Hình 3.21. Trang chi tiết phiếu mượn

Hình 3.22. Popup gửi email nhắc nhở khách hàng

Các API sử dụng:

- ◆ **Endpoint:** `/admin/borrow-ticket-management/get-borrow-ticket`

Method: GET**Request data:**borrow_ticket_id: *string***Response data:**Is_success: *boolean*message: *string*borrow_ticket: *object*

Vai trò, mục đích: lấy thông tin chi tiết phiếu mượn sách theo Mã phiếu mượn.

- ◆ **Endpoint:**

`/admin/borrow-ticket-management/send-email-for-late-borrow-ticket`**Request data:**message: *string*customer_email: *string***Response data:**Is_success: *boolean*message: *string*

Vai trò, mục đích: gửi email nhắc nhở khi phiếu mượn sách trễ hẹn.

- ◆ **Endpoint:** `/admin/borrow-ticket-management/finish-borrow-ticket`

Method: POST**Request data:**borrow_ticket_id: *string***Response data:**

Is_success: boolean

message: string

borrow_ticket: object

Vai trò, mục đích: trả sách, hoàn thành phiếu mượn.

3.2.13.Trang danh sách tài khoản hệ thống

- Quản lý danh sách tài khoản của hệ thống cùng với các chức năng thêm xóa, sửa, tìm kiếm tài khoản...

Nhập mã tài khoản	Nhập tên tài khoản	Chọn vai trò	Tìm kiếm
MÃ TÀI KHOẢN	TÊN TÀI KHOẢN	VAI TRÒ	GHI CHÚ
#000008	PPZORBBRXK	Khách hàng	-
#000007	quangtruong	Khách hàng	-
#000006	thanhphong	Quản lý	-
#000005	kimhai	Quản lý	-

Hình 3.23. Trang danh sách tài khoản hệ thống

Các API sử dụng:

Endpoint: /admin/account-management/get-accounts

Method: POST

Request data:

page: int

`per_page: int`

Response data:

`has_next: boolean`

`has_prev: boolean`

`items: array`

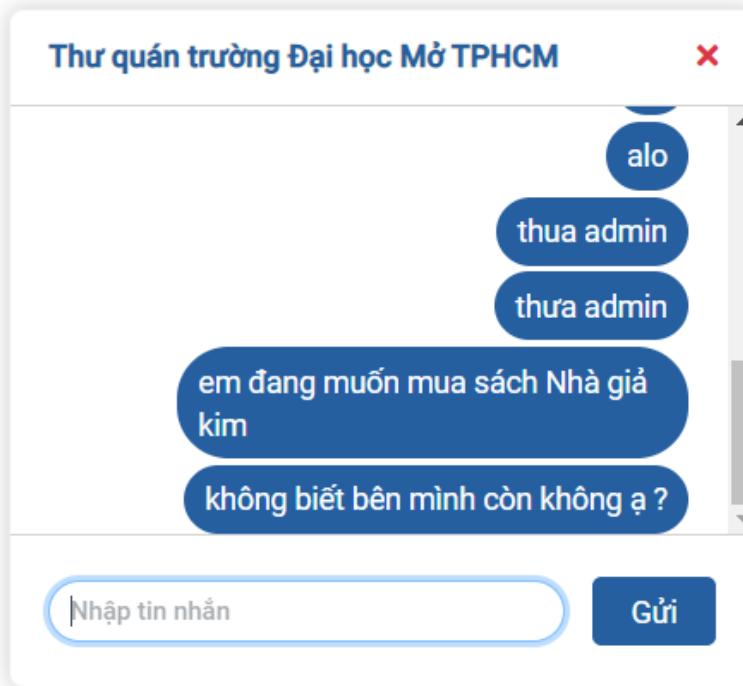
`current_page: int`

Vai trò, mục đích: lấy danh sách tài khoản của hệ thống, có phân trang

3.3 Các giao diện về phía người dùng (User)

3.3.1. Khung chat yêu cầu hỗ trợ, tư vấn của khách hàng

- Khi khách hàng đăng nhập tài khoản, khách hàng có thể sử dụng chức năng tư vấn khách hàng thông qua chat box.



Hình 3.24. Khung chat khách hàng

Các API sử dụng:

- ◆ **Endpoint:** `/message/get-conversation-by-customer-account-id`

Method: GET

Request data:

Customer_account_id: *string*

Response data:

conversation: *object*

Vai trò, mục đích: lấy đoạn tin nhắn, hội thoại của KH với nhân viên theo Mã tài khoản của khách hàng

- ◆ **Endpoint:** `/message/get-messages`

Request data:

Page: *int*

per_page: *int*

conversation_id: *string*

Response data:

has_next: *boolean*

has_prev: *boolean*

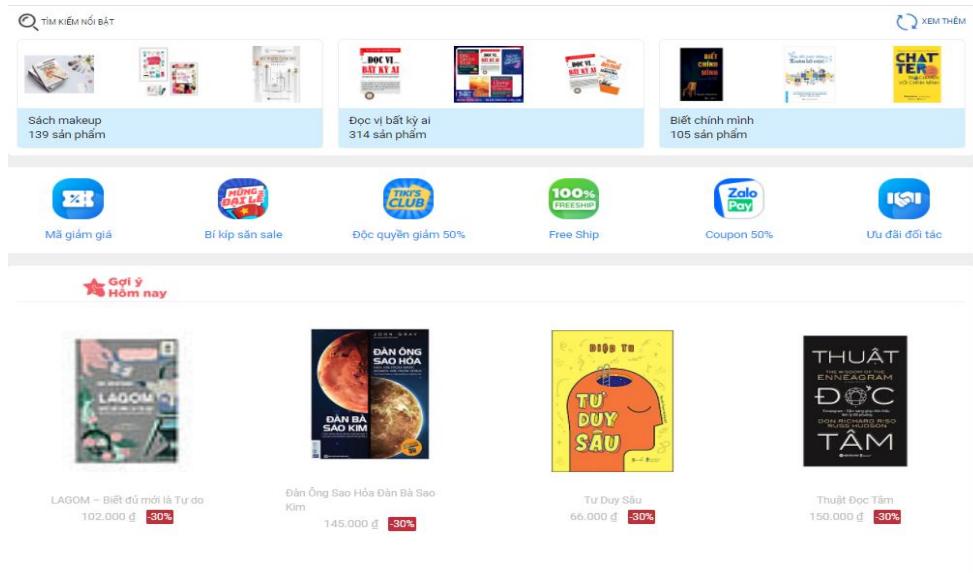
items: *array*

current_page: *int*

Vai trò, mục đích: lấy thêm tin nhắn theo Mã cuộc trò chuyện.

3.3.2. Trang chủ nhà sách

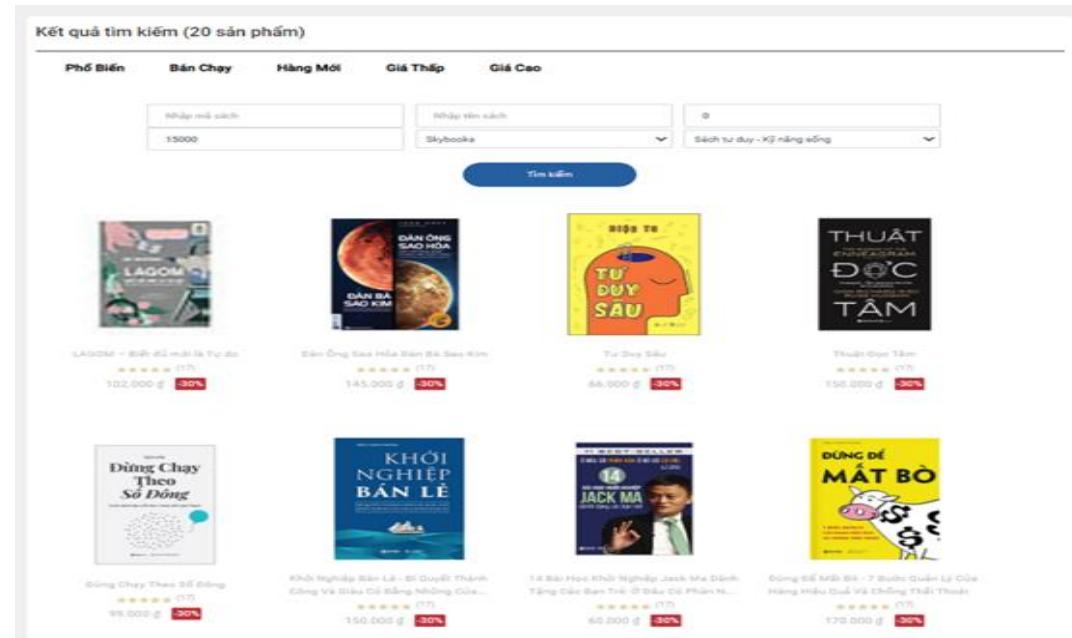
- Trang chủ của nhà sách với các chức năng cơ bản như tìm kiếm sách, xem chi tiết sách.....



Hình 3.25. Trang chủ nhà sách

3.3.3. Trang tìm kiếm sản phẩm

- Trang tìm kiếm sách theo nhiều tiêu chí như giá cả, tên sách, nhà cung cấp, thể loại.....



Hình 3.26. Trang tìm kiếm sản phẩm

Các API sử dụng:

- ◆ **Endpoint:** `/admin/book-management/search-books`

Method: POST

Request data:

Category_id: string

Book_name: string

Book_id: string

Supplier_id: string

Response data:

Books: array

Vai trò, mục đích: Lấy, tìm kiếm thông tin các sách dựa trên các tiêu chí tìm kiếm như tên sách, thể loại sách, giá cả.....

3.3.4. Trang thanh toán

- Trang thanh toán đơn hàng của khách hàng. Khách hàng có thể thanh toán trực tiếp tiền mặt khi giao hàng hoặc có thể trả trước bằng tài khoản Paypal của mình.

The screenshot shows a payment page with the following details:

- Giỏ hàng (1 sản phẩm):**
 - Tư Duy Sâu (Tái Bản 2020)** - GIAO NHANH 2H
 - Tác giả: Nguyễn Thành Hương
 - Sách có thể bọc bằng BookCare
 - Xóa Đã đánh mua sau
 - Dàn Ông Sao Hỏa Dàn Bà Sao Kim (Tái Bản 2020)** - GIAO NHANH 2H
 - Tác giả: Niki Brantmark
 - Sách có thể bọc bằng BookCare
 - Xóa Đã đánh mua sau
- Thông tin giao hàng:**
 - Địa chỉ nhận hàng:**
 - Tên người nhận
 - Số điện thoại người nhận
 - Địa chỉ người nhận
 - Tỉnh thành (*)
 - Quận huyện (*)
 - Phường xã (*)
 - Khuyến Mãi:** Chọn hoặc nhập Khuyến mãi
 - Tạm tính:** 277.000 đ
 - Thành tiền:** 277.000 đ (Đã bao gồm VAT nếu có)
- Thanh toán:**
 - Thanh toán hóa đơn
 - PayPal** Skip login on this device

Hình 3.27. Trang thanh toán đơn hàng

Các API sử dụng:

- ◆ **Endpoint:** */admin/book-management/create-order*

Method: POST

Request data:

```
customer_id: string
employee_id: string
order_date: datetime
type: string
total: float
order_detail_list: array
```

Response data:

```
Is_success: boolean
message: string
```

Vai trò, mục đích: tạo mới, lập đơn hàng.

3.3.5. Trang chi tiết sản phẩm

- Trình bày các chi tiết của sách như tên sách, giá cả... Ngoài ra khách hàng còn có thể bình luận, đánh giá chất lượng sản phẩm cũng như so sánh với sách tương tự ở các hệ thống nhà sách khác.

Thương hiệu: Blueup

Đứng thứ 2 trong [Top 1000 sản phẩm được yêu thích nhất](#)

Đàn Ông Sao Hỏa Đàn Bà Sao Kim

(Xem 3 đánh giá)

[Đánh giá](#) [So sánh](#)

145.000 đ +88.500đ -30%

[RẺ HƠN HOÀN TIỀN](#)

Hoàn tiền 15% tối đa 600k/tháng

1 Mã Giảm Giá

Giảm 10%

Bạn hãy NHẬP ĐỊA CHỈ nhận hàng để được dự báo thời gian & chi phí giao hàng một cách chính xác nhất.

Số Lượng:

- +

[Chọn mua](#)

Hình 3.28. Trang chi tiết sách

Các API sử dụng:

- ◆ **Endpoint:** /admin/book-management/get-book-by-id
- Method:** GET

Request data:

Book_id: string

Response data:

Books: object

Vai trò, mục đích: Lấy thông tin chi tiết của sách từ mã sách

- ◆ **Endpoint:** /admin/book-management/rate-star
- Method:** POST

Request data:

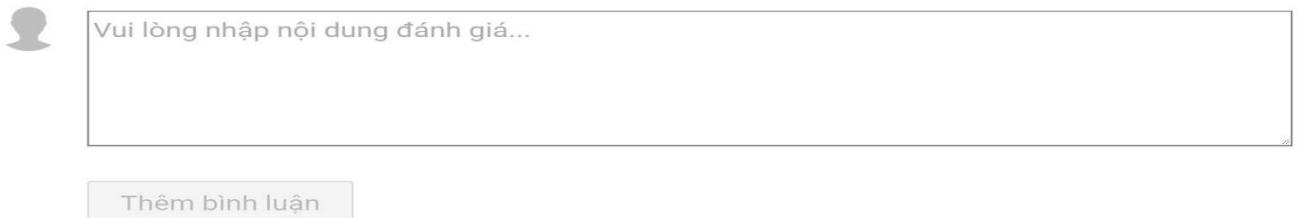
Book_id: string

Star: Float

Response data:

Book: object

Vai trò, mục đích: Đánh giá chất lượng sản phẩm, sách.



Hình 3.29. Danh sách bình luận của sách

Các API sử dụng:

- ◆ **Endpoint:** /admin/book-management/get-comments

Method: GET

Request data:

Book_id: string

Response data:

Comments: array

Vai trò, mục đích: Lấy danh sách các bình luận của người dùng về sách.

- ◆ **Endpoint:** /admin/book-management/create-comment

Method: POST

Request data:

Book_id: string

Content: string

Response data:

Is_success: boolean

Vai trò, mục đích: Tạo, thêm mới bình luận về sách.

So sánh sản phẩm		
Thương hiệu	OU Store	Tiki
Giá bán	145.000 ₫	227.000 ₫
Đánh giá	★★★★★	★★★★★
Số lượt đánh giá	3 đánh giá	2 đánh giá
Số lượt bình luận	18 bình luận	2 bình luận

Đóng

Hình 3.30. Modal so sánh sản phẩm với sản phẩm tương tự tại các trang khác

Các API sử dụng:

- ♦ **Endpoint:** /admin/book-management/compare-book

Method: POST**Request data:**

Book_id: string

Response data:

Books: array

Vai trò, mục đích: Lấy thông tin chi tiết của sản phẩm từ các trang thương mại điện tử khác (Tiki, Shopee)

CHƯƠNG 4. TỔNG KẾT VÀ HƯỚNG PHÁT TRIỂN

Chương này tổng hợp kết quả có được sau khi ta đã hoàn thiện đồ án, ngoài ra cũng nêu ra các nhược điểm vẫn đang còn tồn tại trong dự án, để từ đó, ta có thể đưa ra những giải pháp, những hướng phát triển để có thể hoàn thiện dự án hơn trong tương lai.

4.1 Kết quả đạt được

Sau khi quan sát tìm hiểu, nghiên cứu, khảo sát cũng như thực hiện đồ án, chúng em đã đạt được một số kết quả sau:

- ◆ Xây dựng được website có giao diện, chức năng, cơ sở dữ liệu tương ứng với các bản thiết kế đã được phân tích.
- ◆ Xây dựng được cơ sở dữ liệu phù hợp bằng hệ quản trị cơ sở dữ liệu MySQL.
- ◆ Tìm hiểu được cách thức làm việc với một bên thứ ba (Call API từ bên thứ 3).

4.2 Nhược điểm

Vì thời gian thực hiện đồ án, khóa luận gấp rút cũng như kiến thức của bản thân em còn nhiều hạn chế, thiếu xót, vì thế đề tài sẽ còn nhiều hạn chế, sai sót:

- Chưa hoàn chỉnh chức năng phân quyền.
- Tại giao diện, chương trình chưa xây dựng đầy đủ tất cả các chức năng như quản lý kho, lịch làm.
- Chức năng tìm kiếm của hệ thống còn đơn giản, chưa sort theo nhiều thuộc tính.
- Chức năng thanh toán bằng ví điện tử Momo, Paypal còn kém ổn định (do phụ thuộc vào server của MOMO).
- Khách hàng chỉ mới có chức năng nhắn tin, liên hệ với nhân viên, chưa có nhiều chức năng khác như nhắn tin nội bộ nhân viên.

4.3 Hướng phát triển

Với những nhược điểm trên của hệ thống, để có thể hoàn thiện và phát triển hơn trong tương lai, em dự định:

- Phát triển giao diện cho các chức năng của hệ thống như đã mô tả.
- Tích hợp các chức năng AI vào dự án.
- Hoàn thiện giao diện trang quản lý kho, ca làm.
- Phát triển, hoàn thiện chức năng liên kết đến Chat box.
- Đánh giá tình trạng trên từng quyển sách mượn.

CHƯƠNG 5. TÀI LIỆU THAM KHẢO

Chương này giới thiệu về các trang, nội dung tài liệu tham khảo để xây dựng, hoàn thành đồ án, khóa luận này.

Tham khảo tài liệu trên các website, sách tham khảo:

- [1] – Nate Murray (2016) – “The Complete Book on Angular 2 (Volume 2)”.
- [2] – Adam Freeman – “Pro Angular 9: Build Powerful and Dynamic Web Apps 4th”.
- [3] – Doguhan Uluca Z – “Angular for Enterprise-Ready Web Applications: Build and deliver production-grade and cloud-scale evergreen web apps with Angular 9 and beyond, 2nd Edition”.
- [4] – Greg Lim – “Beginning Angular with Typescript”.
- [5] – Dương Hữu Thành – “Công nghệ phần mềm”.
- [6] – RXJS – <https://www.learnrxjs.io/>
- [7] – Akita Store, State Management – <https://github.com/datorama/akita>
- [8] – Python – <https://www.python.org/>
- [9] – Angular – <https://www.angular.io/>
- [10] – Flask – <https://flask.palletsprojects.com/en/2.0.x/>