

TRƯỜNG ĐẠI HỌC KIẾN TRÚC ĐÀ NẴNG
KHOA CÔNG NGHỆ THÔNG TIN



ĐỒ ÁN TỐT NGHIỆP
NGÀNH: CÔNG NGHỆ THÔNG TIN

ĐỀ TÀI:

**XÂY DỰNG HỆ THỐNG QUẢN LÝ ĐẶT VÉ XEM
PHIM TRÊN NỀN TẢNG REACTJS VÀ NODEJS**

Người hướng dẫn: **TS. LÂM TÙNG GIANG**

Sinh viên thực hiện: **LÊ MINH PHÚ**

Mã số sinh viên: **2151220272**

Lớp: **21CT6**

Đà Nẵng, 12/2025

TRƯỜNG ĐẠI HỌC KIẾN TRÚC ĐÀ NẴNG

KHOA CÔNG NGHỆ THÔNG TIN



ĐỒ ÁN TỐT NGHIỆP

NGÀNH: CÔNG NGHỆ THÔNG TIN

ĐỀ TÀI:

XÂY DỰNG HỆ THỐNG QUẢN LÝ ĐẶT VÉ XEM PHIM TRÊN NỀN TẢNG REACTJS VÀ NODEJS

Người hướng dẫn: TS. LÂM TÙNG GIANG

Sinh viên thực hiện: LÊ MINH PHÚ

Mã số sinh viên: 2151220272

Lớp: 21CT6

Đà Nẵng, 12/2025

TÓM TẮT

Tên đề tài : Xây dựng hệ thống quản lý đặt vé xem phim nền tảng reactjs và nodejs

Sinh viên thực hiện : Lê Minh Phú

Mã số SV : 2151220272

Lớp : 21CT6

Đề tài tập trung hướng đến việc xây dựng một hệ thống quản lý đặt vé xem phim nhằm hỗ trợ rạp chiếu phim tự động hóa quá trình bán vé và nâng cao trải nghiệm người dùng. Hệ thống cho phép khách hàng tra cứu phim xem thông tin phim, lựa chọn suất chiếu (gồm: ngày chiếu và giờ chiếu), chọn ghế và thanh toán trực tuyến một cách nhanh chóng. Về phía quản trị viên, hệ thống hỗ trợ quản lý phim, lịch chiếu, tài khoản người dùng, doanh thu và báo cáo doanh thu.

Hệ thống phục vụ cho 2 đối tượng:

- Khách hàng: Xem thông tin phim, thông tin cá nhân, đặt vé, thanh toán online, lịch sử đặt vé.
- Quản Lý/Admin: quản lý danh sách phim chiếu, quản lý vé, quản lý quyền tài khoản, doanh thu.

Đề tài áp dụng công nghệ web ReactJs (frontend), NodeJs và ExpressJs (backend) hiện đại kết hợp cơ sở dữ liệu MySQL cùng JavaScript để đảm bảo tính chính xác, an toàn và tốc độ xử lý. Việc xây dựng hệ thống không chỉ giúp giảm tải nhân sự, hạn chế sai sót khi đặt vé thủ công mà còn mang đến sự tiện lợi, minh bạch và chuyên nghiệp trong quá trình vận hành rạp chiếu phim.

LỜI NÓI ĐẦU

Để hoàn thành được Đồ án tốt nghiệp này em xin chân thành cảm ơn quý thầy cô trong khoa Công Nghệ Thông Tin - Trường Đại Học Kiến Trúc Đà Nẵng đã tận tình truyền đạt kiến thức trong những năm học vừa qua. Với vốn kiến thức được tiếp thu trong quá trình học không chỉ là nền tảng cho quá trình làm đồ án tốt nghiệp mà còn là hành trang quý báu để em bước vào đời một cách vững chắc và tự tin.

Em cũng xin gửi lời cảm ơn chân thành và sự tri ân sâu sắc của thầy **TS.Lâm Tùng Giang** trong suốt thời gian vừa qua đã nhiệt tình chỉ dạy, giúp đỡ em để có thể hoàn thành tốt đồ án tốt nghiệp.

Trong quá trình làm bài đồ án tốt nghiệp, do trình độ lý luận cũng như kinh nghiệm thực tiễn còn hạn chế nên không thể tránh khỏi những thiếu sót, em rất mong nhận được ý kiến đóng góp của quý báu của các thầy cô để em có thể học hỏi thêm được nhiều kinh nghiệm, kiến thức.

Em xin trân trọng cảm ơn!

CAM ĐOAN

Tôi Lê Minh Phú sinh viên khóa 2021, chuyên ngành Công nghệ thông tin, xin cam đoan rằng:

1. Đồ án tốt nghiệp này là kết quả nghiên cứu và thực hiện của riêng tôi, được tiến hành trên cơ sở kiến thức, tư duy khoa học và sự hướng dẫn của giảng viên hướng dẫn.
2. Đồ án không sao chép hoặc đạo văn từ bất kỳ công trình, tài liệu, hay nguồn thông tin nào khác mà không có trích dẫn, tham khảo rõ ràng và hợp lệ.
3. Các nguồn tài liệu, dữ liệu, và thông tin sử dụng trong quá trình thực hiện đồ án đều được chọn lọc từ những nguồn chính thống, đáng tin cậy và tuân thủ quy định về bản quyền.
4. Các số liệu, kết quả nghiên cứu và kết luận trong đồ án là trung thực, chính xác, không bị chỉnh sửa, bóp méo hay làm sai lệch kết quả nghiên cứu.
5. Tôi nhận thức rõ tầm quan trọng của liêm chính học thuật và chịu trách nhiệm hoàn toàn trước nhà trường, giảng viên hướng dẫn và pháp luật về tính trung thực trong nội dung của đồ án.

Tôi xin cam kết thực hiện nghiêm túc các quy định của nhà trường về đạo đức và liêm chính học thuật, và chịu hoàn toàn trách nhiệm nếu có bất kỳ vi phạm nào xảy ra.

Sinh viên thực hiện

Lê Minh Phú

MỤC LỤC

TÓM TẮT

LỜI NÓI ĐẦU.....	i
------------------	---

MỤC LỤC	iii
---------------	-----

DANH SÁCH CÁC BẢNG	v
--------------------------	---

DANH SÁCH HÌNH VẼ	vi
-------------------------	----

DANH SÁCH CÁC KÝ HIỆU, CHỮ VIẾT TẮT	ix
---	----

MỞ ĐẦU	1
--------------	---

CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI.....	2
---------------------------------	---

1.1 Mục tiêu thực hiện đề tài.....	2
------------------------------------	---

1.2 Đối tượng nghiên cứu	2
--------------------------------	---

1.3 Phạm vi nghiên cứu	3
------------------------------	---

1.4 Phương pháp nghiên cứu	4
----------------------------------	---

1.4.1 Nghiên cứu tài liệu	4
---------------------------------	---

1.4.2 Phân tích và thiết kế hệ thống	4
--	---

1.4.3 Phương pháp thực nghiệm	4
-------------------------------------	---

1.4.4 Phương pháp kiểm thử.....	4
---------------------------------	---

1.5 Cấu trúc đồ án.....	5
-------------------------	---

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT	6
---------------------------------	---

2.1 Giới thiệu mô hình	6
------------------------------	---

2.1.1 Mô hình Client – Server trong phát triển website	6
--	---

2.1.2 Vai trò và chức năng của Client	7
---	---

2.1.3 Vai trò và chức năng của Server	7
---	---

2.1.4 Ưu điểm của mô hình Client – Server	8
---	---

2.2 Giới thiệu về công nghệ	8
-----------------------------------	---

2.2.1 Giới thiệu ReactJs	8
--------------------------------	---

2.2.2 Giới thiệu Vite.....	10
----------------------------	----

2.2.3 Giới thiệu React Router.....	12
------------------------------------	----

2.2.4 Giới thiệu Scss module	13
------------------------------------	----

2.2.5	<i>Context API</i>	14
2.2.6	<i>NodeJS</i>	15
2.2.7	<i>Express.Js</i>	16
2.3	Kết nối Frontend – Backend	17
2.4	Cơ sở dữ liệu MySQL	18
CHƯƠNG 3. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG		20
3.1	Mô tả và phân tích dự án	20
3.1.1	<i>Đặc tả dự án</i>	20
3.1.2	<i>Phân tích dự án</i>	21
3.2	Phân tích hệ thống	22
3.2.1	<i>Danh sách tác nhân và sơ đồ usecase tổng quát</i>	22
3.2.2	<i>Đặc tả usecase</i>	23
3.2.3	<i>Sơ đồ hoạt động</i>	28
3.2.4	<i>Sơ đồ tuần tự</i>	31
3.2.5	<i>Sơ đồ ERD</i>	34
3.2.6	<i>Thiết kế cơ sở dữ liệu MySQL</i>	35
3.3	Kết luận	36
CHƯƠNG 4. TRIỂN KHAI HỆ THỐNG VÀ GIAO DIỆN CHỨC NĂNG		38
4.1	Module chức năng của hệ thống	38
4.1.1	<i>Chức năng người dùng</i>	38
4.1.2	<i>Chức năng admin</i>	51
4.2	Giao diện sản phẩm	59
4.2.1	<i>Giao diện của khách hàng</i>	59
4.2.2	<i>Giao diện của Admin</i>	65
4.3	Kết luận chương 4	69
KẾT LUẬN		70
TÀI LIỆU THAM KHẢO		71

DANH SÁCH CÁC BẢNG

Bảng 2.1 Ưu điểm của Vite	11
Bảng 3.1 Bảng danh sách tác nhân	22
Bảng 3.2 Bảng đặc tả usecase đặt vé	24
Bảng 3.3 Bảng đặc tả usecase tìm kiếm	25
Bảng 3.4 Bảng đặc tả usecase quản lý phim	26
Bảng 3.5 Bảng đặc tả usecase quản lý vé	27
Bảng 3.6 Bảng thông tin ghế được đặt	35
Bảng 3.7 Bảng thông tin người dùng	35
Bảng 3.8 Bảng thông tin thanh toán	36
Bảng 3.9 Bảng thông tin phim	36
Bảng 3.10 Bảng thông tin danh mục phim	36
Bảng 3.11 Bảng đánh giá phim	36
Bảng 3.12 Bảng dữ liệu OTP của user	36
Bảng 3.13 Bảng dữ liệu messagebot	36

DANH SÁCH HÌNH VẼ

Hình 2.1 Mô hình client - server	6
Hình 2.2 ReactJs là gì?	8
Hình 2.3 Vite là gì?	10
Hình 2.4 Scss là gì?	13
Hình 2.5 NodeJs là gì?.....	15
Hình 2.6 MySQL là gì?	19
Hình 3.1 Sơ đồ usecase tổng quát.....	23
Hình 3.2 Sơ đồ hoạt động chức năng đặt vé.....	28
Hình 3.3 Sơ đồ hoạt động chức năng tìm kiếm	28
Hình 3.4 Sơ đồ hoạt động chức năng quản lý phim	29
Hình 3.5 Sơ đồ hoạt động chức năng quản lý vé.....	29
Hình 3.6 Sơ đồ hoạt động chức năng thanh toán.....	30
Hình 3.7 Sơ đồ hoạt động chức năng messagebot.....	30
Hình 3.8 Sơ đồ tuần tự chức năng đặt vé	31
Hình 3.9 Sơ đồ tuần tự chức năng tìm kiếm.....	31
Hình 3.10 Sơ đồ tuần tự quản lý vé	32
Hình 3.11 Sơ đồ tuần tự chức năng quản lý phim.....	33
Hình 3.12 Sơ đồ tuần tự chức năng messagebot	34
Hình 3.13 Sơ đồ ERD.....	34
Hình 3.14 Lược đồ quan hệ.....	35
Hình 4.1 Form hiển thị thông tin đăng ký	38
Hình 4.2 Hàm xử lý thông tin form đăng ký	39
Hình 4.3 Form đăng nhập	39
Hình 4.4 Hàm xử lý thông tin đăng nhập	40
Hình 4.5 Hàm hiển thị danh mục và phim.....	41
Hình 4.6 Hàm hiển thị thông tin phim.....	41

Hình 4.7 Hàm xử lý lấy dữ liệu phim từ cơ sở dữ liệu.....	42
Hình 4.8 Form hiển thị thông tin chi tiết phim.....	42
Hình 4.9 Hàm lấy và xử lý logic dữ liệu phim.....	43
Hình 4.10 Luồng xử lý đặt vé.....	43
Hình 4.11 Luồng xử lý đặt vé.....	44
Hình 4.12 Luồng xử lý thanh toán.....	44
Hình 4.13 Hàm xử lý lấy danh sách ghế theo phim	44
Hình 4.14 Form hiển thị thông tin người dùng.....	45
Hình 4.15 Hàm xử lý cập nhật thông tin người dùng.....	46
Hình 4.16 Xử lý hiển thị thanh toán và đánh giá bình luận	46
Hình 4.17 Form hiển thị lịch sử đặt vé	47
Hình 4.18 Hàm xử lý truy vấn lấy danh sách lịch sử đặt vé.....	47
Hình 4.19 Xử lý thanh toán	48
Hình 4.20 Hàm xác nhận thông tin thanh toán.....	48
Hình 4.21 Xử lý phương thức thanh toán.....	49
Hình 4.22 Xử lý thông tin thanh toán.....	49
Hình 4.23 Form hiển thị thông tin thanh toán	50
Hình 4.24 Hàm xử lý tìm kiếm.....	50
Hình 4.25 Xử lý logic tìm kiếm với Gemini	51
Hình 4.26 Xử lý từng phần trong thống kê.....	51
Hình 4.27 Thiết kế form hiển thị quản lý thống kê	52
Hình 4.28 Hàm xử lý lấy thông tin thống kê	52
Hình 4.29 Cài đặt xử lý logic các hàm quản lý phim.....	53
Hình 4.30 Thiết kế form hiển thị ở quản lý phim.....	53
Hình 4.31 Xử lý logic chức năng của quản lý phim.....	54
Hình 4.32 Logic BE quản lý phim.....	54
Hình 4.33 Xử lý và hiển thị chức năng danh mục.....	55
Hình 4.34 Xử lý logic chức năng danh mục.....	55

Hình 4.35 Xử lý và hiển thị danh sách vé theo form.....	56
Hình 4.36 Lọc vé và xem chi tiết vé theo form	56
Hình 4.37 Xử lý logic các chức năng ở quản lý vé	57
Hình 4.38 form hiển thị thông tin và chỉnh sửa quyền người dùng	58
Hình 4.39 Xử lý logic danh sách và quyền người dùng	58
Hình 4.40 Luồng xử lý và form hiển thị chatbot.....	59
Hình 4.41 Giao diện đăng ký.....	59
Hình 4.42 Giao diện đăng nhập	60
Hình 4.43 Giao diện trang chủ	60
Hình 4.44 Giao diện tìm kiếm phim	61
Hình 4.45 Giao diện chi tiết phim	61
Hình 4.46 Giao diện phim	62
Hình 4.47 Giao diện chọn suất chiếu.....	62
Hình 4.48 Giao diện chọn ghế và phương thức thanh toán	63
Hình 4.49 Tích hợp thanh toán online Momo	63
Hình 4.50 Giao diện thông tin cá nhân.....	64
Hình 4.51 Giao diện chức năng cập nhật thông tin	64
Hình 4.52 Giao diện lịch sử đặt vé	65
Hình 4.53 Giao diện messagebot ở trang chủ.....	65
Hình 4.54 Giao diện trang quản lý thống kê hệ thống	66
Hình 4.55 Giao diện quản lý phim	67
Hình 4.56 Giao diện quản lý danh mục phim.....	67
Hình 4.57 Giao diện quản lý vé.....	68
Hình 4.58 Giao diện quản lý người dùng	68

DANH SÁCH CÁC KÝ HIỆU, CHỮ VIẾT TẮT

STT	Chữ viết tắt	Tên đầy đủ	Ý Nghĩa
1	API	Application Programming Interface	Giao diện lập trình ứng dụng
2	UI	User Interface	Giao diện người dùng
3	UX	User Experience	Trải nghiệm người dùng
4	JWT	JSON Web Token	Chuỗi mã hóa dùng để xác thực và phân quyền người dùng
5	AI	Artificial Intelligence.	Trí tuệ nhân tạo
6	ERD	Entity–Relationship Diagram	Sơ đồ thực thể – mối quan hệ
7	HTTP	HyperText Transfer Protocol	Giao thức truyền tải siêu văn bản (truyền dữ liệu giữa client và server).
8	HTTPS	HyperText Transfer Protocol Secure	Phiên bản “bảo mật” của HTTP (dùng SSL/TLS để mã hóa dữ liệu).
9	SCSS	Sassy Cascading Style Sheets	Cú pháp mở rộng của CSS, thuộc ngôn ngữ Sass.
10	JSX	JavaScript XML	Cú pháp mở rộng của JavaScript
11	DOM	Document Object Model	Mô hình đối tượng tài liệu.
12	SDK	Software Development Kit	Bộ công cụ hỗ trợ lập trình và phát triển phần mềm.
13	CORS	Cross-Origin Resource Sharing	Cơ chế chia sẻ tài nguyên giữa các nguồn khác nhau
14	CRUD	Create – Read – Update – Delete	Tạo mới dữ liệu - đọc / lấy dữ liệu - cập nhật dữ liệu - xóa dữ liệu.
15	NPM	Node Package Manager	Trình quản lý thư viện (package manager)

MỞ ĐẦU

Trong bối cảnh công nghệ thông tin phát triển mạnh mẽ và thương mại điện tử đang trở thành xu hướng tất yếu, nhu cầu mua sắm trực tuyến ngày càng gia tăng. Một trong những lĩnh vực chịu tác động rõ rệt của xu thế này là ngành giải trí, đặc biệt là dịch vụ xem phim. Vấn đề thiếu thông tin chi tiết về các bộ phim, khó khăn trong việc so sánh giá cả thị trường, cũng như sự bất tiện khi phải đến xếp hàng chờ ở các cửa hàng truyền thống đã đặt ra yêu cầu cấp thiết về một giải pháp mua sắm tiện lợi, nhanh chóng và hiệu quả.

Trước thực trạng đó, việc xây dựng một hệ thống đặt vé xem phim trực tuyến là nhu cầu cần thiết nhằm hỗ trợ rạp chiếu phim tự động hóa quy trình bán vé, đồng thời mang lại sự tiện lợi cho người dùng trong việc tra cứu thông tin và lựa chọn suất chiếu phù hợp.

Đề tài “***Xây dựng hệ thống đặt vé xem phim nền tảng ReactJs và NodeJs***” được thực hiện với mục tiêu phát triển một nền tảng giúp quản lý toàn bộ quy trình từ xem thông tin phim, chọn suất chiếu, chọn ghế, đặt vé cho đến thanh toán trực tuyến và chatbot hỗ trợ khách hàng thông tin phim trên hệ thống. Hệ thống hướng đến việc giảm tải công việc cho nhân viên rạp, hạn chế sai sót khi bán vé thủ công và giúp khách hàng chủ động hơn trong việc chọn thời gian xem phim.

Mục đích của việc thực hiện đề tài là tạo ra một hệ thống đặt vé hiện đại, thân thiện và dễ sử dụng, hỗ trợ người dùng đặt vé mọi lúc, mọi nơi. Đồng thời, hệ thống cũng cho phép nhà quản lý theo dõi số lượng vé bán ra, tình trạng phòng chiếu, doanh thu theo từng suất hoặc từng ngày, từ đó giúp tối ưu hóa hoạt động kinh doanh của rạp.

Bên cạnh đó, hệ thống còn mang đến giao diện trực quan, thao tác đơn giản, góp phần nâng cao trải nghiệm của người dùng và tăng khả năng tiếp cận dịch vụ của rạp chiếu phim. Đây sẽ là giải pháp giúp rạp phim phát triển theo hướng hiện đại hóa, chuyên nghiệp hóa và gia tăng lợi nhuận trong tương lai.

CHƯƠNG 1. TỔNG QUAN ĐỀ TÀI

1.1 Mục tiêu thực hiện đề tài

Xây dựng hệ thống website đặt vé xem phim trực tuyến với giao diện thân thiện, hỗ trợ người dùng dễ dàng đăng ký, đăng nhập, lựa chọn phim, suất chiếu, ghế ngồi, thanh toán trực tuyến và chatbot AI hỗ trợ người dùng thông tin phim trên hệ thống.

Ứng dụng công nghệ ReactJS trong việc thiết kế giao diện người dùng (Frontend), đảm bảo tính trực quan, tốc độ phản hồi nhanh và khả năng tương thích đa nền tảng (máy tính, điện thoại, máy tính bảng).

Sử dụng NodeJS để phát triển hệ thống xử lý phía máy chủ (Backend), đáp ứng nhanh, quản lý dữ liệu hiệu quả và hỗ trợ kết nối với cơ sở dữ liệu.

Triển khai cơ sở dữ liệu MySQL nhằm lưu trữ, quản lý thông tin về phim, suất chiếu, người dùng, vé và các giao dịch, đảm bảo tính toàn vẹn dữ liệu và khả năng mở rộng.

Tích hợp các chức năng quản trị cho phép quản lý danh mục phim, suất chiếu, quản lý tài khoản tìm kiếm và lọc phim, vé bán ra và theo dõi thống kê doanh thu, phục vụ cho công tác quản lý rạp chiếu nắm bắt tình hình kịp thời đưa ra một kế hoạch chiến lược phù hợp.

Đề tài không chỉ dừng lại ở việc xây dựng một website đặt vé đơn thuần, mà còn hướng tới khả năng ứng dụng thực tế trong lĩnh vực giải trí – điện ảnh, góp phần nâng cao trải nghiệm khách hàng, tối ưu hóa quy trình bán vé và hỗ trợ công tác quản lý của doanh nghiệp.

1.2 Đối tượng nghiên cứu

Đề tài tập trung vào việc nghiên cứu và phát triển một hệ thống website đặt vé xem phim trực tuyến dựa trên doanh nghiệp vừa và nhỏ tập trung vào những chức năng cốt lõi (quản lý phim, đặt vé, thanh toán, quản lý vé).

Đối tượng người dùng (khách hàng / user):

- Thực hiện đăng ký, đăng nhập, quản lý thông tin tài khoản cá nhân.
- Tra cứu danh sách phim, lịch chiếu, giá vé.
- Thực hiện thao tác đặt vé: chọn phim, vị trí ghế và thanh toán online.

- Nhận thông tin xác nhận vé sau khi hoàn tất giao dịch.
- Chatbot AI hỗ trợ khách hàng thông tin phim ở hệ thống

Quản lý hệ thống (admin / rạp chiếu):

- Quản lý danh sách phim: thêm mới, chỉnh sửa, xóa phim.
- Quản lý phòng chiếu, lịch chiếu và suất chiếu.
- Quản lý đơn đặt vé, theo dõi thống kê số lượng vé bán ra và doanh thu.
- Quản lý thông tin người dùng, phân quyền và xử lý sự cố phát sinh.

Công nghệ sử dụng:

- ReactJS: xây dựng giao diện người dùng, tối ưu trải nghiệm, tăng tính tương tác.
- NodeJS: phát triển hệ thống xử lý phía máy chủ, quản lý logic nghiệp vụ.
- MySQL: thiết kế và quản lý cơ sở dữ liệu quan hệ, lưu trữ thông tin về phim, suất chiếu, vé, người dùng và giao dịch.

1.3 Phạm vi nghiên cứu

Kế thừa và phát triển các hệ thống đặt vé trực tuyến hiện có (CGV, Galaxy, Lotte Cinema...) để phân tích tính năng, ưu điểm và hạn chế.

Nghiên cứu các mô hình giao diện thân thiện với người dùng, ứng dụng kỹ thuật Responsive Design nhằm tối ưu trên nhiều thiết bị (PC, laptop, tablet, smartphone).

Kế thừa các thư viện, framework sẵn có của ReactJS và NodeJS để phát triển tính năng nhanh chóng, ổn định.

Xây dựng chức năng chính:

Hệ thống người dùng: đăng ký/đăng nhập/đăng xuất, quản lý hồ sơ thông tin cá nhân, xem lịch sử đặt vé, hủy vé.

Hệ thống đặt vé: Hiển thị thông tin phim (gồm: tên, thể loại, thời lượng, thông tin diễn viên, mô tả phim, trailer và có đánh giá người dùng trước), lựa chọn suất chiếu (ngày chiếu, ghế ngồi), thanh toán trực tuyến.

Hệ thống quản trị: Thêm/sửa/xóa phim, theo dõi thống kê doanh thu, số vé được bán ra, quản lý phân quyền người dùng.

Đề tài tập trung xây dựng website mẫu áp dụng cho một cụm rạp chiếu phim vừa và nhỏ, chưa đi sâu vào tích hợp các cổng thanh toán quốc tế. Hệ thống bảo mật ở mức

cơ bản (mã hóa mật khẩu, phân quyền tài khoản), chưa triển khai đầy đủ các chuẩn bảo mật cao cấp. Chức năng gợi ý phim theo sở thích cá nhân và chương trình tích điểm khách hàng mới dừng lại ở mức ý tưởng, chưa được triển khai chi tiết.

1.4 Phương pháp nghiên cứu

1.4.1 Nghiên cứu tài liệu

Phương pháp này thu thập các tài liệu kiến thức nền tảng và tham khảo các sản phẩm sẵn có trên thị trường nhằm phục vụ cho việc xây dựng hệ thống:

- Tìm hiểu các mô hình đặt vé trực tuyến hiện có (CGV, Galaxy, Lotte Cinema...) để phân tích quy trình nghiệp vụ, giao diện và trải nghiệm người dùng.
- Nghiên cứu tài liệu về các công nghệ sử dụng trong đề tài: ReactJS, NodeJS, MySQL, kiến trúc API, mô hình Client–Server, JWT Authentication...
- Tham khảo các tài liệu chuyên môn, bài báo khoa học, diễn đàn lập trình để lựa chọn công nghệ phù hợp với mục tiêu và phạm vi đề tài.

1.4.2 Phân tích và thiết kế hệ thống

Phân tích yêu cầu: xác định yêu cầu chức năng và phi chức năng dựa trên đối tượng người dùng (User & Admin).

Mô hình hóa hệ thống bằng các sơ đồ:

- Use Case Diagram (mô tả các chức năng người dùng có thể thực hiện).
- Activity Diagram (luồng hoạt động khi đặt vé, quản trị phim...).
- Sequence Diagram (tương tác giữa client – server khi xử lý đặt vé).
- ERD (sơ đồ thực thể – quan hệ cho cơ sở dữ liệu).

1.4.3 Phương pháp thực nghiệm

Xây dựng giao diện FE bằng Reactjs.

Xây dựng Backend bằng NodeJs – ExpressJs.

Thiết kế cơ sở dữ liệu MySQL.

1.4.4 Phương pháp kiểm thử

Kiểm thử chức năng (Functional Testing): kiểm thử các chức năng cơ bản như đăng kí, đăng nhập và các chức năng chính như đặt vé, thanh toán, quản lí phim đảm bảo hoạt động đúng theo yêu cầu đặt ra cho từng nhóm đối tượng.

Kiểm thử giao diện (UI/UX Testing): đánh giá mức độ thân thiện, dễ sử dụng và sự thống nhất các bố cục trong nhiều loại thiết bị mang đến trải nghiệm tốt nhất cho người dùng.

Kiểm thử hiệu năng cơ bản(Performance Testing): kiểm tra tốc độ tải trang, thời gian phản hồi API.

Kiểm thử người dùng(User Testing/Beta Testing): thu thập phản hồi trải nghiệm của người dùng hay quản trị để nâng cao phát triển trải nghiệm tốt nhất trong tương lai.

1.5 Cấu trúc đồ án

Đồ án tốt nghiệp xây dựng hệ thống quản lý đặt vé xem phim với bố cục 3 chương:

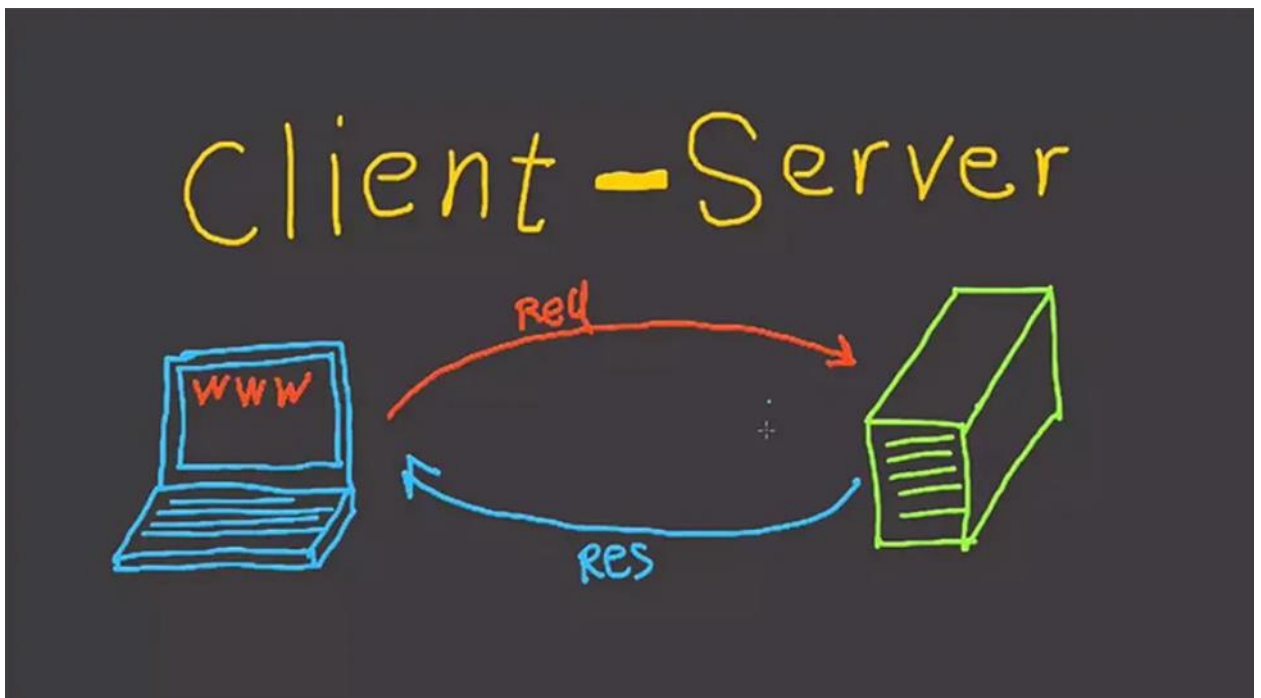
- Chương 1 (Cơ sở lý thuyết): Chuyên đề giới thiệu về những công nghệ đã sử dụng liên quan đến các chủ đề client, server, framework, server, database, và một số công nghệ khác.
- Chương 2 (Phân tích / thiết kế hệ thống): Chuyên đề đưa ra các lược đồ cơ sở dữ liệu, quan hệ giữa các lớp trong biểu đồ và một số biểu đồ tuần tự.
- Chương 3 (kết quả thực nghiệm): Đưa ra các cấu trúc thư mục code, hướng dẫn cài đặt và kết quả thực nghiệm sản phẩm.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

2.1 Giới thiệu mô hình

2.1.1 Mô hình Client – Server trong phát triển website

Trong lĩnh vực phát triển phần mềm nói chung và lập trình web nói riêng, mô hình Client – Server [1] là một kiến trúc cơ bản và phổ biến nhất được sử dụng để xây dựng các ứng dụng web hiện đại. Kiến trúc này phân chia rõ ràng vai trò giữa hai thành phần chính: phía Client (máy khách) và Server (máy chủ), giúp hệ thống hoạt động hiệu quả, linh hoạt và dễ dàng mở rộng. Client (máy khách): Là phía người dùng tương tác trực tiếp, thường là trình duyệt web. Client gửi yêu cầu (request) tới Server thông qua các giao thức như HTTP hoặc HTTPS. Giao diện này có thể được phát triển bằng các công nghệ như HTML, CSS, JavaScript, và các thư viện/framework như React, Vue hoặc Angular.



Hình 2.1 Mô hình client - server

Mô hình Client – Server là một mô hình phân tán, trong đó Client là bên gửi yêu cầu dịch vụ và Server là bên cung cấp dịch vụ đó. Khi người dùng thao tác trên một website (ví dụ như xem phim, đăng nhập, đặt hàng), các hành động đó sẽ được gửi dưới dạng yêu cầu HTTP tới Server. Sau khi xử lý yêu cầu, Server sẽ gửi phản hồi về cho Client, thường dưới dạng dữ liệu JSON hoặc HTML, và dữ liệu này sẽ được Client hiển thị cho người dùng.

Đặc điểm của mô hình này là Client không trực tiếp truy cập vào cơ sở dữ liệu, mà toàn bộ tương tác với dữ liệu đều phải thông qua Server, đảm bảo tính bảo mật và quản lý dữ liệu tập trung.

2.1.2 Vai trò và chức năng của Client

Client là phần giao diện người dùng (UI – User Interface), nơi người dùng tương tác trực tiếp với hệ thống. Trong chuyên đề này, phần Client được xây dựng bằng React.js một thư viện JavaScript mạnh mẽ chuyên dùng để xây dựng các giao diện người dùng tương tác nhanh và mượt mà. Các chức năng chính của Client bao gồm:

- Hiển thị giao diện thân thiện, dễ sử dụng.
- Gửi các yêu cầu HTTP đến Server thông qua các phương thức như GET, POST, PUT, DELETE.
- Nhận dữ liệu phản hồi từ Server và cập nhật giao diện tương ứng.
- Thực hiện điều hướng trang, quản lý trạng thái giao diện (state), và xử lý sự kiện người dùng.

Với việc sử dụng công nghệ Single Page Application (SPA), ứng dụng React chỉ tải một lần duy nhất và sau đó cập nhật nội dung động mà không cần tải lại toàn bộ trang, mang đến trải nghiệm mượt mà cho người dùng.

2.1.3 Vai trò và chức năng của Server

Server là nơi tiếp nhận, xử lý các yêu cầu từ Client và gửi phản hồi tương ứng. Trong đồ án này, Server được xây dựng bằng Node.js kết hợp với Express.js – một framework mạnh mẽ và linh hoạt cho ứng dụng web.

Các chức năng chính của Server bao gồm:

- Xử lý logic nghiệp vụ (business logic) như đăng nhập, xử lý giỏ hàng, đặt hàng...
- Quản lý và tương tác với cơ sở dữ liệu (thêm, sửa, xóa, truy vấn dữ liệu).
- Cung cấp API RESTful để các client có thể dễ dàng giao tiếp.
- Bảo mật hệ thống bằng các phương pháp xác thực (authentication) và phân quyền (authorization).
- Xử lý các tệp tin (uploads), mã hóa thông tin, gửi email, v.v.
- Server thường hoạt động trên một cổng (port) riêng, lắng nghe các yêu cầu đến và phản hồi dữ liệu phù hợp với định dạng JSON hoặc HTML.

2.1.4 Ưu điểm của mô hình Client – Server

Tính phân tách rõ ràng: Tách biệt giao diện và xử lý logic, dễ bảo trì và nâng cấp

Tái sử dụng cao: Backend có thể dùng chung cho nhiều loại client (web, mobile, desktop).

Dễ mở rộng: Có thể tăng hiệu năng Server hoặc triển khai thêm nhiều Client dễ dàng.

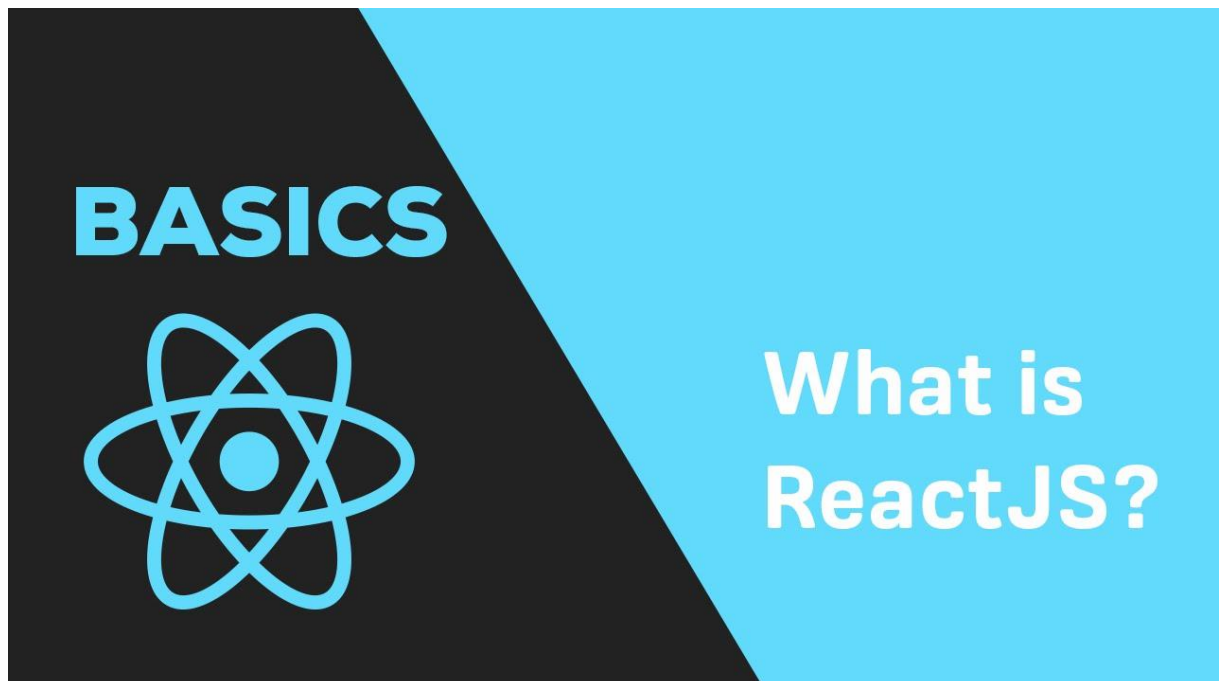
Bảo mật: Dữ liệu được quản lý tập trung tại Server, dễ kiểm soát và bảo vệ.

2.2 Giới thiệu về công nghệ

2.2.1 Giới thiệu ReactJs

React.js (gọi tắt là React [2]) là một thư viện JavaScript mã nguồn mở được phát triển bởi Facebook, lần đầu được ra mắt vào năm 2013. React được sử dụng để xây dựng giao diện người dùng (UI – User Interface) trong các ứng dụng web và mobile. Thay vì thao tác trực tiếp với DOM truyền thống, React sử dụng một cơ chế gọi là Virtual DOM để tối ưu hóa hiệu suất cập nhật giao diện.

React không phải là một framework toàn diện như Angular hay Vue, mà chỉ tập trung vào “V” (View) trong mô hình MVC (Model – View – Controller), giúp lập trình viên dễ dàng tạo ra các thành phần giao diện tái sử dụng.



Hình 2.2 ReactJs là gì?

Đặc điểm nổi bật của React.js

Virtual DOM: React sử dụng Virtual DOM để theo dõi sự thay đổi giao diện và chỉ cập nhật những phần cần thiết trên DOM thật, giúp tăng hiệu suất và tốc độ phản hồi của ứng dụng.

Component-Based Architecture (Kiến trúc thành phần): Giao diện trong React được chia nhỏ thành các thành phần (component) riêng biệt và độc lập, giúp tái sử dụng và dễ bảo trì.

JSX (JavaScript XML): Cho phép viết HTML bên trong JavaScript, giúp code dễ đọc, dễ viết và dễ hiểu hơn.

One-Way Data Binding (Ràng buộc dữ liệu một chiều): Dữ liệu được truyền từ component cha xuống component con, tạo ra luồng dữ liệu rõ ràng và dễ kiểm soát.

Hooks (Từ phiên bản 16.8): React cung cấp các hook như useState, useEffect, useContext... cho phép sử dụng state và các tính năng khác mà không cần viết class component.

Ưu điểm của React

Giao diện mượt mà, tốc độ cao nhờ Virtual DOM.

Dễ bảo trì và mở rộng do thiết kế theo thành phần.

Thích hợp cho các ứng dụng lớn, có nhiều tương tác người dùng.

Cộng đồng phát triển lớn, tài liệu đầy đủ.

Dễ tích hợp với các thư viện và công nghệ khác như Redux, Tailwind CSS, Axios, React Router...

Ứng dụng của React trong chuyên đề

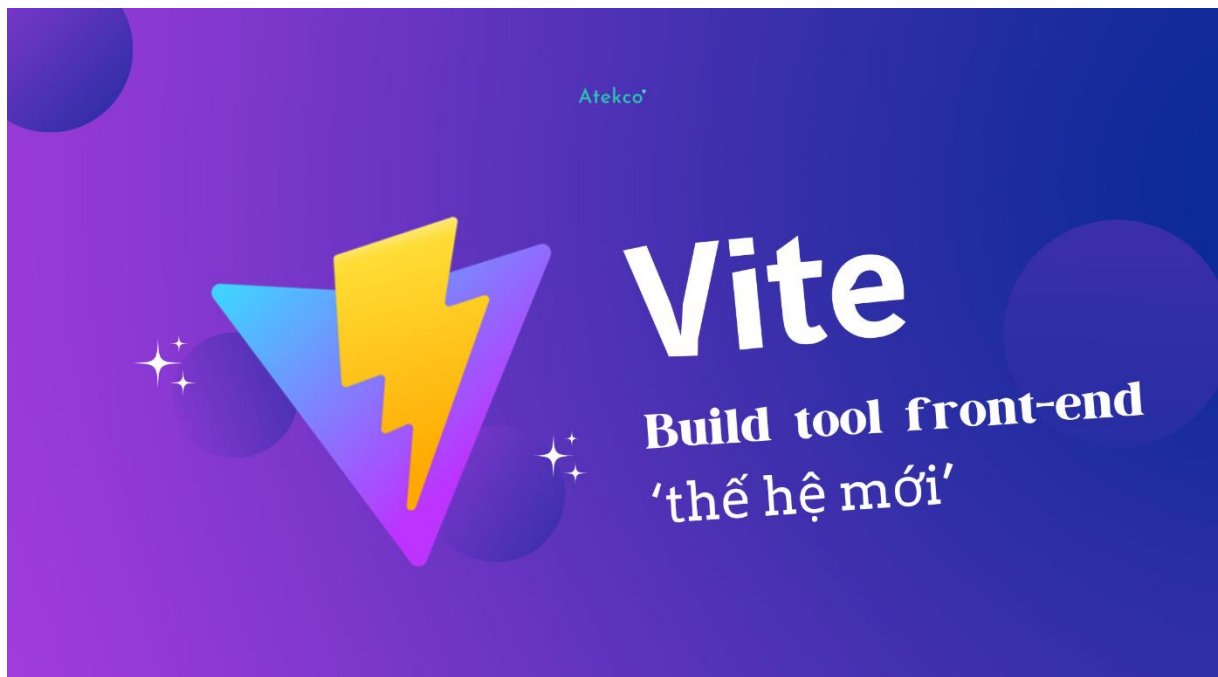
Trong đồ án xây dựng website bán Vé Xem Phim, React.js được sử dụng để:

- Thiết kế các giao diện người dùng như: trang chủ, trang chi tiết phim, giỏ hàng, đăng nhập, đăng ký, trang quản trị.
- Tạo các component tái sử dụng như: Header, Footer, ProductCard, FormInput, Modal...
- Gọi dữ liệu từ backend thông qua Axios và hiển thị lên giao diện.
- Điều hướng trang bằng thư viện react-router-dom.

2.2.2 Giới thiệu Vite

Vite [3] là một công cụ build (build tool) thế hệ mới được phát triển bởi Evan You – tác giả của Vue.js. Vite được thiết kế để giúp quá trình phát triển ứng dụng frontend trở nên nhanh hơn và nhẹ hơn, đặc biệt là khi làm việc với các framework hiện đại như React, Vue, hoặc Svelte.

Từ "Vite" trong tiếng Pháp nghĩa là "nhanh" – phản ánh đúng mục tiêu mà công cụ này hướng tới: tăng tốc độ khởi động dự án và rút ngắn thời gian build phim.



Hình 2.3 Vite là gì?

Đặc điểm nổi bật của Vite

Khởi động siêu nhanh: Vite sử dụng ES Modules (ESM) để phục vụ mã nguồn trực tiếp từ trình duyệt trong môi trường phát triển, thay vì phải bundle toàn bộ project như Webpack, giúp khởi động gần như tức thì.

Hot Module Replacement (HMR) hiệu suất cao: Vite hỗ trợ HMR cực kỳ nhanh và chính xác, giúp nhà phát triển thấy ngay thay đổi khi cập nhật mã mà không cần reload trang.

Tối ưu khi build: Khi build cho production, Vite sử dụng Rollup – một bundler tối ưu, giúp tạo ra các file JavaScript gọn nhẹ, hiệu quả cao và dễ dàng cấu hình.

Hỗ trợ sẵn TypeScript, JSX, CSS Modules, PostCSS mà không cần cấu hình phức tạp.

Tương thích tốt với React, Vue và các thư viện frontend hiện đại.

Cách hoạt động của Vite

Vite chia quy trình làm việc thành hai pha chính:

- Pha phát triển (Development):
 - + Vite sử dụng native ESM và một server nhẹ được viết bằng Node.js.
 - + Các tệp JavaScript, CSS, TS... được phân phối trực tiếp đến trình duyệt mà không cần bundle, giúp thời gian phản hồi cực nhanh.
 - + Khi một tệp thay đổi, chỉ riêng tệp đó được xử lý lại thay vì rebuild toàn bộ dự án.
- Pha build (Production):
 - + Vite sử dụng Rollup để build toàn bộ project thành các file tối ưu.
 - + Tách code thành nhiều chunk, hỗ trợ lazy loading và tree shaking để giảm kích thước file.

Ứng dụng của Vite trong đồ án

Trong chuyên đề xây dựng website bán vé xem phim, Vite được sử dụng để:

- Khởi tạo dự án React nhanh chóng với lệnh `npm create vite@latest`.
- Tăng tốc độ khởi động và thay đổi giao diện trong quá trình phát triển frontend.
- Kết hợp với Tailwind CSS để build giao diện nhanh chóng, nhẹ và tối ưu.
- Tạo ra bản build nhỏ gọn, dễ deploy cho trang web bán vé xem phim khi triển khai thực tế.

Ưu điểm khi sử dụng Vite thay cho Webpack

Tiêu chí	Vite	Webpack
Tốc độ khởi động	Rất nhanh	Chậm hơn do cần bundle
HMR	Cập nhật tức thì	Có độ trễ khi dự án lớn
Cấu hình	Đơn giản, gần như không cần	Phức tạp, nhiều file cấu hình
Build production	Sử dụng Rollup, nhẹ và hiệu quả	Dùng chính Webpack, chậm hơn

Bảng 2.1 Ưu điểm của Vite

2.2.3 Giới thiệu React Router

React Router [4] là một thư viện định tuyến (routing library) được sử dụng phổ biến trong các ứng dụng React để quản lý việc điều hướng giữa các trang (route) một cách mượt mà mà không cần tải lại toàn bộ trang web. Nó cho phép các ứng dụng đơn trang (SPA – Single Page Application) hoạt động như các ứng dụng đa trang, bằng cách điều hướng giữa các component dựa trên URL trình duyệt.

React Router được phát triển bởi Remix Software và hiện nay phiên bản phổ biến là React Router v6, cung cấp cú pháp đơn giản, tối ưu hiệu suất, dễ sử dụng hơn các phiên bản trước.

Cách hoạt động

React Router hoạt động bằng cách:

- Theo dõi sự thay đổi của URL trong trình duyệt mà không làm reload trang.
- Gắn kết các URL tương ứng với component React cần hiển thị.
- Quản lý lịch sử điều hướng thông qua History API.

Các thành phần chính trong React Router:

- `<BrowserRouter>`: Bao bọc toàn bộ ứng dụng, quản lý lịch sử điều hướng.
- `<Routes>`: Container cho các route.
- `<Route>`: Định nghĩa từng tuyến đường, ánh xạ giữa đường dẫn và component.
- `<Link>`: Thay thế cho thẻ `<a>` để chuyển trang mà không reload.
- `useNavigate()`, `useParams()`, `useLocation()`: Các hook hỗ trợ điều hướng, lấy tham số, và vị trí hiện tại của trang.

Ưu điểm của React Router

Tối ưu cho SPA: Điều hướng nhanh chóng, không cần tải lại trang.

Tích hợp sâu với React: Giao diện được cập nhật theo component tương ứng.

Hỗ trợ nested routes: Có thể lồng các route bên trong nhau.

Route động: Hỗ trợ đường dẫn có tham số, ví dụ `/product/:id`.

Điều hướng lập trình (programmatic navigation): Có thể điều hướng bằng hàm mà không cần click thủ công.

Ứng dụng trong đồ án website bán vé xem phim

Trong chuyên đề xây dựng website bán vé xem phim, React Router được sử dụng để:

- Tạo hệ thống định tuyến gồm các trang: trang chủ, chi tiết phim, lịch sử đặt vé, đăng nhập/dăng ký, trang quản trị, trang quên mật khẩu, trang thông tin người dùng.
- Cho phép chuyển trang mượt mà giữa các giao diện mà không tải lại toàn bộ ứng dụng.
- Hiển thị thông tin phim cụ thể thông qua route động như /product/123.
- Sử dụng useNavigate() để điều hướng người dùng sau khi đăng nhập hoặc mua hàng thành công.

2.2.4 Giới thiệu Scss module

SCSS Modules [5] là sự kết hợp giữa SCSS (Sassy CSS) – phần mở rộng của CSS với cú pháp nâng cao, và CSS Modules – kỹ thuật viết CSS với phạm vi riêng biệt cho từng component. Việc kết hợp này giúp quản lý style trong các dự án frontend, đặc biệt là khi sử dụng React, một cách khoa học, rõ ràng và tránh trùng lặp.

SCSS cung cấp các tính năng như biến, mixins, nesting và import, giúp người phát triển có thể viết mã CSS một cách rõ ràng, dễ đọc và dễ bảo trì. Trong khi đó, CSS Modules đảm bảo rằng các lớp (class) chỉ áp dụng trong phạm vi component cụ thể, tránh xung đột toàn cục.



Hình 2.4 Scss là gì?

Ưu điểm của SCSS Modules

Phạm vi riêng biệt (local scope): Tránh xung đột tên class giữa các component trong toàn bộ ứng dụng.

Cấu trúc rõ ràng, dễ bảo trì: Mỗi component có thể gắn liền với file style riêng biệt.

Tận dụng toàn bộ sức mạnh của SCSS như biến, mixin, nesting, kế thừa...

Tương thích tốt với các dự án React + Vite, không cần cấu hình phức tạp.

Hỗ trợ modular hóa và tái sử dụng: Giúp tách nhỏ các phần giao diện mà không lo trùng lặp style.

Ứng dụng trong chuyên đề

Trong chuyên đề xây dựng website bán vé xem phim, SCSS Modules được sử dụng để thiết kế giao diện cho từng thành phần (component) như phần hiển thị phim, menu điều hướng, form đăng nhập, v.v. Mỗi phần có thể có file .module.scss riêng để đảm bảo tính độc lập và dễ bảo trì. Điều này giúp quá trình phát triển frontend diễn ra thuận tiện, linh hoạt và đảm bảo tính mở rộng cao cho hệ thống.

2.2.5 Context API

Context API là một cơ chế được tích hợp sẵn trong thư viện React từ phiên bản 16.3 trở lên, cho phép truyền dữ liệu giữa các component mà không cần phải sử dụng phương pháp truyền props theo từng cấp (prop drilling). Thay vì truyền dữ liệu qua nhiều lớp component trung gian, Context API cho phép chia sẻ dữ liệu trực tiếp từ một nơi trung tâm đến các component con có nhu cầu sử dụng.

Context API phù hợp để quản lý những dữ liệu cần dùng ở nhiều nơi trong ứng dụng như: thông tin người dùng, ngôn ngữ hiển thị, chủ đề giao diện (theme), trạng thái đăng nhập, giỏ hàng...

Ưu điểm của Context API

Tránh lặp đi lặp lại việc truyền props qua nhiều cấp component.

Quản lý trạng thái đơn giản, đặc biệt đối với ứng dụng nhỏ hoặc vừa.

Tích hợp sẵn trong React, không cần cài thêm thư viện ngoài như Redux.

Tăng tính mô-đun và dễ mở rộng của ứng dụng.

Kết hợp tốt với React Hooks (như useContext) để làm cho mã ngắn gọn và dễ hiểu.

Ứng dụng trong chuyên đề

Trong chuyên đề xây dựng website bán vé xem phim, Context API được sử dụng để:

- Quản lý trạng thái người dùng: Lưu thông tin đăng nhập và quyền truy cập.
- Quản lý trạng thái giỏ hàng: Lưu danh sách vé đã chọn và thông tin thanh toán.
- Quản lý trạng thái toàn cục: Chia sẻ dữ liệu như danh mục phim, lịch sử đặt vé, v.v.

Việc áp dụng Context API giúp giảm độ phức tạp trong mã nguồn và tăng hiệu quả trong quá trình phát triển frontend.

2.2.6 NodeJS

Node.js là một nền tảng chạy JavaScript phía máy chủ (server-side) được xây dựng trên V8 JavaScript Engine của Google. Không giống như JavaScript trong trình duyệt (chỉ chạy ở client), Node.js cho phép lập trình viên viết các ứng dụng máy chủ bằng JavaScript, đặc biệt phù hợp cho các hệ thống web hiện đại như RESTful API, real-time application và microservices.

Node.js [6] sử dụng mô hình non-blocking I/O (nhập/xuất không chặn) và kiến trúc event-driven (hướng sự kiện), giúp xử lý đồng thời nhiều kết nối một cách hiệu quả, đặc biệt thích hợp với các ứng dụng web cần hiệu suất cao.



Hình 2.5 NodeJs là gì?

Ưu điểm của Node.js

Tốc độ cao nhờ sử dụng V8 engine – biên dịch mã JavaScript sang mã máy.

Non-blocking I/O giúp xử lý nhiều request cùng lúc mà không bị chặn.

Sử dụng JavaScript toàn bộ từ frontend đến backend, giúp đồng bộ ngôn ngữ giữa client và server.

Thư viện phong phú nhờ hệ sinh thái NPM.

Thích hợp cho API hoặc microservice nhờ khả năng mở rộng linh hoạt.

Ứng dụng trong chuyên đề

Trong chuyên đề xây dựng website bán vé xem phim, Node.js đóng vai trò là nền tảng backend chính để:

- Xây dựng RESTful API cho frontend giao tiếp.
- Xử lý các chức năng như đăng ký, đăng nhập, quản lý người dùng, phim và vé.
- Gửi email: Gửi email quên mật khẩu và xác nhận thanh toán.
- Tương tác với cơ sở dữ liệu (MySQL) thông qua Sequelize.
- Quản lý các route API một cách hiệu quả và rõ ràng.
- Node.js giúp hệ thống backend xử lý nhanh, hiệu quả và đồng thời cho phép phát triển nhanh chóng với cùng một ngôn ngữ là JavaScript.

2.2.7 Express.js

Express.js (thường gọi là Express) là một framework web tối giản và linh hoạt dành cho Node.js, được sử dụng để xây dựng các ứng dụng web và API nhanh chóng, hiệu quả. Đây là một trong những framework phổ biến nhất trong hệ sinh thái Node.js và là nền tảng quan trọng trong mô hình phát triển ME(N)N Stack (MongoDB, Express, Node, và thường kết hợp với React/Angular/Vue ở phía frontend).

Express cung cấp các công cụ mạnh mẽ để định nghĩa route, xử lý middleware, tương tác với HTTP request và response, cũng như tích hợp dễ dàng với các cơ sở dữ liệu và công cụ xác thực.

Ưu điểm của Express.js

Cấu trúc đơn giản, rõ ràng: Dễ tiếp cận đối với cả người mới và lập trình viên có kinh nghiệm.

Xử lý route linh hoạt: Hỗ trợ các phương thức HTTP (GET, POST, PUT, DELETE...) và route động.

Middleware mạnh mẽ: Cho phép xử lý tuần tự các yêu cầu qua nhiều bước, như kiểm tra xác thực, ghi log, xử lý lỗi...

Tương thích tốt với Node.js: Tận dụng triệt để ưu điểm non-blocking và event-driven của Node.js.

Cộng đồng lớn, tài liệu phong phú: Dễ tìm kiếm giải pháp và mở rộng chức năng.

Ứng dụng trong chuyên đề

Trong chuyên đề xây dựng website bán vé xem phim, Express.js được sử dụng để phát triển hệ thống RESTful API kết nối giữa frontend và backend. Một số chức năng cụ thể:

- Định nghĩa các API để quản lý người dùng, phim, danh mục, đặt vé,...
- Kiểm tra quyền truy cập của người dùng (qua middleware xác thực JWT).
- Kết nối cơ sở dữ liệu MySQL để truy xuất và cập nhật dữ liệu.
- Xử lý upload ảnh phim, phân trang danh sách,...
- Express giúp backend trở nên dễ tổ chức, dễ mở rộng và đáp ứng nhanh với các yêu cầu từ client.

2.3 Kết nối Frontend – Backend

Trong mô hình phát triển ứng dụng web hiện đại, Frontend và Backend là hai phần hoạt động độc lập nhưng liên kết chặt chẽ với nhau. Frontend là giao diện người dùng (UI) được xây dựng bằng các công nghệ như HTML, CSS, JavaScript (thường là React, Angular, hoặc Vue). Backend là nơi xử lý logic nghiệp vụ, kết nối với cơ sở dữ liệu và cung cấp API cho frontend tương tác.

Kết nối Frontend – Backend là quá trình giao tiếp giữa client (trình duyệt người dùng) và server (máy chủ backend), thường thông qua các API (Application Programming Interface), đặc biệt là RESTful API hoặc GraphQL API.

Cơ chế kết nối phổ biến

Giao tiếp qua HTTP request: Frontend gửi yêu cầu HTTP (GET, POST, PUT, DELETE) đến backend để truy xuất hoặc thao tác dữ liệu.

Dữ liệu trao đổi theo định dạng JSON: Backend trả dữ liệu dưới dạng JSON để frontend dễ dàng xử lý.

Axios hoặc Fetch API thường được dùng ở frontend để gửi request.

CORS phải được cấu hình ở backend nếu frontend và backend chạy ở hai domain/port khác nhau.

Vai trò trong dự án chuyên đề

Trong dự án xây dựng website bán vé xem phim, việc kết nối frontend và backend đóng vai trò thiết yếu:

- Giao tiếp động trao đổi dữ liệu: Frontend gửi yêu cầu (request) đến backend để lấy dữ liệu hoặc thực hiện các thao tác (ví dụ: đăng nhập, đặt vé, lấy danh sách phim). Backend xử lý yêu cầu, tương tác với cơ sở dữ liệu, và trả về phản hồi (response) cho frontend.
- Xác thực người dùng: Frontend nhận token từ backend sau khi đăng nhập và lưu trữ để sử dụng cho các request sau.
- Hiển thị lỗi và thông báo: Backend gửi phản hồi thành công/thất bại, giúp frontend thông báo rõ ràng cho người dùng.

Công cụ và công nghệ sử dụng

Frontend: React.js kết hợp với thư viện Axios để gửi HTTP requests.

Backend: Express.js xây dựng hệ thống API RESTful.

Token (JWT): Dùng để xác thực và phân quyền giữa frontend và backend.

CORS Middleware: Được sử dụng tại backend để cho phép frontend truy cập từ domain khác.

2.4 Cơ sở dữ liệu MySQL

MySQL [7] là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) mã nguồn mở rất phổ biến, sử dụng ngôn ngữ SQL (Structured Query Language) để quản lý, truy vấn và thao tác dữ liệu. Nó được phát triển ban đầu bởi công ty MySQL AB và hiện thuộc sở hữu của Oracle Corporation.

MySQL được sử dụng rộng rãi trong các ứng dụng web, hệ thống quản lý nội dung (CMS), thương mại điện tử, và các nền tảng có quy mô từ nhỏ đến lớn. Nó hoạt

động ổn định, có hiệu suất cao và hỗ trợ tốt khi kết hợp với các ngôn ngữ như PHP, Python, Node.js...



Hình 2.6 MySQL là gì?

Đặc điểm nổi bật

- Dữ liệu được lưu trữ dưới dạng bảng (table) với các cột và dòng, tuân thủ cấu trúc quan hệ.
- Hỗ trợ chuẩn SQL với đầy đủ các thao tác CRUD (Create, Read, Update, Delete).
- Hỗ trợ nhiều công cụ và giao diện trực quan như MySQL Workbench, phpMyAdmin giúp dễ quản lý.
- Tính bảo mật cao, hỗ trợ phân quyền người dùng và mã hóa dữ liệu.
- Tương thích đa nền tảng, hoạt động trên cả Windows, macOS, Linux.

Vai trò của MySQL trong dự án

- Lưu trữ dữ liệu: Người dùng, phim, danh mục, đặt vé, lịch sử giao dịch, tin nhắn.
- Quản lý dữ liệu: Thêm, sửa, xóa, và truy vấn dữ liệu.
- Tương tác với backend: Kết nối thông qua Sequelize ORM.
- Xử lý logic: Xác thực, hiển thị danh sách phim, quản lý đặt vé.
- Bảo mật dữ liệu: Mã hóa mật khẩu và phân quyền truy cập.

CHƯƠNG 3. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

3.1 Mô tả và phân tích dự án

3.1.1 Đặc tả dự án

Trong bối cảnh chuyển đổi số mạnh mẽ hiện nay, nhu cầu giải trí của người dân ngày càng tăng, kéo theo yêu cầu số hóa các hoạt động quản lý trong lĩnh vực rạp chiếu phim. Hình thức mua vé truyền thống như xếp hàng tại quầy, ghi nhận đặt vé thủ công hoặc quản lý suất chiếu, ghế ngồi và doanh thu bằng sổ sách hay bảng tính đơn giản không chỉ tốn thời gian mà còn dễ dẫn đến sai sót, trùng vé, khó kiểm soát số lượng ghế và hiệu quả kinh doanh tổng thể. Điều này ảnh hưởng trực tiếp đến trải nghiệm của khách hàng cũng như khả năng vận hành của rạp.

Đề tài “Xây dựng hệ thống quản lý đặt vé xem phim nền tảng ReactJs và NodeJs” được thực hiện nhằm khắc phục những hạn chế nêu trên bằng cách phát triển một hệ thống đặt vé thông minh, trực quan và hiệu quả, giúp số hóa toàn bộ quy trình từ xem thông tin phim, lựa chọn suất chiếu, chọn ghế, thanh toán cho đến quản lý vé và thống kê doanh thu. Hệ thống hướng tới việc mang lại trải nghiệm thuận tiện cho khách hàng, đồng thời hỗ trợ rạp chiếu phim quản lý hoạt động một cách khoa học, thống nhất và minh bạch.

Hệ thống cho phép:

- Khách hàng có thể xem danh sách phim đang chiếu/sắp chiếu, xem thông tin chi tiết phim, lựa chọn suất chiếu, chọn ghế trực quan trên sơ đồ phòng chiếu, thanh toán trực tuyến (ví điện tử, thẻ ngân hàng, v.v.), nhận vé điện tử về email cá nhân, theo dõi lịch sử đặt vé và có AI chatbot hỗ trợ khách hàng.
- Admin hệ thống quản lý tài khoản và phân quyền người dùng (khách hàng, admin quản lý rạp), giám sát hoạt động của hệ thống, quản lý phim, doanh thu, vé và theo dõi log hệ thống hỗ trợ xử lý sự cố kỹ thuật nhằm đảm bảo hệ thống vận hành ổn định và an toàn.

Hệ thống được xây dựng dựa trên nền tảng website với công nghệ hiện đại ReactJs xây dựng giao diện người dùng UI hiện đại trực quan, Axios gửi yêu cầu kết nối với backend, React Router điều hướng giữa các trang như trang chủ, chi tiết phim, đặt vé.... Về backend có NodeJs kết hợp Express Js xây dựng hệ thống RESTFul API để xử lý logic, Socket.IO đảm bảo tính thời gian thực như trạng thái ghế ngồi, Sequelize ORM tương tác với cơ sở dữ liệu MySQL.

3.1.2 Phân tích dự án

Hệ thống quản lý đặt vé xem phim được thiết kế phục vụ 2 nhóm đối tượng chính là khách hàng và quản lý/admin. Mỗi đối tượng có vai trò, chức năng và nghiệp vụ riêng biệt đảm bảo cho quá trình đặt vé – thanh toán – quản lý hệ thống được vận hành một cách hiệu quả và đồng bộ.

Đối tượng khách hàng

Vai trò: là người sử dụng dịch vụ, có thể xem thông tin phim, đặt vé và thanh toán online thông qua website

Các chức năng chính:

- Chức năng Đăng ký/Đăng nhập với tài khoản người dùng để sử dụng chức năng của người dùng bao gồm đặt vé, theo dõi lịch sử đặt vé, thông tin cá nhân và thanh toán.
- Chức năng đặt vé sau khi xem thông tin phim thì khách hàng thực hiện chọn suất chiếu là ngày chiếu và giờ chiếu sau đó chọn ghế thích hợp.
- Chức năng đánh giá bình luận phim sau khi đặt vé thành công thì người dùng có thể đánh giá và bình luận cho bộ phim đó.
- Chức năng thanh toán khách hàng thanh toán qua 2 cổng thanh toán online được tích hợp ở website là Momo và Vnpay.
- Nhận thông tin vé đã đặt qua email và xem lịch sử vé đã đặt ở hồ sơ thông tin cá nhân.
- ChatbotAI hỗ trợ thêm cho khách hàng về thông tin phim trong quá trình đặt vé.

Đối tượng quản lý admin

Vai trò: là người quản lý, giám sát và điều hành toàn bộ hoạt động của hệ thống đặt vé.

Các chức năng chính:

- Chức năng đăng ký/đăng nhập với tài khoản admin để sử dụng các chức năng của admin.
- Quản lý phim: thêm, sửa, xóa các bộ phim.
- Quản lý danh mục: thêm, sửa, xóa các danh mục phim mới hoặc cũ.

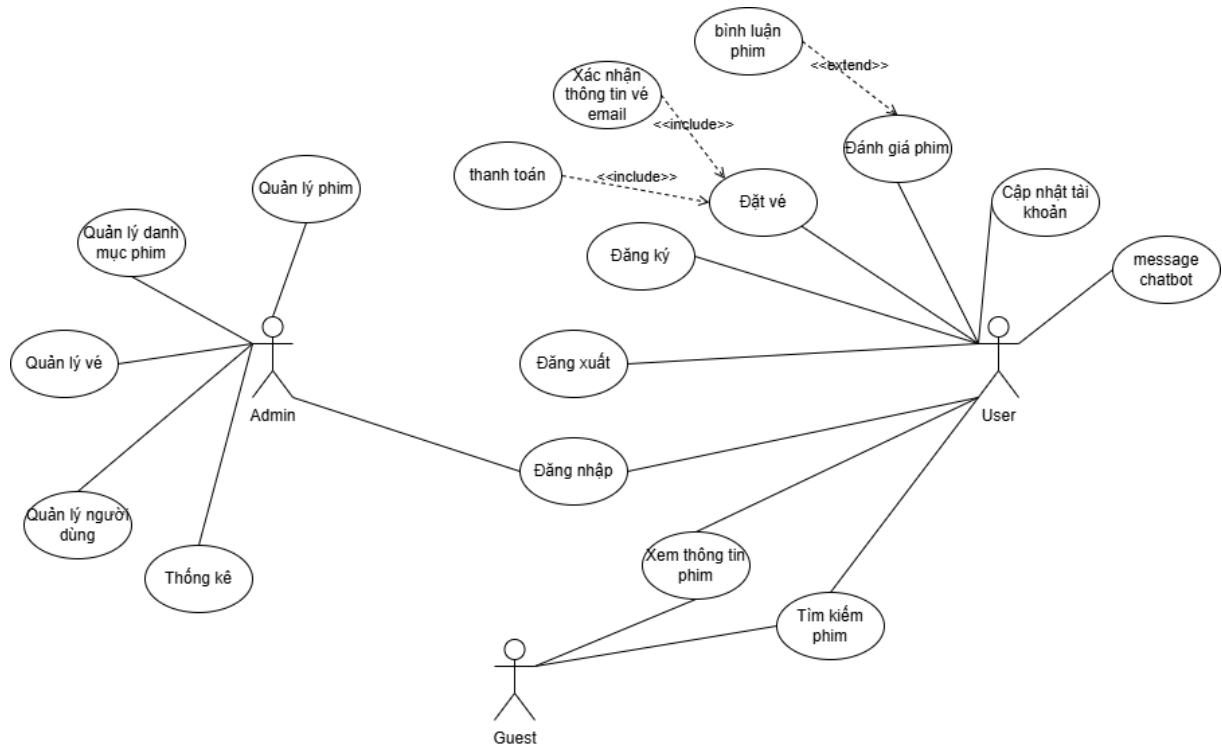
- Theo dõi hoạt động hệ thống: Xem doanh thu, thống kê số lượng người dùng, lượt vé bán ra, thống kê số lượng phim ở trên hệ thống.
- Quản lý vé: thống kê các trạng thái vé bán ra(chờ xử lý/thành công/thất bại), xem chi tiết thông tin vé bán.
- Quản lý người dùng: Phân quyền các tài khoản user hay admin.

3.2 Phân tích hệ thống

3.2.1 Danh sách tác nhân và sơ đồ usecase tổng quát

STT	Tác nhân	Mô tả
1	Guest	Đây là tác nhân đại diện cho những khách hàng truy cập website chưa có tài khoản đăng nhập. Khách hàng có thể tìm kiếm phim, xem thông tin phim
2	User	Đây là tác nhân đại diện cho những khách hàng đã có tài khoản đăng nhập. Khi là thành viên, khách hàng có thể thực hiện đầy đủ các tính năng của website như tìm kiếm, xem thông tin phim, đặt vé, thanh toán trực tuyến, theo dõi tình trạng vé cùng với chatbot h. Thành viên cũng có thể cập nhật thông tin cá nhân, xem lịch sử đặt vé.
3	Admin	Đây là tác nhân đại diện cho người quản trị hệ thống. Khi truy cập vào trang quản trị, Admin có thể xem các báo cáo thống kê, quản lý thông tin phim, danh mục, các đơn hàng, mã giảm giá và tài khoản người dùng.

Bảng 3.1 Bảng danh sách tác nhân



Hình 3.1 Sơ đồ usecase tổng quát

3.2.2 Đặc tả usecase

Tên Usecase	Đặt vé xem phim
Mô tả	Chức năng cho phép User (khách hàng) lựa chọn phim, suất chiếu, ghế ngồi và thanh toán trực tuyến để đặt vé xem phim. Sau khi đặt thành công, hệ thống gửi email xác nhận thông tin vé.
Actor chính	Khách hàng.
Tiền điều kiện	User đã đăng ký và đăng nhập vào hệ thống. Thông tin phim, suất chiếu và sơ đồ ghế đã được thêm vào hệ thống.
Hậu điều kiện	Thông tin vé (phim, suất chiếu, ghế, giá tiền, mã vé...) được lưu vào CSDL. Số lượng ghế tương ứng được cập nhật trạng thái đã đặt. Hệ thống đã thực hiện use case “Thanh toán” và “Xác nhận thông tin vé qua email”.
Luồng hoạt động	B1: User chọn phim muốn đặt từ trang chủ (sau khi xem thông tin chi tiết của phim). B2: Hệ thống hiển thị danh sách suất chiếu bao gồm (ngày, giờ, ghế).

	B3: User chọn suất chiếu và ghế theo mong muốn. Hệ thống kiểm tra ghế có còn trống, hiển thị thông tin vé(ngày, giờ, số ghế chọn, số tiền vé, tổng tiền). B4: User chọn phương thức thanh toán (Momo, Vnpay). B5: Hệ thống thực hiện thanh toán nếu thành công hệ thống lưu thông tin vé, cập nhật ghế sang trạng thái đã đặt. B6: Hệ thống thực hiện gửi email xác nhận vé đã đặt. B7: Hệ thống hiển thị thông báo “đặt vé thành công” cùng thông tin vé đã đặt.
Luồng thay thế	Trường hợp 1: Nếu tại bước 4 ghế đã được người khác đặt hệ thống sẽ hiển thị màu (color) sang trạng thái đã đặt và user không được chọn ghế đấy. Trường hợp 2: Nếu tại bước 2 suất chiếu không còn hiệu lực hết giờ chiếu hay suất chiếu bị hủy thì hệ thống cũng tự động báo suất chiếu không còn nữa.

Bảng 3.2 Bảng đặc tả usecase đặt vé

Tên Usecase	Tìm kiếm phim
Mô tả	Cho phép người dùng tìm kiếm phim theo từ khóa (tên phim, mô tả, thể loại, diễn viên, đạo diễn, quốc gia, năm phát hành) thông qua giao diện tìm kiếm. Kết quả tìm kiếm sẽ hiển thị danh sách các phim phù hợp.
Actor chính	Khách hàng, Guest.
Tiền điều kiện	Hệ thống đã có danh sách phim được lưu trữ trong cơ sở dữ liệu. Người dùng đã truy cập vào giao diện có chức năng tìm kiếm
Hậu điều kiện	Danh sách phim phù hợp với từ khóa được hiển thị trên giao diện. Nếu không có phim nào phù hợp, hệ thống hiển thị thông báo "Không tìm thấy phim".
Luồng hoạt động	B1: Người dùng nhập từ khóa vào ô tìm kiếm. B2: Hệ thống gửi yêu cầu đến API backend. B3: Backend xử lý yêu cầu: Truy vấn cơ sở dữ liệu để tìm các phim phù hợp với từ khóa và trả về danh sách phim dưới dạng JSON.

	B4: Frontend nhận kết quả từ API và hiển thị danh sách phim trên giao diện. B5: Người dùng có thể nhấp vào một phim trong danh sách để xem chi tiết.
Luồng thay thế	Trường hợp 1: Không có phim phù hợp: Backend trả về danh sách rỗng, Frontend hiển thị thông báo "Không tìm thấy phim". Trường hợp 2: Lỗi kết nối API: Backend không phản hồi hoặc xảy ra lỗi, Frontend hiển thị thông báo lỗi "Không thể tìm kiếm phim, vui lòng thử lại sau". Trường hợp 3: Từ khóa rỗng: Nếu người dùng không nhập từ khóa, hệ thống không gửi yêu cầu tìm kiếm.

Bảng 3.3 Bảng đặc tả usecase tìm kiếm

Tên Usecase	Quản lý phim
Mô tả	Chức năng cho phép admin quản lý danh sách phim trong hệ thống, bao gồm các thao tác như thêm mới phim, chỉnh sửa thông tin phim, xóa phim, và xem danh sách phim. Chức năng này đảm bảo rằng thông tin về các bộ phim luôn được cập nhật và chính xác.
Actor chính	Admin
Tiền điều kiện	Admin đã đăng nhập vào hệ thống.
Hậu điều kiện	Thông tin phim được cập nhật thành công trong cơ sở dữ liệu. Admin nhận được thông báo xác nhận thao tác thành công.
Luồng hoạt động	B1: Admin truy cập vào trang quản lý phim. B2: Hệ thống hiển thị danh sách các phim hiện có trong hệ thống. B3: Admin thực hiện các thao tác: Thêm mới phim: Nhấn nút "Thêm phim mới" và điền thông tin phim (tên phim, thể loại, đạo diễn, diễn viên, năm phát hành, giá vé, mô tả, ảnh bìa, trailer v.v.). Chỉnh sửa phim: Nhấn nút "Chỉnh sửa" trên một phim cụ thể và cập nhật thông tin cần thiết. Xóa phim: Nhấn nút "Xóa" trên một phim cụ thể để xóa phim khỏi hệ thống. B4: Admin nhấn nút "Lưu thay đổi" hoặc xác nhận thao tác.

	B5: Frontend gửi yêu cầu API đến backend với dữ liệu tương ứng. B6: Backend xác thực thông tin (ví dụ: kiểm tra định dạng dữ liệu, đảm bảo không trùng lặp tên phim, v.v.). B7: Backend thực hiện thao tác (thêm, sửa, xóa) trên cơ sở dữ liệu. B8: Backend trả về phản hồi thành công và frontend hiển thị thông báo: "Thêm phim thành công." "Cập nhật phim thành công." "Xóa phim thành công."
Luồng thay thế	Trường hợp 1: Nếu admin nhập thông tin không hợp lệ (ví dụ: thiếu trường bắt buộc, định dạng không đúng), hệ thống sẽ hiển thị thông báo lỗi tương ứng. Trường hợp 2: Nếu admin xóa phim, hệ thống sẽ yêu cầu xác nhận trước khi thực hiện thao tác xóa. Trường hợp 3: Nếu xảy ra lỗi trong quá trình xử lý (ví dụ: lỗi kết nối cơ sở dữ liệu), hệ thống sẽ hiển thị thông báo lỗi và yêu cầu admin thử lại.

Bảng 3.4 Bảng đặc tả usecase quản lý phim

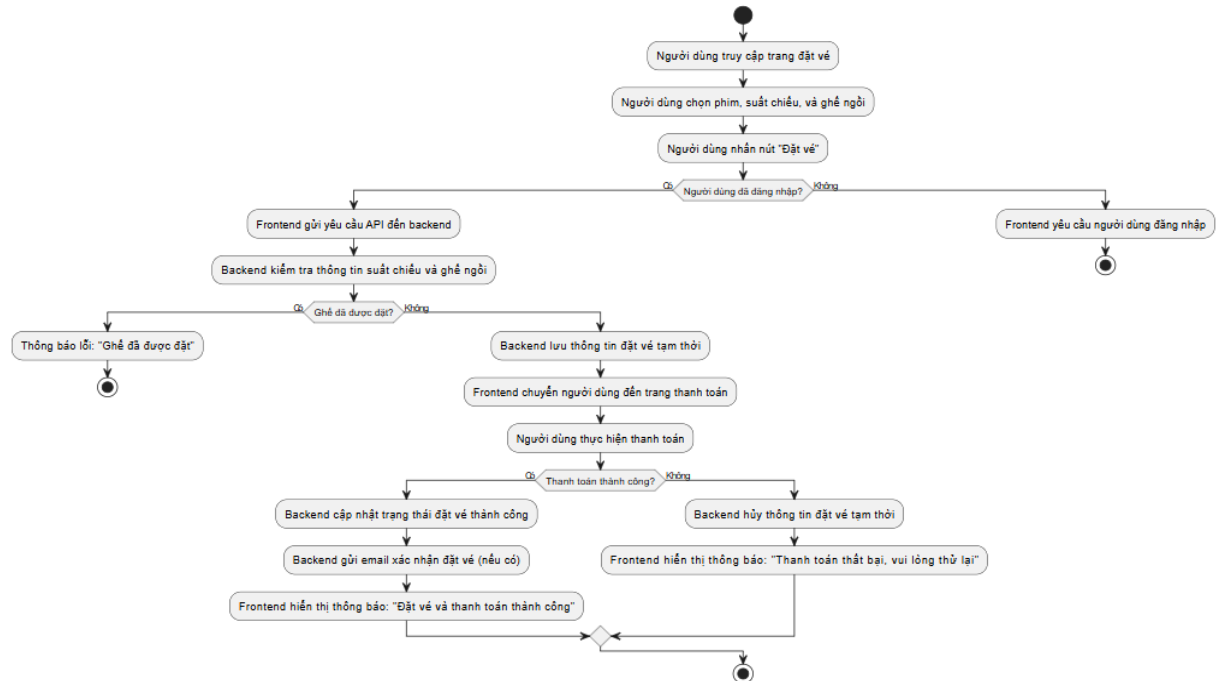
Tên Usecase	Quản lý vé
Mô tả	Chức năng cho phép admin quản lý thông tin vé trong hệ thống, bao gồm các thao tác như xem danh sách vé, chỉnh sửa thông tin vé, hủy vé, và kiểm tra trạng thái vé. Chức năng này đảm bảo rằng thông tin vé được quản lý chính xác và minh bạch.
Actor chính	Admin
Tiền điều kiện	Admin đã đăng nhập vào hệ thống.
Hậu điều kiện	Thông tin vé được cập nhật thành công trong cơ sở dữ liệu. Admin nhận được thông báo xác nhận thao tác thành công.
Luồng hoạt động	B1: Admin truy cập vào trang quản lý vé. B2: Hệ thống hiển thị danh sách các vé hiện có trong hệ thống, bao gồm thông tin chi tiết như: Mã vé, Tên phim, Tên người dùng,

	Số ghế, Thời gian chiếu, Trạng thái vé (đã thanh toán, đang xử lý, đã hủy, v.v.). B3: Admin thực hiện các thao tác: Cập nhật thông tin vé Hủy vé: Nhấn nút "Hủy vé" trên một vé cụ thể để hủy vé. Xem chi tiết vé: Nhấn vào một vé để xem thông tin chi tiết. B4: Admin xác nhận thao tác (nếu cần). B5: Frontend gửi yêu cầu API đến backend với dữ liệu tương ứng. B6: Backend xác thực thông tin (ví dụ: kiểm tra trạng thái vé, đảm bảo vé tồn tại trong hệ thống). B7: Backend thực hiện thao tác (chỉnh sửa, hủy) trên cơ sở dữ liệu. B8: Backend trả về phản hồi thành công và frontend hiển thị thông báo: "Cập nhật vé thành công." "Hủy vé thành công."
Luồng thay thế	Trường hợp 1: Nếu admin nhập thông tin không hợp lệ (ví dụ: mã vé không tồn tại, trạng thái không hợp lệ), hệ thống sẽ hiển thị thông báo lỗi tương ứng. Trường hợp 2: Nếu admin hủy vé, hệ thống sẽ yêu cầu xác nhận trước khi thực hiện thao tác hủy. Trường hợp 3: Nếu xảy ra lỗi trong quá trình xử lý (ví dụ: lỗi kết nối cơ sở dữ liệu), hệ thống sẽ hiển thị thông báo lỗi và yêu cầu admin thử lại.

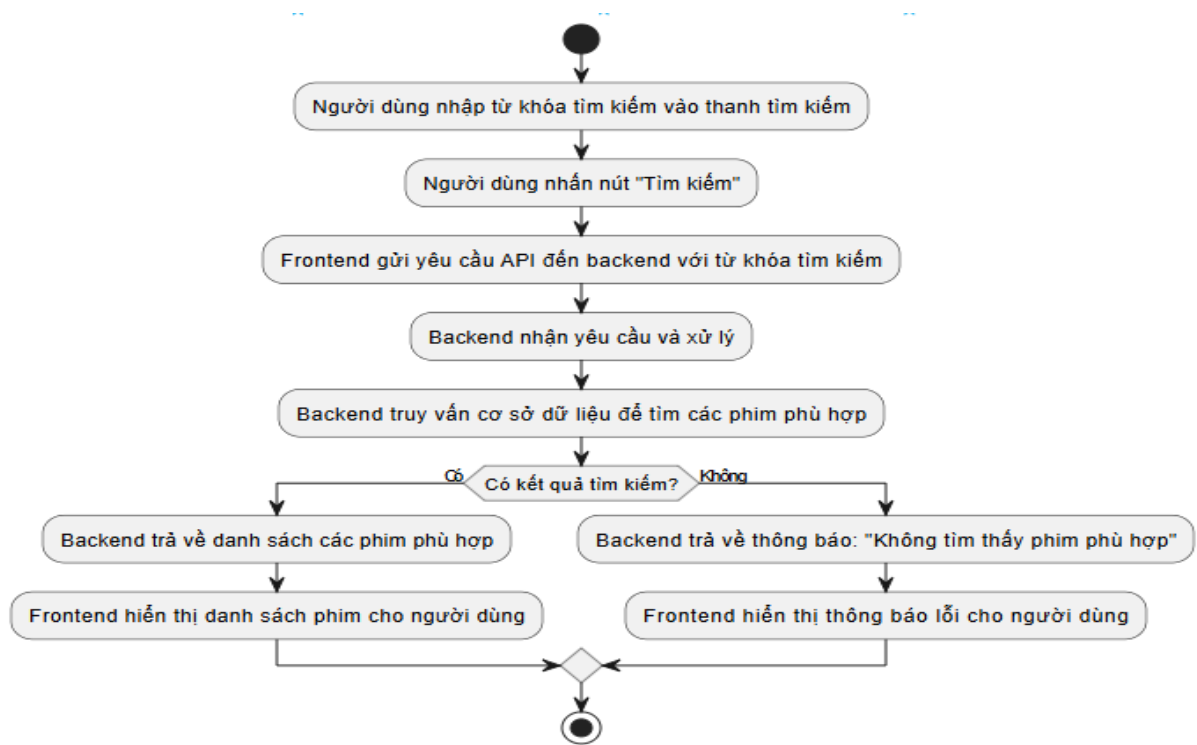
Bảng 3.5 Bảng đặc tả usecase quản lý vé

3.2.3 Sơ đồ hoạt động

Sơ đồ hoạt động chức năng đặt vé trực tuyến

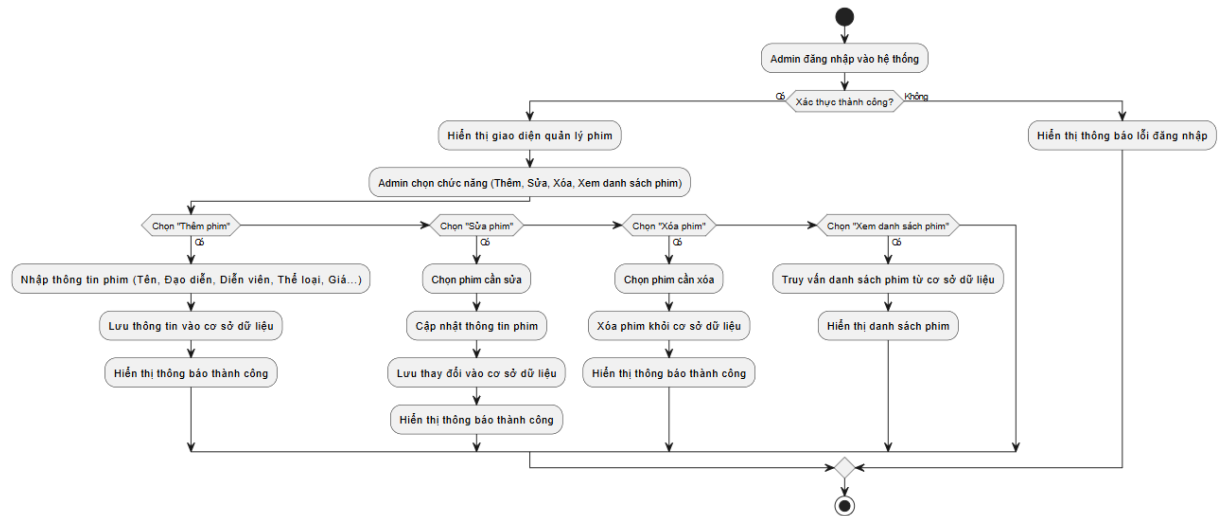


Hình 3.2 Sơ đồ hoạt động chức năng đặt vé
Sơ đồ hoạt động chức năng tìm kiếm phim



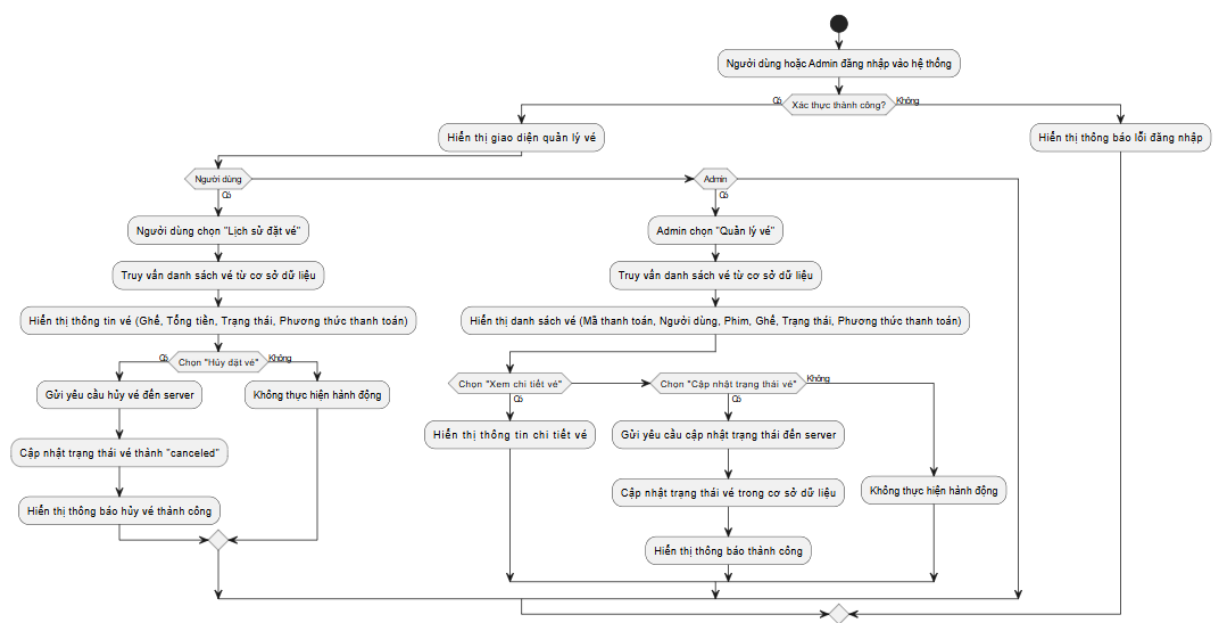
Hình 3.3 Sơ đồ hoạt động chức năng tìm kiếm

Sơ đồ hoạt động chức năng quản lý phim admin



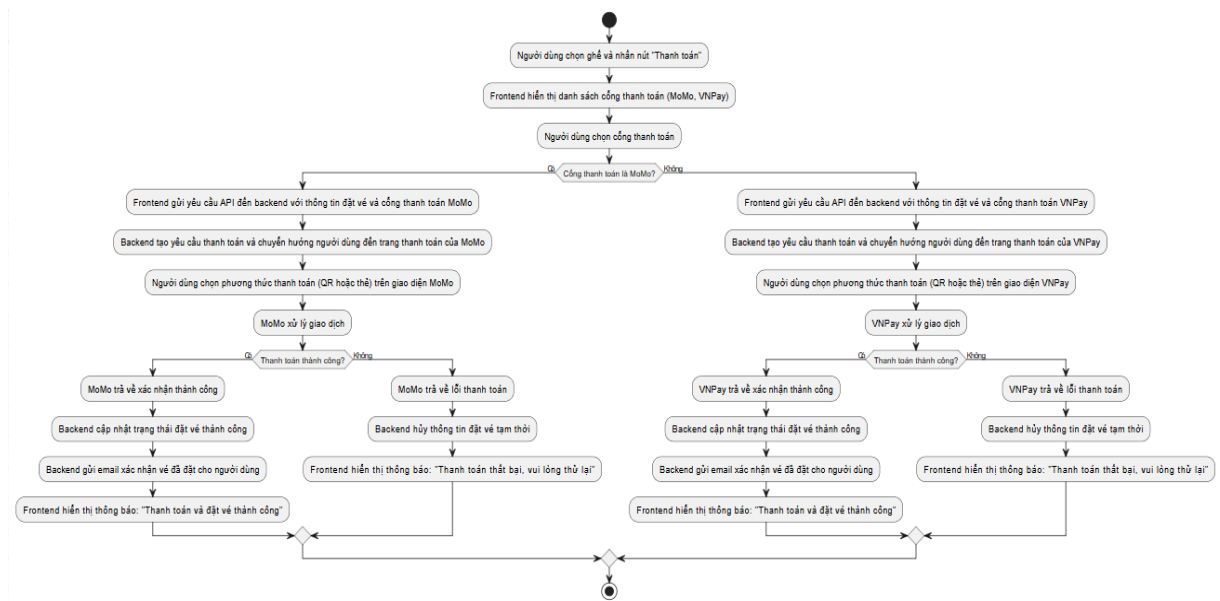
Hình 3.4 Sơ đồ hoạt động chức năng quản lý phim

Sơ đồ hoạt động chức năng quản lý vé



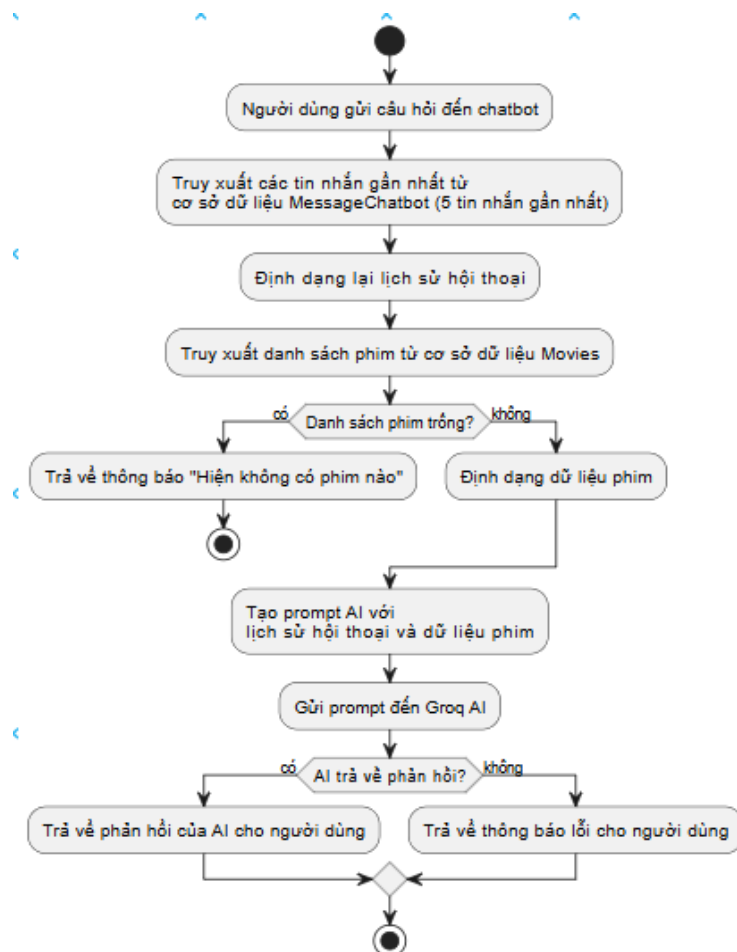
Hình 3.5 Sơ đồ hoạt động chức năng quản lý vé

Sơ đồ hoạt động chức năng thanh toán công điện tử



Hình 3.6 Sơ đồ hoạt động chức năng thanh toán

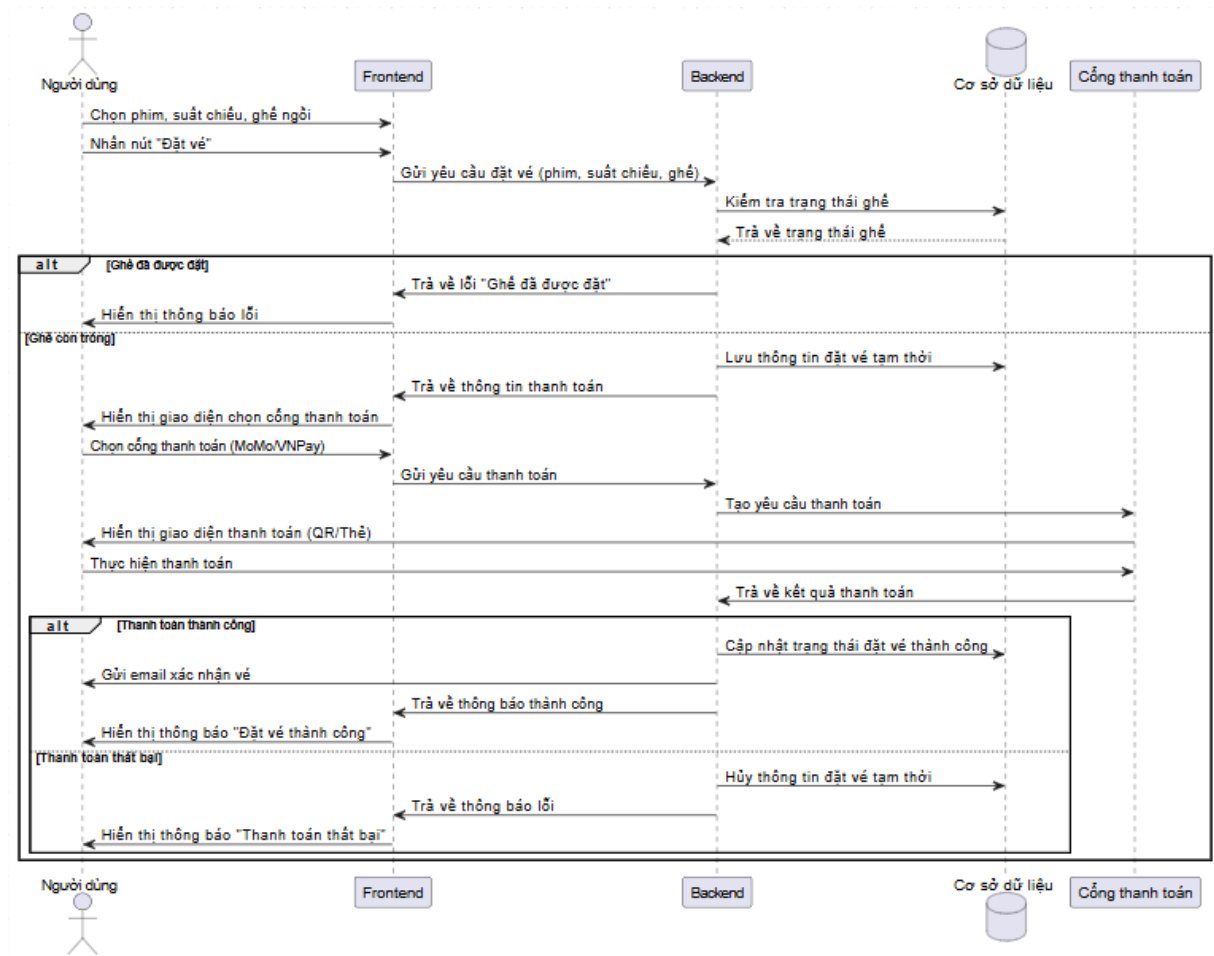
Sơ đồ tuần tự chức năng messagebot



Hình 3.7 Sơ đồ hoạt động chức năng messagebot

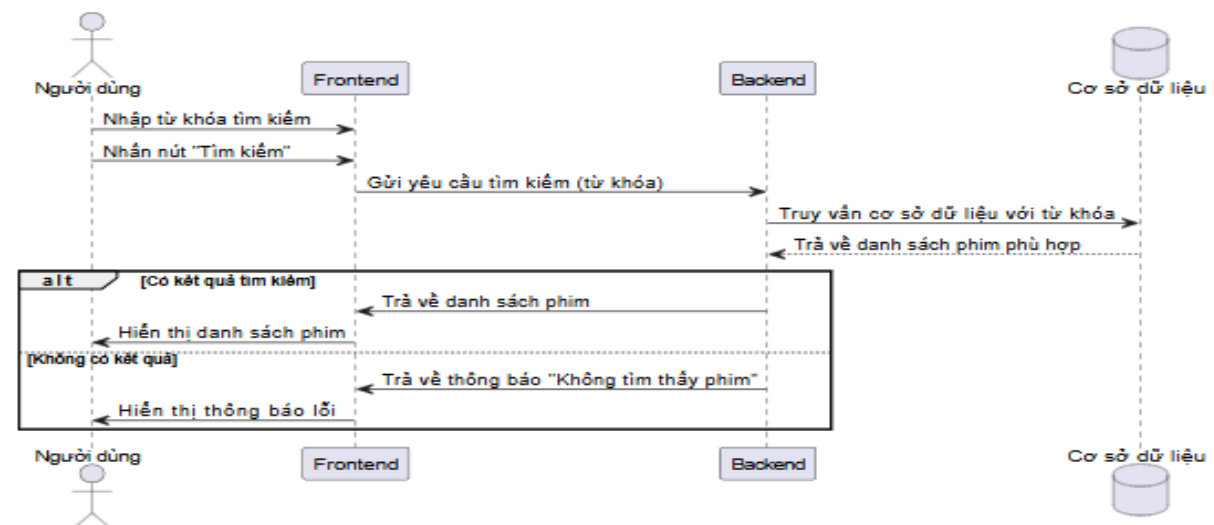
3.2.4 Sơ đồ tuần tự

Sơ đồ tuần tự chức năng đặt vé



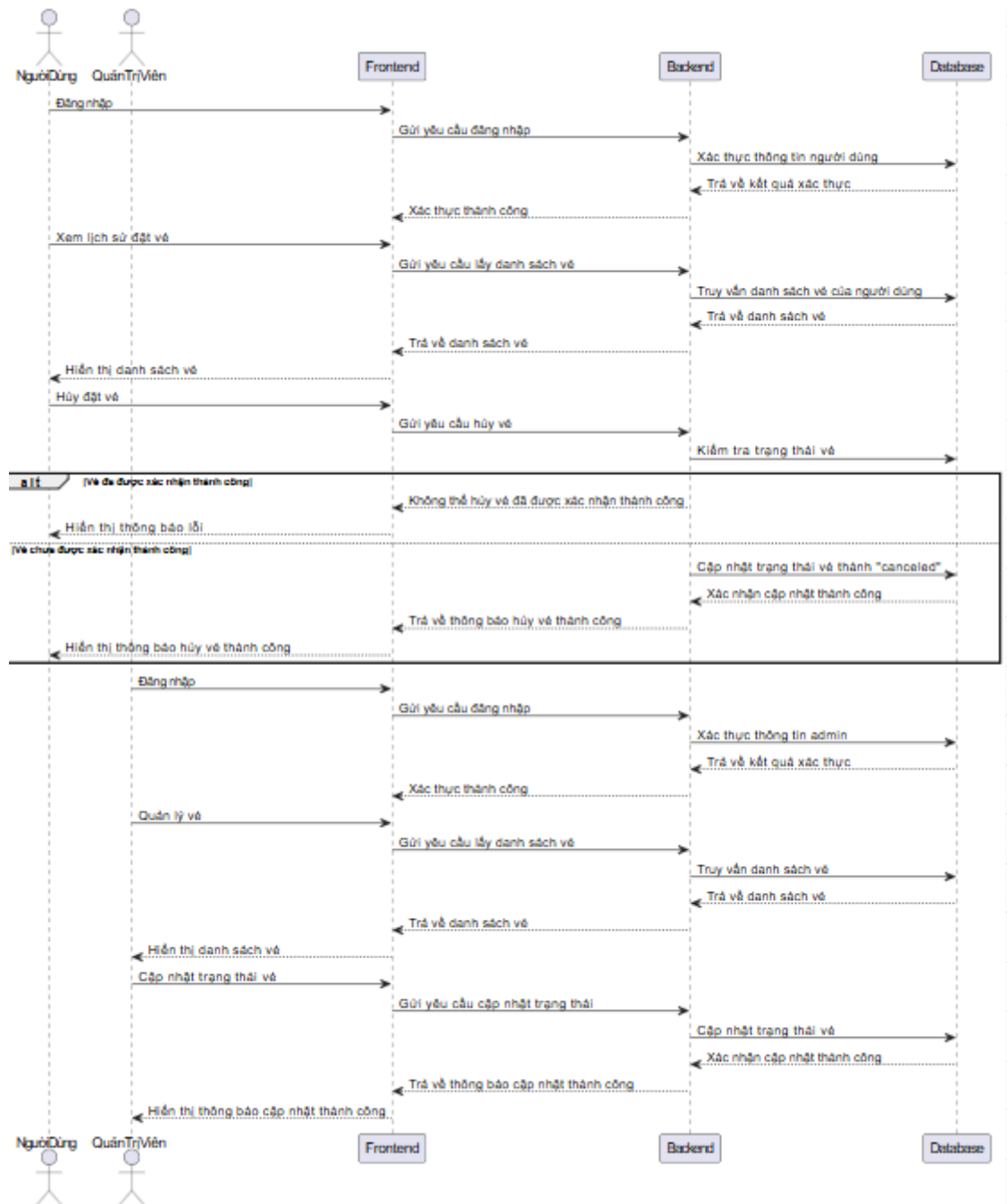
Hình 3.8 Sơ đồ tuần tự chức năng đặt vé

Sơ đồ tuần tự chức năng tìm kiếm



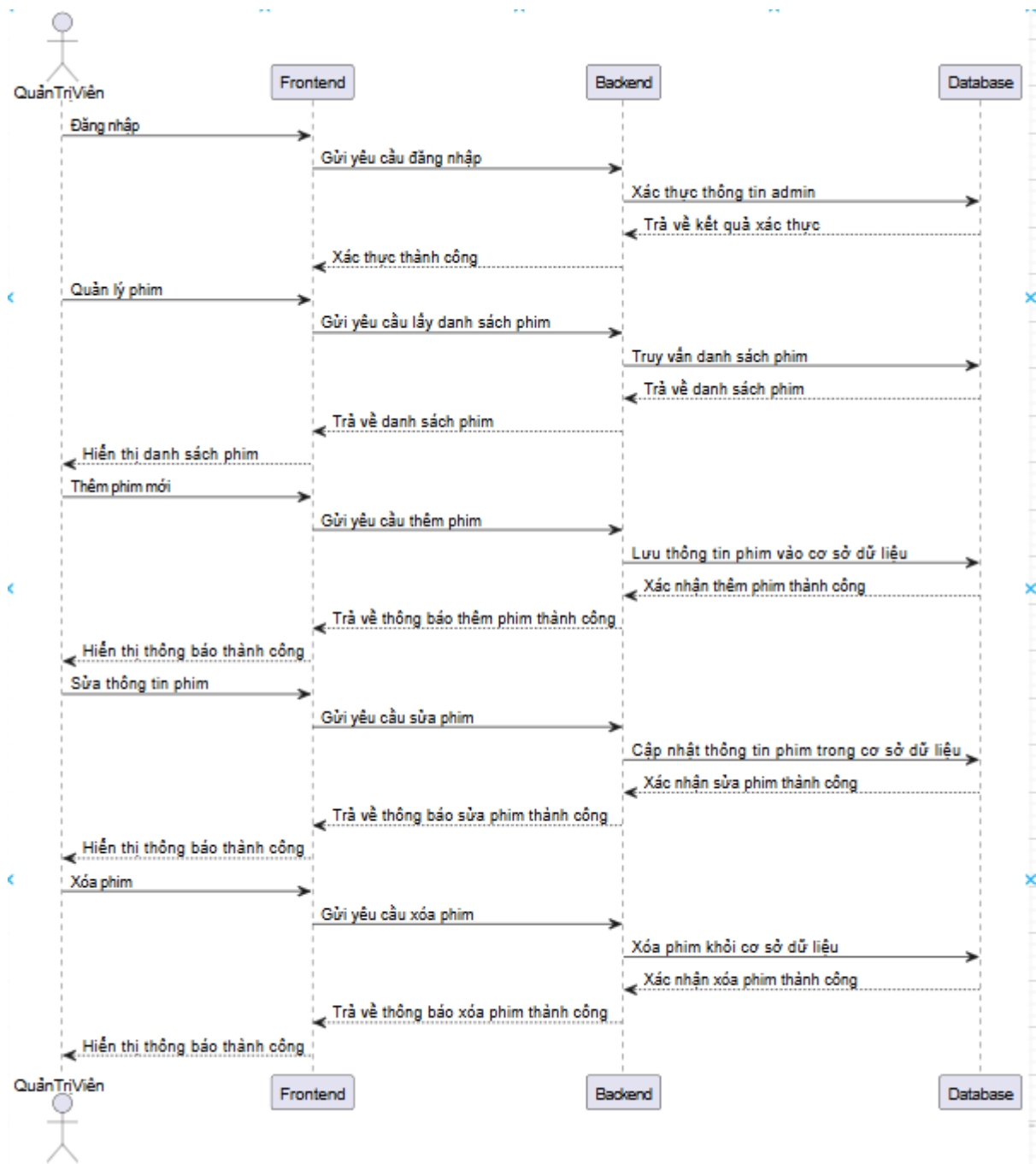
Hình 3.9 Sơ đồ tuần tự chức năng tìm kiếm

Sơ đồ tuần tự chức năng quản lý vé



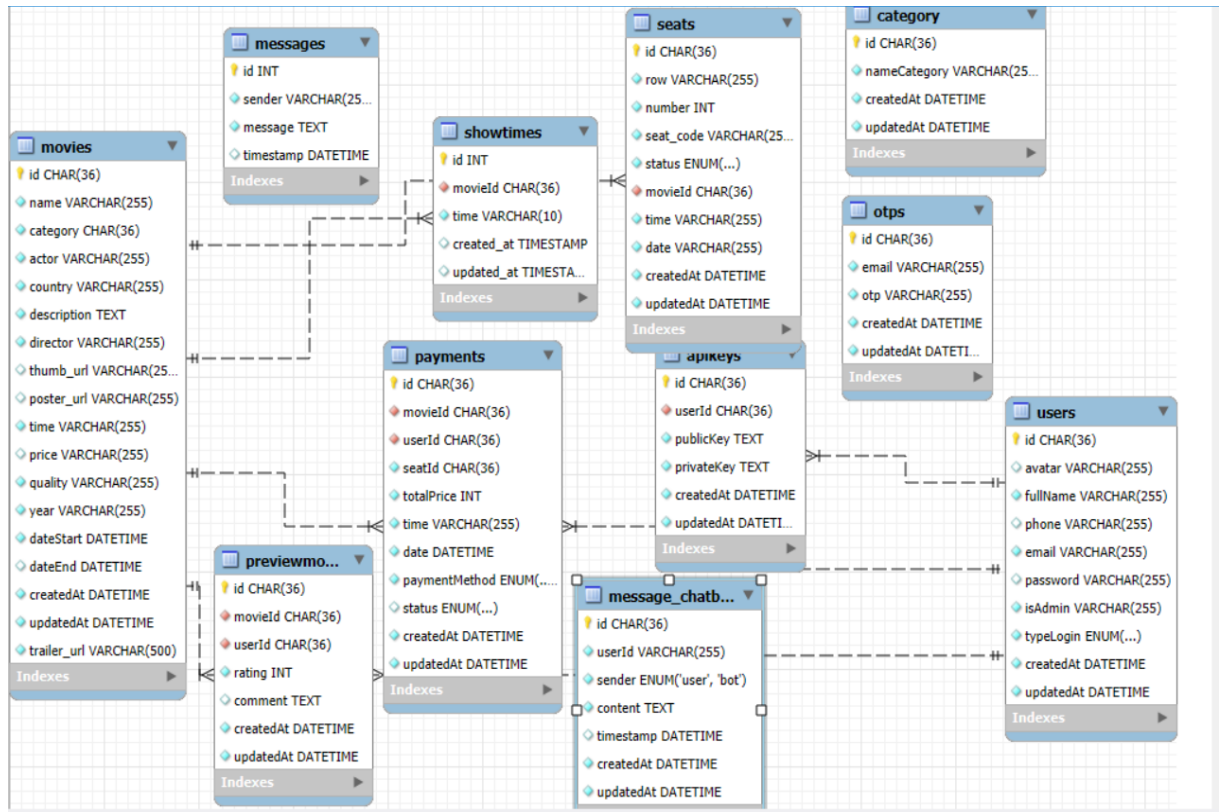
Hình 3.10 Sơ đồ tuần tự quản lý vé

Sơ đồ tuần tự chức năng quản lý phim



Hình 3.11 Sơ đồ tuần tự chức năng quản lý phim

3.2.6 Thiết kế cơ sở dữ liệu MySQL



Hình 3.14 Lược đồ quan hệ

Các bảng trong cơ sở dữ liệu của hệ thống quản lý đặt vé xem phim

id	row	number	seat_code	status	movieId	time	date	createdAt	updatedAt
179f88c0-e7fd-4cbf-9cc1-54e91b7804cc	C	6	C6	booked	6b535437-7c95-43a9-a67a-c7e38af2cbc7	17:30	2025-11-30	2025-11-30 08:21:18	2025-11-30 08:21
1946b8c1-285c-453e-a7a8-96a801175de3	C	5	C5	booked	6b535437-7c95-43a9-a67a-c7e38af2cbc7	17:30	2025-11-30	2025-11-30 08:08:00	2025-11-30 08:08
1cccb2f8-0713-4cff-872b-d34c0633b639	C	6	C6	booked	6b535437-7c95-43a9-a67a-c7e38af2cbc7	15:00	2025-10-13	2025-10-13 05:35:26	2025-10-13 05:35
33486059-0500-4113-8408-8ff4de7b78a0	C	5	C5	booked	6b535437-7c95-43a9-a67a-c7e38af2cbc7	17:30	2025-11-30	2025-11-30 08:14:31	2025-11-30 08:14
3845322d-575f-468e-8acd-b0b76576a778	C	5	C5	booked	6b535437-7c95-43a9-a67a-c7e38af2cbc7	17:30	2025-11-30	2025-11-30 08:11:10	2025-11-30 08:11
4637ee0e-4c0f-4967-a788-c4b6dc406779	C	5	C5	booked	6b535437-7c95-43a9-a67a-c7e38af2cbc7	17:30	2025-11-30	2025-11-30 08:17:53	2025-11-30 08:17
53f9f5b4-425f-4f72-9ca6-1256b4892619	C	5	C5	booked	6b535437-7c95-43a9-a67a-c7e38af2cbc7	17:30	2025-11-30	2025-11-30 08:14:41	2025-11-30 08:14
54f46162-1828-4b7c-9356-7f5869e2f359	C	6	C6	booked	03178c54-cfcf-4a46-a6e8-c66999092362	22:30	2025-11-11	2025-11-11 13:01:36	2025-11-11 13:01
7f070048-e425-494b-8dac-49d077255ae2	B	5	B5	booked	6b535437-7c95-43a9-a67a-c7e38af2cbc7	10:00	2025-10-12	2025-10-11 16:19:56	2025-10-11 16:19
80147b24-165b-4088-8da0-6194e24b2c38	E	5	E5	booked	03178c54-cfcf-4a46-a6e8-c66999092362	22:30	2025-11-13	2025-11-13 13:30:24	2025-11-13 13:30
9de540dc-60a6-4921-b1a5-86502094b480	C	5	C5	booked	6b535437-7c95-43a9-a67a-c7e38af2cbc7	17:30	2025-11-30	2025-11-30 08:08:23	2025-11-30 08:08
b0dd85cb-2ab2-46e9-8290-b4c06f2f245d	B	4	B4	booked	6b535437-7c95-43a9-a67a-c7e38af2cbc7	10:00	2025-10-12	2025-10-11 16:00:53	2025-10-11 16:00

Bảng 3.6 Bảng thông tin ghế được đặt

id	avatar	fullName	phone	email	password	isAdmin	typeLogin	createdAt
095443ff-c2d9-427b-8aab-d02b0ead85b4	NULL	Thao phuong	0916692074	lttp140202@gmail.com	\$2b\$10\$InNCdUr.TTWTrOeU.xJBDeq92UQ2R0...	0	email	2025-10-11 15:
16b1385d-f9ee-488b-b4da-4c5f6455b718	NULL	Minh Phu	0899804328	ttuan113@gmail.com	\$2b\$10\$bUic.Mu9A06MHIERHP2OqGZ0Y2LTrp...	0	email	2025-10-11 12:
346429c4-b067-4561-9583-0195db199ae2	NULL	Phu	NULL	mphct001@gmail.com	\$2b\$10\$PELdgoz5xxD13pfxDXPI2eybrWYxxdB...	1	google	2025-10-11 12:

Bảng 3.7 Bảng thông tin người dùng

id	movieId	userId	seatId	totalPrice	time	date	paymentMethod	status
PAY176019816629506295	03178c54-cfcf-4a46-a6e8-c66999092362	346429c4-b067-4561-9583-0195db199ae2	B3	70000	10:00	2025-10-12 00:00:00	vnmpay	failed
PAY176019845300653006	6b535437-7c95-43a9-a67a-c7e38af2cbc7	095443ff-c2d9-427b-8aab-d02b0ead85b4	B4	70000	10:00	2025-10-12 00:00:00	momo	failed
PAY176019856239442394	6b535437-7c95-43a9-a67a-c7e38af2cbc7	095443ff-c2d9-427b-8aab-d02b0ead85b4	B6	70000	12:30	2025-10-12 00:00:00	vnmpay	failed
PAY176019881344453444	03178c54-cfcf-4a46-a6e8-c66999092362	346429c4-b067-4561-9583-0195db199ae2	B3	70000	10:00	2025-10-12 00:00:00	vnmpay	canceled
PAY176019892428244282	03178c54-cfcf-4a46-a6e8-c66999092362	346429c4-b067-4561-9583-0195db199ae2	B3	70000	10:00	2025-10-12 00:00:00	vnmpay	canceled
PAY176019944999129991	7f6c1757-54bc-4c23-bfeb-1b7dece32ac7	346429c4-b067-4561-9583-0195db199ae2	B4	70000	10:00	2025-10-12 00:00:00	momo	canceled
PAY176019959638956389	6b535437-7c95-43a9-a67a-c7e38af2cbc7	346429c4-b067-4561-9583-0195db199ae2	B5	70000	10:00	2025-10-12 00:00:00	momo	successful
PAY176033372672726727	6b535437-7c95-43a9-a67a-c7e38af2cbc7	346429c4-b067-4561-9583-0195db199ae2	C6	70000	15:00	2025-10-13 00:00:00	momo	successful
PAY176286609618436184	03178c54-cfcf-4a46-a6e8-c66999092362	346429c4-b067-4561-9583-0195db199ae2	C6	70000	22:30	2025-11-11 00:00:00	momo	successful
PAY176304062469024690	03178c54-cfcf-4a46-a6e8-c66999092362	346429c4-b067-4561-9583-0195db199ae2	E5	70000	22:30	2025-11-13 00:00:00	momo	successful
PAY176304111978839788	03178c54-cfcf-4a46-a6e8-c66999092362	346429c4-b067-4561-9583-0195db199ae2	E6	70000	22:30	2025-11-13 00:00:00	momo	pending
PAY176310599684256842	03178c54-cfcf-4a46-a6e8-c66999092362	346429c4-b067-4561-9583-0195db199ae2	E6	70000	15:00	2025-11-14 00:00:00	momo	successful

Bảng 3.8 Bảng thông tin thanh toán

id	name	category	actor	country	description
03178c54-cfcf-4a46-a6e8-c66999092362	TEE YOD: QUỲ ẨM TANG PHẦN 3	0f337553-7999-435a-bfd3-b2e0d91df392	Nadech Kugimiya, Denise Jellicha Kapaun, Mim ...	Thái Lan	Yak và gia đình
18029ce8-e53c-4e30-850b-8aa768587661	CHỊ NGÃ EM NÂNG	d6684b23-1ed6-46e1-8f09-0ee8802a0777	Lê Khánh, Quốc Trường, Thuận Nguyễn, Uyên Ân	Việt Nam	Chị Ngã Em N
58ee27ee-8e9d-49a7-a7c1-39cc2800464b	Tee Yod: Quỳ Ẩn Tang	d6684b23-1ed6-46e1-8f09-0ee8802a0777	Arisara Wongchalee, Denise Jellicha Kapaun, Ja...	Thái Lan	Khi linh hồn k
614375eb-6fd3-4d06-9c20-e8450fed0ed6	Tee Yod: Quỳ Ẩn Tang Phần 2	d6684b23-1ed6-46e1-8f09-0ee8802a0777	Arisara Wongchalee, Denise Jellicha Kapaun, Ka...	Thái Lan	Ba năm sau c
6b535437-7c95-43a9-a67a-c7e38af2cbc7	TỬ CHIẾN TRÊN KHÔNG	0f337553-7999-435a-bfd3-b2e0d91df392	Thái Hòa, Kaity Nguyễn, Thanh Sơn, Xuân Phúc...	Việt Nam	Tử Chiến Trê
7f6c1757-54bc-4c23-bfeb-1b7dece32ac7	MƯA ĐỎ	0f337553-7999-435a-bfd3-b2e0d91df392	Đỗ Nhật Hoàng, Phương Nam, Lâm Thanh Nhã, ...	Việt Nam	"Mưa Đỏ" - Ph
816fcfa2-3198-40bb-b26c-d35b6de60538	TÂY ANH GIỮ MỘT VÍ SAO	d6684b23-1ed6-46e1-8f09-0ee8802a0777	Lee Kwang Soo, Hoàng Hà, Duy Khánh, Cù Thị ...	Hàn Quốc	Siêu sao Kang
b738f528-4fa3-4346-85db-48d93e9334f8	Thám Tử Kiên: Kỳ Ẩn Không Đầu	0f337553-7999-435a-bfd3-b2e0d91df392	Minh Anh, Mỹ Uyên, Ngọc Diệp, Phạm Quỳnh A...	Việt Nam	Một chuỗi án
ec8df868-8349-4404-9679-3f28127fa053	LINH MIẾU	d6684b23-1ed6-46e1-8f09-0ee8802a0777	Hồng Đào, Thiên An, Thuý Tiên, Văn Anh, Sam...	Việt Nam	Linh Miếu: Qu
ee5bd7a9-846a-4782-b8b7-8f6fcc4b7ada	NHÂN DIỄN	d6684b23-1ed6-46e1-8f09-0ee8802a0777	Shin Hyun-been, Park Jeong-min, Han Ji-hyun, ...	Hàn Quốc	Im Dong-hwa

Bảng 3.9 Bảng thông tin phim

id	nameCategory	createdAt	updatedAt
0f337553-7999-435a-bfd3-b2e0d91df392	Phim Nổi Bật Tháng	2025-10-11 12:46:28	2025-10-11 12:46:28
d6684b23-1ed6-46e1-8f09-0ee8802a0777	Phim Đang Chiếu	2025-10-11 12:46:46	2025-10-12 07:11:10

Bảng 3.10 Bảng thông tin danh mục phim

id	movieId	userId	rating	comment
7ef97ad5-e8ed-4d9a-80c1-45f2c8212dc7	03178c54-cfcf-4a46-a6e8-c66999092362	346429c4-b067-4561-9583-0195db199ae2	4	Không hay bằng phần 2

Bảng 3.11 Bảng đánh giá phim

id	email	otp	createdAt	updatedAt
b4eb0420-e646-41c1-a3a5-653e3b6b048a	mphct001@gmail.com	\$2b\$10\$BxXWVo0jPPb9GD8S24z5XuJe7abR28X...	2025-11-14 07:22:40	2025-11-14 07:22:40

Bảng 3.12 Bảng dữ liệu OTP của user

id	userId	sender	content	timestamp	createdAt
03dc63d2-db9d-4826-9dc3-422d09c55c8b	346429c4-b067-4561-9583-0195db199ae2	user	tôi chưa đặt hãy cho tôi đường dẫn tới phim để ...	2025-12-05 05:32:55	2025-12-05 05:32:55
0a47c8b8-3ced-4c0e-858c-ae651f46bc65	346429c4-b067-4561-9583-0195db199ae2	user	cho tôi thông tin phim mưa đỏ	2025-12-05 05:30:47	2025-12-05 05:30:47
0dee4c1e-2baf-43d7-a88e-35d937901294	346429c4-b067-4561-9583-0195db199ae2	bot	Xin chào! Bạn cần tìm phim gì không?	2025-12-05 05:30:03	2025-12-05 05:30:03
1ad54b07-b65c-42ea-ab6f-92de3f6ef074	346429c4-b067-4561-9583-0195db199ae2	bot	Xin chào! Tháng này có một số phim nổi bật như...	2025-12-05 05:30:27	2025-12-05 05:30:27
1fa47454-2f0f-454e-9038-33d421537da8	346429c4-b067-4561-9583-0195db199ae2	user	1 người vào lúc 15h	2025-12-05 05:31:17	2025-12-05 05:31:17
244578c3-5361-49fa-a3ec-3dfcc4haf506	346429c4-b067-4561-9583-0195db199ae2	bot	Để đặt vé cho phim "MƯA ĐỎ", bạn có thể tìm k...	2025-12-05 05:32:55	2025-12-05 05:32:55
2953f9d2-d61e-428f-b0c5-1ad4f8c39a43	346429c4-b067-4561-9583-0195db199ae2	user	tôi muốn đặt vé	2025-12-05 05:31:05	2025-12-05 05:31:05
29b1f8f5-189d-464e-bf32-757e0dd1dc84d	346429c4-b067-4561-9583-0195db199ae2	bot	Để đặt vé, bạn cần chọn một suất chiếu và số l...	2025-12-05 05:31:05	2025-12-05 05:31:05
5ee6e946-4e1a-429a-ac4b-b2b6317111bb	346429c4-b067-4561-9583-0195db199ae2	user	alo	2025-12-05 05:30:03	2025-12-05 05:30:03
8457cb7d-a92b-4955-ac56-2f3de59dcbf9	346429c4-b067-4561-9583-0195db199ae2	user	phim nổi bật tháng này	2025-12-05 05:30:27	2025-12-05 05:30:27
8eb8aa52-f027-4b61-86e4-afd86c139e78	346429c4-b067-4561-9583-0195db199ae2	bot	Bạn muốn đặt vé cho phim "MƯA ĐỎ" với 1 ngư...	2025-12-05 05:31:17	2025-12-05 05:31:17

Bảng 3.13 Bảng dữ liệu messagebot

3.3 Kết luận

Chương 3 đã trình bày quy trình phân tích hệ thống một cách toàn diện nhằm xây dựng nền tảng vững chắc cho việc thiết kế và phát triển hệ thống quản lý đặt vé xem

phim trực tuyến. Trước hết, chương này mô tả bối cảnh, mục tiêu và yêu cầu thực tiễn của dự án, qua đó chỉ ra sự cần thiết của việc số hóa quy trình đặt vé, quản lý suất chiếu, ghế ngồi và hoạt động kinh doanh của rạp chiếu phim. Đồng thời, chương cũng phân tích hai nhóm đối tượng chính của hệ thống — khách hàng và admin quản lý, kèm theo vai trò và các chức năng nghiệp vụ tương ứng của mỗi nhóm.

Tiếp theo, chương đã xây dựng mô hình use case tổng quát giúp xác định các tác nhân tham gia hệ thống và phạm vi chức năng chính như tìm kiếm phim, xem thông tin phim, cập nhật tài khoản, đặt vé và thanh toán trực tuyến. Từng use case quan trọng được đặc tả chi tiết về luồng hoạt động, điều kiện trước – sau và các tình huống thay thế, đảm bảo mô tả rõ ràng nghiệp vụ mà hệ thống cần thực hiện. Bên cạnh đó, chương 3 cũng trình bày các sơ đồ hoạt động và sơ đồ tuần tự cho các chức năng trọng tâm. Những mô hình này giúp mô tả trực quan dòng chảy xử lý, tương tác giữa người dùng – giao diện – backend – cơ sở dữ liệu và các cổng thanh toán. Qua đó, nắm bắt được logic hệ thống, xác định chính xác thứ tự xử lý và các điểm có thể phát sinh lỗi hoặc xung đột trong quá trình đặt vé và thanh toán.

Tổng hợp lại chương 3 đã hoàn thành mục tiêu phân tích nghiệp vụ và yêu cầu hệ thống, tạo tiền đề quan trọng cho việc thiết kế kiến trúc, mô hình dữ liệu và triển khai chức năng toàn diện ở chương thực nghiệm tiếp theo. Đây là bước phân tích nền tảng giúp đảm bảo hệ thống đáp ứng đúng nhu cầu người dùng, hoạt động ổn định và hỗ trợ hiệu quả công tác vận hành của rạp chiếu phim.

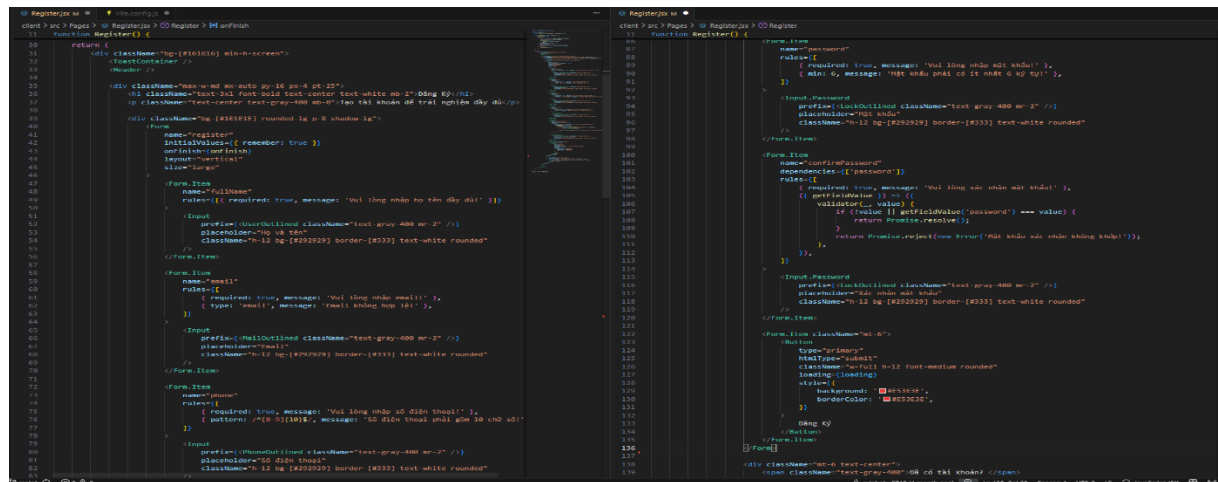
CHƯƠNG 4. TRIỂN KHAI HỆ THỐNG VÀ GIAO DIỆN CHỨC NĂNG

4.1 Module chức năng của hệ thống

4.1.1 Chức năng người dùng

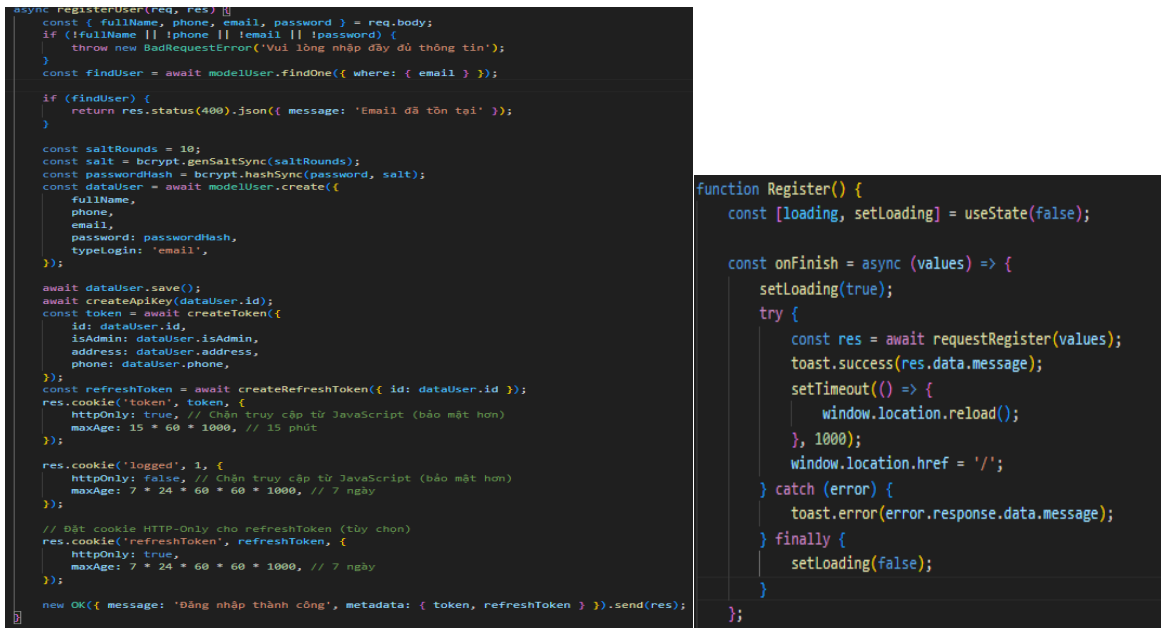
Chức năng đăng ký

Tạo một component Register, form sẽ bao gồm các trường như username, email, phone, password, và nút đăng ký.



Hình 4.1 Form hiển thị thông tin đăng ký

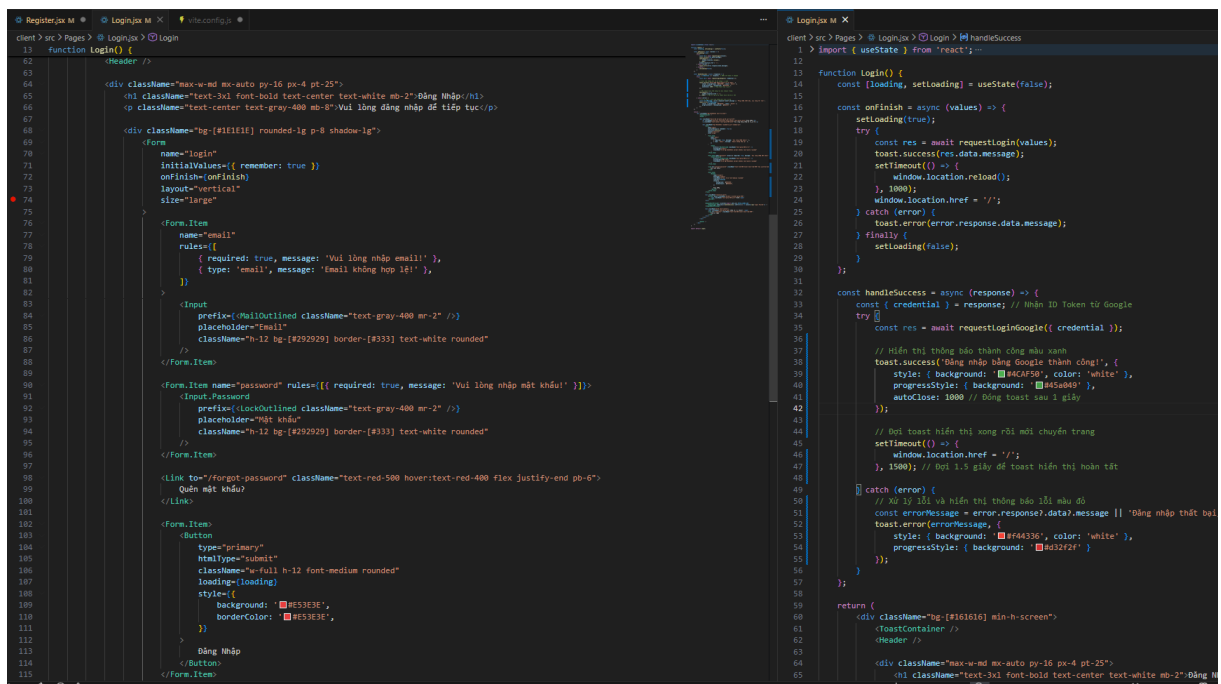
Sau khi người dùng hoàn tất form thì hàm onfinish ở register được gọi ở đây thực hiện đẩy dữ liệu đăng ký tới API backend thông qua hàm requestRegister phương thức post và ở đây router nhận dữ liệu gọi tới hàm registerUser để kiểm tra xử lý logic thông tin đăng ký. Sau đó phản hồi nếu đăng ký thành công hoặc thất bại.



Hình 4.2 Hàm xử lý thông tin form đăng ký

Chức năng đăng nhập

Tạo một component Login, form sẽ bao gồm các trường như email, password và nút đăng nhập.



Hình 4.3 Form đăng nhập

Sau khi người dùng đăng nhập thì hàm onfinish ở login được gọi tới và đẩy dữ liệu đi thông qua requestLogin bằng phương thức post ở đây router nhận dữ liệu và gọi hàm loginUser để kiểm tra xử lý thông tin đăng nhập và trả về kết quả cho frontend.

```
async loginUser(req, res) {
  const { email, password } = req.body;
  if (!email || !password) {
    throw new BadRequestError('Vui lòng nhập đầy đủ thông tin');
  }
  const findUser = await modelUser.findOne({ where: { email } });
  if (!findUser) {
    throw new AuthFailureError('Tài khoản hoặc mật khẩu không chính xác');
  }
  const isPasswordValid = bcrypt.compareSync(password, findUser.password);
  if (!isPasswordValid) {
    throw new AuthFailureError('Tài khoản hoặc mật khẩu không chính xác');
  }
  await createApiKey(findUser.id);
  const token = await createToken({ id: findUser.id, isAdmin: findUser.isAdmin });
  const refreshToken = await createRefreshToken({ id: findUser.id });
  res.cookie('token', token, {
    httpOnly: true, // Chặn truy cập từ JavaScript (bảo mật hơn)
    maxAge: 15 * 60 * 1000, // 15 phút
  });
  res.cookie('logged', 1, {
    httpOnly: false, // Chặn truy cập từ JavaScript (bảo mật hơn)
    maxAge: 7 * 24 * 60 * 60 * 1000, // 7 ngày
  });
  res.cookie('refreshToken', refreshToken, {
    httpOnly: true,
    maxAge: 7 * 24 * 60 * 60 * 1000, // 7 ngày
  });
  new OK({ message: 'Đăng nhập thành công', metadata: { token, refreshToken } }).send(res);
}
```

Hình 4.4 Hàm xử lý thông tin đăng nhập

Chức năng trang chủ

Khi người dùng truy cập trang chủ của hệ thống đặt vé thì ở đây hàm fetchdata được gọi khi component Homepage được render lần đầu tiên sau đó gửi yêu cầu tới API thông qua hàm requestGetAllCategory để lấy danh mục và phim router gọi hàm getAllMovie lấy dữ liệu phim và sau đó dữ liệu trả về lưu vào Categories thông qua setCategories. Tiếp theo duyệt từng danh mục (item) trong categories để hiển thị danh mục phim lên và tương tự duyệt qua danh sách phim thuộc danh mục đó hiển thị phim thông qua component Cardbody.

```
function HomePage() {
  const [categories, setCategories] = useState([]);

  useEffect(() => {
    const fetchData = async () => {
      const resCategories = await requestGetAllCategory();
      setCategories(resCategories.metadata);
    };
    fetchData();
  }, []);

  return (
    <div className="w-[90%] mx-auto">
      {categories.map((item) => (
        <div>
          <div className="flex justify-between items-center text-white mb-10 border-b border-white pb-5">
            <h1 className="text-2xl font-bold">{item.nameCategory}</h1>
            <Link to={`/${category}/${item.id}`}>
              <button className="bg-[#e5091f] rounded-full px-8 py-2 text-white text-sm hover:bg-[#33373b] transition-all duration-300">
                Xem tất cả
              </button>
            </Link>
          </div>
          <div className="grid grid-cols-6 gap-4">
            {item.movies.map((movie) => (
              <Link key={movie.id} to={`/${movie}/${movie.id}`}>
                <Cardbody item={movie} />
              </Link>
            ))}
          </div>
        </div>
      ))}
    </div>
  );
}
```

Hình 4.5 Hàm hiển thị danh mục và phim

```
function Cardbody({ item }) {  
    return (  
        <div className="group relative overflow-hidden rounded-lg transition-transform duration-300">  
            /* Image contains with overlay */  
            <div className="aspect-[2/3] overflow-hidden relative">  
                <img  
                    src={item.poster_url}  
                    alt={item.name}  
                    className="w-full h-full object-cover transition-transform duration-500 group-hover:scale-110"  
                />  
                /* Gradient overlay */  
                <div className="absolute inset-0 bg-gradient-to-t from-black/80 via-black/20 to-transparent opacity-0 group-hover:opacity-100 transition-opacity duration-500"></div>  
                /* Quality Badge */  
                <div className="absolute top-2 right-2">  
                    <span className="bg-red-600 text-white text-xs font-medium py-0.5 px-1.5 rounded">  
                        {item.quality || 'HD'}  
                    </span>  
                </div>  
                /* Hover Info */  
                <div className="absolute bottom-0 left-0 right-0 p-3 transform translate-y-full group-hover:translate-y-0 transition-transform duration-300">  
                    <div className="flex justify-between items-center">  
                        <span className="text-white text-xs font-medium">{item.time || '90'} phút</span>  
                        <span className="bg-yellow-500/90 text-black text-xs font-bold py-0.5 px-1.5 rounded">  
                            {(parseFloat(item.price) || 0).toLocaleString('vi-VN')} đ  
                        </span>  
                    </div>  
                    <div className="bg-red-600/80 mt-2 py-1 px-2 rounded text-center">  
                        <span className="text-white text-xs font-medium">Đặt vé</span>  
                    </div>  
                </div>  
            </div>  
            /* Info section */  
            <div className="mt-2 p-2">  
                <h3 className="text-white text-sm font-semibold text-sm line-clamp-1 group-hover:text-red-500 transition-colors duration-300">  
                    {item.name}  
                </h3>  
                /* Year and country */  
                <div className="flex items-center justify-between mt-1.5">  
                    <span className="text-xs px-1.5 py-0.5 bg-gray-700 text-gray-300 rounded">  
                        {item.year || 'N/A'}  
                    </span>  
                    <span className="text-xs text-gray-400 truncate max-w-[70%]>  
                        {item.country || 'Đang cập nhật'}  
                    </span>  
                </div>  
            </div>  
        </div>  
    )  
}
```

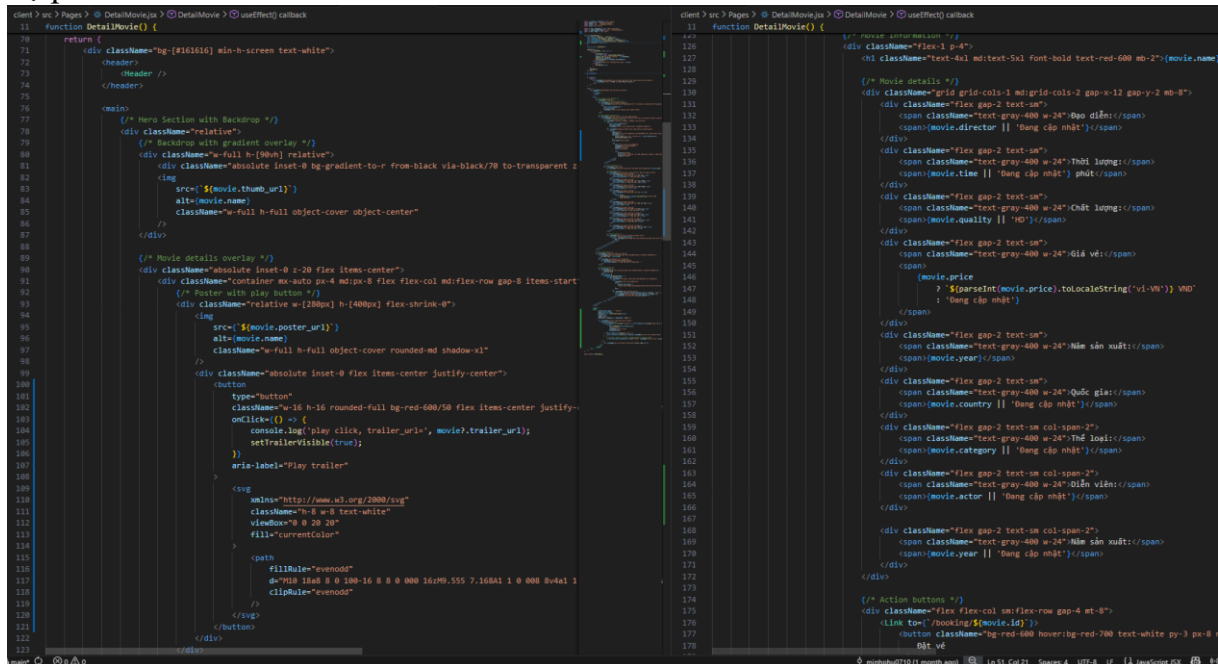
Hình 4.6 Hàm hiển thị thông tin phim

```
async getAllMovie(req, res) {
  const movies = await modelMovie.findAll({});
  const data = await Promise.all(
    movies.map(async (movie) => {
      const findCategory = await modelCategory.findOne({ where: { id: movie?.category } });
      return {
        ...movie.dataValues,
        category: findCategory,
      };
    }),
  );
  new OK({
    message: 'Get all movie successfully',
    metadata: data,
  }).send(res);
}
```

Hình 4.7 Hàm xử lý lấy dữ liệu phim từ cơ sở dữ liệu

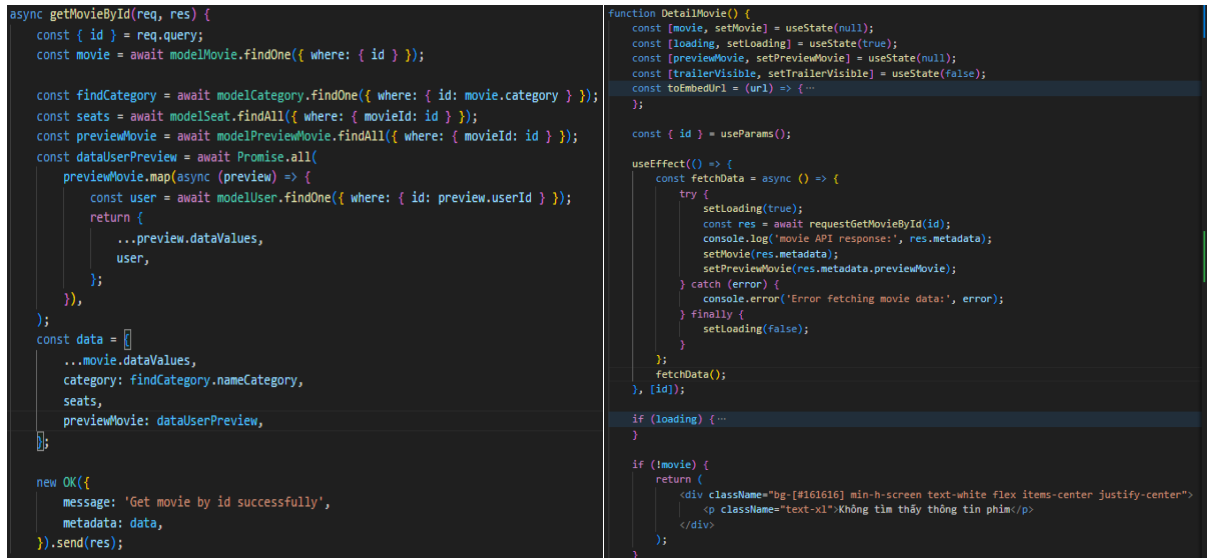
Chức năng hiển thị thông tin phim

Tạo một component DetailMovie trang này sẽ hiển thị thông tin chi tiết của một bộ phim.



Hình 4.8 Form hiển thị thông tin chi tiết phim

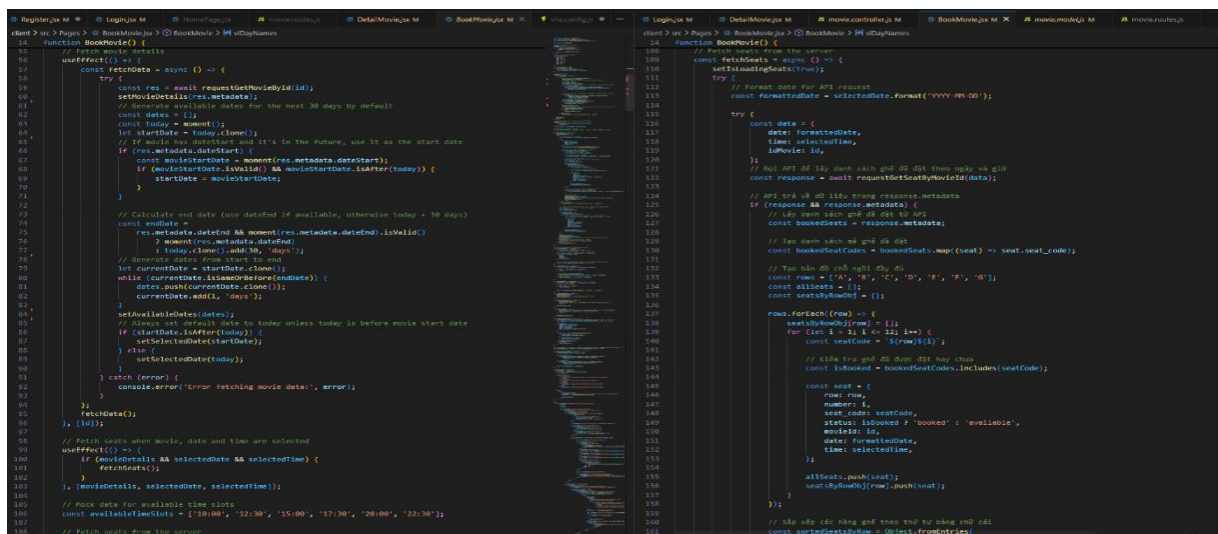
Ở form hàm fetchData được thực thi và hàm requestMovieById gửi yêu cầu tới BE dùng để lấy thông tin chi tiết của phim ở đây BE xử lý gọi tới hàm getMovieById lấy dữ liệu theo ID phim tra lại thông tin chi tiết của phim từ cơ sở dữ liệu sau đó trả về lại cho FE.



Hình 4.9 Hàm lấy và xử lý logic dữ liệu phim

Chức năng đặt vé

Tạo component BookMovie trang này sẽ hiển thị thông tin phim, danh sách ghế và xử lý logic đặt vé bao gồm việc chọn ngày, giờ, ghế ngồi và thanh toán. Đầu tiên thì các trạng thái selectedDate: Ngày chiếu phim được chọn, selectedTime: Suất chiếu phim được chọn, selectedSeats: Danh sách ghế ngồi được chọn, paymentMethod: Phương thức thanh toán (MoMo hoặc VNPAY), movieDetails: Thông tin chi tiết của phim, seats: Danh sách ghế ngồi, seatsByRow: Danh sách ghế ngồi được sắp xếp theo hàng, isLoadingSeats: Trạng thái tải dữ liệu ghế ngồi. Người dùng chọn phim và hàm requestMovieById được gọi gửi yêu cầu tới BE để lấy danh sách ghế theo ngày giờ sau khi kiểm tra hết logic như ghế đó đã được chọn chưa hay giờ chiếu đã qua chưa (dựa trên thời gian hiện tại) thì trả về danh sách ghế đã đặt và xử lý thanh toán.



Hình 4.10 Luồng xử lý đặt vé

```

client > src > Pages > BookMovie.js > BookMovie > useEffect callback > fetchData
14 function BookMovie() {
15   // ...
16   const fetchSeats = async () => {
17     // Sắp xếp các hàng ghế theo thứ tự bảng chữ cái
18     const sortedSeatsByRow = Object.fromEntries(
19       Object.entries(seatsByRow).sort(([,rowA], [,rowB]) => rowA.localeCompare(rowB)),
20     );
21     // Chỉ một state
22     setSeats(allSeats);
23     setSeatsByRow(sortedSeatsByRow);
24     setSelectedSeats([]); // Reset selected seats when changing time/date
25     setisLoadingSeats(false);
26     return;
27   }
28   } catch (error) {
29     console.error('Error fetching seats from API!', error);
30   }
31   // Nếu API thất bại hoặc không có dữ liệu, tạo bản đồ chỗ ngồi mẫu
32   const rows = ['A', 'B', 'C', 'D', 'E', 'F', 'G'];
33   const mockSeats = [];
34   const mockSeatsByRow = {};
35   rows.forEach((row) => {
36     mockSeatsByRow[row] = [];
37     for (let i = 1; i <= 12; i++) {
38       // Chỉ sử dụng trạng thái "available" và "booked" theo API
39       const randomStatus = Math.random() < 0.85 ? 'available' : 'booked';
40       const seat = {
41         row: row,
42         number: i,
43         seat_code: `${row}${i}`,
44         status: randomStatus,
45         movieId: id,
46         date: formattedDate,
47         time: selectedTime,
48       };
49       mockSeats.push(seat);
50       mockSeatsByRow[row].push(seat);
51     }
52   });
53   setSeats(mockSeats);
54   setSeatsByRow(mockSeatsByRow);
55   setSelectedSeats([]); // Reset selected seats when changing time/date
56   } catch (error) {
57     console.error('Error fetching seats!', error);
58     message.error('Không thể tải dữ liệu ghế ngồi!');
59   } finally {
60     setisLoadingSeats(false);
61   }
62 }

```

```

client > src > Pages > BookMovie.js > BookMovie > vDayTimes
14 function BookMovie() {
15   // ...
16   // Check if a time slot is already past for today
17   const isTimeSlotPast = (timeSlot) => {
18     // Only apply time restrictions for today
19     if (selectedDate.isSame(moment(), 'day')) return false;
20     const [hours, minutes] = timeSlot.split(':').map(Number);
21     const slotTime = moment().set({
22       hours,
23       minutes,
24       seconds: 0,
25       milliseconds: 0,
26     });
27     // Add 10 minutes buffer (cannot book less than 10 minutes before showtime)
28     return slotTime.isBefore(moment().add(10, 'minutes'));
29   };
30   const handleSeatClick = (seatCode) => {
31     const seat = seats.find(s => s.seat_code === seatCode);
32     if (!seat || seat.status !== 'available') {
33       return;
34     }
35     if (selectedSeats.includes(seatCode)) {
36       setselectedSeats(selectedSeats.filter(code => code !== seatCode));
37     } else {
38       setselectedSeats([...selectedSeats, seatCode]);
39     }
40   };
41   const getSeatStatus = (seatCode) => {
42     const seat = seats.find(s => s.seat_code === seatCode);
43     if (!seat) return 'booked';
44     if (seat.status === 'booked') {
45       return 'booked';
46     }
47     return selectedSeats.includes(seatCode) ? 'selected' : 'available';
48   };
49   const handleDateChange = (date) => {
50     setselectedDate(date); // Reset time when date changes
51     setselectedSeats([]); // Reset selected seats
52   };
53   const handleTimeSelection = (time) => {
54     if (isTimeSlotPast(time)) {
55       message.error('Suất chiếu này đã bắt đầu. Vui lòng chọn suất chiếu khác!');
56     }
57   };
58 }

```

Hình 4.11 Luồng xử lý đặt vé

```

client > src > Pages > BookMovie.js > BookMovie > DetailMovie.js
14 function BookMovie() {
15   // ...
16   const handleTimeSelection = (time) => {
17     // ...
18     setselectedTime(time);
19   };
20   const handlePayment = () => {
21     if (!selectedDate || !selectedTime || selectedSeats.length === 0 || !paymentMethod) {
22       message.error('Vui lòng chọn đầy đủ thông tin đặt vé!');
23       return;
24     }
25     setIsModalVisible(true);
26   };
27   const handlePaymentConfirm = async () => {
28     const dateOnly = new Date(selectedDate).toISOString().split('T')[0]; // "2023-04-27"
29     const combined = new Date(`${dateOnly}${selectedTime}:00`); // Kết hợp với giờ
30     const data = {
31       type: paymentMethod,
32       idMovie: movieDetails.id,
33       idSeats: selectedSeats,
34       time: selectedTime,
35       date: selectedDate.format('YYYY-MM-DD'),
36     };
37     try {
38       const res = await request(createPayment(data));
39       if (paymentMethod === 'bank') {
40         window.open(res.metadata.payUrl);
41       }
42       if (paymentMethod === 'credit') {
43         window.open(res.metadata.payUrl);
44       }
45       setisModalVisible(false);
46     } catch (error) {
47       console.log(error);
48     }
49   };
50   const totalAmount = selectedSeats.length * (movieDetails?.price ? parseInt(movieDetails.price) : 0);
51   const formattedPrice = `price: toLocaleString('vi-VN') + ' VND';
52   // Format a date display string based on movie dates
53   const getShowTimeDateRange = () => {
54     if (!movieDetails) return 'Đang cập nhật';
55     const startDateTime = moment(movieDetails.dateStart).isValid()
56       ? moment(movieDetails.dateStart).format('DD/MM/YYYY') : 'Hôm nay';
57     const endDateTime =
58       movieDetails.dateEnd && moment(movieDetails.dateEnd).isValid()
59         ? moment(movieDetails.dateEnd).format('DD/MM/YYYY') : 'Hôm nay';
60   };
61 }

```

```

client > src > Pages > BookMovie.js > BookMovie > handleTimeSelection
14 function BookMovie() {
15   // ...
16   const getShowTimeDateRange = () => {
17     // ...
18     return `Start: ${startDateTime} - End: ${endDateTime}`;
19   };
20   // Render seat by seat status
21   const renderSeat = (seat) => {
22     const status = getSeatStatus(seat.seat_code);
23     let colorClass = '';
24     switch (status) {
25       case 'available':
26         colorClass = 'bg-[#d4d4] text-white hover:bg-[#555] cursor-pointer';
27         break;
28       case 'selected':
29         colorClass = 'bg-red-600 text-white cursor-pointer';
30         break;
31       case 'booked':
32         colorClass = 'bg-gray-700 text-gray-500 cursor-not-allowed';
33         break;
34       default:
35         colorClass = 'bg-gray-700 text-gray-500 cursor-not-allowed';
36     }
37     return (
38       <div>
39         {seat.seat_code}
40         {status === 'available' ? 'Available' : status}
41       </div>
42     );
43   };
44   return (
45     <div className="bg-[#161616] min-h-screen text-white">
46       <header>
47         <h1>Movie Details</h1>
48       </header>
49       <main>
50         <div>
51           <div>
52             <div>
53               <div>
54                 <div>
55                   <div>
56                     <div>
57                       <div>
58                         <div>
59                           <div>
60                             <div>
61                               <div>
62                                 <div>
63                                   <div>
64                                     <div>
65                                       <div>
66                                         <div>
67                                           <div>
68                                             <div>
69                                             </div>
70                                           </div>
71                                         </div>
72                                       </div>
73                                     </div>
74                                   </div>
75                                 </div>
76                               </div>
77                             </div>
78                           </div>
79                         </div>
75

```

Hình 4.12 Luồng xử lý thanh toán

```

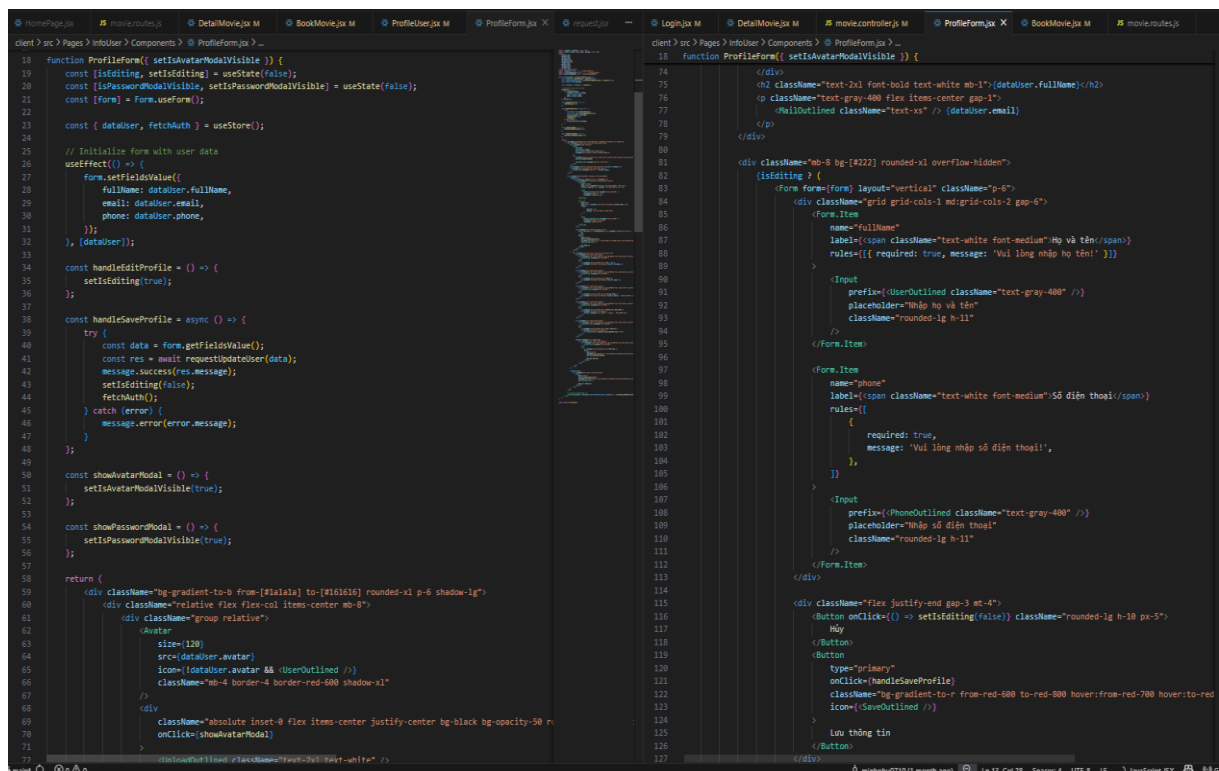
async getSeatByMovieId(req, res) {
  const { date, time, idMovie } = req.query;
  const seats = await modelSeat.findAll({ where: { movieId: idMovie, date, time } });
  new OK({
    message: 'Get seat by movie id successfully',
    metadata: seats,
  }).send(res);
}

```

Hình 4.13 Hàm xử lý lấy danh sách ghế theo phim

Chức năng hiển thị và cập nhật thông tin cá nhân user

Ở đây profileForm làm việc với chức năng hiển thị và chỉnh sửa thông tin cá nhân của người dùng bao gồm các hàm xử lý handleEditProfile: Kích hoạt chế độ chỉnh sửa thông tin, handleSaveProfile: Lưu thông tin đã chỉnh sửa. Gửi dữ liệu lên backend thông qua hàm requestUpdateUser, showAvatarModal: Hiển thị modal để thay đổi ảnh đại diện, showPasswordModal: Hiển thị modal để thay đổi mật khẩu. Đầu tiên sau khi đăng ký và đăng nhập xong thì hàm handleSaveProfile được gọi và thu thập dữ liệu từ form user đã đăng ký hay đăng nhập trước đó khi người dùng thực hiện chỉnh sửa thông tin thì FE sẽ gửi API đến BE lúc này hàm requestUpdateUser được gọi để đưa dữ liệu về BE controller và sẽ xử lý cập nhật thông tin ở đây sau khi xong thì BE sẽ trả kết quả về và hiển thị thành công hoặc lỗi cho người dùng biết.



Hình 4.14 Form hiển thị thông tin người dùng

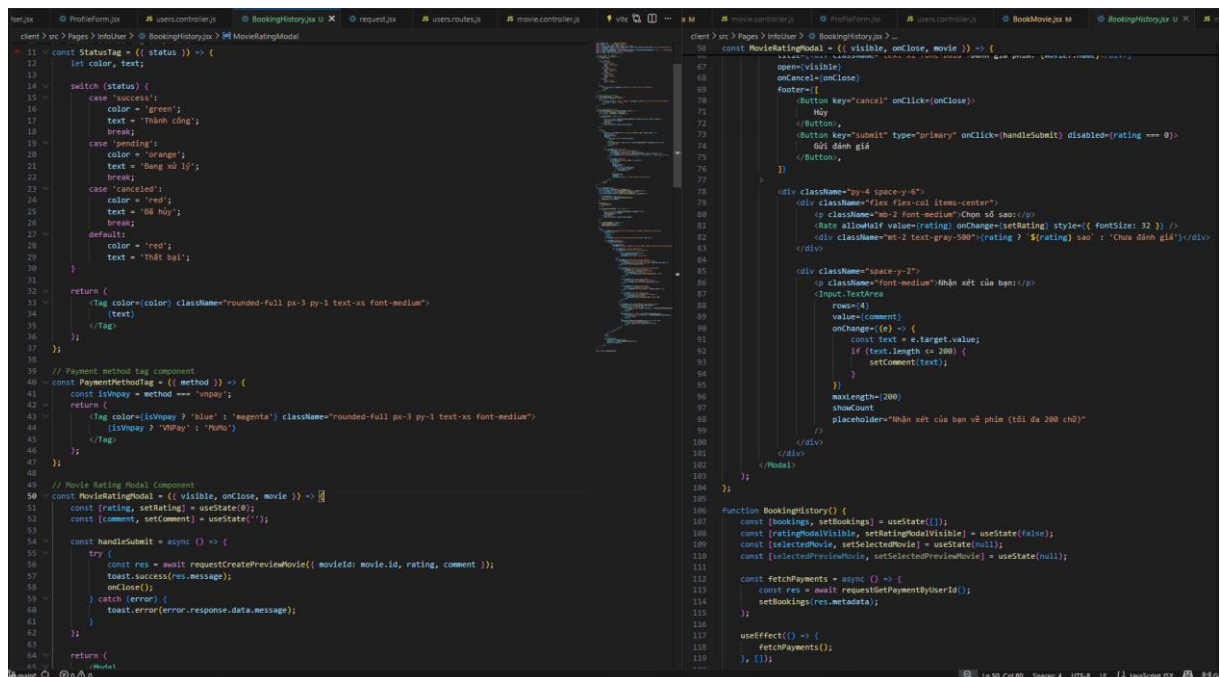
```
async updateUser(req, res) {
  const { id } = req.user;
  const { fullName, phone } = req.body;

  const user = await userModel.findOne({ where: { id } });
  if (!user) {
    throw new AuthFailureError('Tài khoản không tồn tại');
  }
  await user.update({ fullName, phone });
  new OK({ message: 'Cập nhật thông tin tài khoản thành công' }).send(res);
}
```

Hình 4.15 Hàm xử lý cập nhật thông tin người dùng

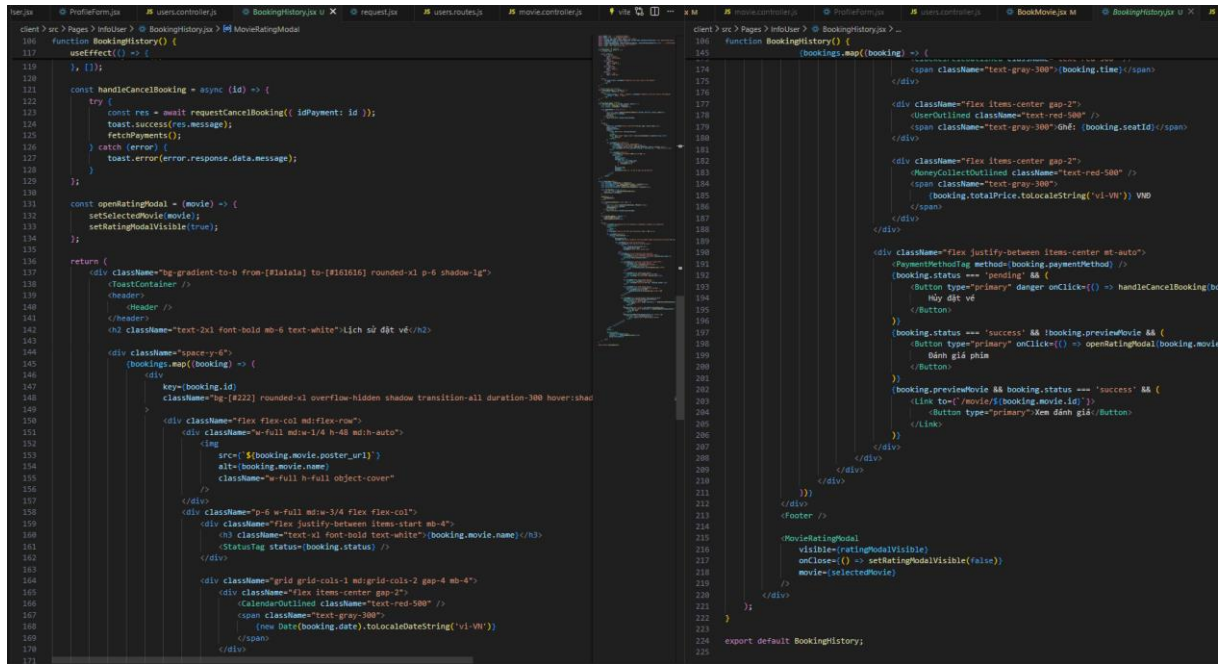
Chức năng lịch sử đặt vé

Ở phần bookingHistory làm việc với chức năng lấy thông tin vé user đã đặt và dữ liệu được trả về dạng danh sách, Khi user yêu cầu xem lịch sử đặt vé thì hàm requestGetPaymentByUserId được gọi tới mục đích để gửi yêu cầu API tới BE lấy danh sách lịch sử đặt vé sau khi tới BE xử lý logic xong thì trả về được lưu vào state booking thông qua setBookings. Sau khi hiển thị được danh sách lịch sử đặt vé thì ở đây khi người dùng thanh toán thành công đã được admin xác nhận thì còn phát triển thêm một chức năng là người dùng có thể đánh giá và bình luận.



Hình 4.16 Xử lý hiển thị thanh toán và đánh giá bình luận

Xây dựng hệ thống quản lý đặt vé xem phim trên nền tảng reactjs và nodejs



Hình 4.17 Form hiển thị lịch sử đặt vé

```

async getDashboard(req, res) {
  try {
    let { startDate, endDate } = req.query;

    // Default to last 7 days if no date range provided
    if (!startDate || !endDate) { ...
    }

    // Get total users
    const [userResults] = await connect.query('SELECT COUNT(*)');
    const totalUsers = userResults[0].total;

    // Get total movies
    const [movieResults] = await connect.query('SELECT COUNT(*)');
    const totalMovies = movieResults[0].total;

    // Get const revenueResults: any count
    const [revenueResults] = await connect.query(` ...
    `);

    const totalRevenue = revenueResults[0].totalRevenue || 0;
    const totalWatching = revenueResults[0].totalCount || 0;

    // Get recent orders
    const [recentOrders] = await connect.query(`
      SELECT
        p.id as idPayment,
        u.fullName,
        p.totalPrice,
        p.status,
        p.paymentMethod as typePayment,
        m.name as movieName
      FROM payments p
      LEFT JOIN users u ON p.userId = u.id
      LEFT JOIN movies m ON p.movieId = m.id
      WHERE p.date BETWEEN '${startDate}' AND '${endDate}'
      ORDER BY p.createdAt DESC
      LIMIT 5
    `);
  }
}

```

Hình 4.18 Hàm xử lý truy vấn lấy danh sách lịch sử đặt vé

Chức năng thanh toán

Phần chức năng thanh toán khi người dùng chọn ngày, giờ, ghế ngồi và công thanh toán thì lúc này ở Bookmovie hàm `handlePaymentConfirm` được gọi để xác nhận thông tin đặt vé của user sau khi xác nhận thì request được gọi chuyển hướng đến cổng khởi tạo quá trình thanh toán lúc này BE xử lý logic thanh toán sau khi đã thanh toán thành công lúc này `fetchPayment` được gọi để lấy thông tin thanh toán và trả về cho FE.

```
const handlePaymentConfirm = async () => {
  const dateOnly = new Date(selectedDate).toISOString().split('T')[0]; // "2025-06-27"
  const combined = new Date(`${dateOnly}T${selectedTime}:00Z`); // Kết hợp với giờ

  const data = {
    typePayment: paymentMethod,
    idMovie: movieDetails.id,
    idSeat: selectedSeats,
    time: selectedTime,
    date: selectedDate.format('YYYY-MM-DD'),
  };

  try {
    const res = await requestCreatePayment(data);
    if (paymentMethod === 'momo') {
      window.open(res.metadata.payUrl);
    }
    if (paymentMethod === 'vnpay') {
      window.open(res.metadata);
    }
    setIsModalVisible(false);
  } catch (error) {
    console.log(error);
  }
};
```

Hình 4.20 Hàm xác nhận thông tin thanh toán

```
useEffect(() => {
  const fetchPayment = async () => {
    try {
      setLoading(true);
      if (id) {
        const res = await requestGetPaymentById(id);
        if (res && res.metadata) {
          setPaymentDetails({
            ...res.metadata,
            movieName: res.metadata.movie.name || '',
          });
        }
      } else {
        // fallback to URL params if no ID
        const queryParams = new URLSearchParams(location.search);
        const data = {};

        if (queryParams.get('movieId')) {
          data.movieId = queryParams.get('movieId');
        }
        if (queryParams.get('movieName')) {
          data.movieName = queryParams.get('movieName');
        }
        if (queryParams.get('totalPrice')) {
          data.totalPrice = queryParams.get('totalPrice');
        }
        if (queryParams.get('seatId')) {
          data.seatId = queryParams.get('seatId');
        }
        if (queryParams.get('time')) {
          data.time = queryParams.get('time');
        }
        if (queryParams.get('date')) {
          data.date = queryParams.get('date');
        }
        if (queryParams.get('paymentMethod')) {
          data.paymentMethod = queryParams.get('paymentMethod');
        }
        if (Object.keys(data).length > 0) {
          setPaymentDetails(data);
        } else {
          // Show mock data for preview
          setPaymentDetails({
            id: 'PAY75185144970497803',
            movieId: '65a250f-40b2-4f9f-97f3-bc348a0b0e5',
            movieName: 'Phù Thủy Tội Thượng 10',
            date: '2025-06-27T10:00:00.000Z',
            time: '20:00',
            seatId: 'A7/02',
            totalPrice: 100000,
            paymentMethod: 'vnpay',
            status: 'success',
          });
        }
      }
    } catch (error) {
      console.error('Error fetching payment data:', error);
    } finally {
      setLoading(false);
    }
  }

  fetchPayment();
}, [id, location]);

// Format price to VND
const formatPrice = (price) => {
  if (!price) return '0 VND';
  return parseInt(price).toLocaleString('vi-VN') + ' VND';
};

// Format date to Vietnamese format
const formatDate = (dateString) => {
  if (!dateString) return '';
  const date = new Date(dateString);
  return date.toLocaleDateString('vi-VN', {
    day: '2-digit',
    month: '2-digit',
    year: 'numeric',
  });
};

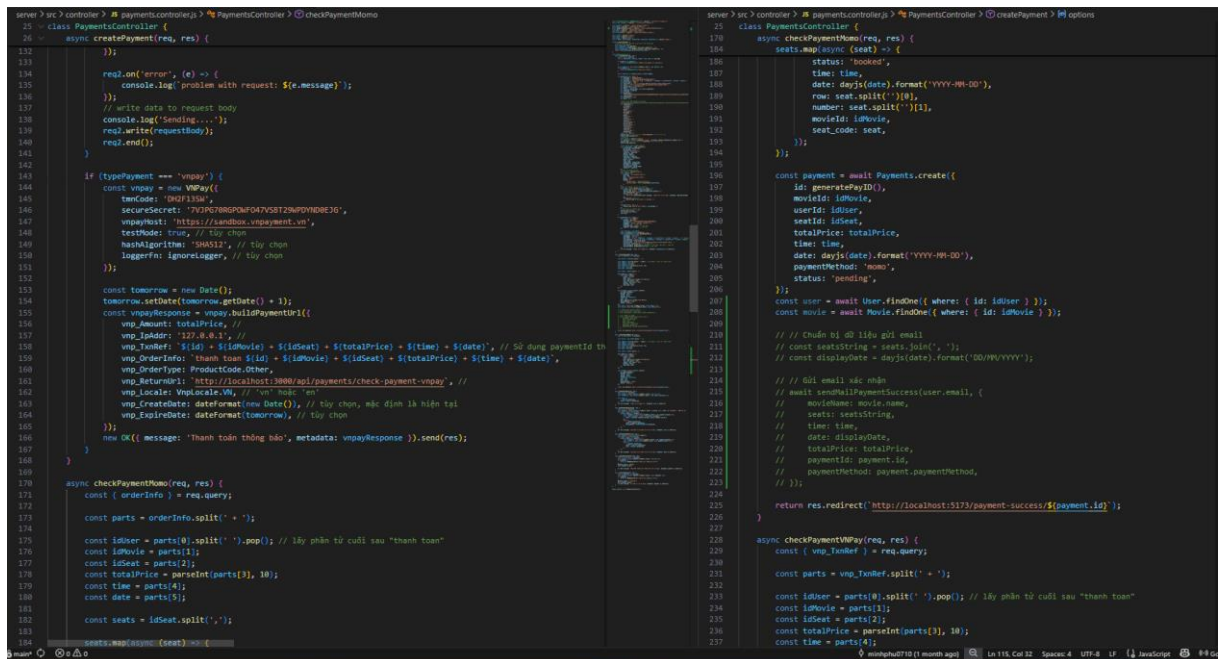
// Convert seatId string to array
const getSeatArray = (seatString) => {
  if (!seatString) return [];
  return seatString.split(',').map((seat) => seat.trim());
};

if (loading) {
  return (
    <div className="bg-[#f3f3f3] min-h-screen flex flex-col">
      <header />
      <div className="flex-grow flex items-center justify-center">
        <div className="animate-spin rounded-full h-12 w-12 border-t-2 border-b-2 border-[#000080]"/>
        <h3 />
      </div>
    </div>
  );
}

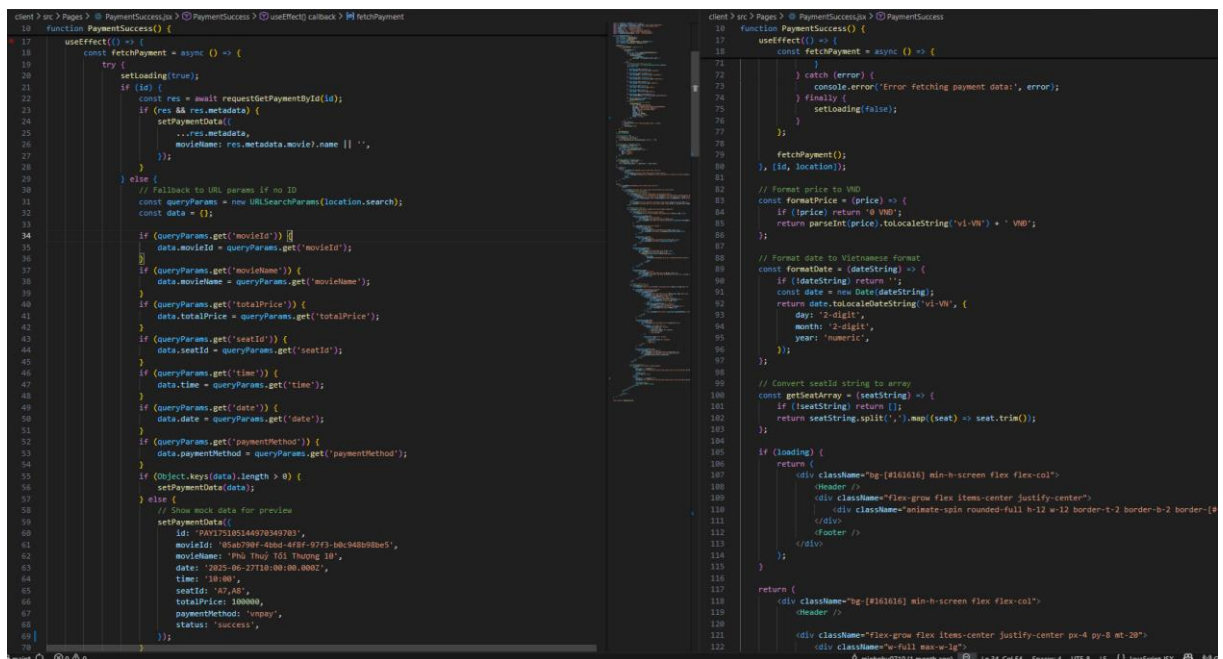
return (
  <div className="bg-[#f3f3f3] min-h-screen flex flex-col">
    <header />
    <div className="flex-grow flex items-center justify-center px-4 py-8 mt-20">
```

Hình 4.19 Xử lý thanh toán

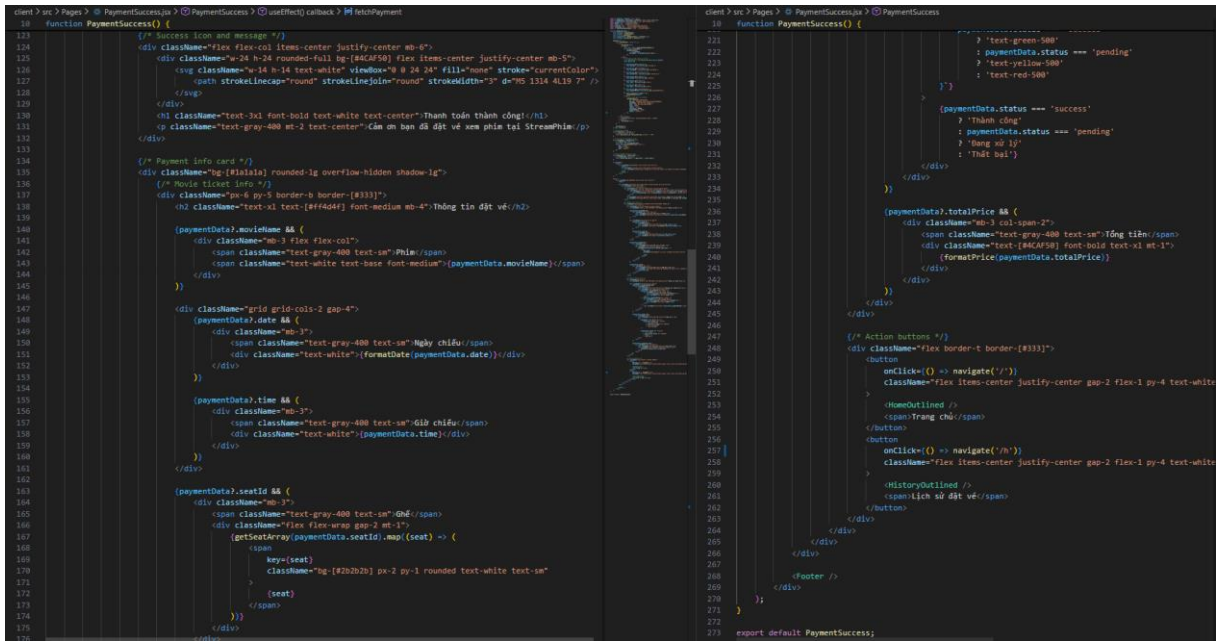
Xây dựng hệ thống quản lý đặt vé xem phim trên nền tảng reactjs và nodejs



Hình 4.21 Xử lý phương thức thanh toán



Hình 4.22 Xử lý thông tin thanh toán



Hình 4.23 Form hiển thị thông tin thanh toán

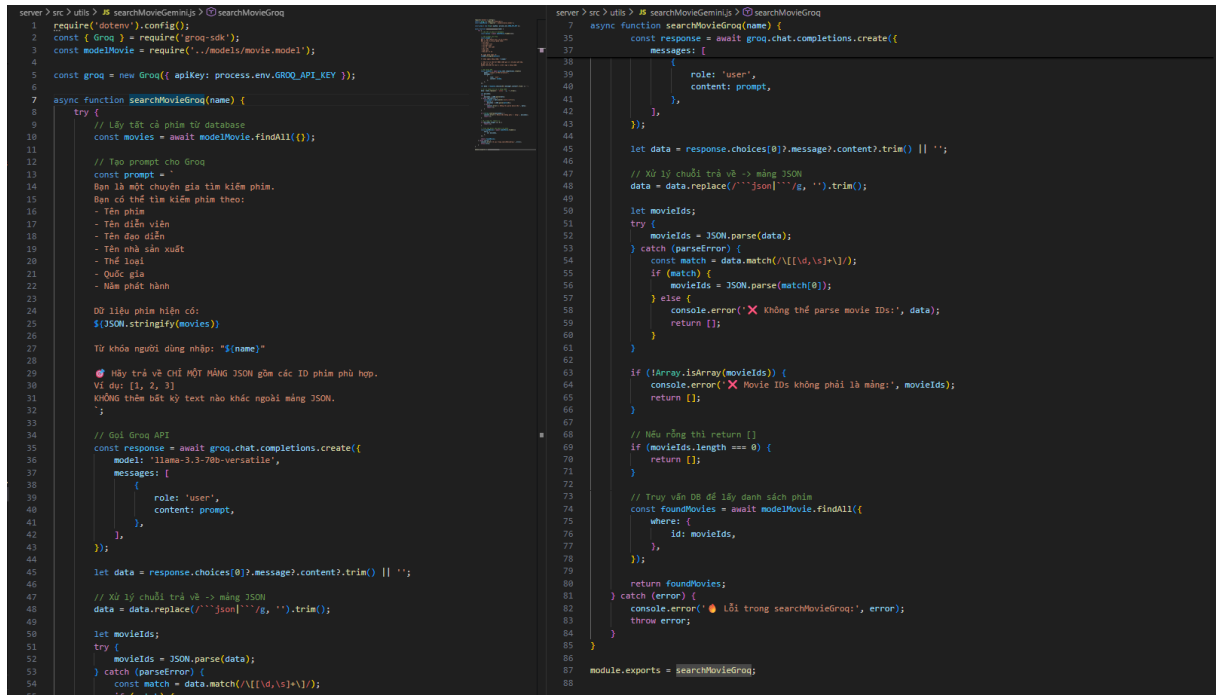
Chức năng tìm kiếm phim

Chức năng được tích hợp AI Gemini để nâng cao khả năng tìm kiếm thông minh hỗ trợ các tiêu chí tìm kiếm như tên phim, diễn viên, đạo diễn, thể loại, quốc gia và năm phát hành bằng cách nhập từ khóa vào thanh tìm kiếm ở trang chủ lúc này từ khóa được tìm kiếm được gửi đến API thông qua request lúc này BE nhận yêu cầu và xử lý hàm searchMovie gọi đến searchMovieGroq để xử lý logic tìm kiếm lấy toàn bộ danh sách phim từ cơ sở dữ liệu ra một prompt chi tiết được xây dựng gửi đến AI Gemini thông qua SDK Groq lúc này kết quả trả về và parse vào mảng movieIds nếu hợp lý sẽ cho kết quả phù hợp về cho FE.

```
async searchMovie(req, res) {
  const { name } = req.query;

  const data = await searchMovieGemini(name);
  new OK({
    message: 'Search movie successfully',
    metadata: data,
  }).send(res);
}
```

Hình 4.24 Hàm xử lý tìm kiếm

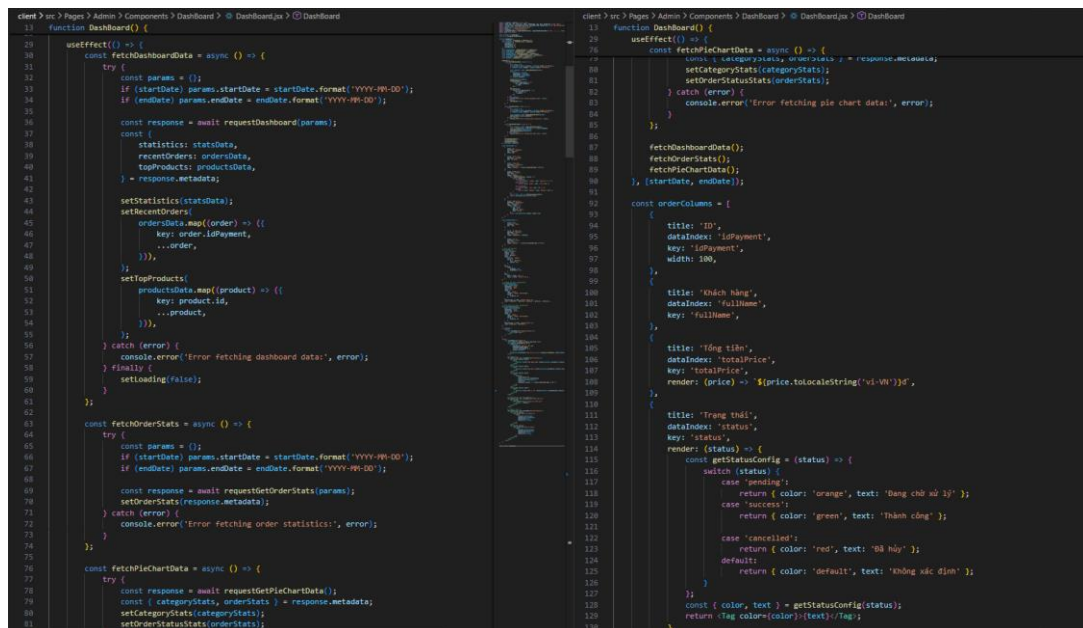


Hình 4.25 Xử lý logic tìm kiếm với Gemini

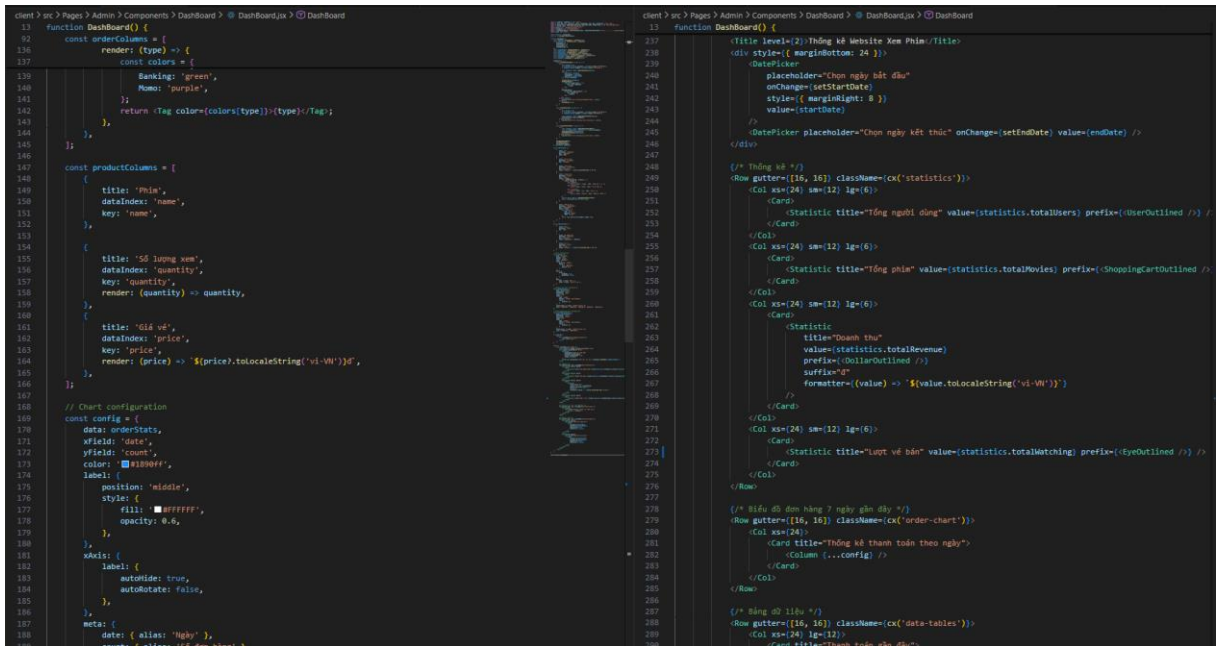
4.1.2 Chức năng admin

Chức năng quản lý thống kê

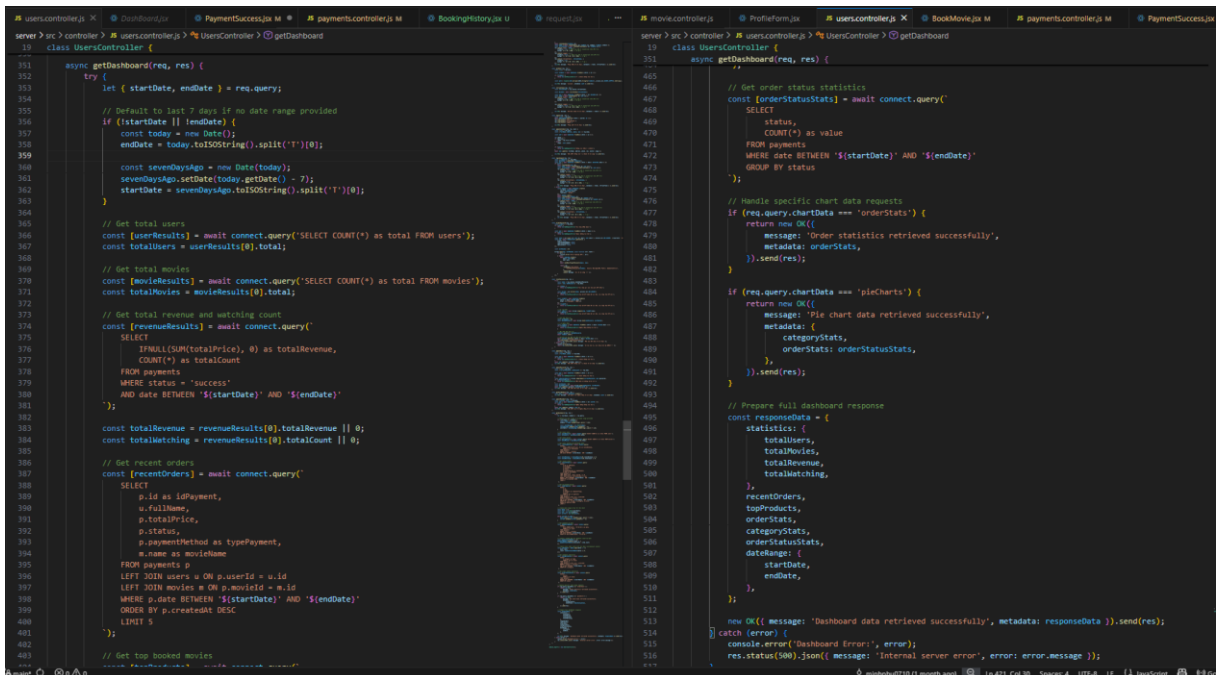
Admin truy cập trang quản lý dành cho admin lúc này phần dashboard được render gọi API lấy dữ liệu thống kê sau khi nhận yêu cầu thì gọi hàm getDashboard truy vấn cơ sở dữ liệu để lấy tất cả thông tin thống kê và trả về hiển thị lên giao diện.



Hình 4.26 Xử lý từng phần trong thống kê



Hình 4.27 Thiết kế form hiển thị quản lý thống kê

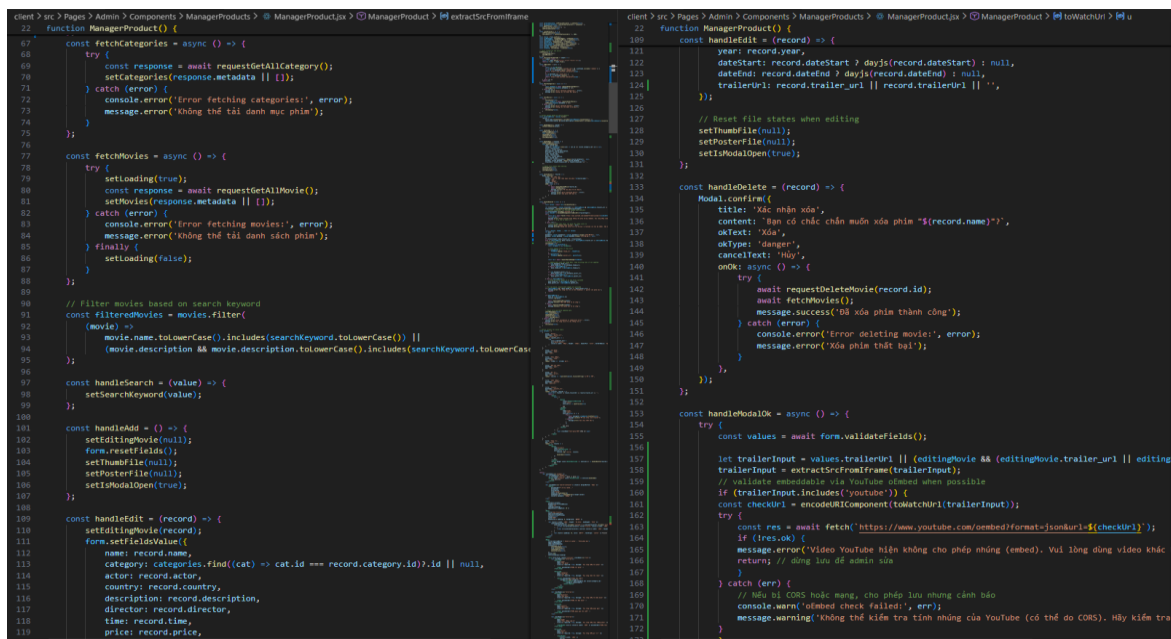


Hình 4.28 Hàm xử lý lấy thông tin thống kê

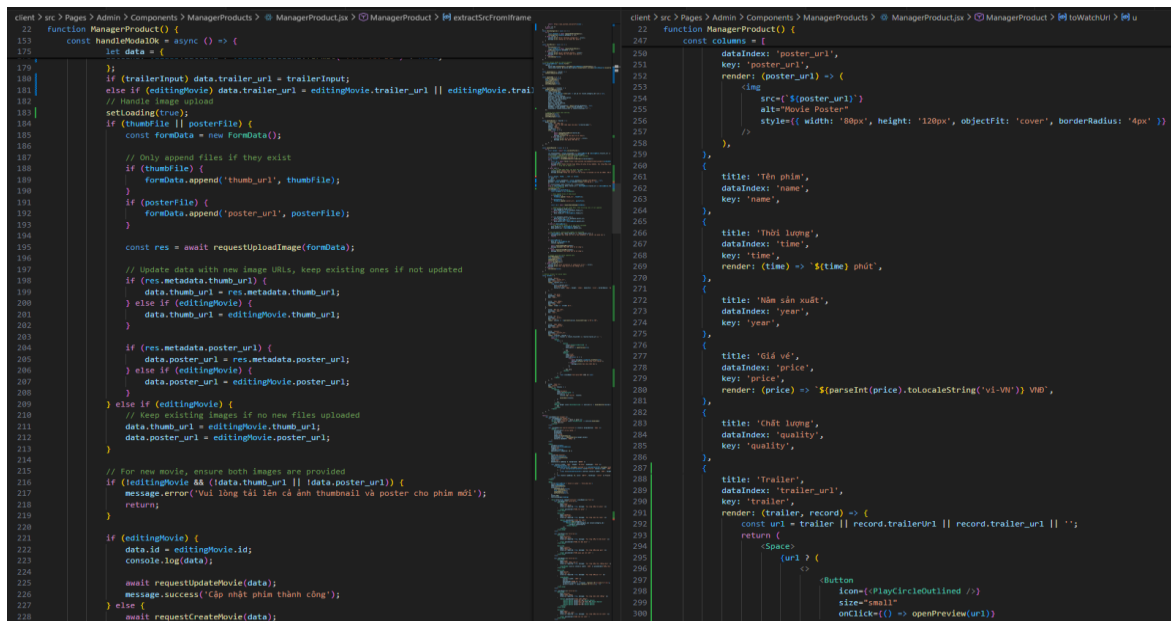
Chức năng quản lý phim

Chức năng này giúp admin quản lý nội dung của phim một cách chặt chẽ có hệ thống nó bao gồm: hiển thị danh sách phim hiện có trên hệ thống, tìm kiếm, thêm phim mới, cập nhật phim và xóa phim. Admin tạo những request tương ứng để có thể gửi yêu cầu lên API thông qua những request ấy để đến BE xử lý API tạo phim (createMovie): Nhận dữ liệu từ frontend, bao gồm thông tin phim và đường dẫn ảnh lưu thông tin phim

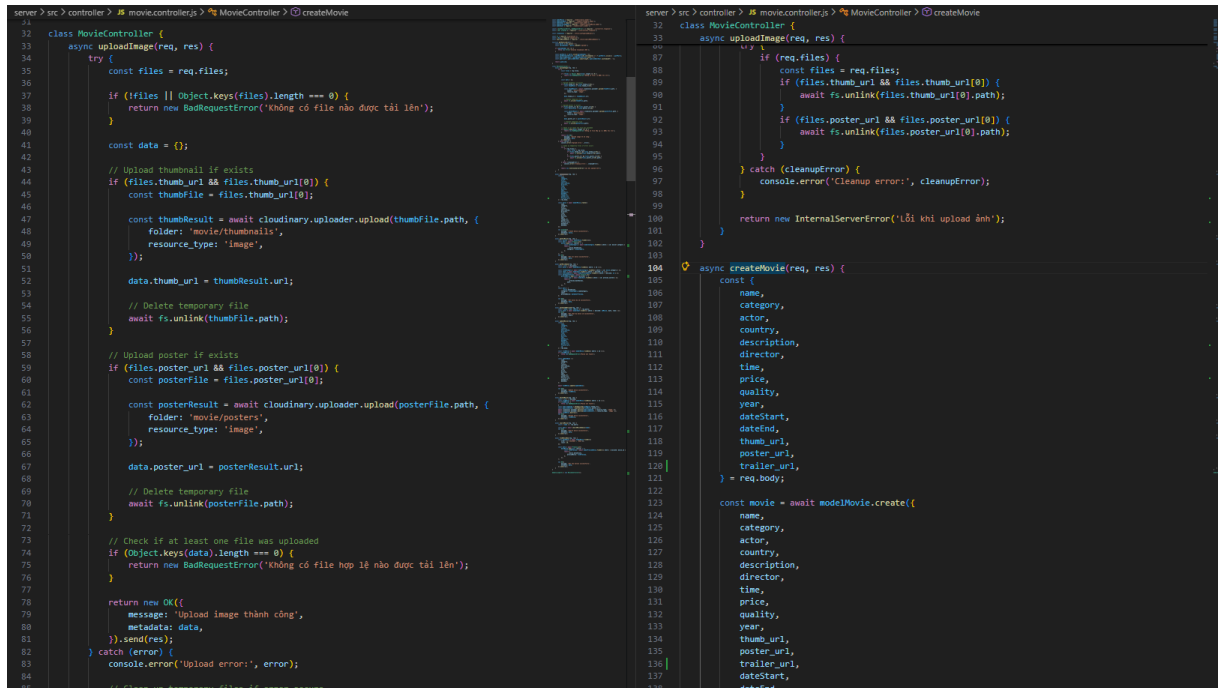
vào cơ sở dữ liệu. API lấy danh sách phim (getAllMovie): Truy vấn cơ sở dữ liệu để lấy danh sách phim, trả về dữ liệu cho frontend. API chỉnh sửa phim (updateMovie): Nhận dữ liệu chỉnh sửa từ frontend, cập nhật thông tin phim trong cơ sở dữ liệu. API xóa phim (deleteMovie): Nhận ID phim từ frontend xóa phim khỏi cơ sở dữ liệu. API tìm kiếm phim (searchMovie): Nhận từ khóa tìm kiếm từ frontend, truy vấn cơ sở dữ liệu để tìm phim phù hợp. Upload ảnh (uploadImage): Sử dụng multer để xử lý upload ảnh thumbnail và poster, lưu ảnh lên Cloudinary và trả về URL.



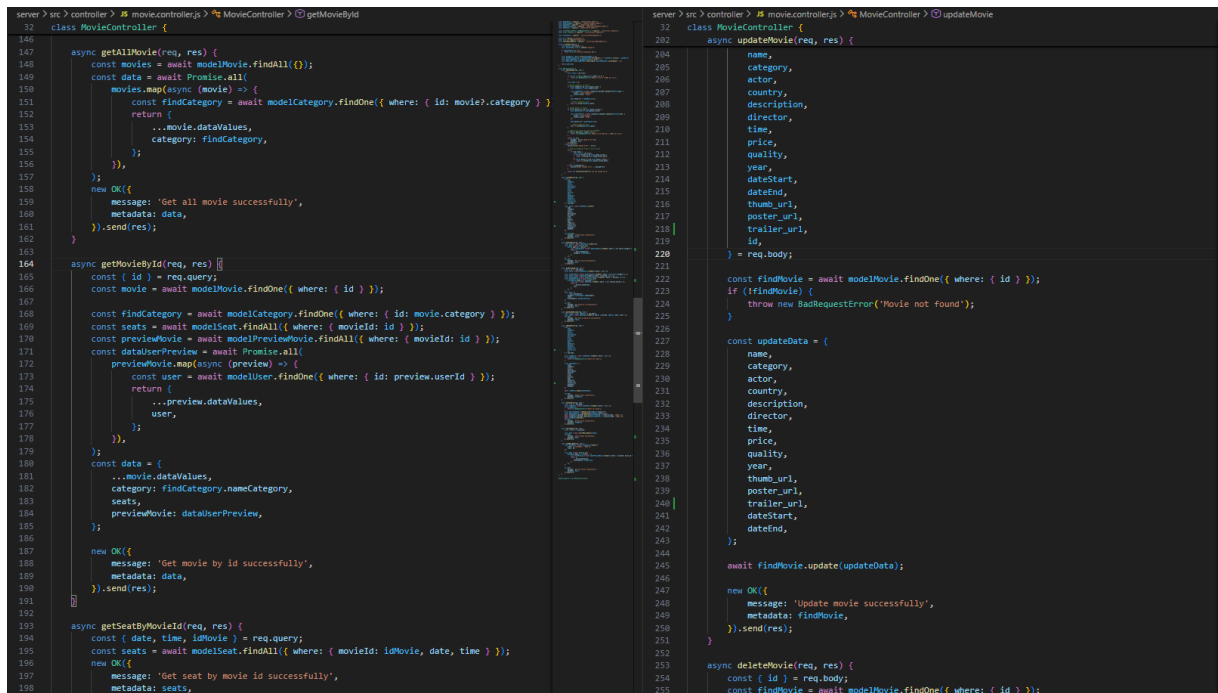
Hình 4.29 Cài đặt xử lý logic các hàm quản lý phim



Hình 4.30 Thiết kế form hiển thị ở quản lý phim



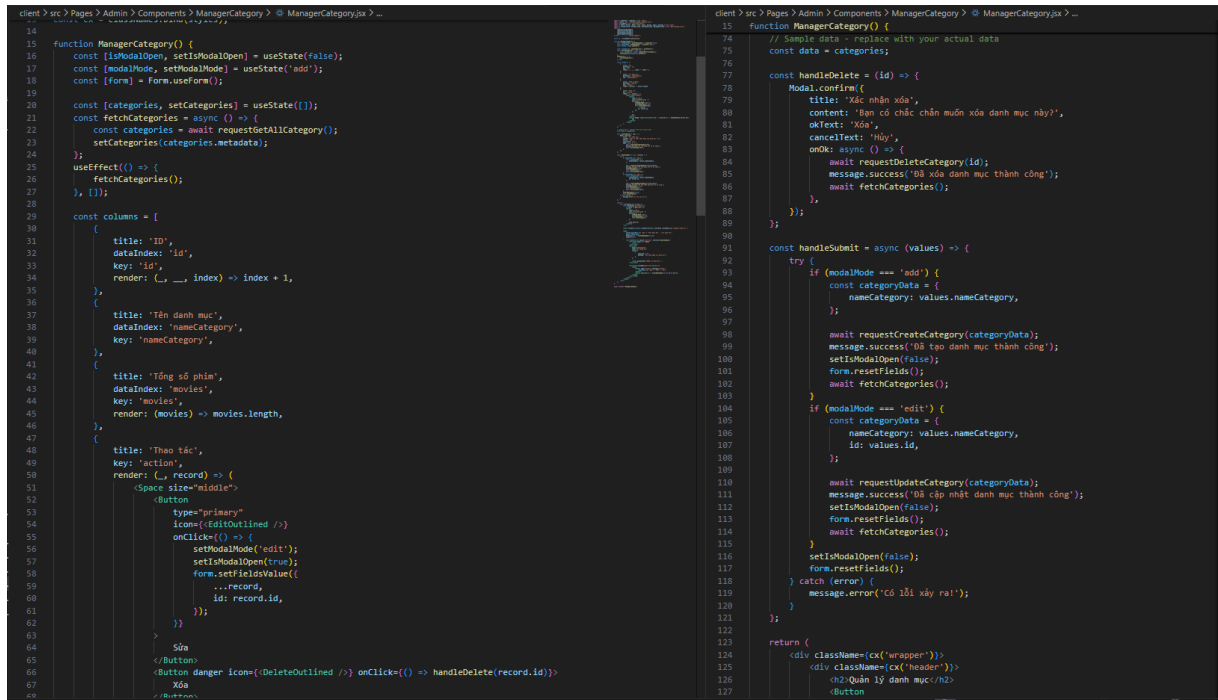
Hình 4.31 Xử lý logic chức năng của quản lý phim



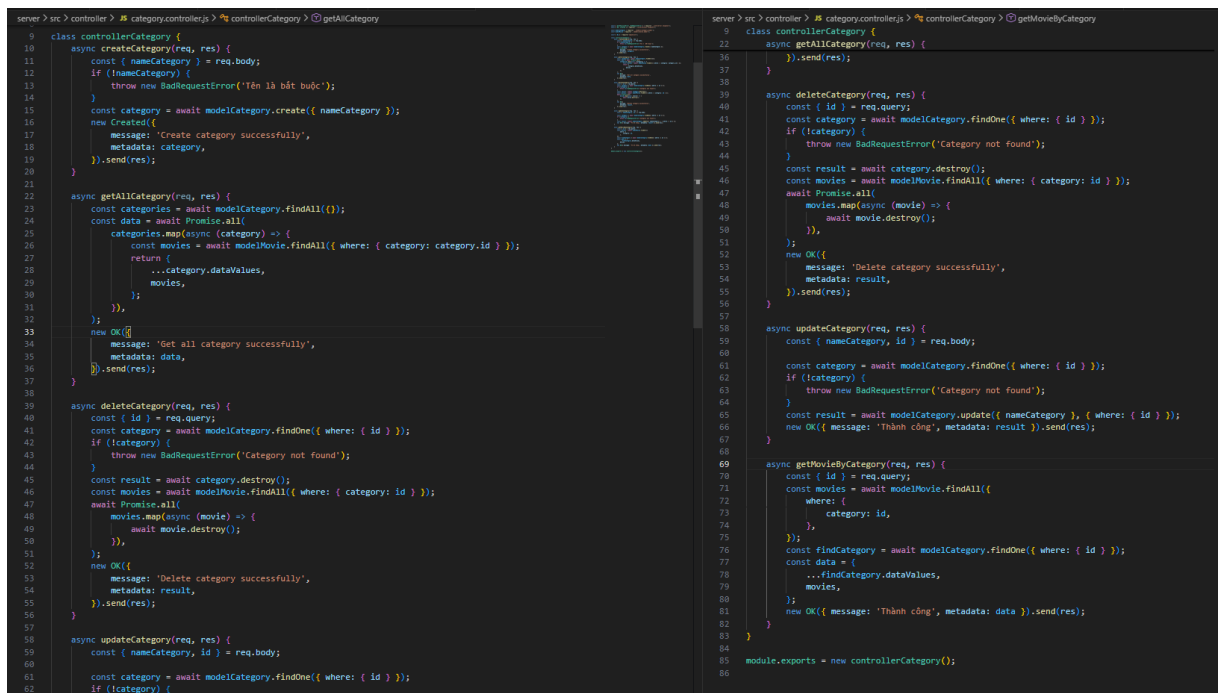
Hình 4.32 Logic BE quản lý phim

Chức năng quản lý danh mục phim

Chức năng này giúp admin kiểm soát việc lấy danh sách số lượng phim theo từng danh mục rõ ràng minh bạch và thêm, sửa, xóa các danh mục bằng cách khởi tạo các API tương ứng với từng chức năng.



Hình 4.33 Xử lý và hiển thị chức năng danh mục

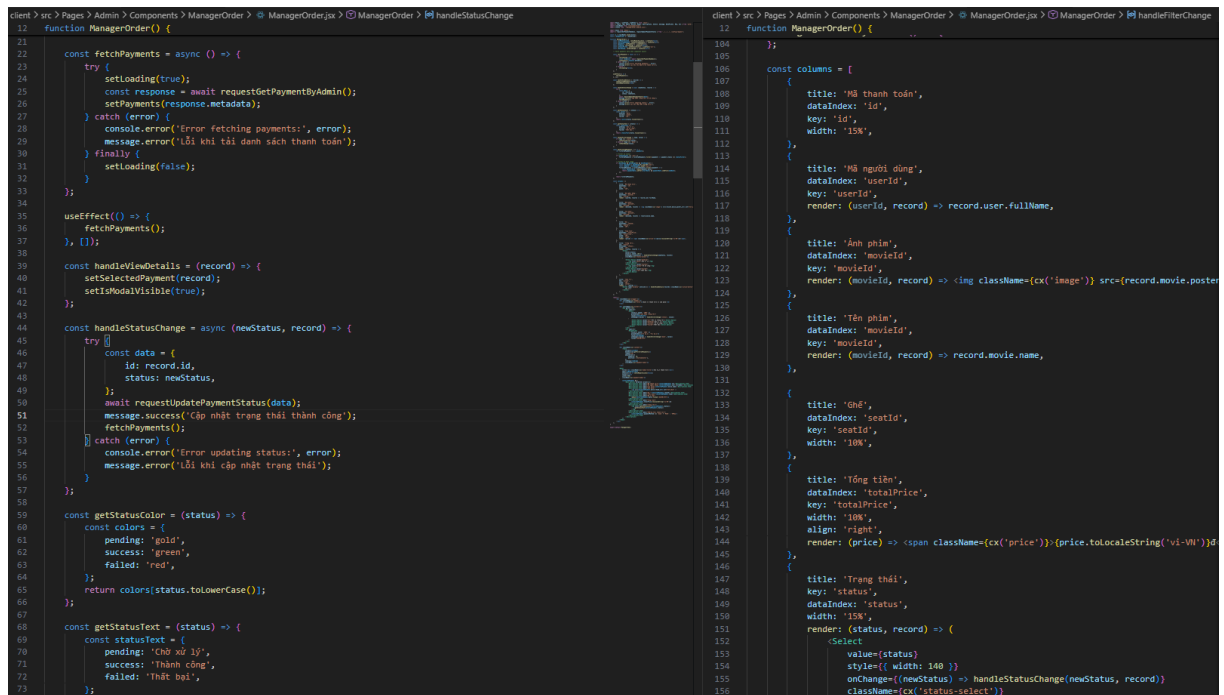


Hình 4.34 Xử lý logic chức năng danh mục

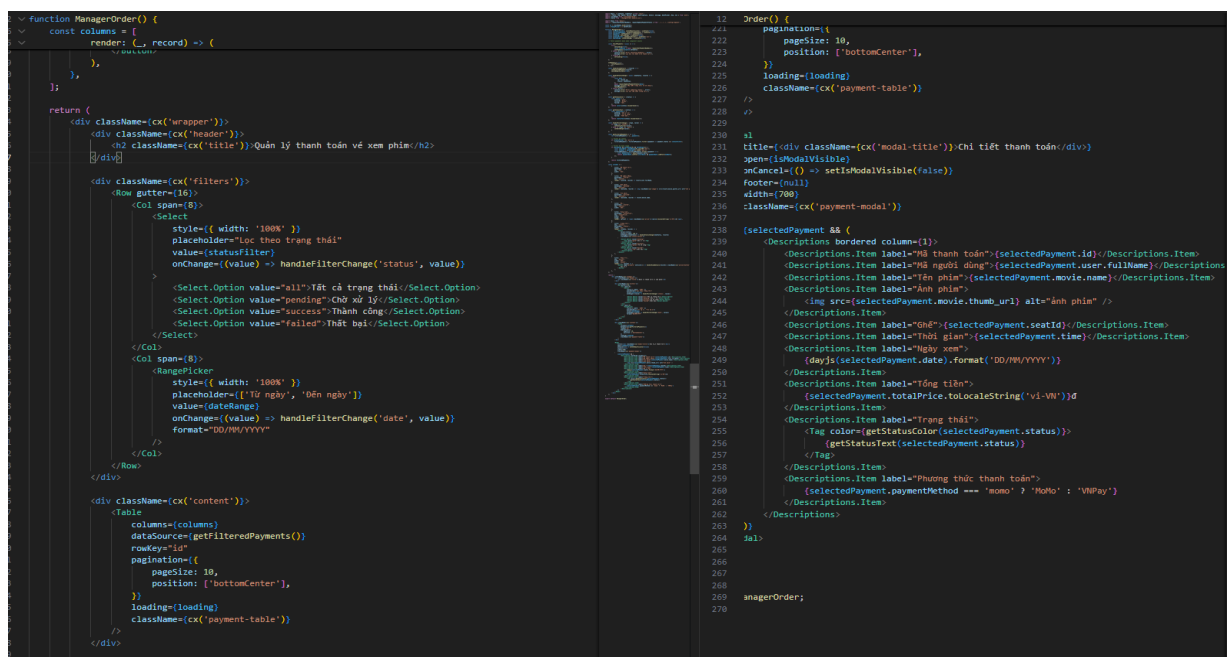
Chức năng quản lý vé

Chức năng quản lý vé giúp admin kiểm soát lượng vé đặt hàng ngày/tháng/năm, lọc vé theo các trạng thái thanh toán (đang xử lý, đã thanh toán, hủy), có thể xem được thông tin chi tiết vé của user đã đặt để nắm chi tiết mã vé thuộc người dùng nào và cập nhật trạng thái vé hiển thị vé cho user bằng cách khởi tạo các API tương ứng để sử dụng.

từng chức năng trong quản lý vé mỗi lần thao tác là gửi yêu cầu đến API tương ứng và BE nhận yêu cầu và xử lý dữ liệu.



Hình 4.35 Xử lý và hiển thị danh sách vé theo form



Hình 4.36 Lọc vé và xem chi tiết vé theo form

```
async getPaymentByUserId(req, res) {
  const { id } = req.user;
  const payments = await Payments.findAll({ where: { userId: id }, order: [['createdAt', 'DESC']] });
  const data = await Promise.all(
    payments.map(async (payment) => {
      const findMovie = await Movie.findOne({ where: { id: payment.movieId } });
      const findPreviewMovie = await PreviewMovie.findOne({
        where: { movieId: payment.movieId, userId: payment.userId },
      });
      return {
        ...payment.dataValues,
        movie: findMovie.dataValues,
        previewMovie: findPreviewMovie?.dataValues,
      };
    })
  );
  new OK({ message: 'Lấy danh sách thanh toán thành công', metadata: data }).send(res);
}

async getPaymentByAdmin(req, res) {
  const payments = await Payments.findAll();
  const data = await Promise.all(
    payments.map(async (payment) => {
      const findMovie = await Movie.findOne({ where: { id: payment.movieId } });
      const findUser = await User.findOne({ where: { id: payment.userId } });
      return {
        ...payment?.dataValues,
        movie: findMovie?.dataValues,
        user: findUser?.dataValues,
      };
    })
  );
  new OK({ message: 'Lấy danh sách thanh toán thành công', metadata: data }).send(res);
}

async updatePaymentStatus(req, res) {
  const { id, status } = req.body;
  const payment = await Payments.findOne({ where: { id: id } });
  if (!payment) {
    throw new BadRequestError('Thanh toán không tồn tại');
  }
  payment.status = status;
  await payment.save();

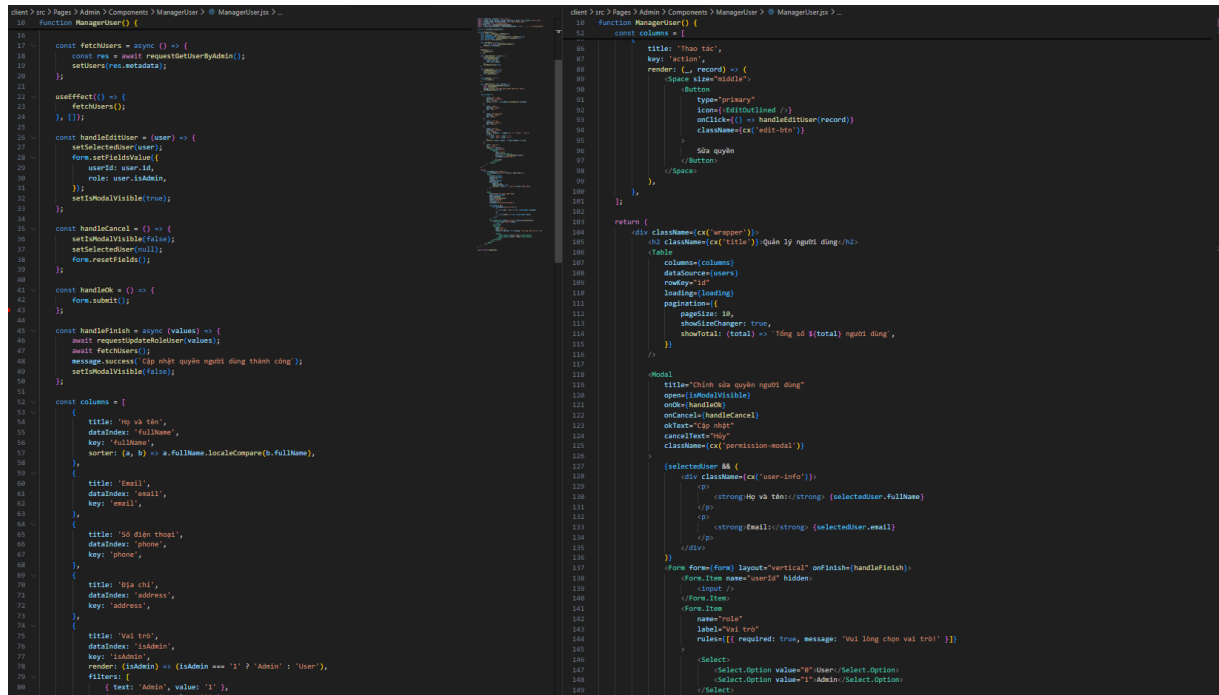
  new OK({ message: 'Cập nhật trạng thái thanh toán thành công', metadata: payment }).send(res);
}

async cancelBooking(req, res) {
  const { idPayment } = req.body;
  const payment = await Payments.findOne({ where: { id: idPayment } });
  if (!payment) {
    throw new BadRequestError('Thanh toán không tồn tại');
  }
  payment.status = 'canceled';
  await payment.save();
  new OK({ message: 'Hủy đặt vé thành công', metadata: payment }).send(res);
}
```

Hình 4.37 Xử lý logic các chức năng ở quản lý vé

Chức năng quản lý người dùng

Chức năng quản lý người dùng giúp admin kiểm soát lượng người dùng truy cập vào hệ thống nắm được thông tin từng user và có thể cấp quyền admin cho user khi hệ thống đủ lớn cần thiết thêm đội ngũ admin kiểm soát hệ thống.



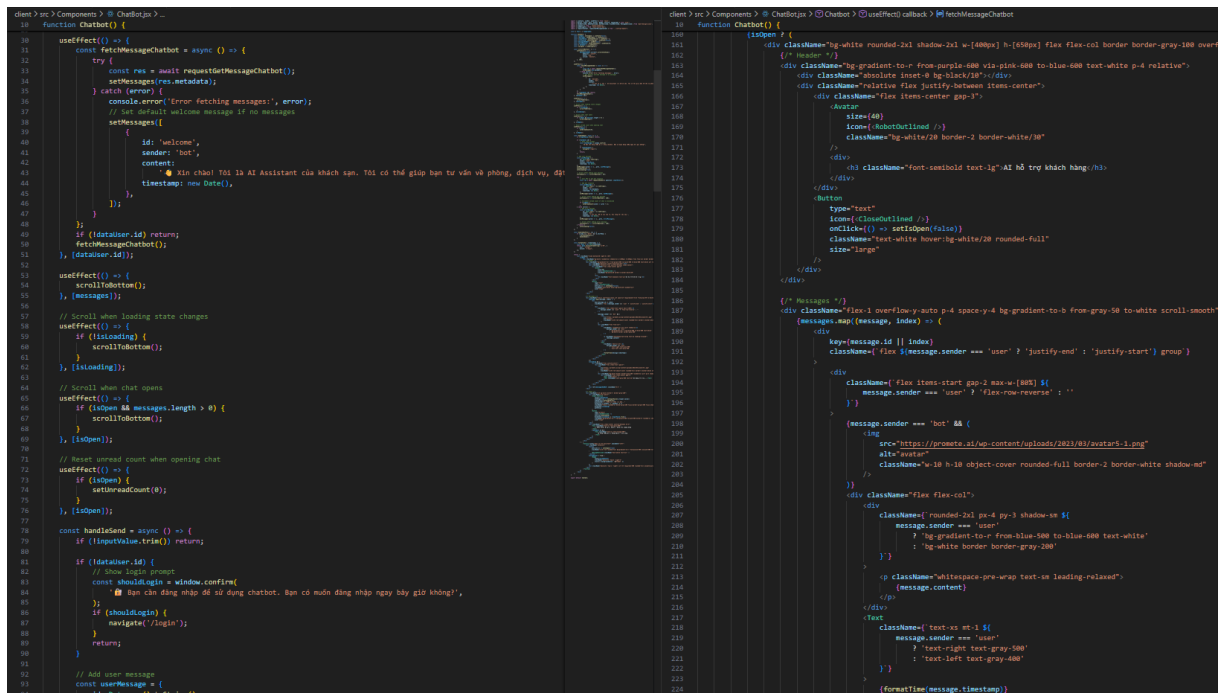
Hình 4.38 form hiển thị thông tin và chỉnh sửa quyền người dùng



Hình 4.39 Xử lý logic danh sách và quyền người dùng

Chức năng chatbot AI

Người dùng gửi một tin nhắn đến chatbot lúc này nhận câu hỏi và service xử lý logic lấy lịch sử tin nhắn lấy danh sách phim từ cơ sở dữ liệu tạo prompt và gửi tới Goq AI sau đó BE trả phản hồi từ AI về FE



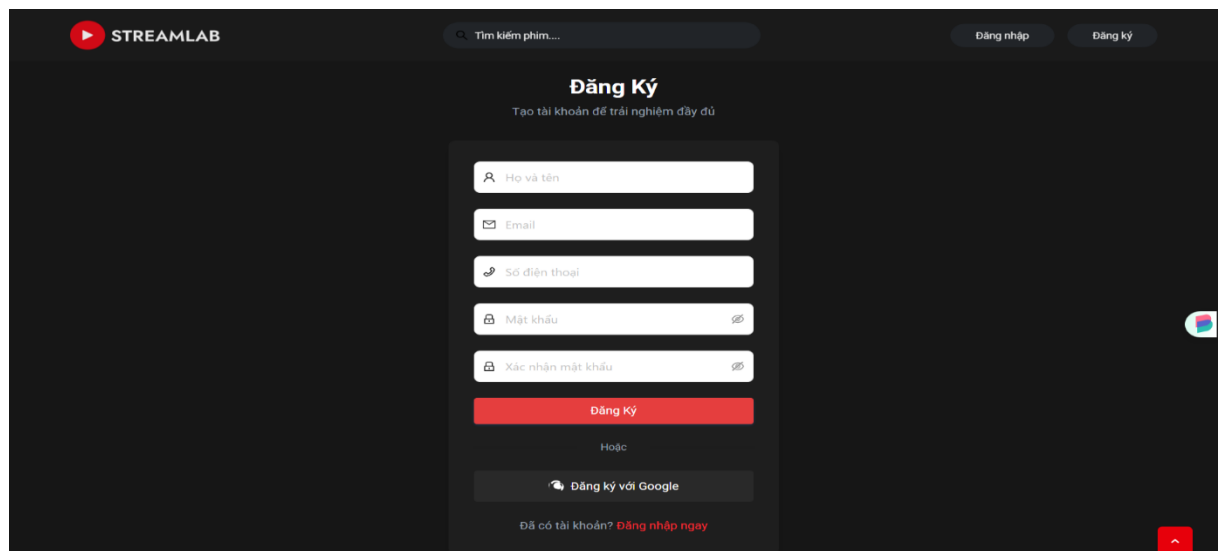
Hình 4.40 Luồng xử lý và form hiển thị chatbot

4.2 Giao diện sản phẩm

4.2.1 Giao diện của khách hàng

Giao diện chức năng đăng ký

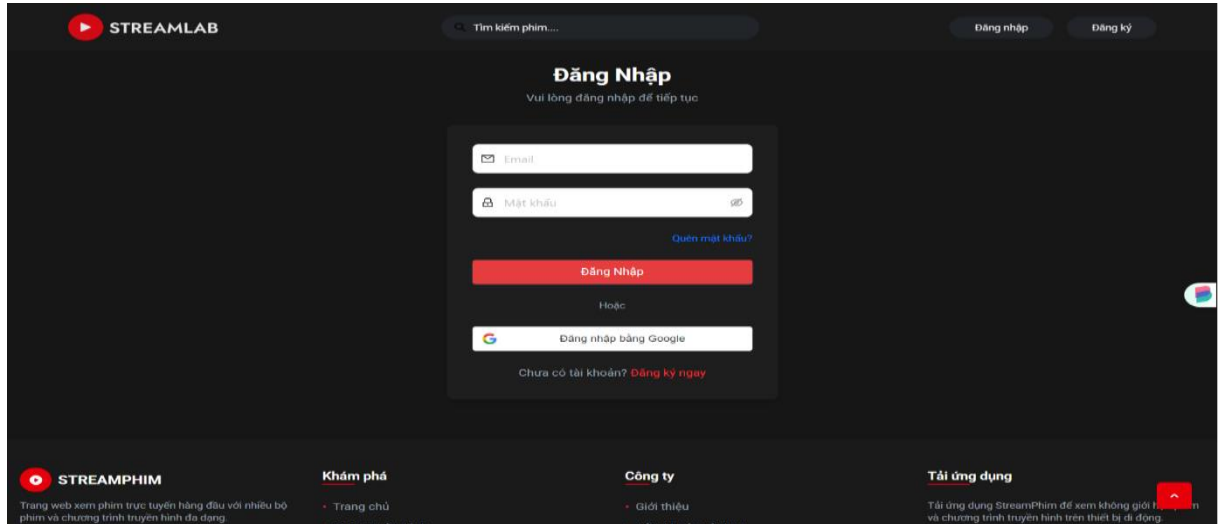
Mục đích chức năng này cho phép người dùng đăng ký tài khoản mới bằng cách điền các thông tin cá nhân(họ tên, email, số điện thoại và mật khẩu), người dùng vào trang web và ở trên header của trang web sẽ chọn button đăng ký và trang đăng ký hiện lên.



Hình 4.41 Giao diện đăng ký

Giao diện chức năng đăng nhập

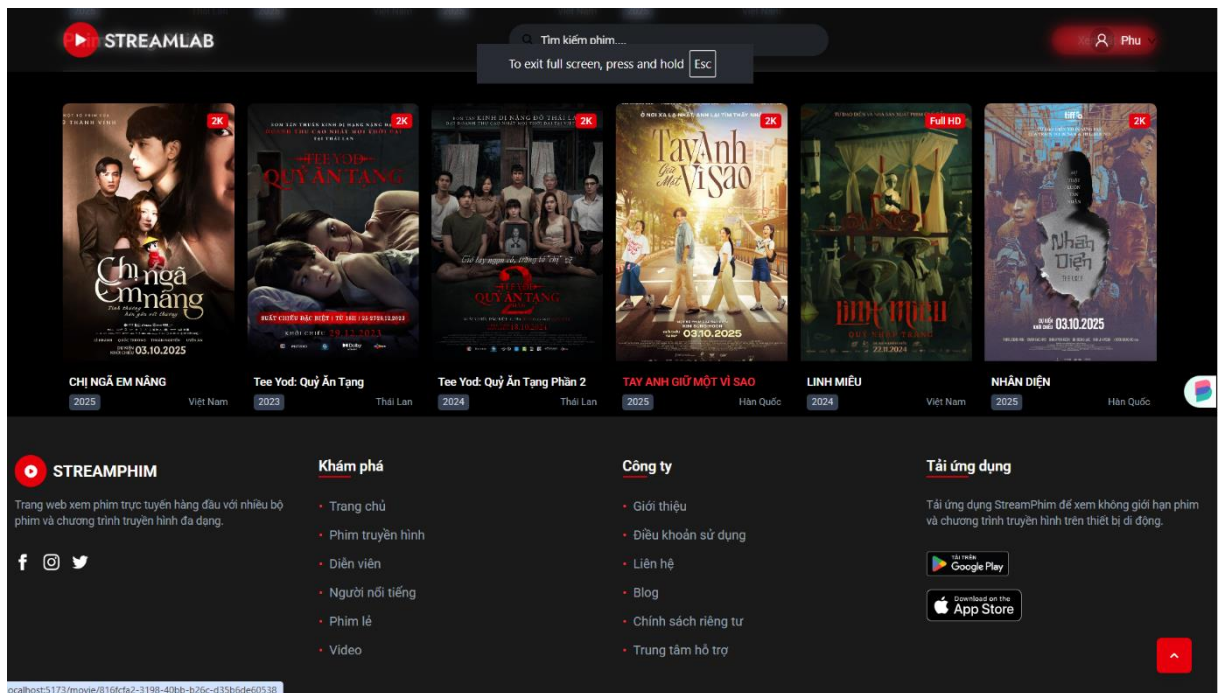
Mục đích của chức năng đăng nhập cho phép người dùng đăng nhập vào tài khoản của họ sau khi đã có tài khoản trên hệ thống (hoặc đăng nhập trực tiếp bằng tài khoản google), người dùng thao tác chọn button đăng nhập và form đăng nhập sẽ hiện lên.



Hình 4.42 Giao diện đăng nhập

Giao diện trang chủ

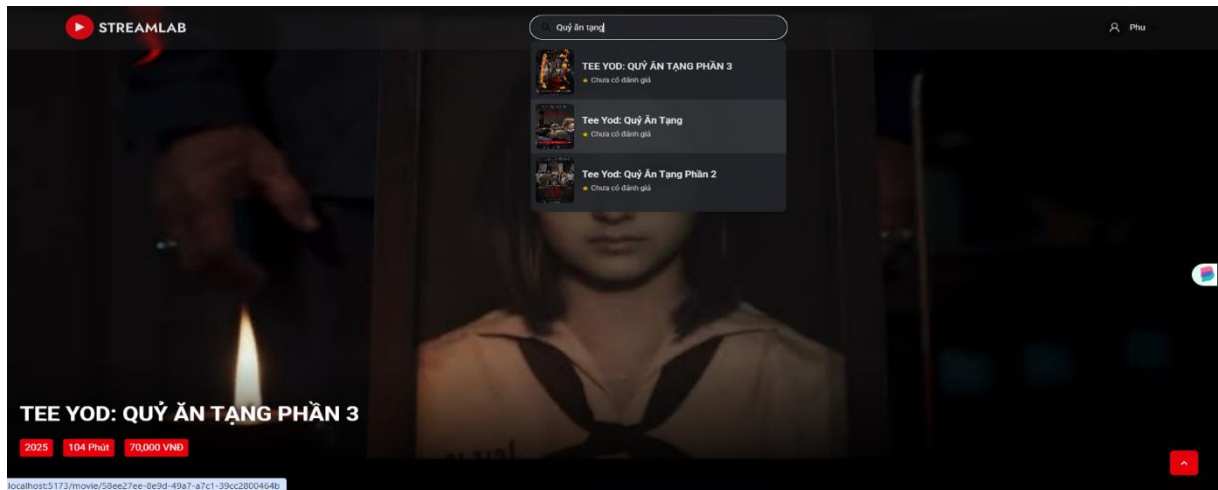
Người dùng truy cập trang chủ với vai trò khách lạ của hệ thống sẽ xem được những sản phẩm hiện có ở website và có thể xem thông tin chi tiết từng phim ở đây.



Hình 4.43 Giao diện trang chủ

Giao diện trang tìm kiếm phim

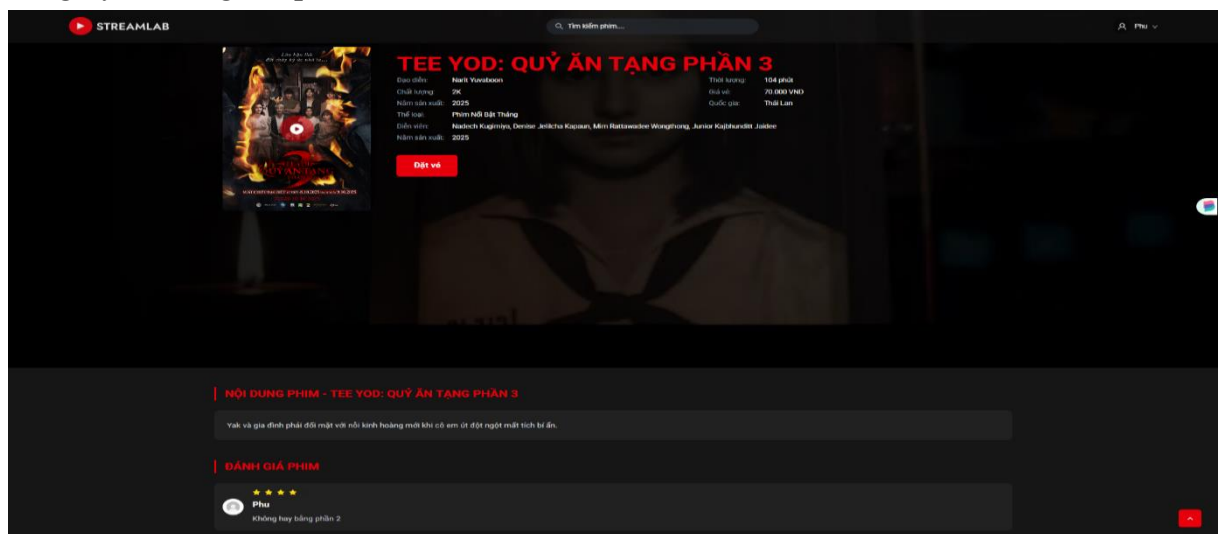
Mục đích của giao diện tìm kiếm là cung cấp cho người dùng một công cụ nhanh chóng và tiện lợi để tìm kiếm các bộ phim theo từ khóa (tên phim, tên đạo diễn hay cả tên diễn viên), người dùng thao tác truy cập trang web ở header sẽ có thanh tìm kiếm người dùng nhập từ khóa cần tìm kiếm hệ thống sẽ trả về kết quả phù hợp.



Hình 4.44 Giao diện tìm kiếm phim

Giao diện thông tin phim

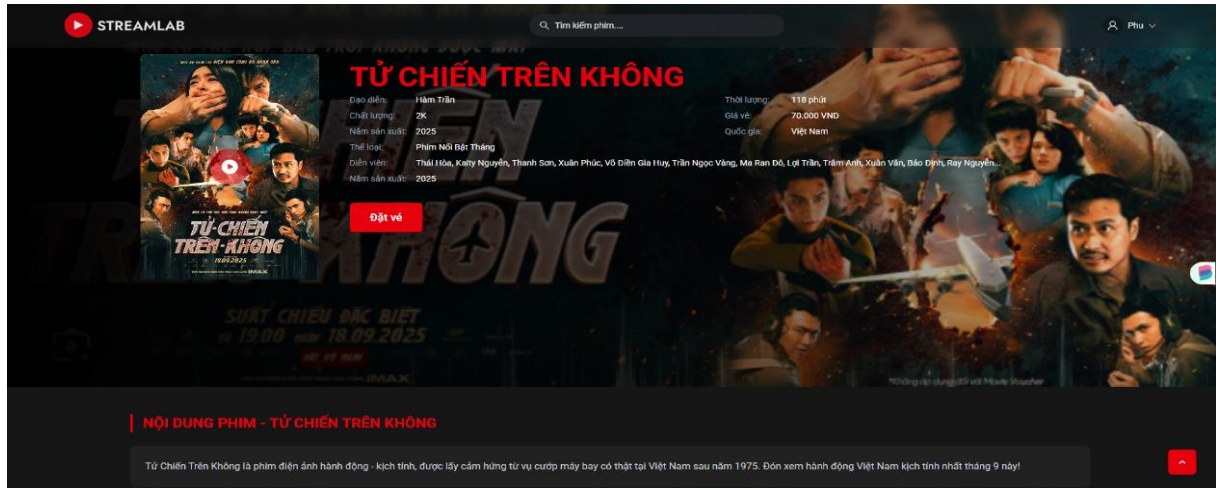
Mục đích giao diện thông tin phim dùng để cung cấp cho người dùng đầy đủ về thông tin của phim cũng như đánh giá bình luận ở đó khi người dùng thao tác truy cập hệ thống với vai trò khách lạ thì có thể xem thông tin nhưng khi đặt vé thì bắt buộc phải đăng ký và đăng nhập.



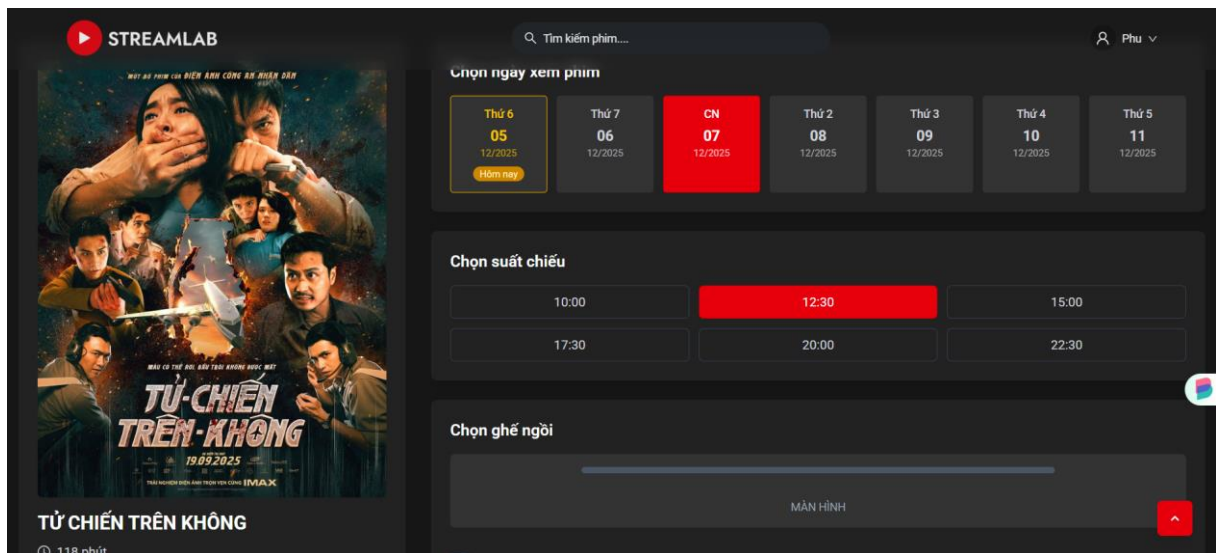
Hình 4.45 Giao diện chi tiết phim

Giao diện đặt vé và thanh toán

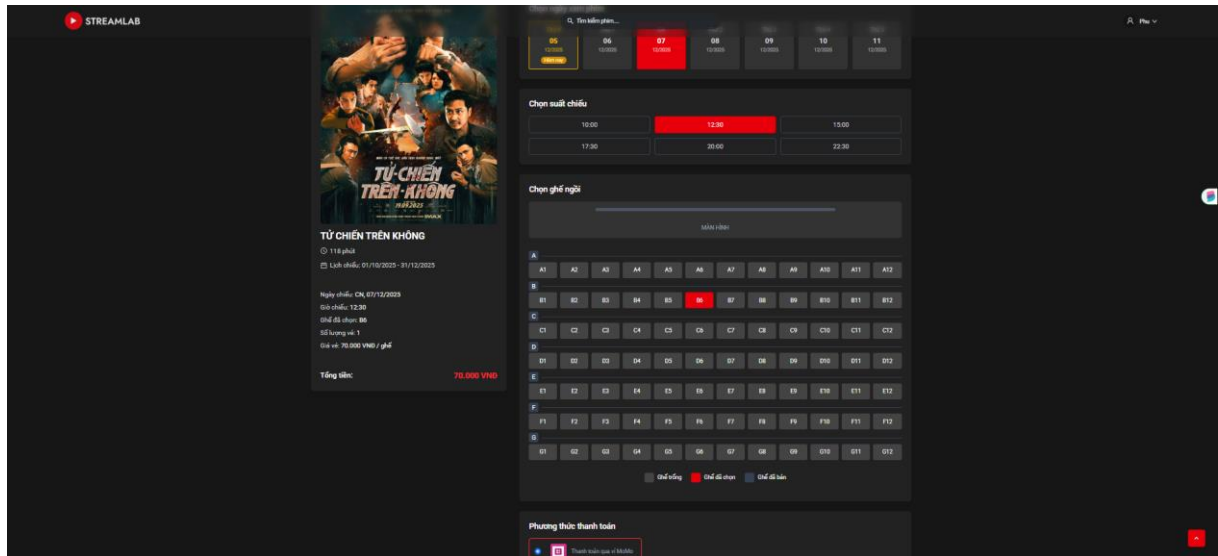
Mục đích của chức năng đặt vé cung cấp cho người dùng trải nghiệm đặt vé trực tuyến có thể chọn suất chiếu ngày, giờ, ghế ngồi theo sở thích mong muốn. Sau khi đặt vé thanh toán thì sẽ có thông báo rõ ràng cho người sử dụng. người dùng thao tác chọn phim sau đó chọn đặt vé thì ở đây đầu tiên suất chiếu ngày, giờ hiện ra sau khi thao tác chọn ngày giờ thì sơ đồ ghế hiện lên và người dùng chọn số ghế phù hợp và thông tin vé và phương thức thanh toán sẽ được đưa ra.



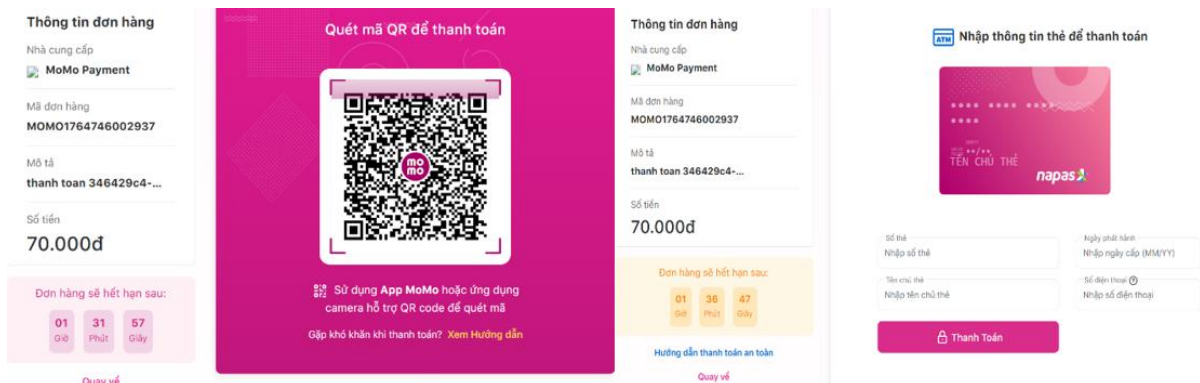
Hình 4.46 Giao diện phim



Hình 4.47 Giao diện chọn suất chiếu



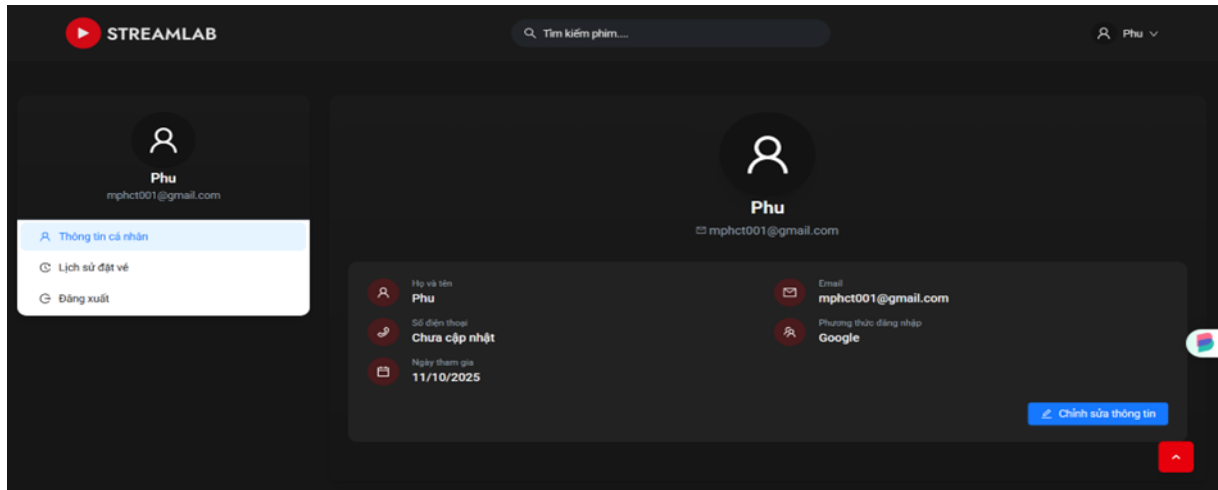
Hình 4.48 Giao diện chọn ghế và phương thức thanh toán



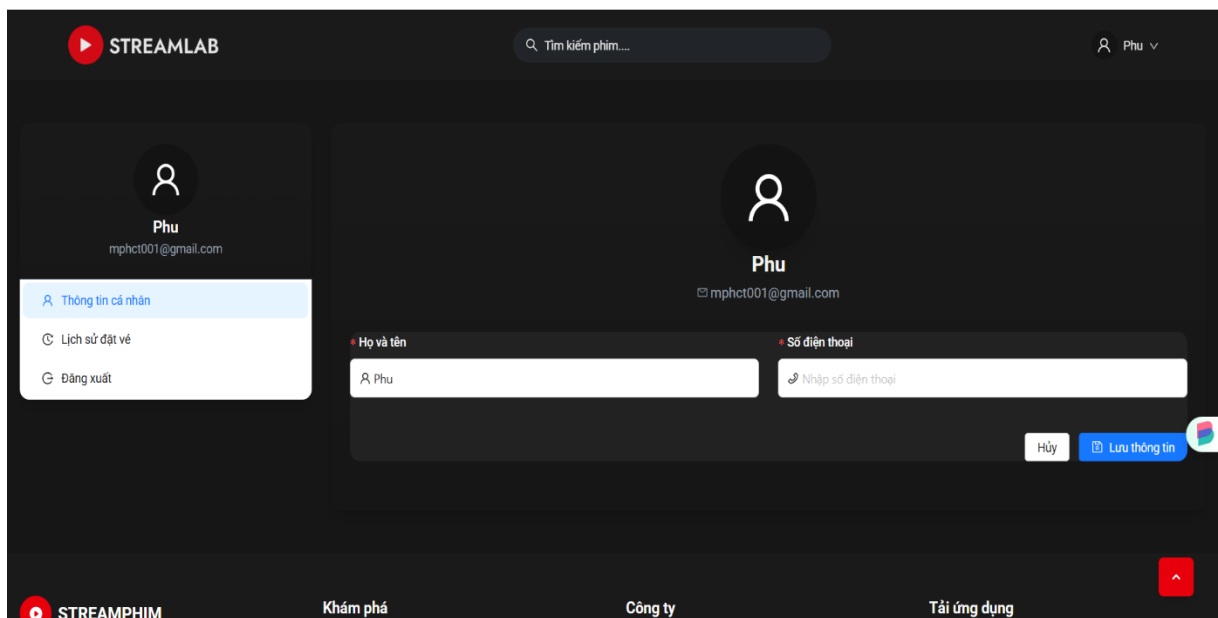
Hình 4.49 Tích hợp thanh toán online Momo

Giao diện thông tin cá nhân và chức năng cập nhật thông tin cá nhân user

Mục đích của giao diện thông tin cá nhân và chức năng cập nhật thông tin user là cung cấp cho người dùng một giao diện trực quan để quản lý thông tin cá nhân của mình. Khi đã có tài khoản trên hệ thống sẽ hiển thị góc phải màn hình avatar và tên của người dùng chỉ cần click vào đó trang thông tin sẽ hiện ra và khi muốn thao tác đổi thông tin chỉ cần click vào nút chỉnh sửa thông tin thì người dùng sẽ được sửa thông tin.



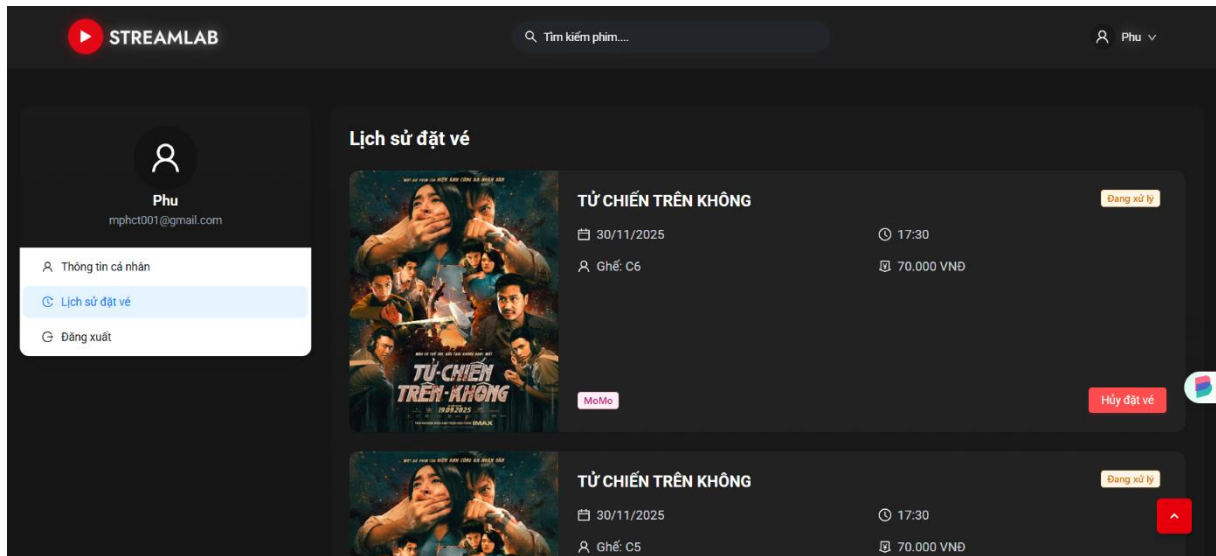
Hình 4.50 Giao diện thông tin cá nhân



Hình 4.51 Giao diện chức năng cập nhật thông tin

Giao diện lịch sử đặt vé

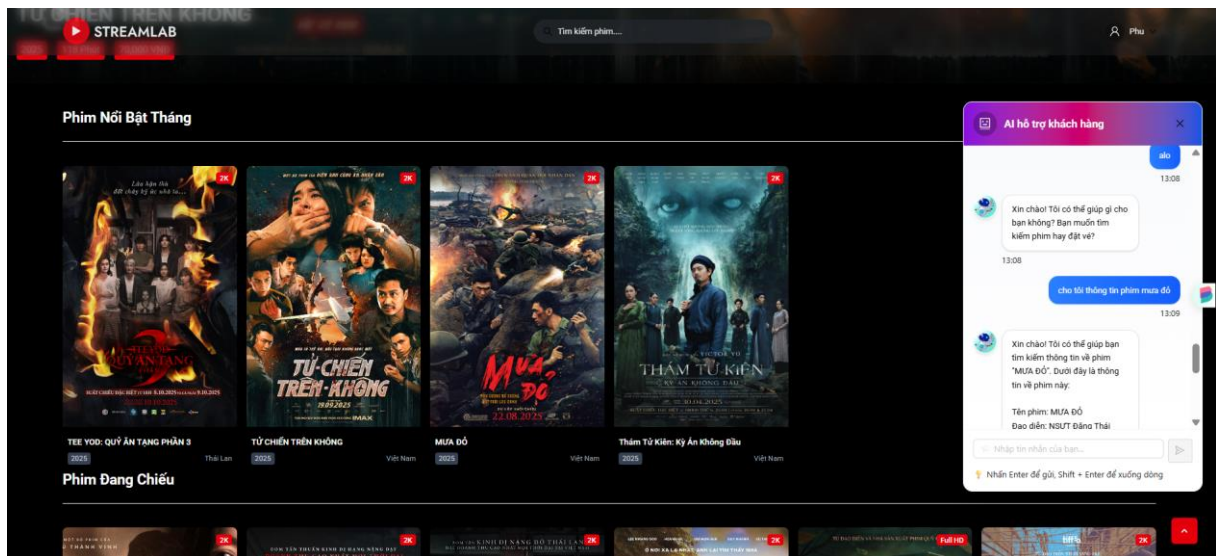
Mục đích của giao diện lịch sử đặt vé là cung cấp cho người dùng một nơi để xem lại toàn bộ các giao dịch đặt vé đã thực hiện. Người dùng có thể dễ dàng kiểm tra thông tin chi tiết của từng vé đã đặt, khi đã đặt vé và có thông báo thành công thì user cần truy cập vào thông tin cá nhân ở đây sẽ có mục lịch sử đặt vé và có thể thao tác xem lịch sử dễ dàng.



Hình 4.52 Giao diện lịch sử đặt vé

Giao diện messagebot AI

Mục đích của giao diện này là mang lại sự thân thiện gần gũi của hệ thống tới khách hàng giúp họ tăng sự tương tác với hệ thống, khi muốn tiếp cận thông tin phim mới ở trang chủ sẽ có phần icon tin nhắn người dùng vào đây để thực hiện chat hỏi AI những phim mới và sắp có trên hệ thống.



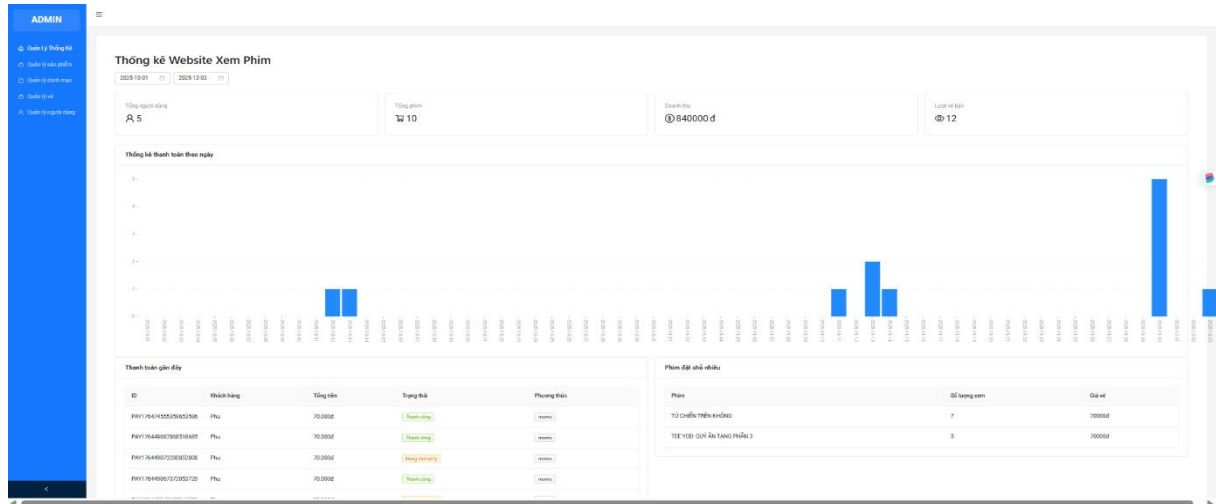
Hình 4.53 Giao diện messagebot ở trang chủ

4.2.2 Giao diện của Admin

Giao diện trang quản lý thống kê

Mục đích của trang chủ admin là cung cấp một giao diện tổng quan để quản lý và giám sát toàn bộ hệ thống. Trang chủ admin thường hiển thị các thông tin quan trọng

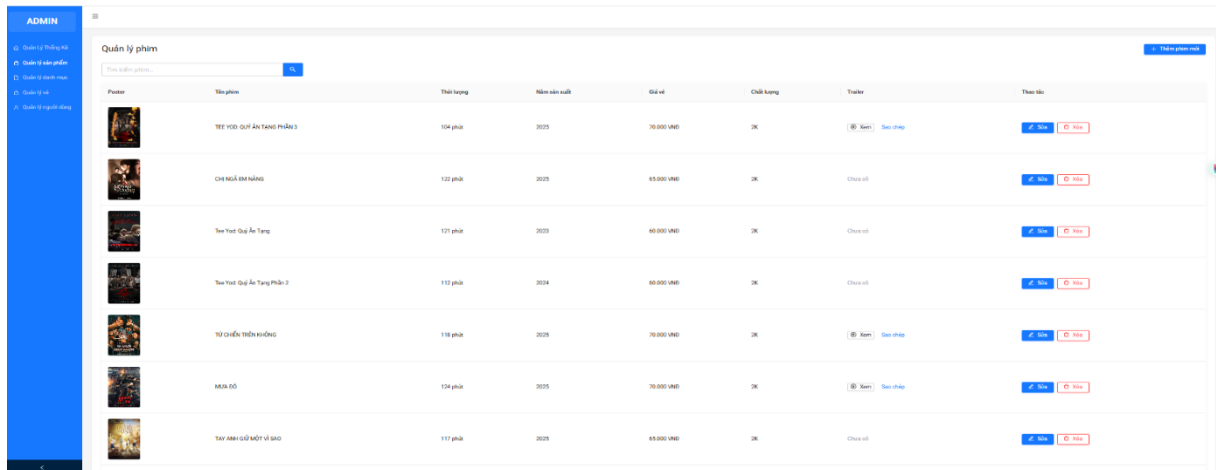
như số lượng người dùng, số lượng phim, trạng thái đơn đặt vé, doanh thu, lượt vé bán và các thông báo hệ thống. Giao diện này giúp admin dễ dàng theo dõi hoạt động, thực hiện các thao tác quản lý như thêm, sửa, xóa dữ liệu, và xử lý các vấn đề phát sinh một cách hiệu quả.



Hình 4.54 Giao diện trang quản lý thống kê hệ thống

Giao diện trang quản lý phim

Mục đích của trang quản lý phim là cung cấp cho admin một giao diện trực quan để thực hiện các thao tác quản lý danh sách phim trong hệ thống. Admin có thể dễ dàng thêm mới, chỉnh sửa, hoặc xóa phim, cũng như xem thông tin chi tiết về từng bộ phim như tên, thời lượng, năm sản xuất, giá vé, chất lượng, và trailer. Trang này giúp admin kiểm soát và cập nhật dữ liệu phim một cách hiệu quả, đảm bảo rằng thông tin hiển thị đến người dùng luôn chính xác và đầy đủ.



Poster	Tên phim	Thời lượng	Năm sản xuất	Giá vé	Chất lượng	Trailer	Thao tác
	TÔI YÊU QUÊ AN TẠNG PHẦN 3	104 phút	2025	70.000 VNĐ	2K	Xem	Thêm Sửa Xóa
	CHỊ NGƯỜI KHÁNH	122 phút	2025	65.000 VNĐ	2K	Xem	Thêm Sửa Xóa
	Tên Việt Quê Ấm Tươi	121 phút	2023	60.000 VNĐ	2K	Xem	Thêm Sửa Xóa
	Tên Việt Quê Ấm Tươi Phần 2	110 phút	2024	60.000 VNĐ	2K	Xem	Thêm Sửa Xóa
	TÔI CHỈ CHÉO TRÊN KINH ĐO	118 phút	2025	70.000 VNĐ	2K	Xem	Thêm Sửa Xóa
	MÙA ĐỎ	124 phút	2025	70.000 VNĐ	2K	Xem	Thêm Sửa Xóa
	TAY ANH GIỎI NHẤT VIỆT NAM	117 phút	2025	65.000 VNĐ	2K	Xem	Thêm Sửa Xóa

Hình 4.55 Giao diện quản lý phim

Giao diện quản lý danh mục phim

Mục đích của trang quản lý danh mục phim là cung cấp cho admin một giao diện trực quan để quản lý các danh mục phim trong hệ thống. Admin có thể dễ dàng thêm mới, chỉnh sửa, hoặc xóa danh mục, cũng như xem tổng số phim thuộc từng danh mục. Trang này giúp tổ chức và phân loại phim một cách hiệu quả, đảm bảo rằng hệ thống hiển thị thông tin phim theo từng danh mục rõ ràng, hỗ trợ người dùng tìm kiếm và truy cập nội dung một cách thuận tiện.

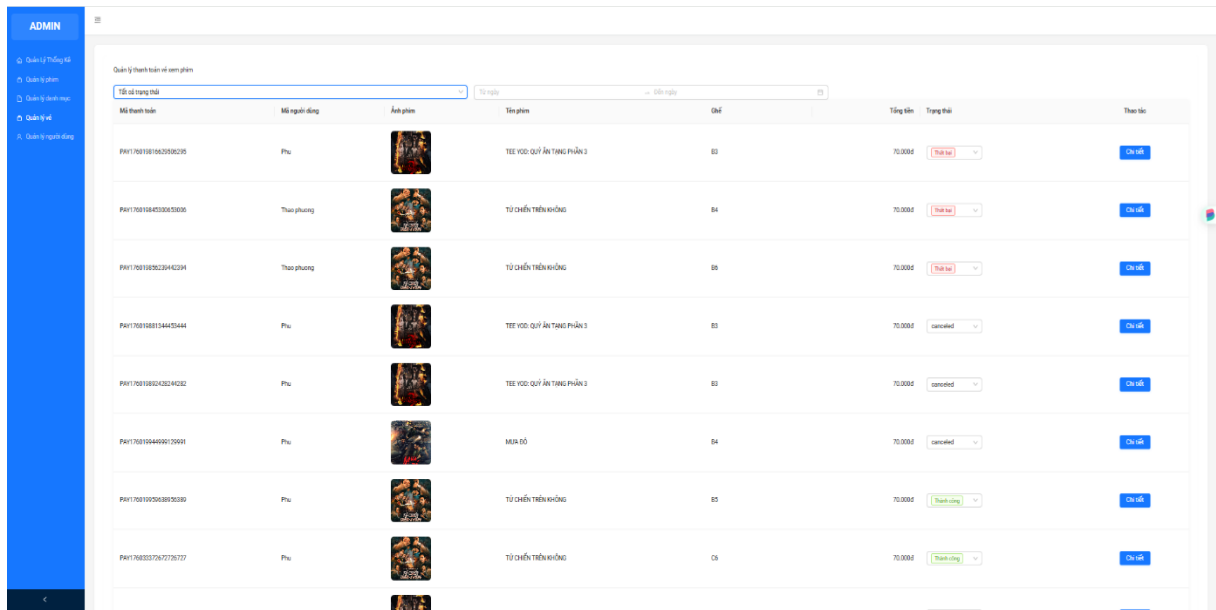


ID	Tên danh mục	Tổng số phim	Thao tác
1	Phim nổi bật tháng	4	Thêm Sửa Xóa
2	Phim đang chiếu	6	Thêm Sửa Xóa

Hình 4.56 Giao diện quản lý danh mục phim

Giao diện quản lý vé

Mục đích của trang quản lý vé là cung cấp cho admin một giao diện trực quan để theo dõi và quản lý các giao dịch đặt vé trong hệ thống. Admin có thể xem chi tiết thông tin vé, bao gồm mã thanh toán, tên người dùng, tên phim, ghế ngồi, tổng tiền, và trạng thái thanh toán (thành công, thất bại, hoặc đã hủy). Trang này giúp admin dễ dàng kiểm soát các giao dịch, xử lý các vấn đề liên quan đến thanh toán, và đảm bảo tính minh bạch trong quản lý hệ thống.

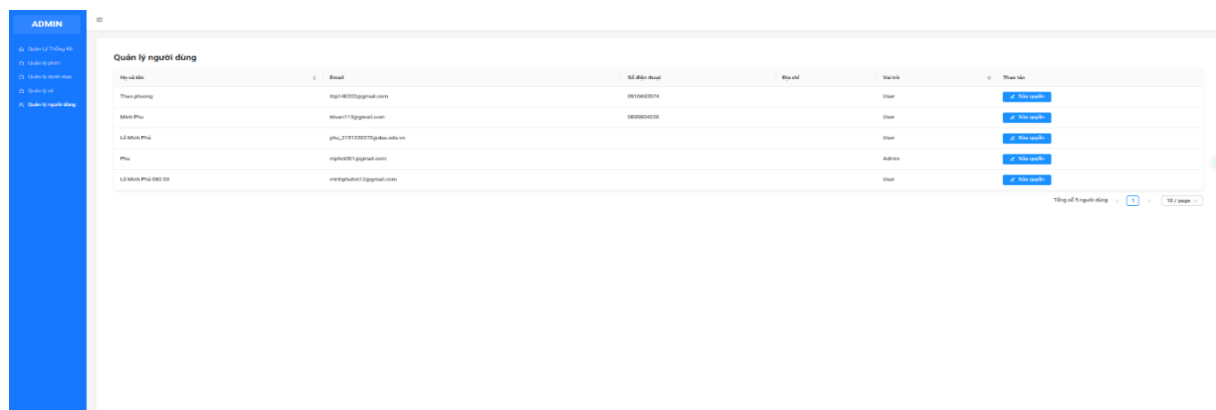


Mã thanh toán	Mã người dùng	Ảnh phim	Tên phim	Ghế	Tổng tiền	Trạng thái	Thao tác
PM170118163500295	Phu		TIE VOI QUY AN TANG PHAN 3	B0	70,000đ	Thất bại	Chi tiết
PM1701181635005006	Thao phuong		TU CHIEN THIEN KHONG	B4	70,000đ	Thất bại	Chi tiết
PM17011816350042391	Thao phuong		TU CHIEN THIEN KHONG	B8	70,000đ	Thất bại	Chi tiết
PM17011816350045444	Phu		TIE VOI QUY AN TANG PHAN 3	B0	70,000đ	cancelled	Chi tiết
PM17011816350042402	Phu		TIE VOI QUY AN TANG PHAN 3	B0	70,000đ	cancelled	Chi tiết
PM17011816350042591	Phu		MEKA BO	B4	70,000đ	cancelled	Chi tiết
PM17011816350050090	Phu		TU CHIEN THIEN KHONG	B0	70,000đ	Thành công	Chi tiết
PM17011816350050090	Phu		TU CHIEN THIEN KHONG	C8	70,000đ	Thành công	Chi tiết

Hình 4.57 Giao diện quản lý vé

Giao diện quản lý người dùng

Mục đích của trang quản lý người dùng là cung cấp cho admin một giao diện trực quan để theo dõi và quản lý thông tin tài khoản của người dùng trong hệ thống. Admin có thể xem danh sách người dùng, bao gồm các thông tin như họ tên, email, số điện thoại, địa chỉ, và vai trò phân quyền (User hoặc Admin).



Họ và tên	Email	Số điện thoại	Địa chỉ	Vai trò	Thao tác
Thao phuong	thao1402@gmail.com	09158600716	Đà Nẵng	User	Chi tiết
Minh Phu	minh171@gmail.com	0909040033	Đà Nẵng	User	Chi tiết
Lê Minh Phú	phu_211220027@gmail.com		Đà Nẵng	User	Chi tiết
Phu	ngphu4007@gmail.com		Đà Nẵng	Admin	Chi tiết
Lê Minh Phú 080 08	minhphut07@gmail.com		Đà Nẵng	User	Chi tiết

Hình 4.58 Giao diện quản lý người dùng

4.3 Kết luận chương 4

Trong Chương 4, hệ thống đặt vé xem phim trực tuyến đã được xây dựng và triển khai thành công với đầy đủ các giao diện và chức năng dành cho cả người dùng và quản trị viên. Thông qua quá trình thực nghiệm, em đã tiến hành thiết kế, cài đặt và kiểm thử các thành phần của hệ thống dựa trên yêu cầu nghiệp vụ thực tế, đảm bảo tính trực quan, dễ sử dụng và đáp ứng đúng mục tiêu của đề tài.

Về phía người dùng, các chức năng cốt lõi như đăng ký, đăng nhập, hiển thị trang chủ, xem thông tin phim, đặt vé, thanh toán, quản lý thông tin cá nhân, xem lịch sử đặt vé và tìm kiếm phim bằng AI Gemini đều được hiện thực hóa rõ ràng và hoạt động ổn định. Đặc biệt, luồng xử lý đặt vé và thanh toán qua MoMo/VNPay được tích hợp đồng bộ, giúp nâng cao trải nghiệm người dùng và đảm bảo tính an toàn – minh bạch trong giao dịch.

Về phía admin, các chức năng quản trị như quản lý thống kê, phim, danh mục, vé và người dùng được xây dựng đầy đủ, hỗ trợ admin giám sát và điều hành hệ thống một cách hiệu quả. Các API phục vụ cho việc thêm, sửa, xóa, tra cứu dữ liệu đều được cài đặt hợp lý và tương tác tốt với giao diện quản trị.

Nhìn chung, các thực nghiệm trong chương đã chứng minh rằng hệ thống vận hành đúng theo thiết kế, đáp ứng được các yêu cầu chức năng đã đặt ra ban đầu. Đây là nền tảng quan trọng để tiếp tục đánh giá, tối ưu và hoàn thiện hệ thống ở các chương tiếp theo.

KẾT LUẬN

Đề tài “Xây dựng hệ thống quản lý đặt vé xem phim trực tuyến” đã hoàn thành các mục tiêu đề ra, từ khảo sát yêu cầu thực tế đến thiết kế, xây dựng và triển khai hệ thống. Hệ thống đã số hóa quy trình đặt vé truyền thống, mang lại trải nghiệm thuận tiện cho khách hàng và hỗ trợ quản lý hiệu quả cho rạp chiếu phim.

Về mặt chức năng, hệ thống cho phép người dùng thực hiện đầy đủ các tác vụ như đăng ký, đăng nhập, xem thông tin phim, đặt vé, thanh toán trực tuyến, quản lý thông tin cá nhân, xem lịch sử giao dịch, hỗ trợ chatbot AI. Phần quản trị (Admin) hỗ trợ các thao tác quan trọng như quản lý phim, danh mục, vé, người dùng và xem thống kê. Các chức năng đều được hiện thực hóa dựa trên mô hình client-server, kết hợp ReactJS ở frontend và NodeJS ở backend, đảm bảo tính ổn định, bảo mật và tốc độ xử lý.

Kết quả thực nghiệm cho thấy hệ thống hoạt động đúng theo yêu cầu nghiệp vụ, giao diện trực quan, dễ sử dụng, logic xử lý rõ ràng, đồng thời tích hợp các công nghệ hiện đại như thanh toán trực tuyến MoMo/VNPay và tìm kiếm thông minh bằng AI Gemini, chatbot AI.

Qua quá trình thực hiện và phát triển hệ thống đề tài còn có hạn chế và sẽ phải hướng phát triển lâu dài trong tương lai:

- Cần phát triển chức năng quản lý suất chiếu mở rộng và ổn định hơn.
- Hạn chế về hiệu năng nếu số lượng người dùng tăng cao cần tối ưu hóa cơ sở dữ liệu, sử dụng caching(Redis, Memcached) kiến trúc microservices để giảm tải.
- Tăng cường bảo mật mã hóa dữ liệu, phân quyền chặt chẽ.
- Cải thiện trải nghiệm người dùng responsive design, thông báo đẩy.

Kết thúc đồ án không chỉ đáp ứng được đúng mục tiêu ban đầu của đề tài. Website đã hình thành nền tảng vững chắc để tiếp tục phát triển thêm các tính năng nâng cao, hướng tới một giải pháp quản lý hệ thống đặt vé trực tuyến an toàn và chuyên nghiệp hơn.

TÀI LIỆU THAM KHẢO

- [1] "Client - Server overview," Mozilla Corporation's, 23 6 2025. [Online]. Available: https://developer.mozilla.org/en-US/docs/Learn_web_development/Extensions/Server-side/First_steps/Client-Server_overview. [Accessed 10 11 2025].
- [2] "React Documentation," React,dev, 2023. [Online]. Available: <https://react.dev/learn>. [Accessed 10 11 2025].
- [3] "Vite Overview," Vite.dev, 2024. [Online]. Available: <https://vite.dev/guide/>. [Accessed 12 11 2025].
- [4] "React router," reactrouter.com, 2024. [Online]. Available: <https://reactrouter.com/home>. [Accessed 11 11 2025].
- [5] "SCSS Overview," sass-lang.com, 2019. [Online]. Available: <https://sass-lang.com/documentation/modules/>. [Accessed 12 11 2025].
- [6] "Introduction to Node.js," nodejs.org, 2023. [Online]. Available: <https://nodejs.org/en/learn/getting-started/introduction-to-nodejs#introduction-to-nodejs>. [Accessed 12 11 2025].
- [7] "Overview of the MySQL Database Management System," dev.mysql.com, [Online]. Available: https://www.w3schools.com/mysql/mysql_intro.asp. [Accessed 12 11 2025].