

**TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ
KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO KẾT THÚC MÔN HỌC
CÔNG NGHỆ PHẦN MỀM**

**ĐỀ TÀI:
XÂY DỰNG HỆ THỐNG
ĐẶT VÉ XEM PHIM TRỰC TUYẾN**

Giáo viên hướng dẫn:

Ts. Nguyễn Bảo Ân

Nhóm sinh viên thực hiện:

Phạm Hữu Luân (110122016 – DA22TTA)

Nguyễn Hữu Anh (110122033 – DA22TTA)

Lâm Thanh Đỉnh (110122051 – DA22TTA)

Trà Vinh, tháng 7 năm 2025

LỜI CẢM ƠN

Lời đầu tiên, em xin trân trọng cảm ơn giảng viên Nguyễn Bảo Ân - người đã trực tiếp chỉ bảo, hướng dẫn nhóm em trong quá trình hoàn thành bài đồ án môn học này.

Nhóm em cũng xin được gửi lời cảm ơn đến quý thầy, cô giáo trường Kỹ thuật và Công nghệ, đặc biệt là các thầy, cô khoa Công nghệ Thông tin - những người đã truyền lửa và giảng dạy kiến thức cho em suốt thời gian qua.

Mặc dù đã có những đầu tư nhất định trong quá trình làm bài song cũng khó có thể tránh khỏi những sai sót, chúng em kính mong nhận được ý kiến đóng góp của quý thầy cô để bài báo cáo được hoàn thiện hơn.

Nhóm chúng em xin chân thành cảm ơn!

Trà Vinh, ngày ... tháng 7 năm 2025

Nhóm thực hiện:

Phạm Hữu Luân - 110122016

Nguyễn Hữu Anh - 110122033

Lâm Thành Đỉnh - 110122051

Sinh viên ký và ghi rõ họ và tên

Sinh viên 1

Sinh viên 2

Sinh viên 3

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN

MỤC LỤC

CHƯƠNG 1: GIỚI THIỆU	6
1.1 Lý do chọn đề tài	6
1.2 Tên đề tài và chủ đề	6
1.3 Mục tiêu ứng dụng.....	6
1.4 Tổng quan về công nghệ phần mềm.....	7
CHƯƠNG 2: PHÂN TÍCH YÊU CẦU HỆ THỐNG	8
2.1 Các chức năng chính của hệ thống (Functional Requirements)	8
2.1.1 Chức năng dành cho khách hàng	8
2.1.2 Chức năng dành cho quản trị viên	9
2.2 Các yêu cầu phi chức năng (Non-functional Requirements).....	10
2.2.1 Hiệu suất	10
2.2.2 Quyền riêng tư	10
2.2.3 Tính sẵn sàng	10
2.2.4 Khả năng mở rộng	10
2.2.5 Tính dễ sử dụng	10
2.2.6 Tính duy trì và phát triển	11
2.2.7 Tính bảo mật và an toàn.....	11
CHƯƠNG 3: THIẾT KẾ HỆ THỐNG	12
3.1 Kiến trúc tổng thể	12
3.1.1 Client (Giao diện người dùng Frontend – ReactJS)	12
3.1.2 Server (Node.js + ExpressJS)	12
3.1.3 Cơ sở dữ liệu (MySQL).....	12
3.2 Thiết kế cơ sở dữ liệu	14
3.2.1 Mô hình thực thể - quan hệ (ERD)	14
3.2.2 Sơ đồ quan hệ dữ liệu MySQL	16
3.3 Thiết kế API.....	18
3.3.1 Tổng quan	18
3.3.2 Nhóm endpoint chính	18
3.3.3 Cấu trúc request/response	19
3.4 Thiết kế giao diện (UI/UX)	23
3.4.1 Trang chủ website.....	23
3.4.2 Giao diện đăng nhập, đăng ký	24

3.4.3	Trang danh mục phim theo trạng thái.....	27
3.4.4	Giao diện trang thông tin chi tiết phim.....	28
3.4.5	Giao diện cho chức năng tìm kiếm	30
3.4.6	Giao diện trang cập nhật thông tin cá nhân	31
3.4.7	Giao diện trang đổi mật khẩu.....	32
3.4.8	Giao diện trang lịch chiếu.....	33
3.4.9	Giao diện trang chọn ghế và lịch sử đặt vé.....	34
3.4.10	Giao diện các trang quản trị	38
CHƯƠNG 4: TRIỂN KHAI VÀ CÔNG NGHỆ SỬ DỤNG	44	
4.1	Danh sách công nghệ sử dụng	44
4.2	Quy trình CI/CD với GitHub Actions	45
4.3	Cấu hình Docker và quy trình triển khai ứng dụng	45
4.3.1	Cấu hình Docker	45
4.3.2	Quy trình triển khai	46
CHƯƠNG 5: QUẢN LÝ DỰ ÁN	47	
5.1	Mô hình phát triển phần mềm	47
5.2	Sử dụng Jira để lập kế hoạch và theo dõi tiến độ	47
5.2.1	Sprint 1.....	47
5.2.2	Sprint 2.....	48
5.2.3	Sprint 3.....	49
5.2.4	Sprint 4.....	51
5.2.5	Sprint 5.....	52
5.2.6	Sprint 6.....	53
5.3	Phân công nhiệm vụ của từng thành viên trong nhóm	55
CHƯƠNG 6: KIỂM THỬ	58	
6.1	Chiến lược kiểm thử	58
6.2	Kết quả kiểm thử	58
CHƯƠNG 7: ĐÁNH GIÁ VÀ KẾT LUẬN.....	63	
7.1	Kết quả đạt được.....	63
7.2	Khó khăn gặp phải	63
7.3	Bài học kinh nghiệm được rút ra	64
7.4	Đề xuất cải thiện	64
TÀI LIỆU THAM KHẢO.....	65	

DANH MỤC HÌNH ẢNH

Hình 2.1. Sơ đồ use-case chức năng.....	8
Hình 3.1. Sơ đồ kiến trúc hệ thống.....	13
Hình 3.2. Sơ đồ ERD được thiết kế trong PowerDesign.....	14
Hình 3.3. Sơ đồ cơ sở dữ liệu quan hệ MySQL	17
Hình 3.4. Request gửi dữ liệu đăng nhập	20
Hình 3.5. Response khi đăng nhập thành công	20
Hình 3.6. Response khi lấy danh sách phim đang chiếu thành công	21
Hình 3.7. Request cập nhật thông tin một rạp phim.....	21
Hình 3.8. Response sau khi cập nhật rạp thành công	22
Hình 3.9. Response sau khi xóa rạp thành công.....	22
Hình 3.10. Bản thiết kế giao diện trang chủ người dùng bằng Figma	23
Hình 3.11. Giao diện trang chủ website	24
Hình 3.12. Bản thiết kế form đăng ký bằng Figma	25
Hình 3.13. Bản thiết kế form đăng nhập bằng Figma	25
Hình 3.14. Giao diện trang đăng ký tài khoản.....	26
Hình 3.15. Giao diện trang đăng nhập.....	26
Hình 3.16. Bản thiết kế giao diện danh sách phim theo danh mục bằng Figma	27
Hình 3.17. Giao diện trang phim đang chiếu	28
Hình 3.18. Bản thiết kế giao diện thông tin chi tiết phim bằng Figma	29
Hình 3.19. Giao diện trang thông tin chi tiết phim.....	29
Hình 3.20. Tab trailer ở giao diện chi tiết phim	30
Hình 3.21. Bản thiết kế giao diện chức năng tìm kiếm phim bằng Figma	30
Hình 3.22. Giao diện cho chức năng kết quả tìm kiếm phim	31
Hình 3.23. Bản thiết kế giao diện xem và cập nhật thông tin cá nhân bằng Figma	31
Hình 3.24. Giao diện trang cập nhật thông tin cá nhân	32
Hình 3.25. Bản thiết kế giao diện đổi mật khẩu bằng Figma	32
Hình 3.26. Giao diện trang đổi mật khẩu	33
Hình 3.27. Bản thiết kế giao diện lịch chiếu bằng Figma	33
Hình 3.28. Giao diện trang xem lịch chiếu.....	34
Hình 3.29. Modal thông báo cần phải đăng nhập trước	34
Hình 3.30. Bản thiết kế giao diện chọn ghế bằng Figma	35

Hình 3.31. Giao diện trang chọn ghế để đặt vé	36
Hình 3.32. Bản thiết kế giao diện lịch sử đặt vé bằng Figma	37
Hình 3.33. Giao diện trang lịch sử đặt vé	37
Hình 3.34. Bản thiết kế giao diện báo cáo thông kê bằng Figma.....	38
Hình 3.35. Giao diện trang thống kê	39
Hình 3.36. Bản thiết kế giao diện quản lý chung bằng Figma	40
Hình 3.37. Giao diện quản lý phim	40
Hình 3.38. Giao diện quản lý suất chiếu	40
Hình 3.39. Modal hiện thông tin chi tiết người dùng	41
Hình 3.40. Modal xác nhận xóa một phim	41
Hình 3.41. Giao diện trang thêm mới phim.....	42
Hình 3.42. Giao diện trang cập nhật thông tin phim	43
Hình 5.1. Sprint Backlog của Sprint 1.....	48
Hình 5.2. Burndown chart Sprint 1	48
Hình 5.3. Sprint Backlog của Sprint 2.....	49
Hình 5.4. Burndown chart Sprint 2	49
Hình 5.5. Sprint Backlog của Sprint 3.....	50
Hình 5.6. Burndown chart Sprint 3	50
Hình 5.7. Sprint Backlog của Sprint 4.....	51
Hình 5.8. Burndown chart Sprint 4	52
Hình 5.9. Sprint Backlog của Sprint 5.....	53
Hình 5.10. Burndown chart Sprint 5	53
Hình 5.11. Sprint Backlog của Sprint 6.....	54
Hình 5.12. Burndown chart Sprint 6	54
Hình 6.1. Test API đăng nhập bằng Postman.....	58
Hình 6.2. Postman trả về status 401 khi thiếu token trong header	59
Hình 6.3. Postman trả về status 200 khi có header token.....	59
Hình 6.4. Postman trả về lỗi 403 khi token không phải của admin.....	60
Hình 6.5. Postman trả về status 200 khi cập nhật người dùng thành công	61
Hình 6.6. Postman trả về status 200 khi xóa một chuỗi rạp thành công	62

DANH MỤC BẢNG BIỂU

Bảng 1. Mô tả các quan hệ giữa các thực thể trong ERD	15
Bảng 2. Cấu hình truy cập API.....	18
Bảng 3. Danh sách nhóm endpoint chính.....	18
Bảng 4. Bảng phân công nhiệm vụ các Sprint 1, 2 và 3 trong Jira	55
Bảng 5. Bảng phân công nhiệm vụ các Sprint 4, 5 và 6 trong Jira	56

CHƯƠNG 1: GIỚI THIỆU

1.1 Lý do chọn đề tài

Trong thời đại số hóa, nhu cầu giải trí ngày càng tăng cao, đặc biệt là việc xem phim tại rạp. Tuy nhiên, hình thức mua vé truyền thống còn tồn tại nhiều hạn chế như mất thời gian xếp hàng, không chủ động trong việc chọn ghế và dễ xảy ra tình trạng quá tải tại quầy vé. Trong khi đó, người dùng hiện nay có xu hướng ưu tiên những trải nghiệm nhanh chóng, tiện lợi thông qua các nền tảng số.

Xuất phát từ thực tế đó, nhóm quyết định lựa chọn đề tài **thiết kế website đặt vé xem phim** nhằm giải quyết những bất cập nói trên, đồng thời tạo ra một hệ thống ứng dụng thực tế, có khả năng triển khai trong môi trường kinh doanh.

Đây cũng là cơ hội để nhóm vận dụng các kiến thức đã học như lập trình web, thiết kế giao diện, quản lý cơ sở dữ liệu và triển khai hệ thống, phần mềm, từ đó nâng cao kỹ năng và kinh nghiệm trong phát triển phần mềm.

1.2 Tên đề tài và chủ đề

- Đề tài: **Movie Ticket Booking System** – Xây dựng hệ thống đặt vé xem phim trực tuyến.
- Chủ đề: Phát triển một hệ thống website cho phép người dùng dễ dàng tra cứu thông tin phim, lịch chiếu, chọn chỗ ngồi và thanh toán trực tuyến. Dự án hướng đến việc đơn giản hóa quá trình đặt vé, nâng cao trải nghiệm người dùng, đồng thời hỗ trợ các rạp phim trong việc quản lý suất chiếu, ghế ngồi và doanh thu một cách hiệu quả. Đây là giải pháp công nghệ thiết thực nhằm hiện đại hóa quy trình vận hành của rạp chiếu phim và thúc đẩy sự phát triển của ngành điện ảnh tại Việt Nam.

1.3 Mục tiêu ứng dụng

Đối với người dùng, hệ thống giúp đặt vé nhanh chóng, cho phép tra cứu thông tin phim, giờ chiếu, chọn ghế trực quan và thanh toán trực tuyến, từ đó tiết kiệm thời gian và tránh phải xếp hàng tại rạp.

Đối với rạp chiếu phim, hệ thống hỗ trợ quản lý lịch chiếu, tình trạng ghế và vé bán theo thời gian thực, giảm tải cho quầy vé và tối ưu kế hoạch vận hành.

Đối với nhà quản lý, hệ thống cung cấp báo cáo doanh thu, hỗ trợ phân tích hành vi khách hàng và dễ dàng mở rộng khi tăng lượng truy cập hoặc thêm cụm rạp.

1.4 Tổng quan về công nghệ phần mềm

Công nghệ phần mềm (Software Engineering) là lĩnh vực nghiên cứu và ứng dụng các phương pháp và công cụ để phát triển, vận hành và bảo trì phần mềm hiệu quả, ổn định và dễ mở rộng. Một nội dung cốt lõi là vòng đời phát triển phần mềm (SDLC), gồm các giai đoạn: khảo sát, phân tích, thiết kế, lập trình, kiểm thử, triển khai và bảo trì. Quy trình này giúp đảm bảo phần mềm đáp ứng đúng yêu cầu, có tính hệ thống và dễ nâng cấp trong tương lai.

Trong dự án này, công nghệ phần mềm được áp dụng xuyên suốt từ phân tích yêu cầu người dùng đến triển khai và vận hành hệ thống. Dự án đáp ứng các nhu cầu cụ thể:

- Người dùng cần một nền tảng đặt vé tiện lợi, có thể xem lịch chiếu, chọn ghế và thanh toán nhanh chóng.
- Rạp chiếu phim cần công cụ quản lý lịch chiếu, tình trạng ghế ngồi và giảm tải cho nhân viên bán vé.
- Nhà quản lý cần hệ thống hỗ trợ thông kê doanh thu, phân tích dữ liệu đặt vé và có khả năng mở rộng khi số lượng rạp hoặc người dùng tăng lên.

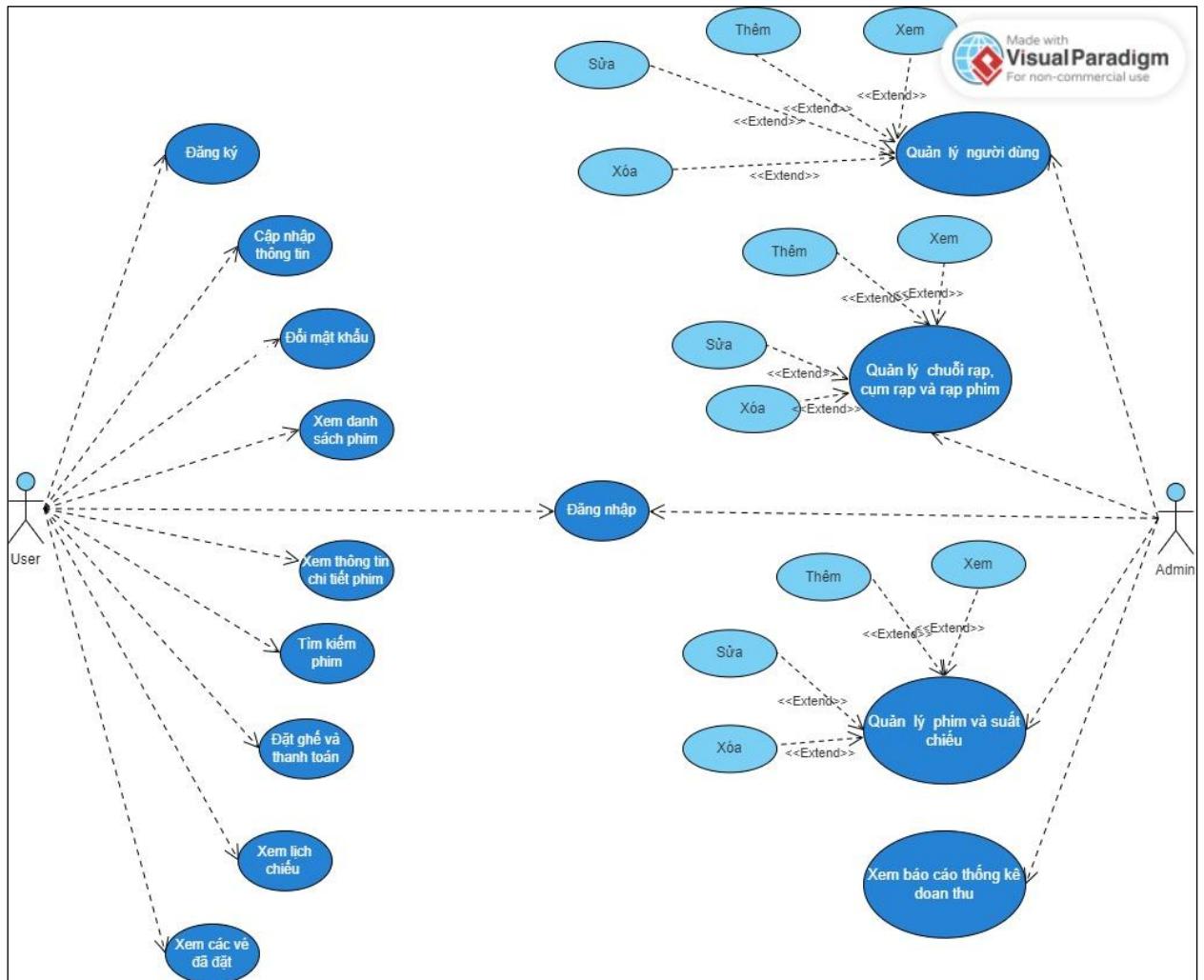
Để đáp ứng các yêu cầu trên, nhóm đã lựa chọn kiến trúc **client–server**, tách biệt rõ ràng giữa giao diện người dùng (frontend) và phần xử lý nghiệp vụ (backend). Các thành phần này được container hóa bằng Docker, dễ dàng triển khai và quản lý. Ngoài ra, backend được thiết kế theo hướng **module service-based**, gần với mô hình **microservices** đơn giản, giúp từng chức năng như người dùng, phim, suất chiếu có thể phát triển và bảo trì độc lập. Việc triển khai bằng **Docker Compose** trên môi trường cloud cũng đảm bảo hiệu suất cao, dễ mở rộng và tiết kiệm chi phí vận hành.

Tổng thể, việc ứng dụng các nguyên lý của công nghệ phần mềm vào dự án giúp nhóm xây dựng được một hệ thống đặt vé hoàn chỉnh, có tính thực tiễn cao và phù hợp với nhu cầu phát triển trong bối cảnh số hóa ngành giải trí tại Việt Nam.

CHƯƠNG 2: PHÂN TÍCH YÊU CẦU HỆ THỐNG

2.1 Các chức năng chính của hệ thống (Functional Requirements)

Hệ thống đặt vé xem phim trực tuyến bao gồm hai nhóm chức năng chính: dành cho người dùng (khách hàng) và dành cho quản trị viên (admin).



Hình 2.1. Sơ đồ use-case chức năng

2.1.1 Chức năng dành cho khách hàng

- Đăng ký tài khoản và đăng nhập
 - + Người dùng có thể tạo tài khoản bằng email và mật khẩu.
 - + Có thể đăng nhập để sử dụng các chức năng như đặt vé, xem lịch sử vé.
- Cập nhật thông tin cá nhân và đổi mật khẩu.
- Xem danh sách phim: Hiển thị thông tin các bộ phim đang chiếu hoặc sắp chiếu (tên phim, poster, mô tả, thời lượng, thể loại...).

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

- Xem chi tiết phim: Khi người dùng nhấp vào một phim, sẽ hiển thị các thông tin chi tiết như trailer, nội dung, diễn viên, suất chiếu...
- Tìm kiếm: Cho phép người dùng tìm phim theo tên, thể loại hoặc thời gian chiếu.
- Xem lịch chiếu theo rạp: Hiển thị các suất chiếu của từng bộ phim tại các cụm rạp, bao gồm giờ chiếu, phòng chiếu và giá vé.
- Chọn suất chiếu và đặt vé
 - + Người dùng chọn suất chiếu mong muốn.
 - + Hiển thị sơ đồ ghế ngồi và người dùng chọn ghế còn trống.
- Thanh toán: Xác nhận đơn đặt vé sau khi thanh toán.
- Xem lại thông tin vé đã đặt: Người dùng có thể xem lịch sử giao dịch và chi tiết vé.

2.1.2 Chức năng dành cho quản trị viên

- Đăng nhập quản trị: Chỉ admin mới được phép truy cập vào giao diện quản trị.
- Quản lý phim:
 - + Xem danh sách phim đang có trong hệ thống.
 - + Thêm, sửa thông tin phim, bao gồm: tiêu đề, nội dung, thời lượng, thể loại, hình ảnh poster, trailer...
 - + Xóa các phim không còn chiếu ở rạp nữa.
- Quản lý chuỗi rạp, cụm rạp và phòng chiếu: Quản lý danh sách và thực hiện các chức năng thêm, sửa xóa cho chuỗi rạp, cụm rạp và phòng chiếu.
- Quản lý suất chiếu: Xem, thêm, sửa, xoá các suất chiếu cho từng phim tại từng rạp và phim, kèm ngày giờ chiếu.
- Quản lý người dùng:
 - + Xem danh sách người dùng đăng ký.
 - + Thêm, sửa, xóa người dùng.
- Xem báo cáo thống kê:
 - + Thống kê số lượng phim theo trạng thái phim giúp hệ thống nắm bắt được số

lượng phim trong toàn hệ thống.

- + Doanh thu và số vé đã bán được lọc theo cụm rạp hoặc theo phim ở một khoảng thời gian bất kì.

2.2 Các yêu cầu phi chức năng (Non-functional Requirements)

Các yêu cầu phi chức năng đề cập đến chất lượng hệ thống, hiệu suất, tính bảo mật và khả năng mở rộng:

2.2.1 Hiệu suất

- Hệ thống cần phản hồi nhanh trong các thao tác tra cứu phim, suất chiếu, chọn ghế và đặt vé.
- Thời gian tải trang không vượt quá 3 giây đối với kết nối mạng trung bình.

2.2.2 Quyền riêng tư

- Dữ liệu cá nhân của người dùng được giới hạn hiển thị và chỉ truy cập bởi chính chủ tài khoản (hướng tiếp cận tương đương nguyên tắc GDPR).
- Chỉ người dùng có vai trò admin mới được phép truy cập giao diện quản trị.

2.2.3 Tính sẵn sàng

- Hệ thống có thể hoạt động liên tục 24/7 (nếu triển khai thật).
- Khả năng xử lý nhiều người dùng cùng lúc đặt vé mà không xung đột dữ liệu.

2.2.4 Khả năng mở rộng

Hệ thống được thiết kế theo mô hình client-server tách biệt (React frontend và Node.js backend), dễ dàng mở rộng thêm chức năng hoặc tích hợp ứng dụng di động trong tương lai. Sử dụng Docker giúp triển khai linh hoạt trên môi trường cloud.

2.2.5 Tính dễ sử dụng

- Giao diện người dùng thân thiện, dễ sử dụng, hiển thị rõ ràng thông tin phim và lịch chiếu.
- Giao diện quản trị đơn giản, phân loại chức năng theo danh mục.

2.2.6 Tính duy trì và phát triển

- Mã nguồn được tổ chức rõ ràng theo mô hình RESTful API, dễ bảo trì và phát triển thêm tính năng.
- Tách biệt rõ frontend – backend để các thành viên nhóm có thể làm việc song song.

2.2.7 Tính bảo mật và an toàn

- Biện pháp kỹ thuật: Sử dụng JWT để xác thực và phân quyền, bcrypt để mã hóa mật khẩu, Sequelize để ngăn SQL injection, đảm bảo an toàn dữ liệu trong các bảng như users và orders.
- Kiểm tra an toàn: Kiểm thử API bằng Postman để phát hiện lỗ hổng (ví dụ: lỗi 400 do thiếu tham số), đảm bảo phản hồi ổn định.
- Tích hợp HTTPS để mã hóa truyền tải dữ liệu và sao lưu định kỳ MySQL để khôi phục khi hệ thống gặp sự cố.

CHƯƠNG 3: THIẾT KẾ HỆ THỐNG

3.1 Kiến trúc tổng thể

Hệ thống được phát triển theo mô hình Client–Server, chia thành 3 thành phần chính:

3.1.1 Client (Giao diện người dùng Frontend – ReactJS)

- Cung cấp giao diện người dùng để tương tác, như xem phim, chọn suất chiếu, chọn ghế, đặt vé, quản lý hệ thống...
- Gửi yêu cầu HTTP tới server thông qua các API RESTful.

3.1.2 Server (Node.js + ExpressJS)

Hệ thống backend được xây dựng theo mô hình phân lớp gồm các thành phần chính như sau:

- Cơ sở dữ liệu :
 - + Lưu trữ thông tin người dùng, phim, suất chiếu, vé, hệ thống rạp, chuỗi rạp, rạp chiếu và ghế ngồi.
 - + Truy xuất dữ liệu thông qua Sequelize ORM giúp dễ dàng thao tác với bảng và mối quan hệ.
- Service Layer: Chứa các logic xử lý nghiệp vụ chính (ví dụ: kiểm tra ghế trống, tính tổng tiền, xử lý thanh toán, lọc phim theo cụm rạp,...). Tách biệt hoàn toàn với controller giúp mã dễ test và tái sử dụng.
- Controller Layer: Nhận request từ phía client thông qua các routes và gọi các hàm xử lý từ service và trả kết quả lại cho frontend dưới dạng JSON.
- Routes (API Endpoints): Định nghĩa các đường dẫn API cho hệ thống như /api/v1/movies, /api/v1/users,... Nó gắn với các hàm controller tương ứng.
- Middleware: Bao gồm các chức năng như kiểm tra xác thực người dùng (JWT), phân quyền (admin/user), kiểm lỗi,...

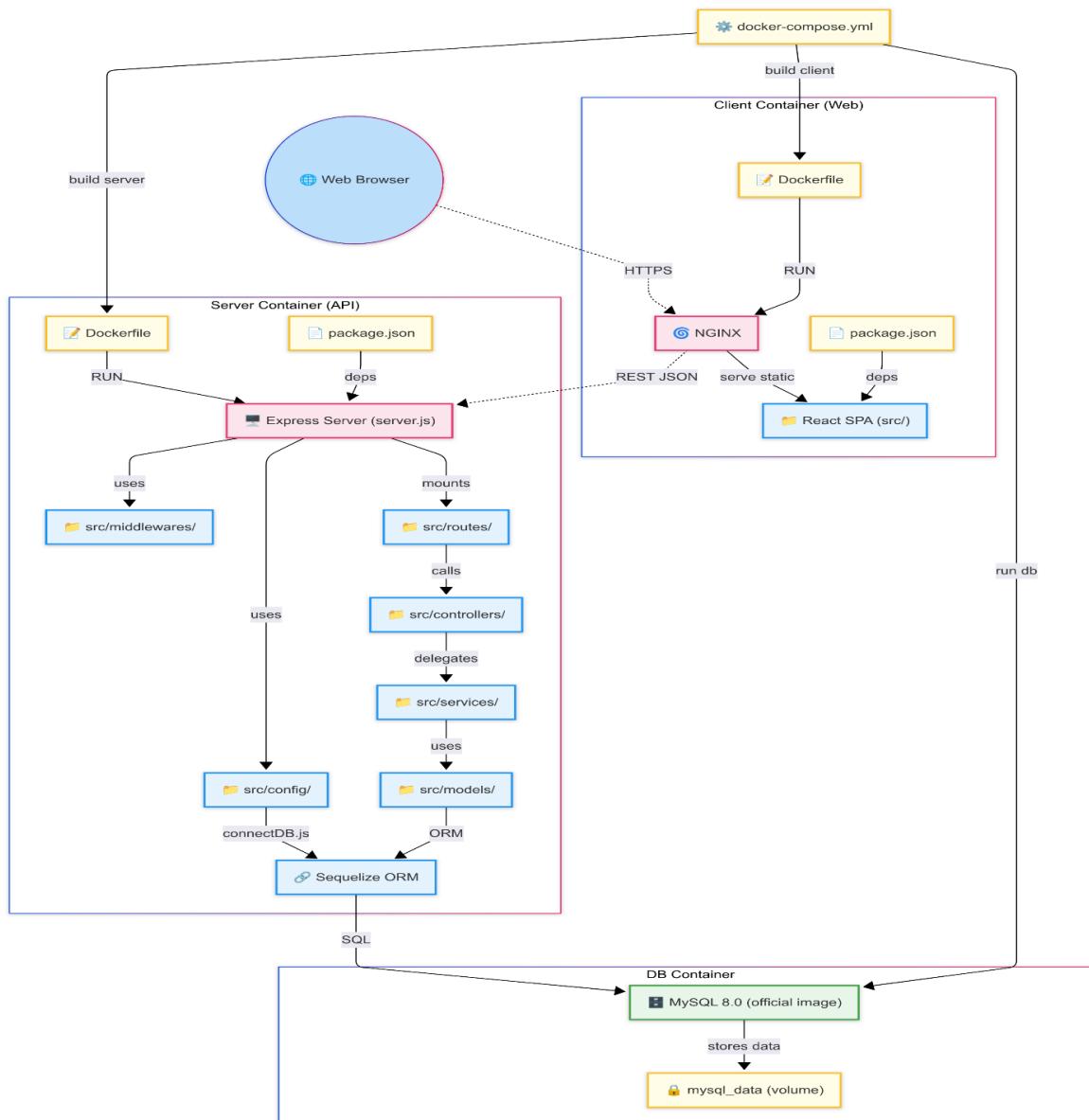
3.1.3 Cơ sở dữ liệu (MySQL)

Sử dụng Sequelize ORM để tương tác giữa server và cơ sở dữ liệu.

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

Sơ đồ mô tả kiến trúc hệ thống sử dụng mô hình Client - Server, trong đó:

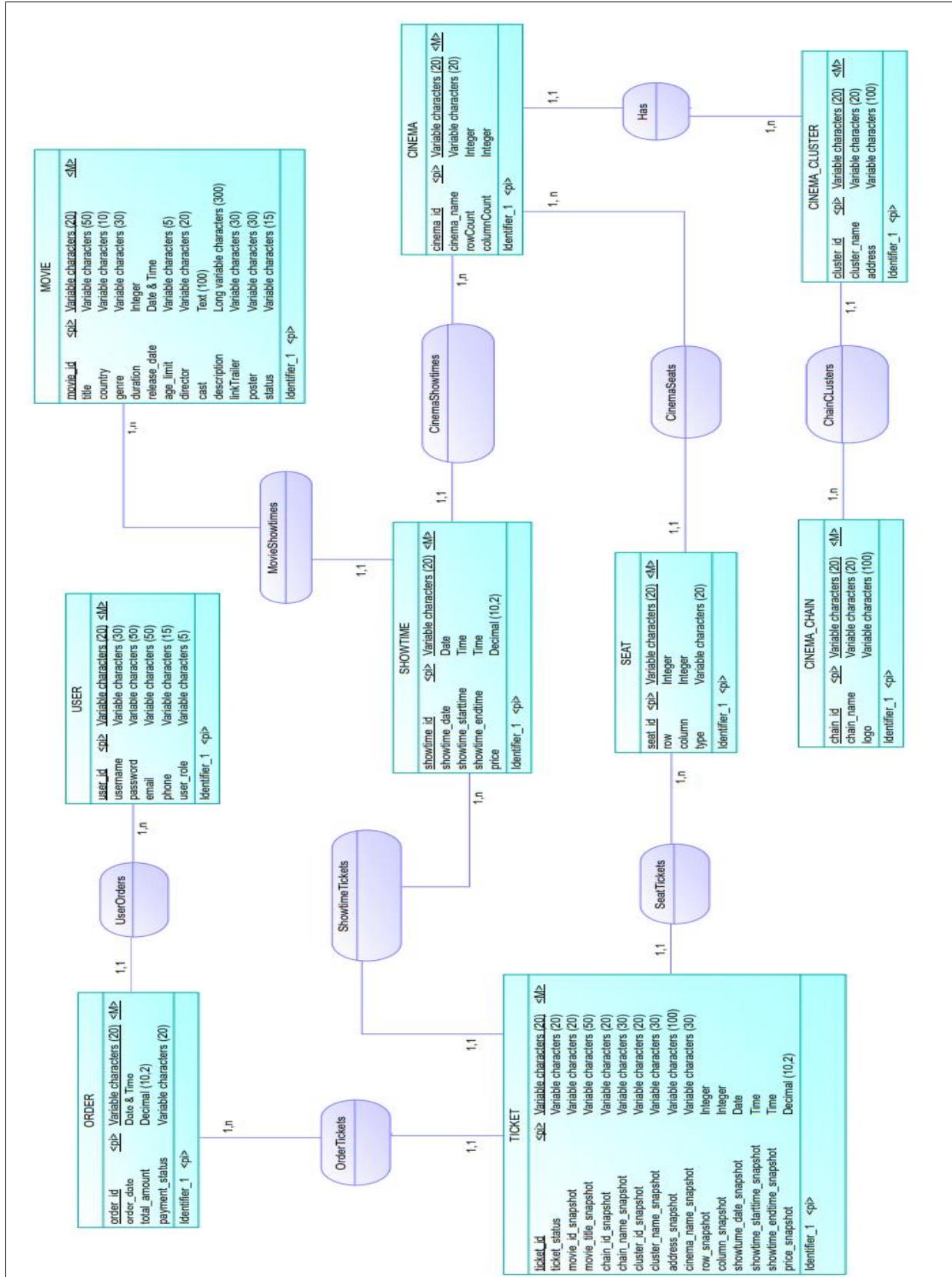
- Client: là một ứng dụng React Single Page Application (SPA) chạy trong container riêng, được phục vụ bởi NGINX.
- Server: là ứng dụng backend Express.js chạy trong container riêng, giao tiếp với MySQL thông qua Sequelize ORM.
- Cơ sở dữ liệu MySQL: là container MySQL riêng, trao đổi dữ liệu với server qua kết nối SQL nội bộ do Docker Compose thiết lập.
- Các thành phần giao tiếp qua REST API (JSON), sử dụng Docker Compose để quản lý và khởi tạo dịch vụ.



Hình 3.1. Sơ đồ kiến trúc hệ thống

3.2 Thiết kế cơ sở dữ liệu

3.2.1 Mô hình thực thể - quan hệ (ERD)



Hình 3.2. Sơ đồ ERD được thiết kế trong PowerDesign

Bảng 1. Mô tả các quan hệ giữa các thực thể trong ERD

Thực thể	Thuộc tính	Khóa chính	Khóa ngoại	Mối quan hệ
1. User	user_id, username, password, email, phone, user_role	user_id		1-n với OrderTable (qua user_id)
2. OrderTable	order_id, user_id, order_date, total_amount, payment_status	order_id	user_id	1-n với Ticket (qua order_id), n-1 với User (qua user_id)
3. Ticket	ticket_id, showtime_id, order_id, seat_id, ticket_status, (các snapshot fields)	ticket_id	showtime_id, order_id, seat_id	n-1 với Showtime (qua showtime_id), n-1 với Seat (qua seat_id), n-1 với OrderTable (qua order_id)
4. Showtime	showtime_id, showtime_date, showtime_starttime, showtime_endtime, price, movie_id, cinema_id	showtime_id	movie_id, cinema_id	1-n với Ticket (qua showtime_id), n-1 với Movie (qua movie_id), n-1 với Cinema (qua cinema_id)
5. Movie	movie_id, title, country, genre, duration, release_date, age_limit, director, cast, description, linkTrailer, poster, status	movie_id		1-n với Showtime (qua movie_id)
6. Seat	seat_id, cinema_id, row, column, type	seat_id	cinema_id	1-n với Ticket (qua seat_id), n-1 với Cinema (qua cinema_id)

Thực thể	Thuộc tính	Khóa chính	Khóa ngoại	Mối quan hệ
7. Cinema	cinema_id, cinema_name, cluster_id, rowCount, columnCount	cinema_id	cluster_id	1-n với Showtime (qua cinema_id), 1- n với Seat (qua cinema_id), n- 1 với CinemaCluster (qua cluster_id)
8. Cinema Cluster	cluster_id, cluster_name, address, chain_id	cluster_id	chain_id	1-n với Cinema (qua cluster_id), n- 1 với CinemaChain (qua chain_id)
9. Cinema Chain	chain_id, chain_name, logo	chain_id		1-n với CinemaCluster (qua chain_id)

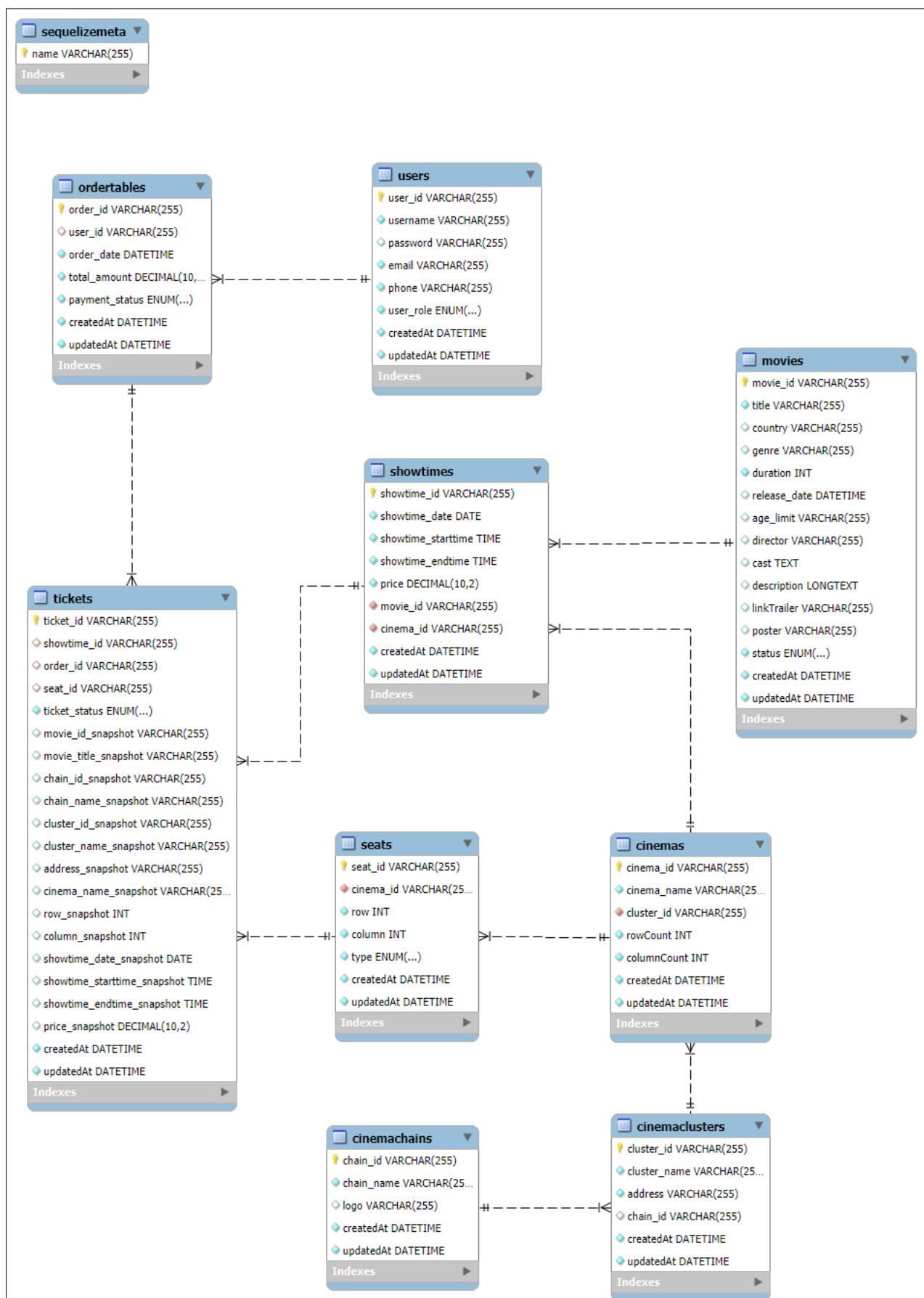
- Mối quan hệ:
 - + 1-n: Quan hệ một-nhiều (ví dụ: một Cinema Cluster có nhiều Cinema).
 - + n-1: Quan hệ nhiều-một (ví dụ: nhiều Ticket thuộc về 1 Order).
 - + onDelete: Quy tắc xóa được áp dụng (CASCADE, SET NULL) theo mã Sequelize.
- Snapshot fields trong Ticket: Lưu trữ thông tin tạm thời (như movie_id_snapshot, price_snapshot) để giữ dữ liệu tại thời điểm đặt vé nhằm vẫn giữ được thông tin vé khi các dữ liệu vé, phim, suất chiếu liên quan bị xóa hoàn toàn khỏi cơ sở dữ liệu.

3.2.2 Sơ đồ quan hệ dữ liệu MySQL

Cơ sở dữ liệu được xây dựng dựa trên mô hình ERD thiết kế ở mục trên bằng MySQL Workbench, sau đó được triển khai thực tế bằng **Sequelize ORM**. Các bảng được tạo ra thông qua lệnh migrate (npx sequelize-cli db:migrate):

Cơ sở dữ liệu bao gồm các bảng chính như: users, movies, cinemas, cinemachains, cinemaclusters, showtimes, tickets, ordertables, seats và bảng hệ thống sequelizemeta.

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN



Hình 3.3. Sơ đồ cơ sở dữ liệu quan hệ MySQL

3.3 Thiết kế API

3.3.1 Tổng quan

Hệ thống cung cấp RESTful API theo chuẩn OpenAPI 3.0, tài liệu hóa bằng Swagger, dưới đây là bảng mô tả tổng quan:

Bảng 2. Cấu hình truy cập API

Mục	Giá trị
Base URL	http://localhost:5000/api/v1
Swagger UI	http://localhost:5000/api-docs
Xác thực	Bearer Token (JWT) – header Authorization: Bearer <token>
Định dạng	application/json (trừ các endpoint upload hình sử dụng multipart/form-data)

Cấu hình Swagger nằm trong src/config/swagger.js; tài liệu được sinh tự động từ các chú thích @swagger trong các file router.

3.3.2 Nhóm endpoint chính

Hệ thống cung cấp các nhóm API chính phục vụ các chức năng CRUD (Create, Read, Update, Delete) như quản lý người dùng, phim, suất chiếu, đặt vé, thông kê,... Mỗi nhóm được tổ chức theo chuẩn RESTful và định nghĩa rõ ràng bằng Swagger để dễ tích hợp frontend và kiểm thử.

Bảng 3. Danh sách nhóm endpoint chính

Nhóm chức năng	Đường dẫn (prefix)	Các thao tác nổi bật
Authentication	/auth	POST /login, POST /register
Người dùng	/users	GET /, GET /me, PUT /me/profile, PUT /me/password, POST /, PUT /:user_id, DELETE /:user_id, GET /roles

Nhóm chức năng	Đường dẫn (prefix)	Các thao tác nổi bật
Phim	/movies	GET /, GET /statuses, GET /:id, POST /, PUT /:id, DELETE /:id
Chỗ ngồi	/seats	POST /:cinema_id, GET / (layout)
Chuỗi rạp / cụm rạp / rạp	/cinemachains, /cinemaclusters, /cinemas	GET /, POST /, GET /:id, PUT /:id, DELETE /:id
Suất chiếu	/showtimes	GET /, POST /:movie_id, GET /:id, PUT /:id, DELETE /:id
Vé đã đặt	/users/me/orders	POST tạo đơn, GET lịch sử
Thống kê	/statistics	GET /movies, GET /clusters
Chỉ số tăng giá vé	/configs	GET / (tăng giá ghế VIP)

3.3.3 Cấu trúc request/response

Dưới đây là một số ví dụ minh họa cách thức giao tiếp giữa client và server, bao gồm phần request từ phía người dùng và phản hồi trả về từ hệ thống.

3.3.3.1 Đăng nhập người dùng

- Phương thức: POST
- Endpoint: api/v1/auth/login
- Request body (yêu cầu):

```
{
  "email": "admin@gmail.com",
  "password": "admin123456789"
}
```

```

Request body required

Edit Value | Schema

{
  "email": "admin@gmail.com",
  "password": "admin123456789"
}

```

Hình 3.4. Request gửi dữ liệu đăng nhập

- Response (phản hồi):

```

{
  "err": 0,
  "msg": "Đăng nhập thành công",
  "token": "eyJhbGciOiJIUzI1NiIsInR... "
}

Code Details
200 Response body
{
  "err": 0,
  "msg": "Đăng nhập thành công",
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJcI2Vyb2lkIjoiNzK3YTEzMjktNTczNS08YzQ5LWIzMGYtNjNjNDk2ZTUyNjg3TiiviGhvbmUiOiIwMzgwMzgwMzgilCJlbWFnbhCIEtmFkhWlunGdtwWlsMnvbSiisinJvbUiOijhZGipbiisImlhdcI6MTc1MjQxNDExmwiZxhwijoxNzUyNTg20TEyfQ._9cqxiopXUy_Aznqa_661QZ-z0FgnowBE3XN3kJcmte"
}

```

Hình 3.5. Response khi đăng nhập thành công

3.3.3.2 Lấy danh sách phim

- Phương thức: POST
- Endpoint: api/v1/movies?status=Now%20Showing
- Response:

```

{
  "err": 0,
  "msg": "OK",
  "response": {
    "count": 1,
    "rows": [
      {
        "movie_id": "466aaadc-b31d-437b-b078-b693f74672e0",
        "title": "SUPERMAN", ...
      }
    ]
  }
}

```

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

The screenshot shows a Postman interface with a 'Code' tab selected. The status code is 200, and the response body is a JSON object representing a movie. The movie details include: id (466aaad-b31d-437b-b078-b693f74672e0), title (SUPERMAN), country (My), genre (Hành Động, Phiêu Lưu), duration (138), release_date (2025-07-10T17:00:00Z), age_limit (T13), director (James Gunn), cast (David Corenswet, Rachel Brosnahan, Nicholas Hoult, ...), description ("Mùa hè tới đây, Warner Bros. Pictures sẽ mang "Superman" - phim điện ảnh đầu tiên của DC Studios đến các rạp chiếu trên toàn cầu. Với phong cách riêng biệt của mình, James Gunn sẽ khắc họa người hùng huyền thoại trong vũ trụ DC hoàn toàn mới, với sự kết hợp đặc đáo của các yếu tố hành động đỉnh cao, hài hước và vô cùng cảm xúc. Một Superman với lòng trắc ánh và niềm tin vào sự thiện lương của con người sẽ xuất hiện đầy hứa hẹn trên màn ảnh."), link_trailer ("https://youtu.be/412qwhKtQ8"), poster ("images/1752415378782-655193250.jpg"), status (Now Showing), created_at (2025-07-13T14:02:58.000Z), and updated_at (2025-07-13T14:02:58.000Z). A 'Download' button is visible at the bottom right.

Hình 3.6. Response khi lấy danh sách phim đang chiếu thành công

3.3.3.3 Cập nhật thông tin rạp phim (chỉ admin)

- Phương thức: PUT
- Endpoint: api/v1/ cinemas/{cinema_id}
- Header:

Authorization: Bearer <admin_token>

Content-Type: application/json

- Request body (chỉ cập nhật số lượng hàng và cột ghế):

```
{  
    "cinema_name": "",  
    "cluster_id": "",  
    "rowCount": 9,  
    "columnCount": 9  
}
```

The screenshot shows the Postman interface with a 'PUT /cinemas/{cinema_id}' endpoint selected. In the 'Parameters' section, there is a single parameter: 'cinema_id' (string, path) with value '4zb3d_pIhd45aQ4KMkI9'. In the 'Request body' section, the content type is set to 'application/json', and the JSON payload matches the one shown in the previous code block: { "cinema_name": "", "cluster_id": "", "rowCount": 9, "columnCount": 9 }.

Hình 3.7. Request cập nhật thông tin một rạp phim

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

- Response:

```
{  
    "err": 0,  
    "msg": "Cập nhật rạp (id: movie_id) thành công!"  
}
```

Request URL
`http://localhost:5000/api/v1/cinemas/4zb3d_pIhd45aQ4KMrkI9`

Server response

Code	Details
200	Response body <pre>{ "err": 0, "msg": "Cập nhật rạp (id: 4zb3d_pIhd45aQ4KMrkI9) thành công!"</pre> Copy Download

Hình 3.8. Response sau khi cập nhật rạp thành công

3.3.3.4 Xóa một rạp phim

- Phương thức: DELETE
- Endpoint: api/v1/ cinemas/{cinema_id}
- Header:

```
Authorization: Bearer <admin_token>  
Content-Type: application/json
```

- Response:

```
{ "err": 0, "msg": "Xóa rạp thành công! }
```

Request URL
`http://localhost:5000/api/v1/cinemas/nI0B54gp-RAhUEtxqvR69`

Server response

Code	Details
200	Response body <pre>{ "err": 0, "msg": "Xóa rạp thành công!"</pre> Copy Download

Hình 3.9. Response sau khi xóa rạp thành công

3.4 Thiết kế giao diện (UI/UX)

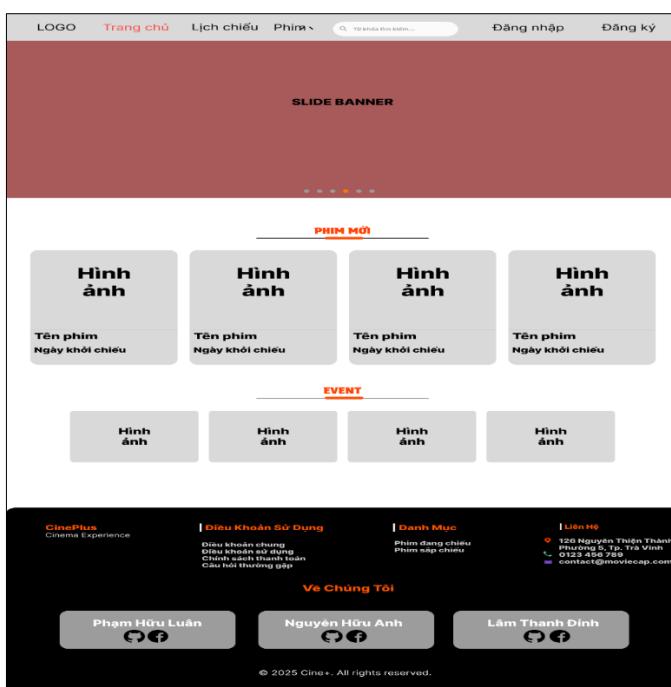
Giao diện người dùng được lên ý tưởng và thiết kế bằng Figma trước khi dùng ngôn ngữ lập trình để xây dựng, đảm bảo trực quan, dễ sử dụng và nhất quán. Các yếu tố như màu sắc, bố cục và tương tác được tối ưu cho trải nghiệm người dùng. Giao diện được chia thành hai nhóm chính: người dùng đặt vé và quản trị viên quản lý hệ thống.

3.4.1 Trang chủ website

Giao diện chính khi truy cập website, hiển thị các slide banner, phim mới, sự kiện,... trên đầu trang. Thanh header và footer dùng chung cho toàn bộ các trang giao diện người dùng, đảm bảo sự nhất quán và tiện lợi.

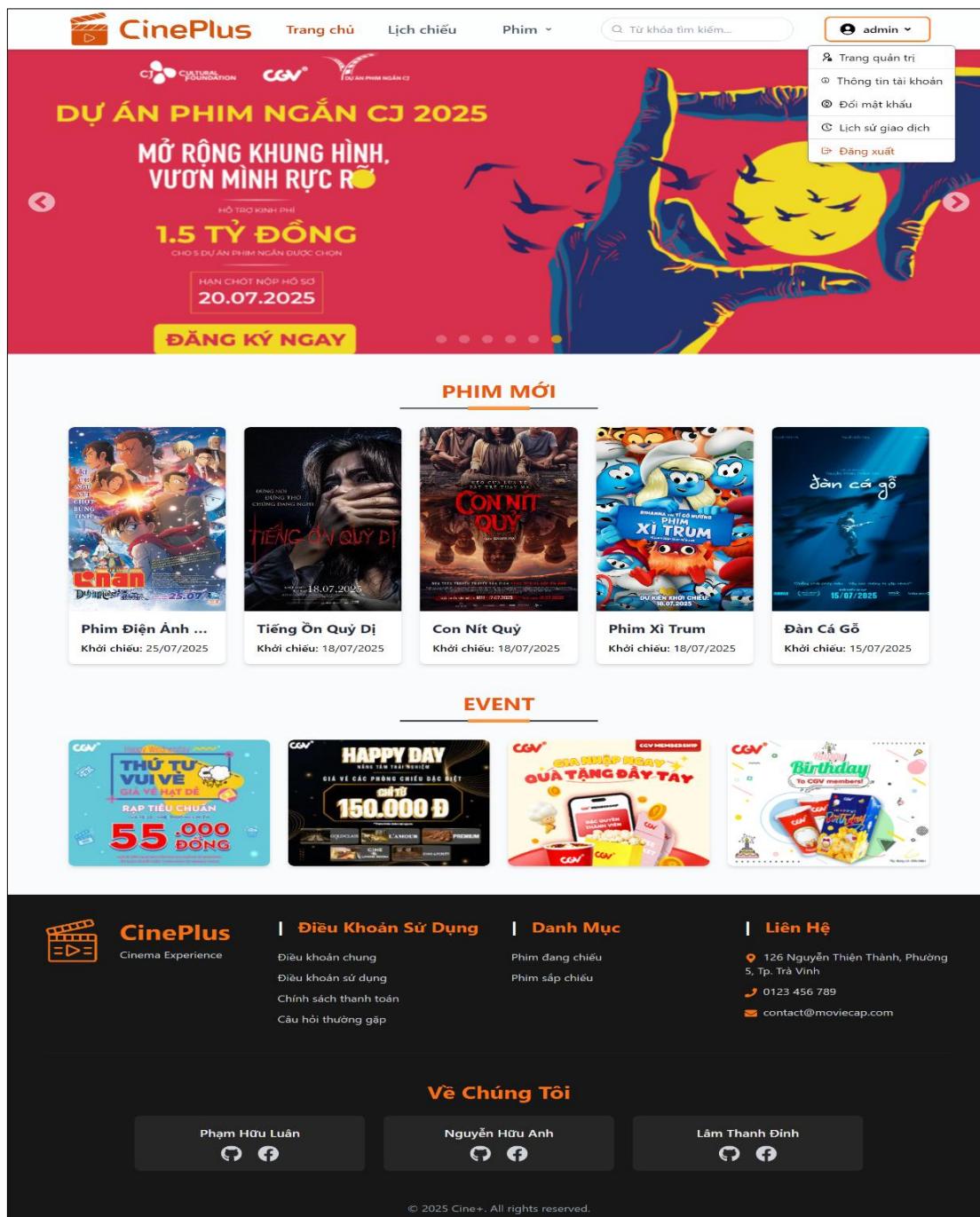
Thanh header gồm các mục:

- “Trang chủ”, “Lịch chiếu”, “Phim” (phân loại thành phim đang chiếu, phim sắp chiếu), ô tìm kiếm.
- Nếu chưa đăng nhập: hiển thị các nút Đăng nhập / Đăng ký.
- Nếu đã đăng nhập: hiển thị tên người dùng kèm dropdown menu gồm: Thông tin cá nhân, Cập nhật thông tin, Đổi mật khẩu, Lịch sử giao dịch và Đăng xuất.
- Nếu tài khoản là admin: thêm mục “Trang quản trị” trong dropdown để điều hướng đến trang quản trị hệ thống.



Hình 3.10. Bản thiết kế giao diện trang chủ người dùng bằng Figma

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN



Hình 3.11. Giao diện trang chủ website

3.4.2 Giao diện đăng nhập, đăng ký

Người dùng bấm vào hai nút Đăng nhập hoặc Đăng ký để chuyển đến trang tương ứng. Nếu ở form đăng ký và sau khi bấm nút “ĐĂNG KÝ” sẽ chuyển sang form đăng nhập.

Sau khi đăng nhập thành công sẽ chuyển đến trang chủ nếu là người dùng và chuyển đến trang quản trị nếu là admin.

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

The wireframe shows a sign-up form titled 'ĐĂNG KÝ' (Sign Up) with a yellow checkmark icon above it. It includes fields for 'Tên tài khoản' (Account name), 'Số điện thoại' (Phone number), 'Email', 'Mật khẩu' (Password), and 'Nhập lại mật khẩu' (Re-enter password). Each password field has an 'eye' icon to show/hide the characters. A large red 'ĐĂNG KÝ' button is at the bottom, with a link below it for existing users: 'Bạn đã có tài khoản? Đăng nhập ngay'.

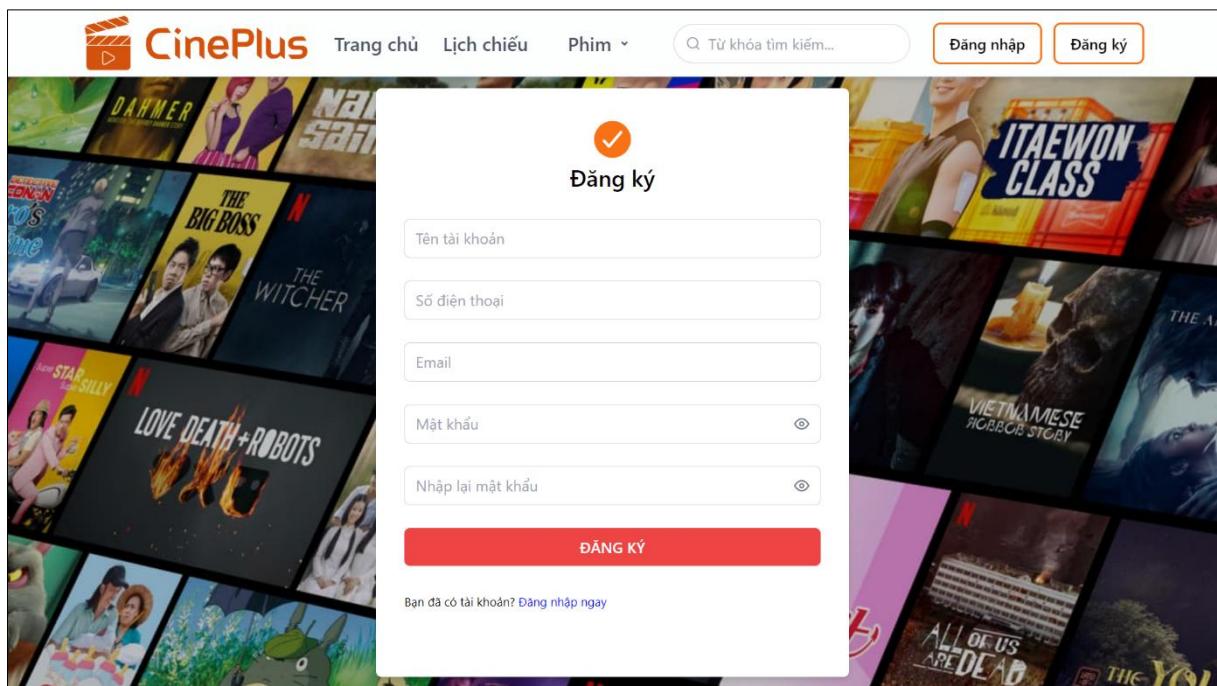
Hình 3.12. Bản thiết kế form đăng ký bằng Figma

The wireframe shows a sign-in form titled 'ĐĂNG NHẬP' (Sign In) with a user icon above it. It includes fields for 'Số điện thoại hoặc email' (Phone number or email) and 'Mật khẩu' (Password). Each password field has an 'eye' icon. A large red 'ĐĂNG NHẬP' button is at the bottom, with links for forgotten passwords ('Bạn quên mật khẩu?') and new accounts ('Bạn chưa có tài khoản? Tạo tài khoản mới').

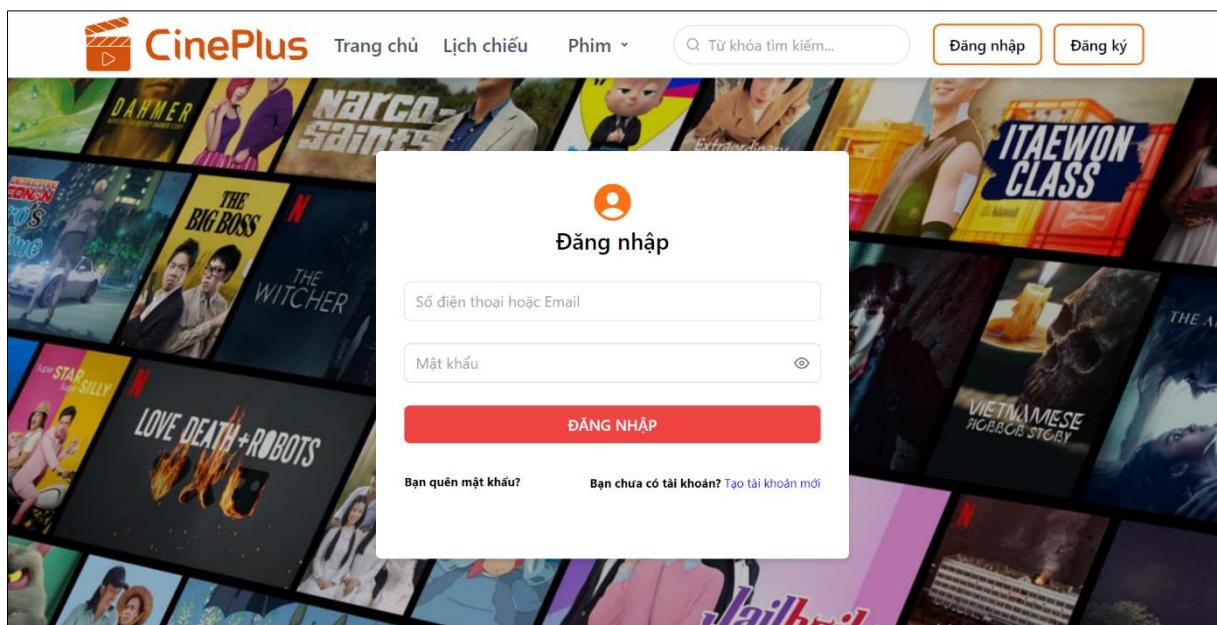
Hình 3.13. Bản thiết kế form đăng nhập bằng Figma

Dựa vào các bản thiết kế cơ bản trên Figma để hiện thực hóa thành các giao diện hoàn chỉnh, sinh động và bắt mắt cho website.

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN



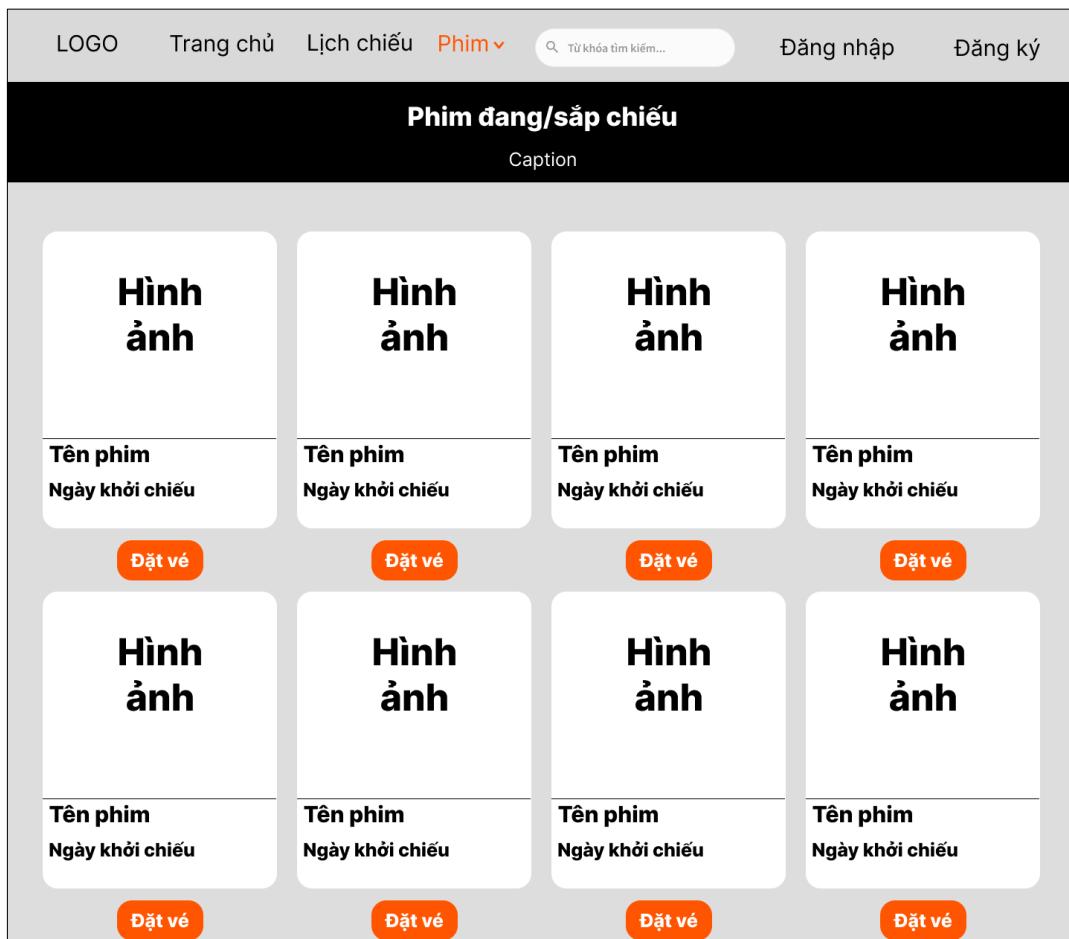
Hình 3.14. Giao diện trang đăng ký tài khoản



Hình 3.15. Giao diện trang đăng nhập

3.4.3 Trang danh mục phim theo trạng thái

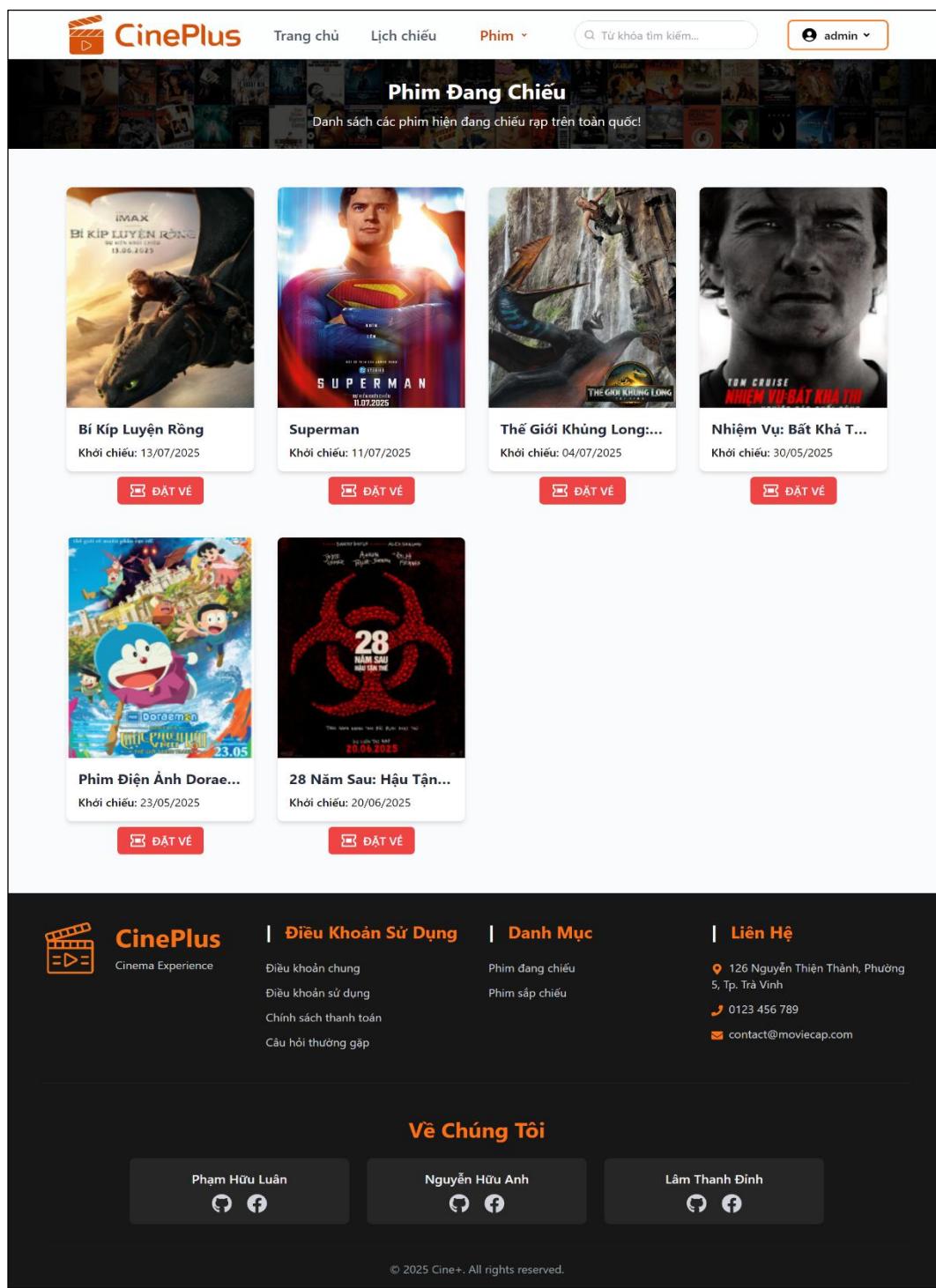
Khi người dùng chọn vào mục “Phim” trên thanh header sẽ hiện dropdown menu gồm hai thể loại phim là “Phim đang chiếu” và “Phim sắp chiếu”. Ở mỗi trang trạng thái phim sẽ dẫn đến trang danh sách phim theo trạng thái đó.



Hình 3.16. Bản thiết kế giao diện danh sách phim theo danh mục bằng Figma

Phim sắp chiếu sẽ có URL là : <http://localhost:3000/movies/coming-soon> và phim đang chiếu sẽ có URL là: <http://localhost:3000/movies/now-showing>.

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

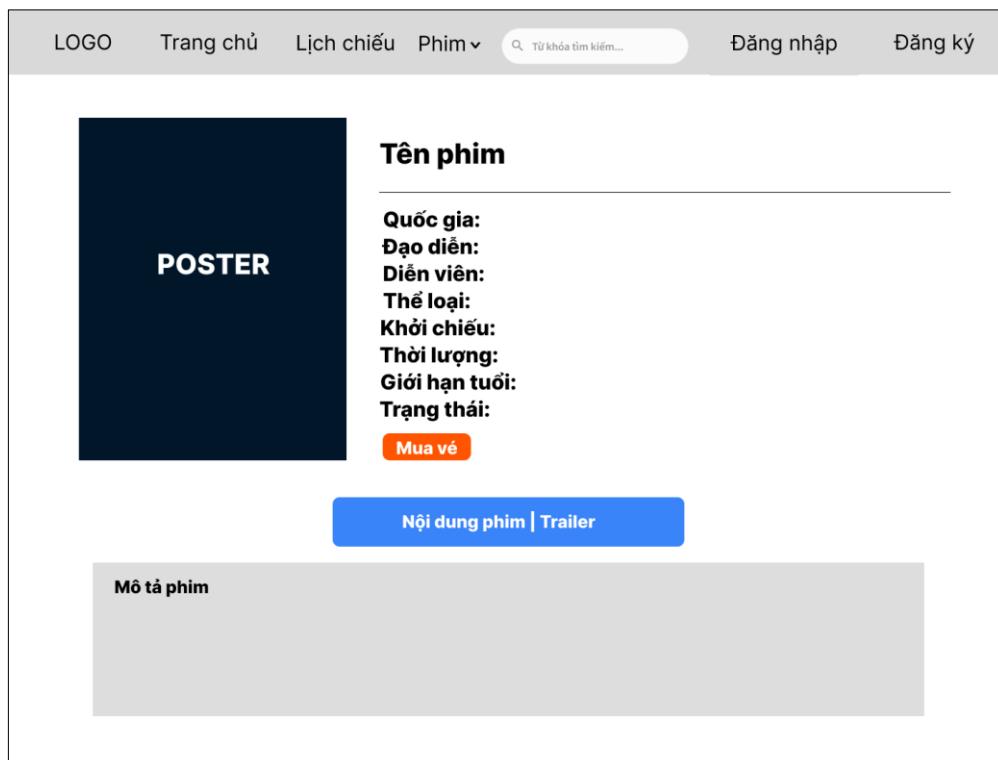


Hình 3.17. Giao diện trang phim đang chiếu

3.4.4 Giao diện trang thông tin chi tiết phim

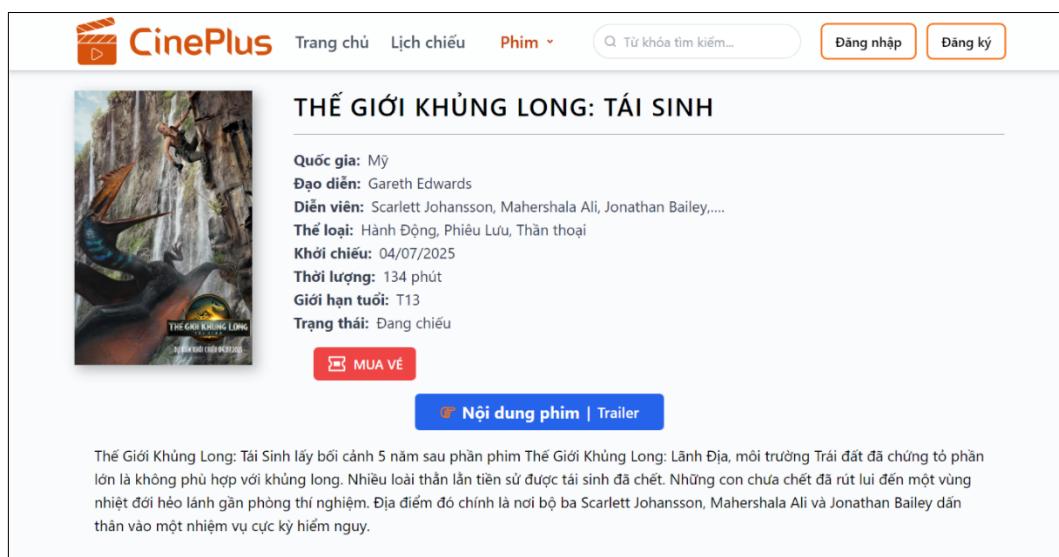
Trang Chi tiết phim hiển thị đầy đủ thông tin về một bộ phim: quốc gia, đạo diễn, diễn viên, thể loại, ngày khởi chiếu, thời lượng, giới hạn tuổi và trạng thái. Giao diện gồm ảnh poster lớn, nút “MUA VÉ” và hai nút chuyển đổi giữa “Nội dung phim” và “Trailer”. Phần mô tả nội dung phim giúp người dùng nắm được cốt truyện trước khi đặt vé. Giao diện rõ ràng, dễ theo dõi và thúc đẩy hành động mua vé.

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

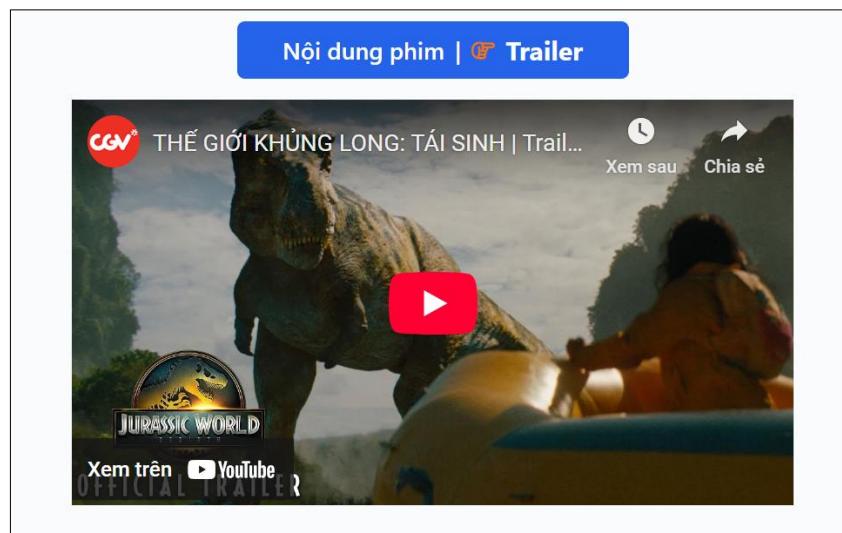


Hình 3.18. Bản thiết kế giao diện thông tin chi tiết phim bằng Figma

URL của trang: <http://localhost:3000/movies/detail/:movieId/title>. Với tham số là `movieId` (mã định danh của phim) và `title` (tên phim).



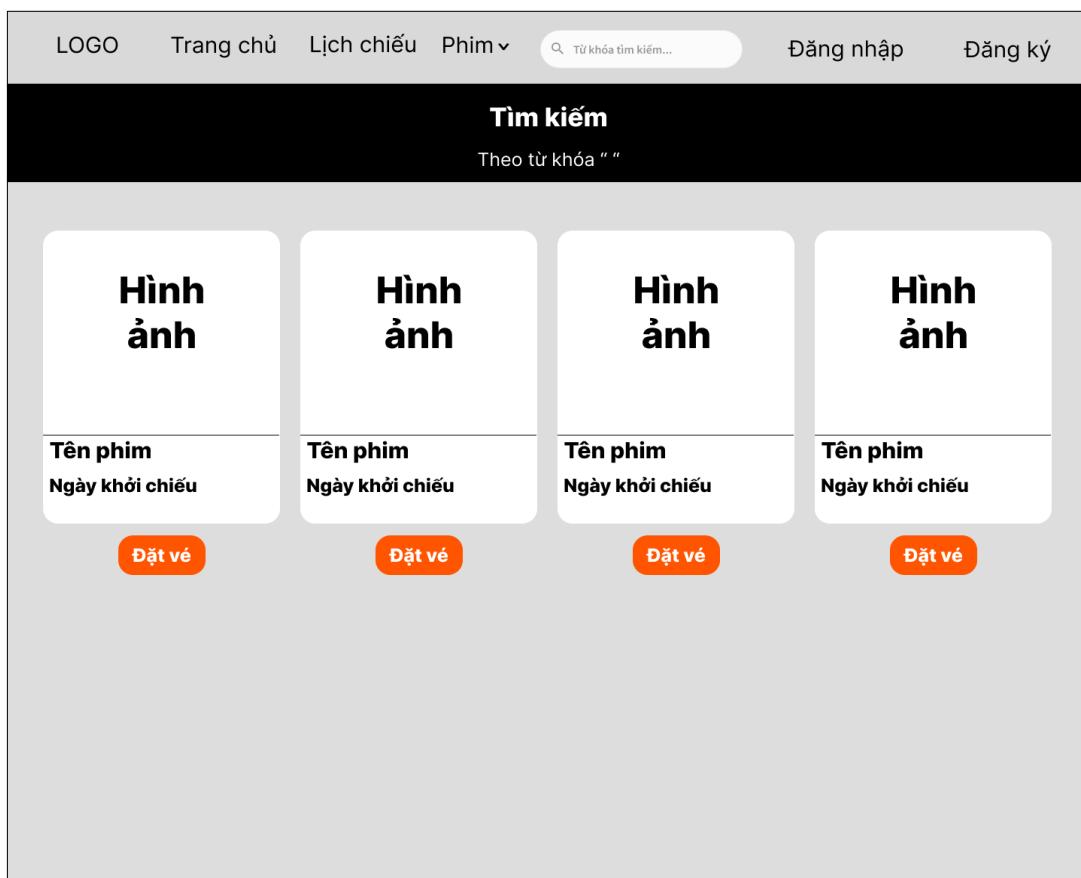
Hình 3.19. Giao diện trang thông tin chi tiết phim



Hình 3.20. Tab trailer ở giao diện chi tiết phim

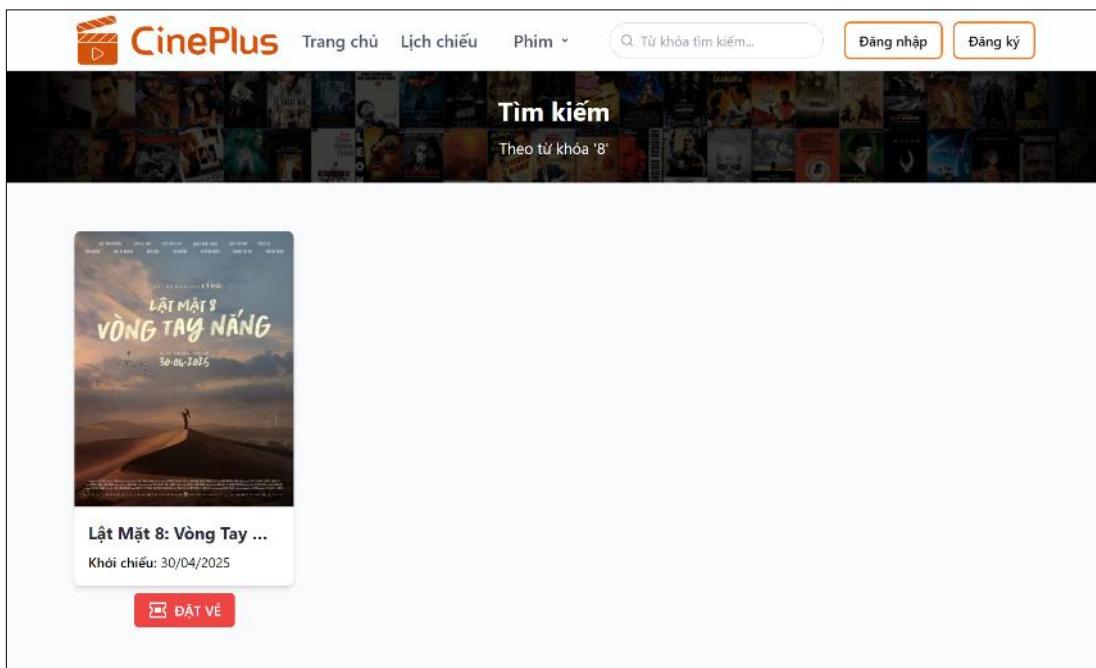
3.4.5 Giao diện cho chức năng tìm kiếm

Người dùng nhập từ khóa tìm kiếm trên header để tìm phim phù hợp với từ khóa đó, ví dụ nhập từ khóa “8” thì URL sẽ là: <http://localhost:3000/search?s=8>.



Hình 3.21. Bản thiết kế giao diện chức năng tìm kiếm phim bằng Figma

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN



Hình 3.22. Giao diện cho chức năng kết quả tìm kiếm phim

3.4.6 Giao diện trang cập nhật thông tin cá nhân

Trang thông tin cá nhân hiển thị và cho phép người dùng cập nhật các thông tin như họ tên, email, số điện thoại và loại tài khoản. Người dùng có thể chỉnh sửa trực tiếp và nhấn nút "CẬP NHẬT THÔNG TIN" để lưu thay đổi. Giao diện đơn giản, rõ ràng, phù hợp cho việc quản lý thông tin cá nhân.

A wireframe of a user profile update page. At the top, there's a header with 'LOGO', 'Trang chủ', 'Lịch chiếu', 'Phim' (with a dropdown arrow), a search bar containing 'Từ khóa tìm kiếm...', and two orange buttons for 'Đăng nhập' and 'Đăng ký'. Below the header, the title 'THÔNG TIN CÁ NHÂN' is centered above a horizontal line. The main form area contains four input fields: 'Họ và tên :', 'Email :', 'Số điện thoại :', and 'Loại tài khoản :'. Each field has a corresponding text input line below it. At the bottom right of the form area is a green button labeled 'CẬP NHẬT THÔNG TIN' (Update Information).

Hình 3.23. Bản thiết kế giao diện xem và cập nhập thông tin cá nhân bằng Figma

URL của trang: <http://localhost:3000/user/profile>.

The screenshot shows a user profile page titled "THÔNG TIN CÁ NHÂN". It contains four input fields with the following data:

- Họ và tên: admin
- Email: admin@gmail.com
- Số điện thoại: 038038038
- Loại tài khoản: admin

A green button labeled "CẬP NHẬT THÔNG TIN" is located at the bottom right of the form.

Hình 3.24. Giao diện trang cập nhật thông tin cá nhân

3.4.7 Giao diện trang đổi mật khẩu

Trang đổi mật khẩu cho phép người dùng thay đổi mật khẩu tài khoản để đảm bảo bảo mật. Giao diện gồm các trường nhập: mật khẩu hiện tại, mật khẩu mới và xác minh lại mật khẩu. Người dùng có thể ẩn/hiện nội dung nhập thông qua biểu tượng con mắt. Nút "ĐỔI MẬT KHẨU" gửi yêu cầu cập nhật đến hệ thống.

The Figma wireframe shows a password change form with the following structure:

- Header: LOGO, Trang chủ, Lịch chiếu, Phim, search bar, Đăng nhập, Đăng ký
- Main Content:
 - TITLE: ĐỔI MẬT KHẨU
 - Fields:
 - Mật khẩu hiện tại
 - Mật khẩu mới
 - Xác minh mật khẩu mới
 - Buttons:
 - ĐỔI MẬT KHẨU (green)

Hình 3.25. Bản thiết kế giao diện đổi mật khẩu bằng Figma

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

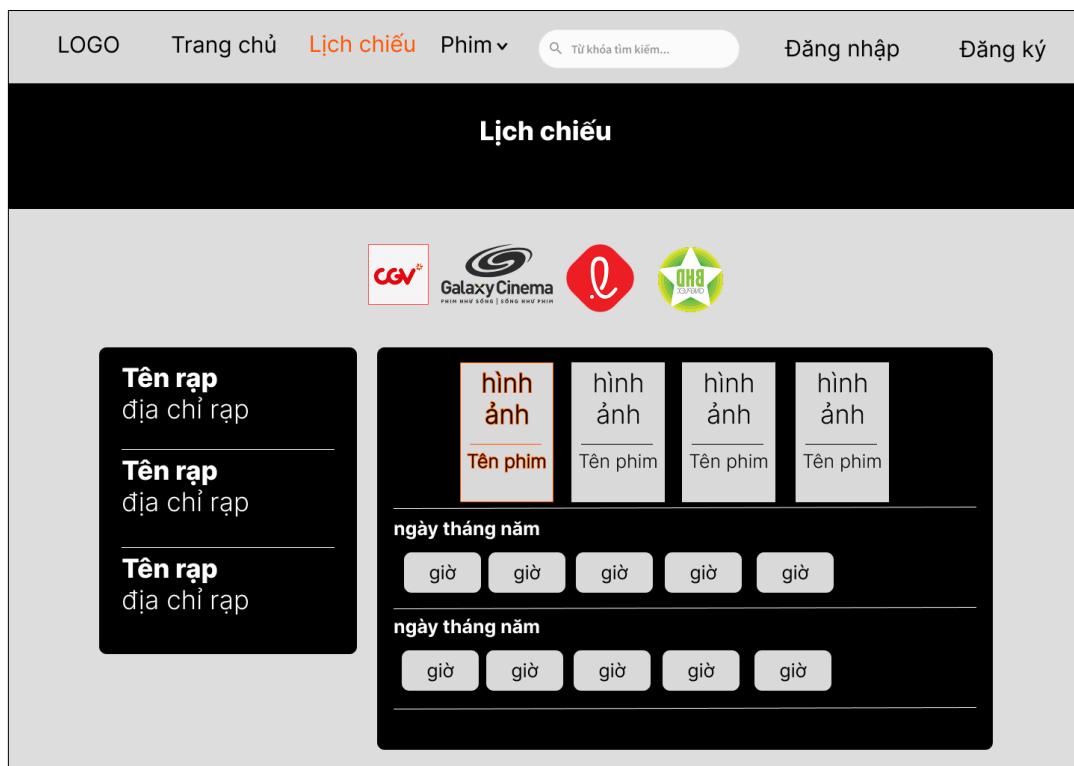
URL của trang người dùng tự đổi mật khẩu: <http://localhost:3000/user/change-password>.



Hình 3.26. Giao diện trang đổi mật khẩu

3.4.8 Giao diện trang lịch chiếu

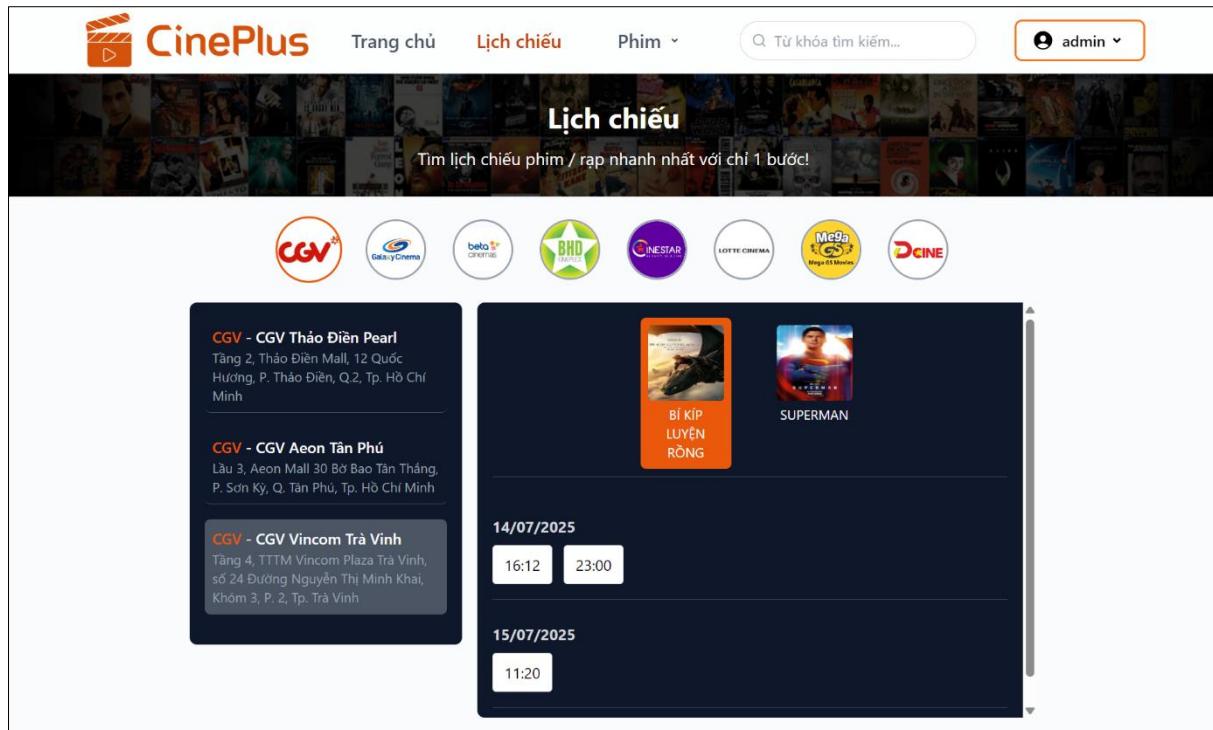
Trang lịch chiếu cho phép người dùng xem nhanh lịch chiếu phim theo từng cụm rạp. Người dùng có thể chọn rạp (như CGV, Galaxy, BHD...) và xem danh sách phim cùng thời gian chiếu tương ứng theo ngày. Giao diện trực quan, dễ thao tác, giúp người dùng nhanh chóng tra cứu thông tin lịch chiếu phù hợp nhu cầu.



Hình 3.27. Bản thiết kế giao diện lịch chiếu bằng Figma

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

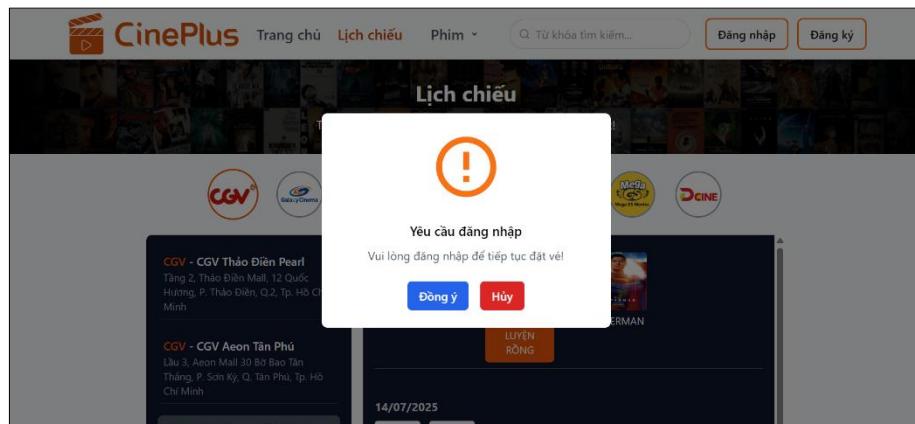
URL của trang: <http://localhost:3000/showtime>.



Hình 3.28. Giao diện trang xem lịch chiếu

3.4.9 Giao diện trang chọn ghế và lịch sử đặt vé

Khi chọn suất chiếu từ trang lịch chiếu, nếu chưa đăng nhập, hệ thống hiển thị modal yêu cầu đăng nhập, sau đó chuyển hướng về đúng trang chọn ghế.

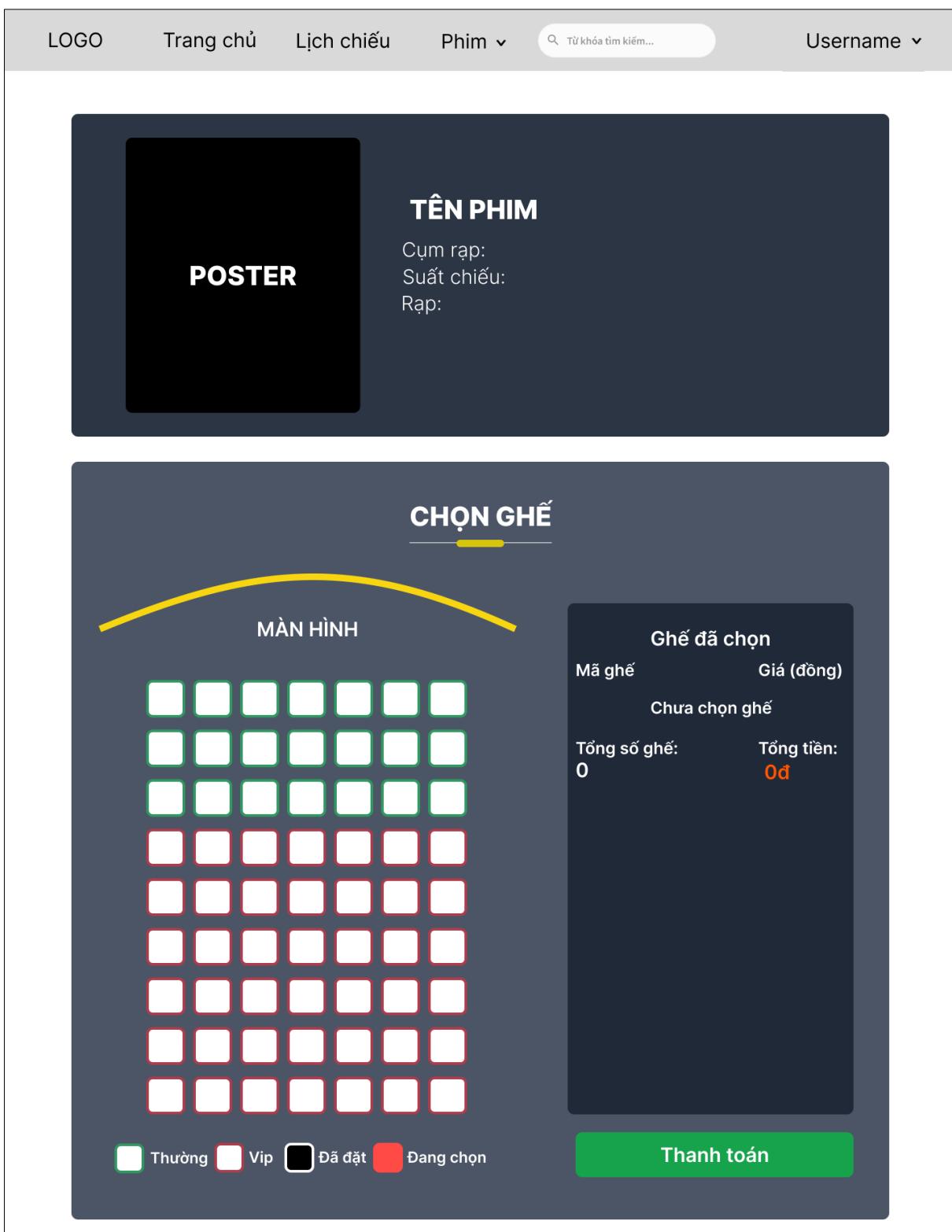


Hình 3.29. Modal thông báo cần phải đăng nhập trước

Tại giao diện chọn ghế, người dùng thấy chi tiết suất chiếu, sơ đồ phòng chiếu với các loại ghế (thường, VIP, đã đặt, đang chọn), cùng thông tin mã ghế, đơn giá, tổng ghế, tổng tiền.

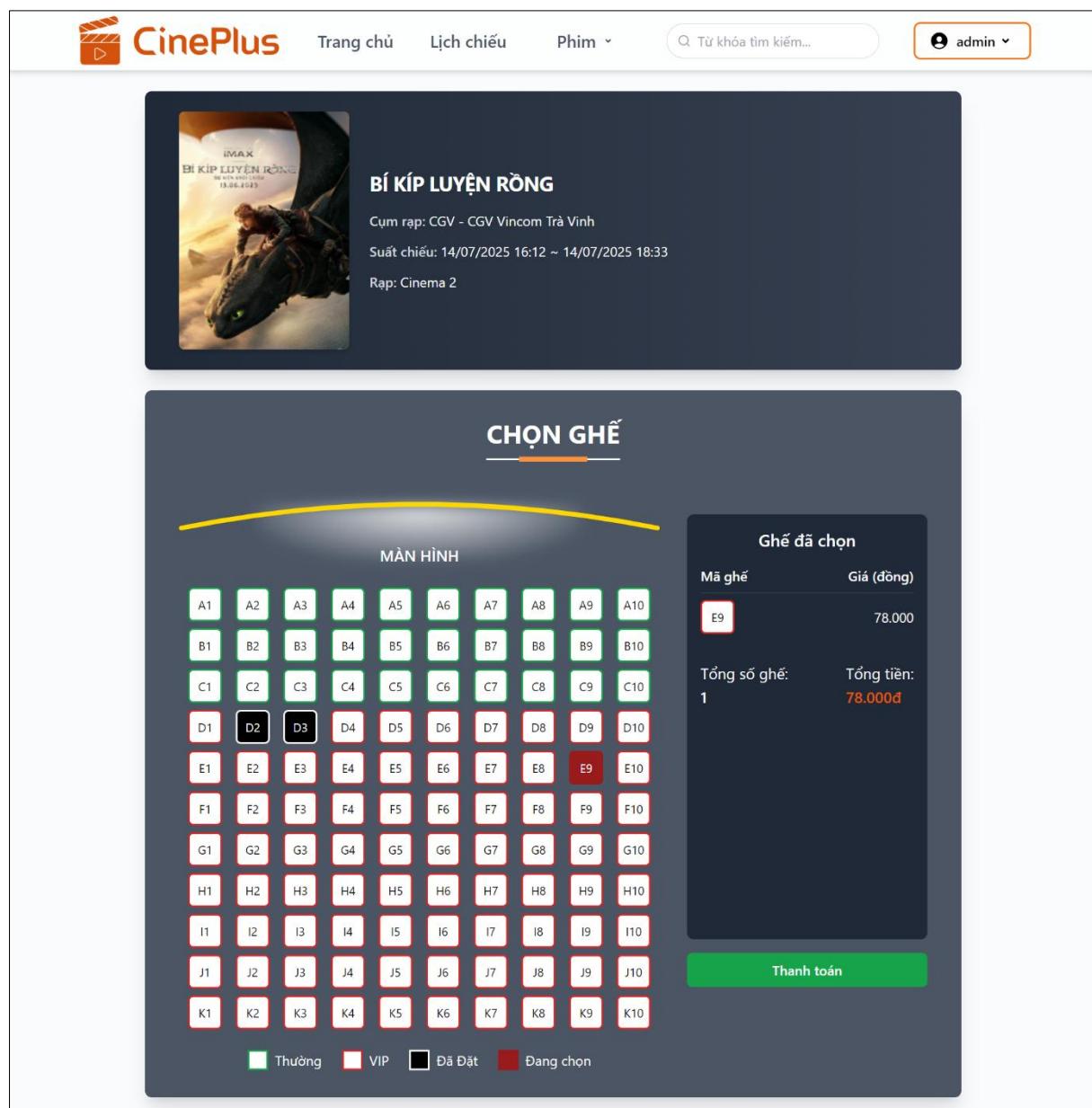
URL của trang chọn ghế: <http://localhost:3000/booking/:showtimeId/select-seat>.

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN



Hình 3.30. Bản thiết kế giao diện chọn ghế bằng Figma

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN



Hình 3.31. Giao diện trang chọn ghế để đặt vé

Sau khi người dùng xác nhận thanh toán thành công, hệ thống chuyển đến trang lịch sử giao dịch. Giao diện hiển thị danh sách các giao dịch đã thực hiện, gồm mã giao dịch, tên phim, ngày đặt và tổng tiền. Người dùng có thể mở rộng từng dòng để xem chi tiết: thời gian chiếu, ghế đã đặt, rạp và địa chỉ. Thiết kế dễ theo dõi, giúp người dùng kiểm tra lại thông tin đặt vé khi cần.

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

MÃ GIAO DỊCH	TÊN PHIM	NGÀY ĐẶT	TỔNG TIỀN (VND)
Đoạn mã	Tên phim	Ngày giờ đặt	Số tiền
Mã giao dịch: Thời gian giao dịch: Phim: Thời gian chiếu: Thời gian kết thúc: Cụm rạp: Rạp: Địa chỉ: Ghế: Tổng tiền:			
Đoạn mã	Tên phim	Ngày giờ đặt	Số tiền

Hình 3.32. Bản thiết kế giao diện lịch sử đặt vé bằng Figma

URL của trang lịch sử đặt vé: <http://localhost:3000/user/orders>.

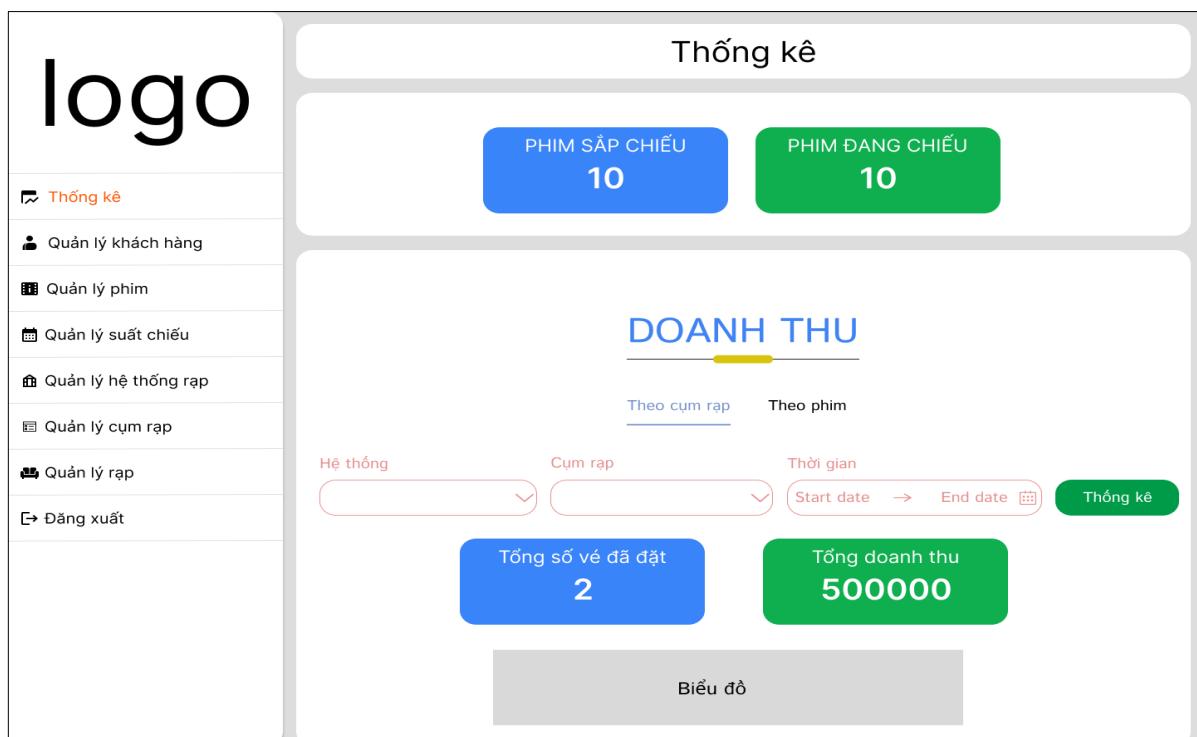
MÃ GIAO DỊCH	TÊN PHIM	NGÀY ĐẶT	TỔNG TIỀN (VND)
(-) d81c70a0-1511-46f2-b27d-46e041ce4174	BÍ KÍP LUYÊN RỒNG	16:21:35 14/7/2025	78.000
Mã giao dịch: d81c70a0-1511-46f2-b27d-46e041ce4174 Thời gian giao dịch: 16:21:35 14/7/2025 Phim: Bí Kíp Luyện Rồng Thời gian chiếu: 14/07/2025 16:12:00 Thời gian kết thúc: 14/07/2025 18:33:00 Cụm rạp: CGV - CGV Vincom Trà Vinh Rạp: Cinema 2 Địa chỉ: Tầng 4, TTTM Vincom Plaza Trà Vinh, số 24 Đường Nguyễn Thị Minh Khai, Khóm 3, P. 2, Tp. Trà Vinh Ghế: E9 Tổng tiền: 78.000đ			
(+) 6d027403-2492-47c8-8a4d-b8a6fb1ebea4	BÍ KÍP LUYÊN RỒNG	16:19:24 14/7/2025	157.000

Hình 3.33. Giao diện trang lịch sử đặt vé

3.4.10 Giao diện các trang quản trị

3.4.10.1 Giao diện trang thống kê

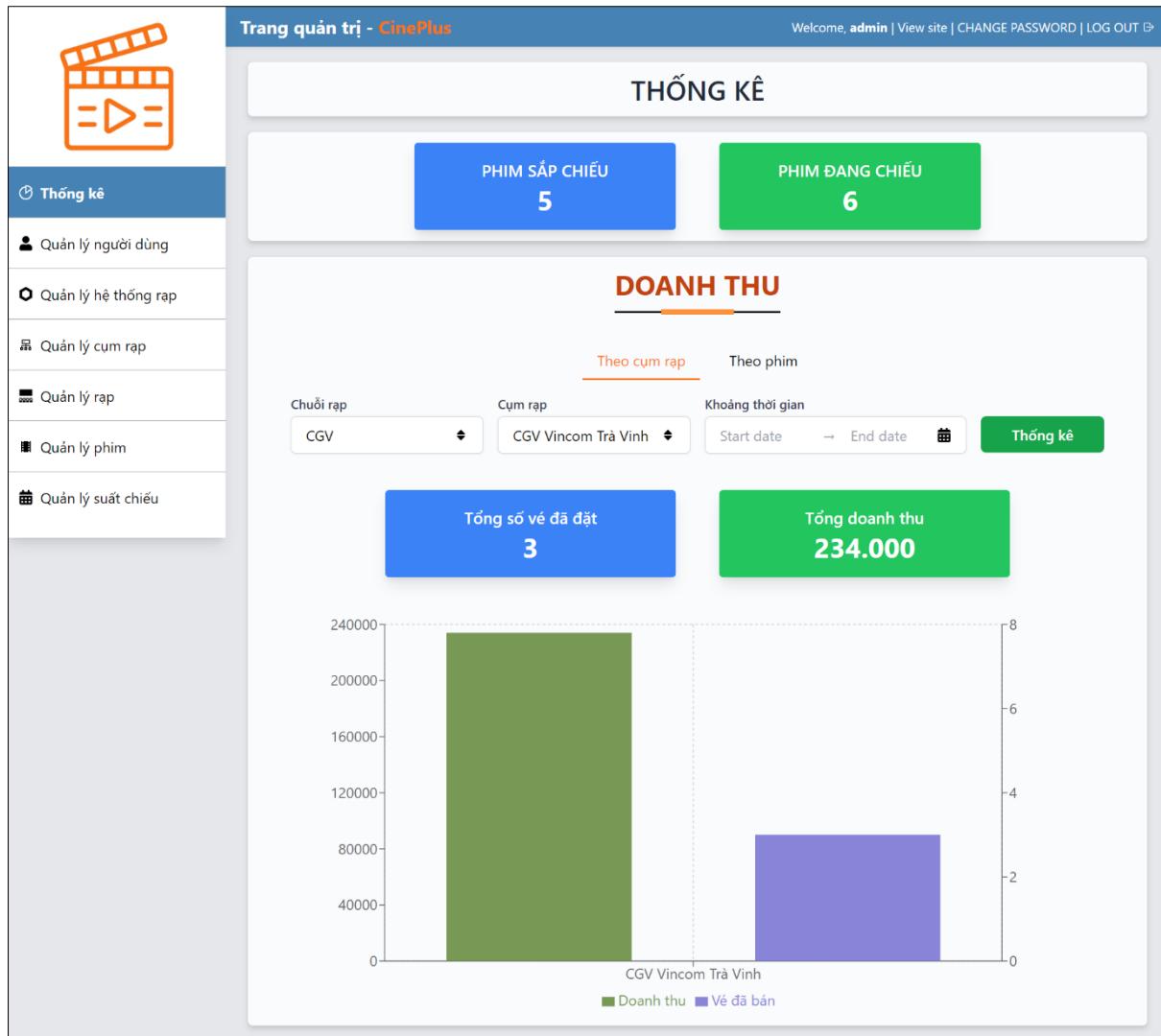
Trang "Thống kê" của hệ thống quản trị là giao diện chính khi đăng nhập bằng tài khoản quản trị, cung cấp cái nhìn tổng quan về hoạt động rạp phim. Giao diện hiển thị số phim sắp chiếu, số phim đang chiếu và tổng doanh thu. Người dùng có thể chọn cụm rạp hoặc phim và khoảng thời gian để xem thông kê chi tiết về doanh thu và vé đã bán, được thể hiện qua biểu đồ trực quan.



Hình 3.34. Bản thiết kế giao diện báo cáo thống kê bằng Figma

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

URL của trang: <http://localhost:3000/admin/dashboard>.



Hình 3.35. Giao diện trang thống kê

3.4.10.2 Giao diện các trang quản lý

Hệ thống quản trị bao gồm các chức năng quản lý người dùng, chuỗi rạp, cụm rạp, rạp, phim và suất chiếu. Giao diện sử dụng thiết kế dạng bảng (table layout) nhất quán, trực quan, dễ thao tác. Mỗi bảng đều được phân trang hợp lý và tích hợp đầy đủ các nút chức năng CRUD (thêm, xem, sửa, xóa). Mỗi bản ghi đều sử dụng mã UUID hoặc nanoid giúp đảm bảo tính duy nhất, nâng cao bảo mật và dễ dàng mở rộng hệ thống về sau.

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

Mã người dùng	Email	Họ và tên	Loại người dùng	Số điện thoại	Thao tác
1	Huuhanh@gmail.com	Nguyễn Hữu Anh	Quản trị viên	0123456789	
2	Huuluan@gmail.com	Phạm Hữu Luân	Quản trị viên	0123456789	
3	ThanhDinh@gmail.com	Lâm Thanh Đinh	Quản trị viên	0123456789	

Hình 3.36. Bản thiết kế giao diện quản lý chung bằng Figma

Mã phim	Tên phim	Quốc gia sản xuất	Năm phát hành	Thời lượng	Thể loại	Trạng thái	Ngày phát hành	Poster	Thao tác
3564037e-c489-4f22-bd5f-5906bf4afbec	Mới	Việt Nam	2025	120 phút	hài	Đang chiếu	02/07/2025		
9a611b30-2624-4fe7-94a6-cee48c3cc866	Nhiệm Vũ Bất Khả Thi 8 - Nghiệp Bảo Cuối	Mỹ	2025	168 phút	Hành Động	Đang chiếu	30/05/2025		

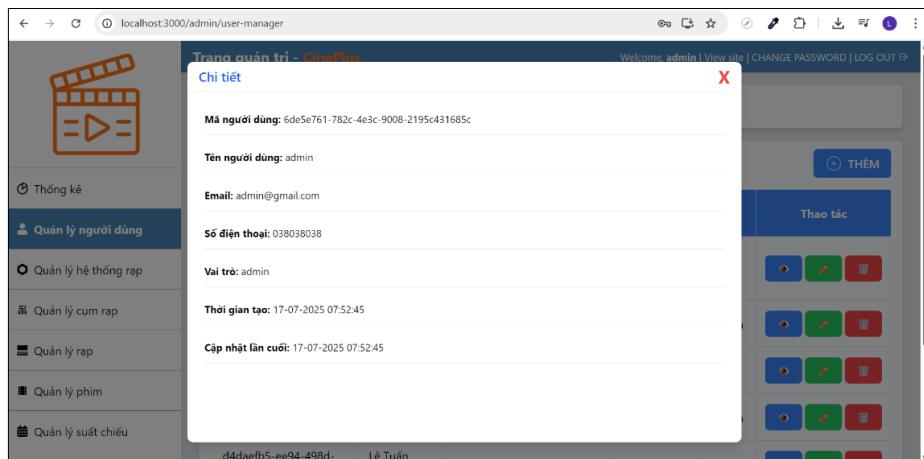
Hình 3.37. Giao diện quản lý phim

Mã phim	Tên phim	Số xuất chiếu	Thời lượng	Trạng thái	Poster	Thao tác
01a50ad7-4f5b-483c-81d7-cbb6757956c4	BÍ KÍP LUYỆN RỘNG	3	126	Đang chiếu		

Hình 3.38. Giao diện quản lý suất chiếu

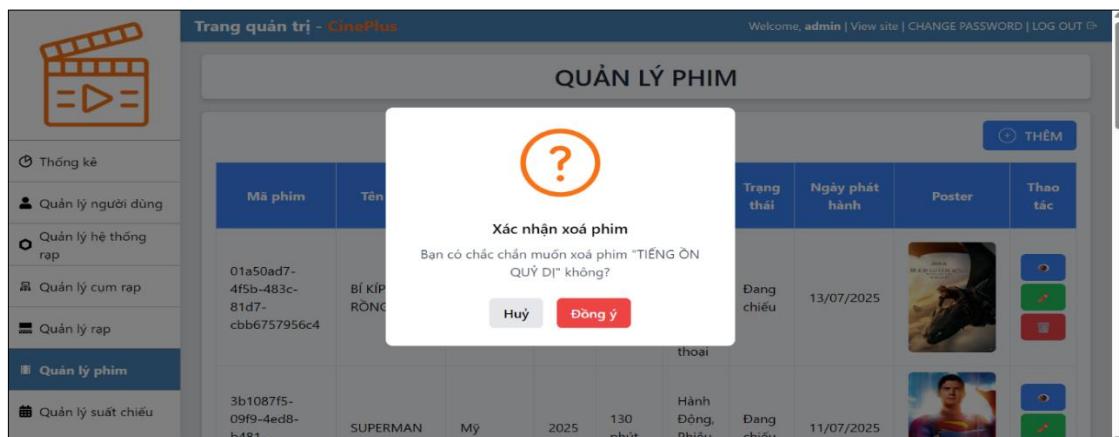
XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

Khi bấm vào biểu tượng nút xem ở cột thao tác mỗi trang quản lý sẽ hiện ra modal thể hiện thông tin chi tiết về đối tượng đó.



Hình 3.39. Modal hiển thị thông tin chi tiết người dùng

Tương tự như chức năng xem thì chức năng xóa sẽ được thực hiện khi nhấn chọn biểu tượng xóa ở cột thao tác sẽ hiện modal xác nhận xóa đối tượng đó.



Hình 3.40. Modal xác nhận xóa một phim

3.4.10.3 Giao diện thêm một đối tượng trong trang quản trị

Trang thêm vào được bằng cách chọn vào nút “Thêm” ở đầu các danh sách mỗi trang quản trị. Trang này dùng để tạo một đối tượng hay bản ghi mới cho các đối tượng như phim, suất chiếu, người dùng,... nhằm bổ sung dữ liệu vào hệ thống quản lý. Khi nhập thông tin mới xong thì chọn nút Submit Form để gửi dữ liệu mới cho server.

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

The screenshot shows the 'Trang quản trị - CinePlus' (Admin Page - CinePlus) interface. On the left is a sidebar with icons for Thống kê (Statistics), Quản lý người dùng (User Management), Quản lý hệ thống rạp (Cinema System Management), Quản lý cụm rạp (Cinema Complex Management), Quản lý rạp (Cinema Hall Management), and Quản lý phim (Movie Management). The 'Quản lý phim' option is selected and highlighted in blue. The main content area is titled 'THÊM PHIM' (Add Movie). The form fields include:

- Tên phim (Movie Name)
- Quốc gia (Country)
- Thể loại (Genre)
- Thời lượng (phút) (Duration in minutes)
- Ngày phát hành (Release Date) with a date picker input.
- Đạo diễn (Director)
- Diễn viên (Actor)
- Giới hạn tuổi (Age Rating)
- Mô tả nội dung (Description)
- Trạng thái (Status) with a dropdown menu showing 'Chọn trạng thái' (Select status).
- Link trailer (Trailer Link)
- Poster (Poster) with a 'Chọn tệp' (Select file) button.

A green 'Submit Form' button is located at the bottom right of the form.

Hình 3.41. Giao diện trang thêm mới phim

3.4.10.4 Giao diện cập nhật một đối tượng trong trang quản trị

Trang cập nhật vào được bằng cách chọn vào nút biểu tượng chỉnh sửa ở cột thao tác mỗi trang quản trị. Trang này dùng để cập nhật thông tin một đối tượng hay bản ghi cho phim, suất chiếu, rạp, cụm rạp, chuỗi rạp, suất chiếu,... nhằm cập nhật dữ liệu vào hệ thống quản lý.

Khi nhập thông tin mới xong thì chọn nút Submit Form để gửi dữ liệu cập nhật cho server xử lí và cơ sở dữ liệu cập nhật lại thông tin của đối tượng đó.

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

The screenshot shows the 'Trang quản trị - CinePlus' (Admin Panel) interface. On the left is a sidebar with icons and labels for: Thống kê, Quản lý người dùng, Quản lý hệ thống rạp, Quản lý cụm rạp, Quản lý rạp, Quản lý phim (highlighted in blue), and Quản lý suất chiếu. The main area is titled 'CẬP NHẬT PHIM' (Update Movie). It contains fields for: Tên phim (BÍ KÍP LUYỆN RỒNG), Quốc gia (Mỹ), Thể loại (Hài, Hành Động, Phiêu Lưu, Thần thoại), Thời lượng (phút) (126), Ngày phát hành (13/07/2025), Đạo diễn (Dean DeBlois), Diễn viên (Mason Thames, Nico Parker, Gerard Butler), Giới hạn tuổi (K), Mô tả nội dung (Câu chuyện về một chàng trai trẻ với ước mơ trở thành thợ săn rồng, nhưng định mệnh lại đưa đẩy anh đến tinh bạn bất ngờ với một chú rồng.), Trạng thái (Phim đang chiếu), Link trailer (https://youtu.be/rvOaNwwDVZk), and Poster (A file path: /images/1752480719926-311758481.jpg). A 'Submit Form' button is at the bottom.

Hình 3.42. Giao diện trang cập nhật thông tin phim

CHƯƠNG 4: TRIỂN KHAI VÀ CÔNG NGHỆ SỬ DỤNG

4.1 Danh sách công nghệ sử dụng

Dự án "Hệ thống đặt vé xem phim" được phát triển dựa trên các công nghệ chính, được gom nhóm theo vai trò và chức năng để đảm bảo hiệu quả trong thiết kế, triển khai và quản lý hệ thống:

- Nhóm công nghệ giao diện và frontend

ReactJS: Thư viện JavaScript hàng đầu dùng để xây dựng giao diện người dùng (frontend) theo mô hình SPA. Nó hỗ trợ tạo các thành phần (component) tái sử dụng, như trang danh sách phim hoặc giao diện chọn ghế, với quản lý trạng thái qua hooks (useState, useEffect). Điều này đảm bảo phản hồi nhanh khi người dùng chọn ghế hoặc cập nhật thông tin, nâng cao trải nghiệm người dùng.

- Nhóm công nghệ backend và API

Node.js/ExpressJS: Node.js là môi trường chạy JavaScript phía server, kết hợp ExpressJS để phát triển backend. ExpressJS cung cấp các API RESTful, xử lý logic như xác thực người dùng (JWT), quản lý suất chiếu và tính toán tổng tiền vé. Nhóm đã tích hợp middleware để kiểm tra quyền (admin/user) và xử lý lỗi, đảm bảo các endpoint như /api/v1/movies hoạt động ổn định. Trong quá trình phát triển, sử dụng Babel để hỗ trợ cú pháp import/export và các tính năng JavaScript hiện đại.

- Nhóm công nghệ cơ sở dữ liệu

+ MySQL: Hệ quản trị cơ sở dữ liệu quan hệ, lưu trữ dữ liệu cốt lõi như thông tin người dùng, phim, suất chiếu, ghế ngồi... MySQL hỗ trợ truy vấn phức tạp, ví dụ thống kê doanh thu theo rạp, với thiết kế bảng như users, movies, showtimes có khóa ngoại đảm bảo tính toàn vẹn.

+ Sequelize: Object-Relational Mapping (ORM) cho Node.js, đơn giản hóa tương tác với MySQL. Nhóm dùng Sequelize để định nghĩa mô hình (model) và mối quan hệ, đồng thời chuyển đổi dữ liệu (migration) qua lệnh để tự động tạo bảng.

- Nhóm công nghệ kiểm thử và quản lý dự án

+ Postman: Kiểm thử các API RESTful. Dùng để kiểm tra phản hồi khi đăng nhập, đặt vé, tạo phim, phân quyền...

- + Jira: Quản lý công việc theo mô hình Agile/Scrum. Dùng để lập kế hoạch Sprint, phân công nhiệm vụ, theo dõi tiến độ.
- **Nhóm công nghệ triển khai và container hóa**

Docker: Công cụ container hóa đóng gói ứng dụng thành các container (React+NGINX, Express, MySQL), tách biệt môi trường phát triển và triển khai. Điều này đảm bảo tính nhất quán, cho phép hệ thống chạy giống nhau trên các máy chủ, đặc biệt hữu ích khi tích hợp với đám mây.

- **Nhóm công nghệ quản lý mã nguồn và làm việc nhóm**

Git/GitHub: Công cụ quản lý mã nguồn phân nhánh (main, dev, feature) giúp theo dõi thay đổi, quản lý phiên bản và hợp tác nhóm. GitHub lưu trữ kho mã nguồn, hỗ trợ branch và pull request, đảm bảo quy trình phát triển chuyên nghiệp.

4.2 Quy trình CI/CD với GitHub Actions

CI/CD, viết tắt của Continuous Integration/Continuous Delivery (hoặc Continuous Deployment), là một phương pháp phát triển phần mềm giúp tự động hóa quá trình tích hợp, kiểm thử và triển khai mã nguồn. Mục tiêu chính là tăng tốc độ phát hành phần mềm, giảm lỗi và cải thiện tính ổn định của sản phẩm.

Nhóm đã triển khai quy trình CI đơn giản bằng GitHub Actions để tự động hóa một số bước cơ bản. Cụ thể, hệ thống tự động build mã nguồn mỗi khi có commit mới, giúp phát hiện lỗi sớm. Quá trình này bao gồm:

- Tự động build: Biên dịch mã nguồn frontend (React) và backend (Node.js).
- Test code: Chạy các kiểm thử đơn vị cơ bản để đảm bảo tính toàn vẹn.

Do giới hạn thời gian, quy trình CD (Continuous Deployment) chưa được triển khai đầy đủ, nhưng CI đã hỗ trợ nhóm tối ưu hóa quy trình phát triển.

4.3 Cấu hình Docker và quy trình triển khai ứng dụng

4.3.1 Cấu hình Docker

Ứng dụng "Hệ thống đặt vé xem phim" được container hóa bằng Docker, với cấu hình được định nghĩa trong tệp docker-compose.yaml. Cấu hình này bao gồm ba container chính:

- Client: Chạy React SPA với NGINX, phục vụ giao diện người dùng, truy cập tại localhost:3000.
- Server: Chạy ExpressJS backend, xử lý các API và kết nối với cơ sở dữ liệu.
- Database: Chạy MySQL, lưu trữ dữ liệu hệ thống như thông tin người dùng, phim, suất chiếu, chuỗi rạp, cụm rạp, rạp, ghế ngồi, vé và đơn hàng dưới dạng bảng.

4.3.2 Quy trình triển khai

- Dockerize ứng dụng: Nhóm đã đóng gói ứng dụng thành các container bằng Docker. Tệp docker-compose.yml định nghĩa các dịch vụ (client, server, database) với các cổng và phụ thuộc (như server kết nối với database). Lệnh **docker-compose up --build** được sử dụng để xây dựng và khởi động các container, đảm bảo môi trường phát triển nhất quán.
- Triển khai và chạy web: Sau khi build thành công, ứng dụng được chạy cục bộ trên máy chủ tại địa chỉ localhost:3000, cho phép kiểm tra giao diện người dùng như trang danh sách phim hoặc chọn ghế. Quá trình này được tự động hóa thông qua CI đơn giản với GitHub Actions, kích hoạt build Docker mỗi khi có commit mới.
- Triển khai lên host: Hiện tại, ứng dụng đã được triển khai thành công trên máy chủ cục bộ. Nhóm đang lập kế hoạch mở rộng lên môi trường cloud bằng cách điều chỉnh cấu hình Docker như thay đổi cổng hoặc tích hợp với dịch vụ như AWS, đảm bảo tính nhất quán giữa các môi trường phát triển và sản xuất trong tương lai.

CHƯƠNG 5: QUẢN LÝ DỰ ÁN

5.1 Mô hình phát triển phần mềm

Phát triển phần mềm thường dựa trên các mô hình khác nhau, mỗi mô hình có ưu điểm và hạn chế tùy thuộc vào yêu cầu dự án. Hai mô hình phổ biến được nhắc đến là Waterfall và Agile. Mô hình Waterfall là phương pháp tuyến tính, trong đó các giai đoạn (yêu cầu, thiết kế, triển khai, kiểm thử) được thực hiện tuần tự và không cho phép quay lại sau khi hoàn thành. Ngược lại, mô hình Agile tập trung vào phát triển lặp lại, chia dự án thành các chu kỳ ngắn gọi là "sprint", cho phép điều chỉnh yêu cầu dựa trên phản hồi và thay đổi thực tế.

Dự án áp dụng mô hình **Agile** do tính linh hoạt và khả năng phản hồi nhanh với thay đổi. Agile phù hợp với hệ thống đặt vé phức tạp gồm nhiều chức năng (đặt vé, quản trị, thống kê). Dự án được chia thành các sprint ngắn, giúp nhóm dễ điều chỉnh theo phản hồi thực tế và giảng viên. Ví dụ, nhóm đã bổ sung tính năng tìm kiếm phim và quản lý admin sau khi hoàn thiện chức năng cơ bản.

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

Nhóm đã sử dụng Jira để quản lý công việc theo cách tiếp cận Agile, tập trung vào lập kế hoạch và theo dõi tiến độ. Dự án được chia thành 6 giai đoạn từ 19/05/2025 đến 19/07/2025, với các story lớn được định nghĩa. Dưới đây là kế hoạch chi tiết cho từng sprint:

5.2.1 Sprint 1

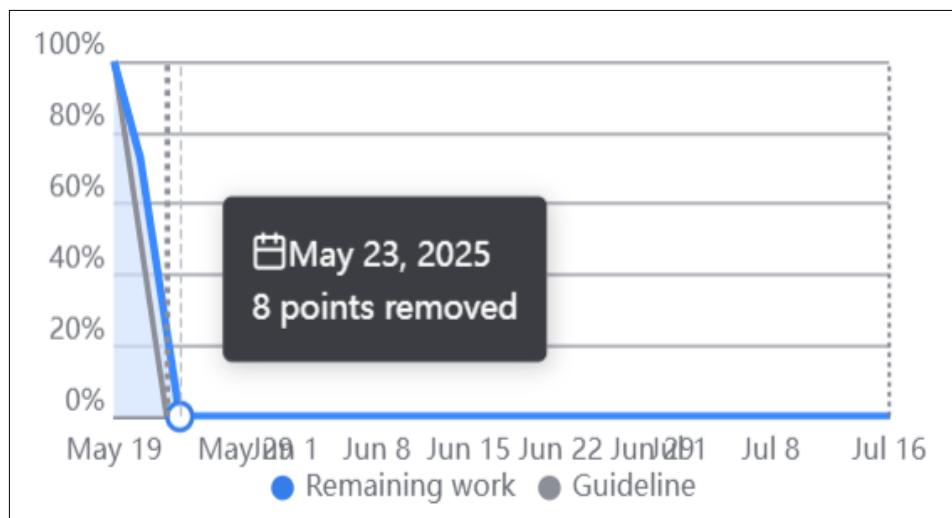
- Thời gian: 19/05/2025 đến 23/05/2025.
- Kế hoạch:
 - + MTB-1: Là một người dùng, tôi muốn giao diện trang chủ được trình bày sinh động và dễ nhìn để dễ dàng tìm phim và trải nghiệm tốt hơn.
 - + MTB-4: Là một người dùng, tôi muốn giao diện đăng nhập và đăng ký rõ ràng, dễ thao tác để có trải nghiệm sử dụng thuận tiện.
 - + MTB-5: Là một người dùng, tôi muốn có thể đăng nhập và đăng ký tài khoản an toàn để sử dụng các tính năng của hệ thống.
- Kết quả: Hoàn thành Sprint 1 với tổng cộng 3 story được thực hiện thành công, được theo dõi qua Jira.

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

☐ MTB Sprint 1 19 May – 23 May (3 work items)		0	0	11	Complete sprint	...
MTB-1	Thiết kế bố cục trang chủ	MOVIE TICKET BOOK...	DONE	3	L	
MTB-4	Thiết kế trang đăng nhập và đăng ký	MOVIE TICKET BOOK...	DONE	3	N/A	
MTB-5	Phát triển chức năng đăng nhập và đăng ký	MOVIE TICKET BOOK...	DONE	5	P/L	

Hình 5.1. Sprint Backlog của Sprint 1

- Burndown Chart dưới đây minh họa tiến độ, với tổng cộng 11 points ban đầu (ngày 19/05) giảm về 0 points (ngày 23/05).



Hình 5.2. Burndown chart Sprint 1

Trong Sprint 1, đường xanh tụt mạnh từ 100% xuống 0% chỉ sau vài ngày đầu, cho thấy nhóm đã hoàn thành toàn bộ công việc sớm hơn kế hoạch. Vào ngày 23/05, hệ thống ghi nhận 8 points được remove, cho thấy các công việc còn lại đã được hoàn tất hoặc đóng lại. Điều này chứng minh nhóm đã có sự phối hợp tốt, xử lý công việc nhanh chóng, không để tồn backlog sau Sprint.

5.2.2 Sprint 2

- Thời gian: 23/05/2025 đến 29/05/2025.
- Kế hoạch:
 - + MTB-6: Là một người dùng, tôi muốn có thể thay đổi và đặt lại mật khẩu để bảo vệ tài khoản của mình khi quên hoặc cần đổi mật khẩu mới.
 - + MTB-7: Là một người dùng, tôi muốn trang thông tin cá nhân có bố cục trực quan và dễ chỉnh sửa để tiện theo dõi và cập nhật thông tin.

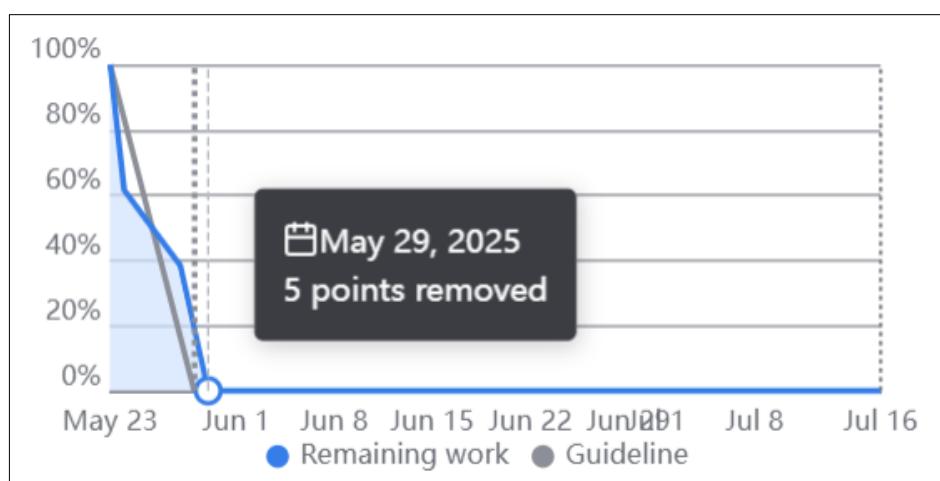
XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

- + MTB-8: Là một người dùng, tôi muốn có thể cập nhật thông tin cá nhân của mình để đảm bảo thông tin luôn chính xác.
- Kết quả: Hoàn thành Sprint 2 với tổng cộng 3 story được thực hiện thành công, được theo dõi qua Jira.

MTB Sprint 2 23 May – 29 May (3 work items)		0	0	13	Complete sprint	...
MTB-6	Phát triển chức năng thay đổi mật khẩu	MOVIE TICKET BOOKI...	DONE	5	PL	
MTB-7	Thiết kế trang thông tin cá nhân	MOVIE TICKET BOOKI...	DONE	3	L	
MTB-8	Phát triển chức năng cập nhập thông tin cá nhân	MOVIE TICKET BOOKI...	DONE	5	NA	

Hình 5.3. Sprint Backlog của Sprint 2

- Burndown Chart dưới đây minh họa tiến độ, với tổng cộng 9 points ban đầu (ngày 23/05) giảm về 0 points (ngày 29/05).



Hình 5.4. Burndown chart Sprint 2

Sprint bắt đầu với 9 story points và kết thúc ở mức 0 đúng ngày 29/05. Không có công việc tồn đọng; không phát sinh thêm điểm ngoài phạm vi. Điều này phản ánh quá trình làm việc hiệu quả: nhóm hoàn thành ba user story MTB-6, MTB-7, MTB-8 sớm hơn guideline, duy trì tiến độ ổn định và không cần điều chỉnh phạm vi giữa chừng.

5.2.3 Sprint 3

- Thời gian: 29/05/2025 đến 08/06/2025.
- Kế hoạch:
 - + MTB-2: Là một người dùng, tôi muốn trang danh sách phim được hiển thị đẹp mắt, dễ phân loại để thuận tiện trong việc tìm kiếm và chọn phim.

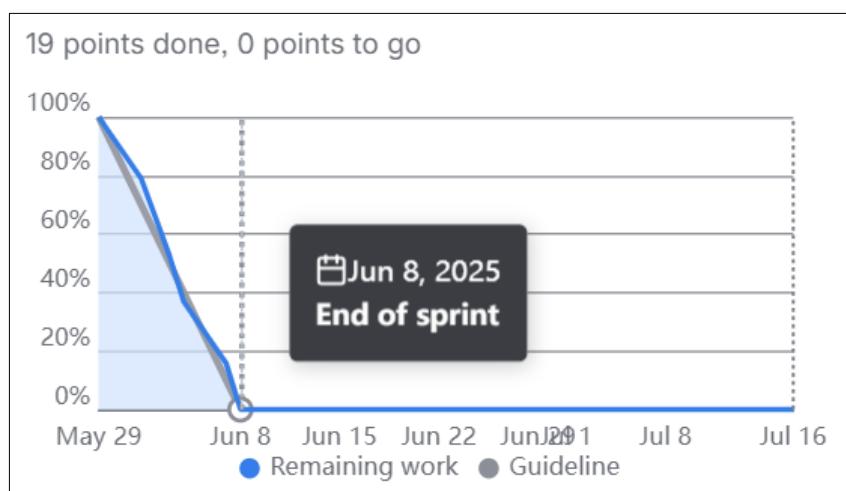
XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

- + MTB-29: Là một quản trị viên, tôi muốn hệ thống cung cấp các API để thêm, sửa, xoá và truy xuất phim nhằm quản lý nội dung phim hiệu quả.
- + MTB-28: Là một người dùng, tôi muốn thấy các bộ phim hiển thị trên giao diện được cập nhật từ hệ thống để tiện theo dõi và lựa chọn.
- + MTB-3: Là một người dùng, tôi muốn trang chi tiết phim có giao diện sinh động, dễ theo dõi các nội dung như trailer, diễn viên và các thông tin khác.
- + MTB-30: Là một người dùng, tôi muốn xem đầy đủ thông tin của một bộ phim cụ thể được lấy từ hệ thống để đưa ra quyết định đặt vé.
- Kết quả: Hoàn thành Sprint 3 với tổng cộng 5 story được thực hiện thành công, được theo dõi qua Jira.

MTB Sprint 3 29 May – 8 Jun (5 work items)		0	0	19	Complete sprint	...
MTB-2 Thiết kế trang danh sách phim	MOVIE TICKET BOOK...	DONE ✓	4	L		
MTB-29 Viết các API về phim	MOVIE TICKET BOOK...	DONE ✓	5	PL		
MTB-28 Hiển thị các phim lấy từ API lên giao diện front-end	MOVIE TICKET BOOK...	DONE ✓	3	L		
MTB-3 Thiết kế trang chi tiết phim	MOVIE TICKET BOOK...	DONE ✓	4	NA		
MTB-30 Hiện thị thông tin chi tiết của phim bằng cách gọi API lấy thông tin một ph...	MOVIE TICKET BOOK...	DONE ✓	3	NA		

Hình 5.5. Sprint Backlog của Sprint 3

- Burndown chart dưới đây minh họa tiến độ, với tổng cộng 19 points ban đầu (ngày 29/05) giảm về 0 points (ngày 08/06).



Hình 5.6. Burndown chart Sprint 3

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

Burndown chart phản ánh hiệu suất ổn định và đều đặn của Sprint 3. Với tổng cộng 19 story points, tất cả được xử lý đúng tiến độ trong khoảng thời gian từ 29/05 đến 08/06. Không có story bị dư, không phát sinh thêm task ngoài kế hoạch. Việc hoàn thành đồng bộ cả frontend (MTB-2, MTB-3, MTB-28, MTB-30) và backend API (MTB-29) cho thấy sự phối hợp tốt giữa các thành viên và tính khả thi của kế hoạch đã đề ra.

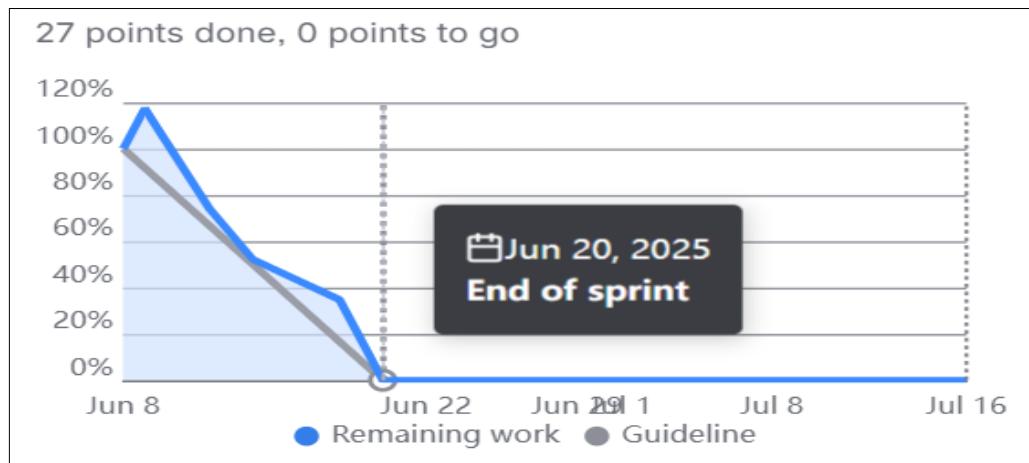
5.2.4 Sprint 4

- Thời gian: 08/06/2025 đến 20/06/2025.
- Kế hoạch:
 - + MTB-32: Là một người dùng, tôi muốn hệ thống hiển thị danh sách vé tôi đã đặt để dễ dàng theo dõi lịch sử giao dịch và kiểm tra thông tin chi tiết của từng vé.
 - + MTB-31: Là một người dùng, tôi muốn có trang lịch sử đặt vé với giao diện rõ ràng, dễ xem để tra cứu các vé đã mua trước đây.
 - + MTB-9: Là một người dùng, tôi muốn có trang tổng hợp lịch chiếu của các phim theo ngày và rạp để dễ dàng lựa chọn suất phù hợp.
 - + MTB-10: Là một người dùng, tôi muốn hệ thống hiển thị lịch chiếu theo dữ liệu thực tế từ backend để đảm bảo thông tin luôn chính xác và cập nhật.
 - + MTB-11: Là một người dùng, tôi muốn có giao diện sơ đồ ghế trực quan để chọn vị trí ngồi yêu thích trước khi thanh toán vé.
 - + MTB-14: Là một người dùng, tôi muốn chọn ghế ngồi và xác nhận đặt vé để hoàn tất quá trình mua vé một cách nhanh chóng và tiện lợi.
- Kết quả: Hoàn thành Sprint 4 với tổng cộng 6 story được thực hiện thành công, được theo dõi qua Jira.

MTB Sprint 4 8 Jun – 20 Jun (6 work items)		0	0	27	Complete sprint	...
	MTB-32 Viết API lấy thông tin vé đã đặt và render lên giao diện	MOVIE TICKET BOOKI...	DONE	4	L	
	MTB-31 Xây dựng trang lịch sử đặt vé	MOVIE TICKET BOOKI...	DONE	4	L	
	MTB-9 Xây dựng trang xem lịch chiếu	MOVIE TICKET BOOKI...	DONE	5	NA	
	MTB-10 Viết các API về lịch chiếu và render lên giao diện	MOVIE TICKET BOOKI...	DONE	5	PL	
	MTB-11 Xây dựng trang chọn ghế để đặt vé	MOVIE TICKET BOOKI...	DONE	5	NA	
	MTB-14 Phát triển chức năng chọn ghế, đặt vé	MOVIE TICKET BOOKI...	DONE	4	PL	

Hình 5.7. Sprint Backlog của Sprint 4

- Burndown chart dưới đây minh họa tiến độ, với tổng cộng 27 points ban đầu (ngày 08/06) giảm về 0 points (ngày 20/06).



Hình 5.8. Burndown chart Sprint 4

Burndown chart cho thấy Sprint 4 đã hoàn thành toàn bộ 27 story points, tuy có vài thời điểm xử lý chậm hơn dự kiến do gặp một số vấn đề nhỏ. Tuy nhiên, toàn bộ công việc vẫn được giải quyết trước ngày kết thúc sprint (20/06). Việc xử lý đồng thời frontend và backend của các chức năng như lịch sử vé đã đặt, lịch chiếu, chọn ghế và đặt vé cho thấy khối lượng công việc lớn nhưng được hoàn thành đầy đủ, phản ánh hiệu quả làm việc nhóm và sự phối hợp tốt giữa các vai trò.

5.2.5 Sprint 5

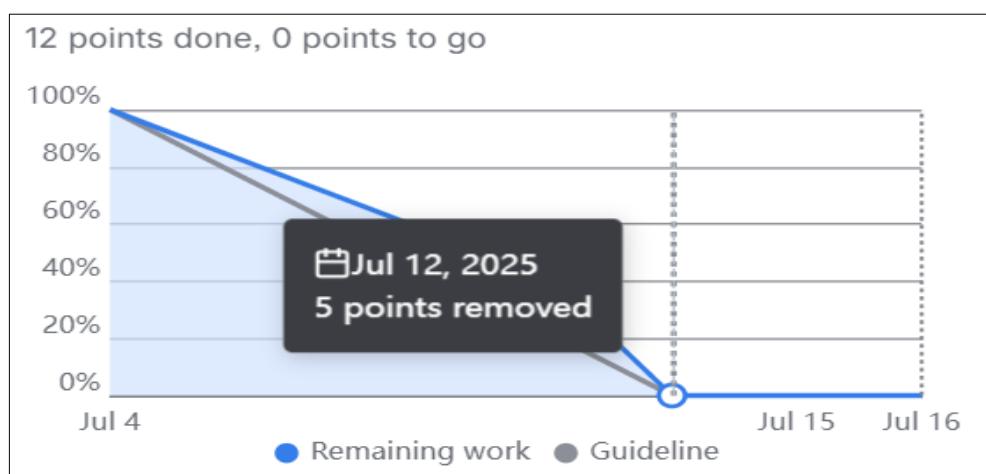
- Thời gian: 04/07/2025 – 13/07/2025.
- Kế hoạch:
 - + MTB-12: Là một người dùng, tôi muốn có thẻ tìm kiếm phim theo tên hoặc thẻ loại để nhanh chóng tìm thấy bộ phim mình quan tâm mà không cần cuộn danh sách dài.
 - + MTB-15: Là một quản trị viên, tôi muốn có thẻ thêm, sửa, xoá phim, suất chiếu, rạp, người dùng... để quản lý nội dung hệ thống một cách linh hoạt và chính xác.
 - + MTB-13: Là một quản trị viên, tôi muốn có một giao diện trực quan để quản lý phim, rạp, người dùng... nhằm thao tác dễ dàng và theo dõi hoạt động hệ thống hiệu quả.
- Kết quả: Hoàn thành Sprint 5 với tổng cộng 3 story được thực hiện thành công, được theo dõi qua Jira.

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

MTB Sprint 5 4 Jul – 13 Jul (3 work items)		0	0	12	Complete sprint	...
MTB-12	Phát triển chức năng tìm kiếm phim	MOVIE TICKET BOOKING	DONE	3	L	
MTB-15	Phát triển các chức năng người quản trị	MOVIE TICKET BOOKING	DONE	5	NA	
MTB-13	Xây dựng trang quản trị	MOVIE TICKET BOOKING	DONE	4	PL	

Hình 5.9. Sprint Backlog của Sprint 5

- Burndown chart dưới đây minh họa tiến độ, với tổng cộng 12 points ban đầu (ngày 04/07) giảm về 0 points (ngày 13/07), phản ánh việc hoàn thành MTB-12, MTB-15 và MTB-13.



Hình 5.10. Burndown chart Sprint 5

Đây là một sprint thành công với tiến độ sát thực tế, thể hiện nhóm đã quen với quy trình Scrum và phối hợp tốt trong việc phát triển cả phía người dùng và quản trị viên.

5.2.6 Sprint 6

- Thời gian: 14/07/2025 đến 19/07/2025.
- Kế hoạch:
 - + MTB-16: Là một quản trị viên hệ thống, tôi muốn có tài liệu API rõ ràng bằng Swagger để dễ dàng kiểm thử, hiểu cách sử dụng và tích hợp các API vào giao diện người dùng hoặc ứng dụng khác.
 - + MTB-33: Là một quản trị viên hệ thống, tôi muốn đóng gói toàn bộ ứng dụng thành các container bằng Docker để đảm bảo môi trường nhất quán, dễ triển khai và bảo trì trên nhiều hệ thống khác nhau.

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

- + MTB-34: Là một quản trị viên hệ thống, tôi muốn tích hợp quy trình CI/CD để mỗi lần cập nhật mã nguồn sẽ tự động build và kiểm thử, giúp giảm lỗi và đẩy nhanh quá trình phát triển cũng như triển khai.
- Kết quả: Hoàn thành Sprint 6 với tổng cộng hai story được thực hiện thành công, và một user story vẫn còn đang thực hiện, được theo dõi qua Jira.

MTB Sprint 6 14 Jul – 19 Jul (3 work items)

Chuẩn hóa quá trình phát triển và triển khai bằng cách viết tài liệu API với Swagger, đóng gói hệ thống bằng Docker và tích hợp quy trình CI/CD để tăng tính tự động hóa và độ tin cậy.

Item	Type	Status	Progress	Label
MTB-16 Viết tài liệu API với Swagger	MOVIE TICKET BOOK...	DONE	4	NA
MTB-33 Dockerize ứng dụng	MOVIE TICKET BOOK...	DONE	5	L
MTB-34 Tích hợp CI/CD	MOVIE TICKET BOOK...	IN PROGRESS	4	PL

Hình 5.11. Sprint Backlog của Sprint 6

- Burndown chart dưới đây minh họa tiến độ:



Hình 5.12. Burndown chart Sprint 6

Dựa trên hình ảnh Burndown Chart, Sprint 6 bắt đầu với 13 story points vào ngày 14/07/2025, nhưng chỉ 9 points đã hoàn thành, còn 4 points chưa xong, với đường "Remaining work" tăng trở lại sau ngày 20/07 do nhóm đã mở lại công việc MTB-34. User story này (tích hợp CI/CD) chưa hoàn tất do nhóm thiếu thời gian nghiên cứu, gây ảnh hưởng đến tiến độ. Sprint thể hiện sự phối hợp tốt với 9/13 points hoàn thành, nhưng lộ hạch chê về phân bổ nguồn lực và kỹ năng.

5.3 Phân công nhiệm vụ của từng thành viên trong nhóm

Bảng 4. Bảng phân công nhiệm vụ các Sprint 1, 2 và 3 trong Jira

Sprint	Công việc (User Story)	Người thực hiện	Point	Sprint goal
1	MTB-1: Thiết kế bố cục trang chủ	Lâm Thanh Đỉnh	3	Hoàn thiện giao diện người dùng cơ bản, bao gồm trang chủ, trang đăng nhập và đăng ký, đồng thời phát triển chức năng xác thực tài khoản giúp người dùng có thể bắt đầu sử dụng hệ thống một cách thuận tiện và an toàn.
	MTB-4: Thiết kế trang đăng nhập và đăng ký	Nguyễn Hữu Anh	3	
	MTB-5: Phát triển chức năng đăng nhập và đăng ký	Phạm Hữu Luân	5	
2	MTB-6: Phát triển chức năng thay đổi và đặt lại mật khẩu	Phạm Hữu Luân	5	Hoàn thiện các chức năng quản lý tài khoản như đổi mật khẩu, chỉnh sửa và cập nhật thông tin cá nhân, giúp người dùng bảo vệ và duy trì thông tin một cách thuận tiện.
	MTB-7: Thiết kế trang thông tin cá nhân	Lâm Thanh Đỉnh	3	
	MTB-8: Phát triển chức năng cập nhật thông tin cá nhân	Nguyễn Hữu Anh	3	
3	MTB-2: Thiết kế trang danh sách phim	Lâm Thanh Đỉnh	4	Hoàn thiện tính năng hiển thị danh sách và chi tiết phim, đảm bảo giao diện trực quan cho người dùng và API quản lý phim hiệu quả cho quản trị viên.
	MTB-29: Viết các API về phim	Phạm Hữu Luân	5	
	MTB-28: Hiển thị các phim lấy từ API lên giao diện frontend.	Lâm Thanh Đỉnh	3	

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

	MTB-3: Thiết kế trang chi tiết phim	Nguyễn Hữu Anh	4	
	MTB-30: Hiện thị thông tin chi tiết của phim bằng cách gọi API lấy thông tin một phim	Nguyễn Hữu Anh	3	

Bảng 5. Bảng phân công nhiệm vụ các Sprint 4, 5 và 6 trong Jira

Sprint	Công việc (SCRUM)	Người thực hiện	Point	Sprint goal
4	MTB-32: Viết API lấy thông tin vé đã đặt và render lên giao diện	Lâm Thanh Đinh	4	Xây dựng các chức năng đặt vé, xem lịch chiếu và lịch sử vé, giúp người dùng dễ dàng tra cứu, lựa chọn suất chiếu và hoàn tất đặt vé với giao diện trực quan, thông tin chính xác.
	MTB-31: Xây dựng trang lịch sử đặt vé	Lâm Thanh Đinh	4	
	MTB-9: Xây dựng trang xem lịch chiếu	Nguyễn Hữu Anh	5	
	MTB-10: Viết các API về lịch chiếu và render lên giao diện	Phạm Hữu Luân	5	
	MTB-11: Xây dựng trang chọn ghế để đặt vé	Nguyễn Hữu Anh	5	
	MTB-14: Phát triển chức năng chọn ghế, đặt vé	Phạm Hữu Luân	4	
5	MTB-12: Phát triển chức năng tìm kiếm phim	Lâm Thanh Đinh	3	Xây dựng các chức năng đặt vé, xem lịch chiếu và lịch

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

	MTB-15: Phát triển các chức năng người quản trị	Nguyễn Hữu Anh	5	sử vé, giúp người dùng dễ dàng tra cứu, lựa chọn suất chiếu và hoàn tất đặt vé với giao diện trực quan, thông tin chính xác.
6	MTB-13: Xây dựng trang quản trị	Phạm Hữu Luân	4	
	MTB-16: Viết tài liệu API với Swagger	Nguyễn Hữu Anh	4	Chuẩn hóa quá trình phát triển và triển khai bằng cách viết tài liệu API với Swagger, đóng gói hệ thống bằng Docker và tích hợp quy trình CI/CD để tăng tính tự động hóa và độ tin cậy.
	MTB-33: Dockerize ứng dụng	Lâm Thanh Đỉnh	5	
	MTB-34: Tích hợp CI/CD	Phạm Hữu Luân	4	

CHƯƠNG 6: KIỂM THỬ**6.1 Chiến lược kiểm thử**

Nhóm đã áp dụng các công cụ và phương pháp sau để kiểm thử dự án "Hệ thống đặt vé xem phim":

- Postman: Sử dụng để kiểm thử API, bao gồm các yêu cầu như đăng nhập, lấy danh sách phim và các endpoint khác.
- GitHub Actions: Tích hợp CI (Continuous Integration) để tự động kiểm thử và triển khai, đảm bảo mã nguồn ổn định qua từng commit.

6.2 Kết quả kiểm thử

Tất cả API endpoint của backend đều được test bằng Postman trước khi tạo tài liệu API với Swagger, một số ví dụ về kết quả kiểm thử:

- **API đăng nhập:** <http://localhost:5000/api/v1/auth/login> (method: POST)

Key	Value	Description	Bulk Edit
phone	038038038		
password	admin123456789		

```

1 {
2   "err": 0,
3   "msg": "Đăng nhập thành công!",
4   "token": "eyJhbGciOiJIUzI1NiIsInR5cCIkIkpXVCJ9.
5       eyJlc2VyX2lkIjoiNmR1NWU3NjEtNzgyYy00ZTNjLTkwMDgtMjE5NWMMzE20DVjIiwicGhvbmUiOiIwMzgwMzgwMzgiLCJ
6       1bwFpbCI6ImFkbWluQGdtYWlsLnNvbSIsInJvbGvioJhZG1pbisImIhdCi6MtclMjcxODQ4NSwiZXhwIjoxNzUyODkxMj
7       g1fQ.Log0fBSiK1SpVynQIKPnVdm18YXQvx_sL161pROWWA"
8 }
  
```

Hình 6.1. Test API đăng nhập bằng Postman

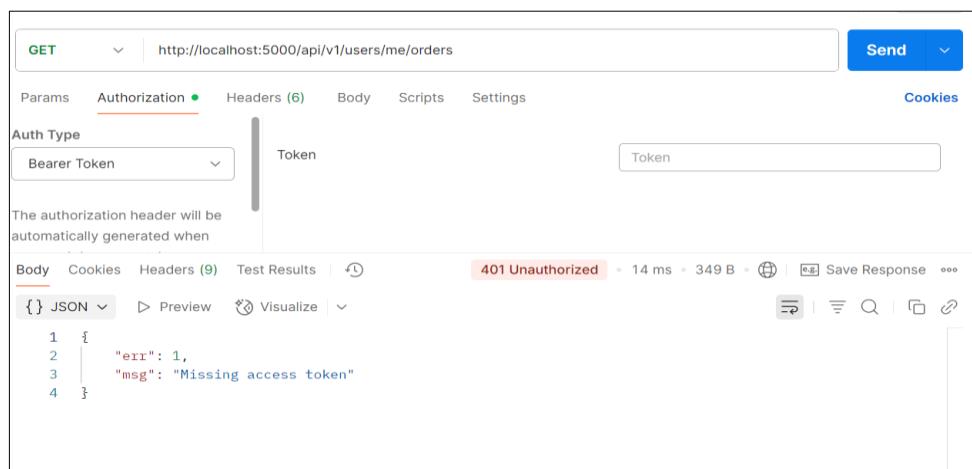
- + Request body:

```
{
  "phone": "038038038",
  "password": "admin123456789"
}
```

- + Phản hồi từ server: Status “**200 OK**” cho thấy yêu cầu đăng nhập đã thành công và trả về thông tin kèm token như mã lỗi err bằng 0, thông báo đăng nhập thành công.

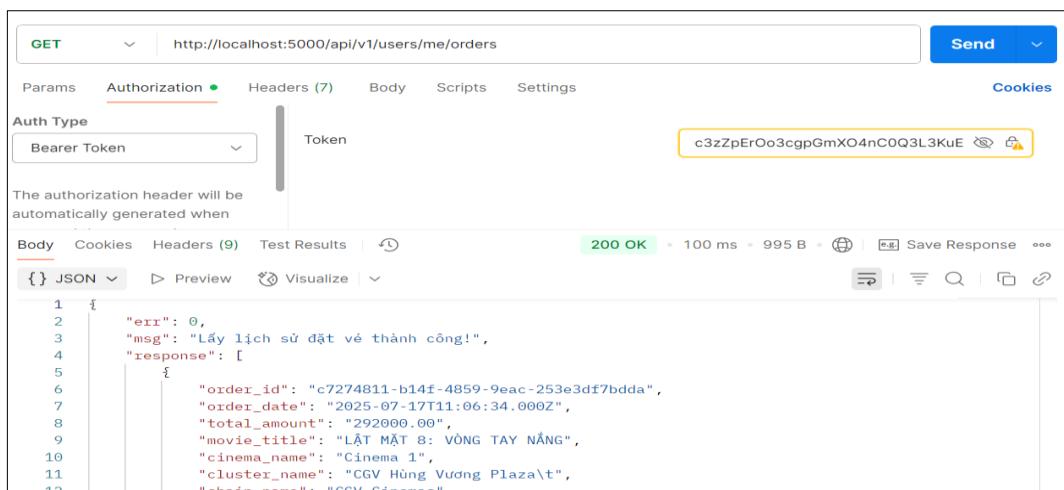
XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

- **API xem lịch sử đặt vé của mình:** <http://localhost:5000/api/v1/users/me/orders> (method GET): Bắt buộc phải có JWT Token đính kèm trong header dưới dạng Bearer Token. Middleware sẽ kiểm tra token này để lấy đúng lịch sử đặt vé của người dùng tương ứng. Vì vậy, người dùng phải đăng nhập để có được token từ API đăng nhập và mới có thể sử dụng API này.
 - + Trường hợp thiếu token hay chưa đăng nhập trên website: Postman sẽ trả về status 401 Unauthorized có ý nghĩa là server từ chối yêu cầu vì người dùng chưa xác thực hoặc xác thực không hợp lệ (do middleware xử lý) và mã lỗi do lập trình viên backend định nghĩa khi có lỗi là 1.



Hình 6.2. Postman trả về status 401 khi thiếu token trong header

- + Trường hợp có cung cấp token thì Postman sẽ trả về status 200 OK và cùng với mảng danh sách các vé đã được đặt bởi người dùng đã xác thực đó.



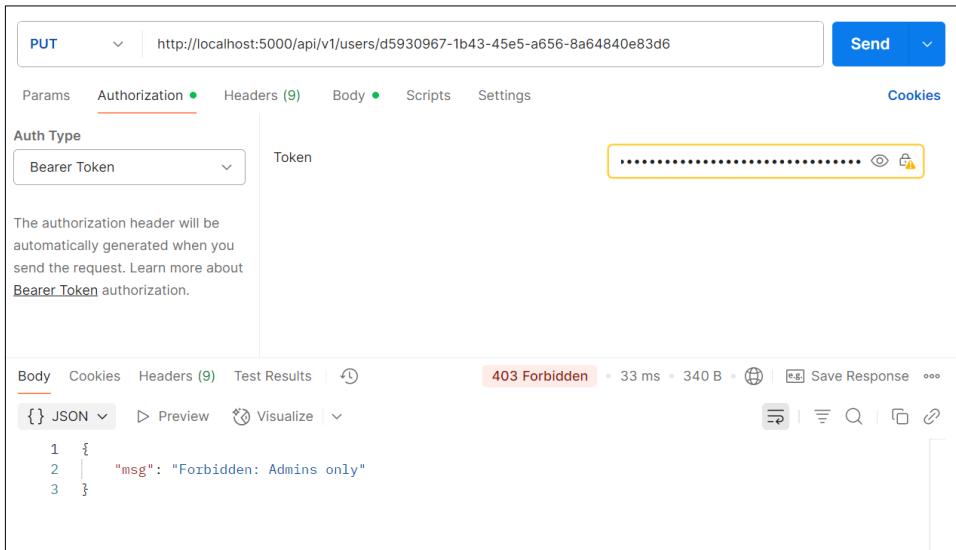
Hình 6.3. Postman trả về status 200 khi có header token

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

- API admin cập nhật thông tin của một người dùng:

- + Route: http://localhost:5000/api/v1/users/:user_id.
- + Tham số: user_id (mã định danh của người dùng được cập nhật thông tin).
- + Method: PUT.
- + Yêu cầu JWT token, trong đó chứa thông tin vai trò (role) của người dùng. Token này chứa thông tin vai trò của người dùng trong đó có vai trò người dùng.
- + Phân quyền: Chỉ người dùng có vai trò admin mới được phép thực hiện API này. Middleware sẽ kiểm tra vai trò trong token, nếu không phải admin sẽ trả về phản hồi:

```
{  
  "msg": "Forbidden: Admins only"  
}
```



Hình 6.4. Postman trả về lỗi 403 khi token không phải của admin

- + Nếu có xác thực token và là admin thì API sẽ trả về thông tin người dùng sau khi cập nhật dựa vào tham số user_id trên route và dữ liệu gửi đi. Ví dụ cập nhật cho người dùng đã tồn tại ở hình 6.2 với user_id là d5930967-1b43-45e5-a656-8a64840e83d6 được tạo tự động bằng chuẩn UUID v4 nhờ thư viện của NodeJs và request body như sau:

```
"username" : "demo"
```

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN

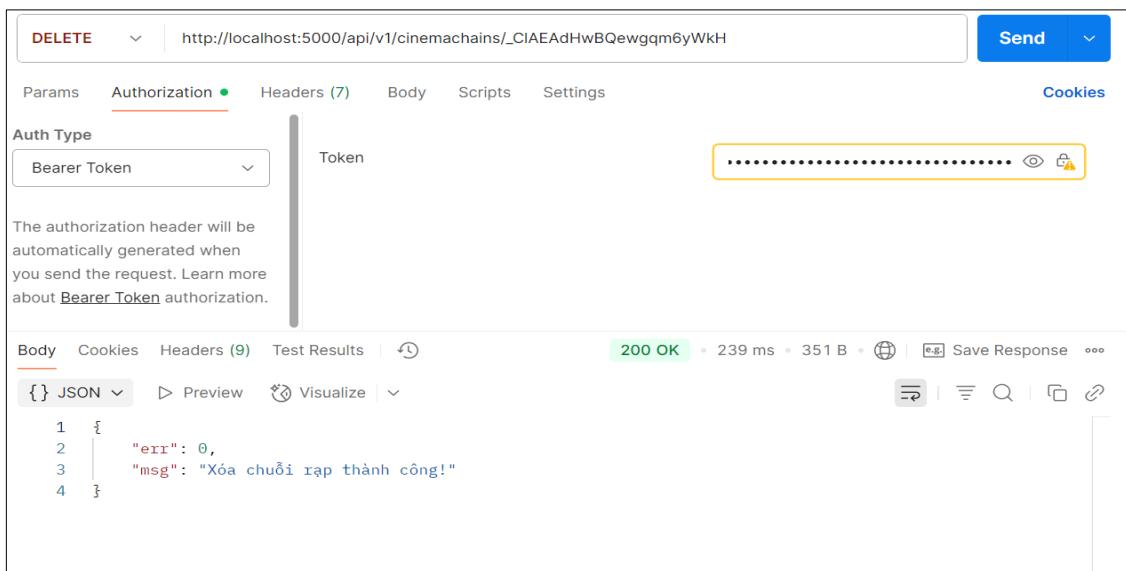
The screenshot shows a Postman interface with a 'PUT' request sent to `http://localhost:5000/api/v1/users/d5930967-1b43-45e5-a656-8a64840e83d6`. The 'Body' tab is selected, showing a JSON payload with 'username' set to 'demo' and 'phone' set to '111111'. The response status is '200 OK' with a response time of 143 ms and a body size of 375 B. The response JSON is displayed as:

```
1 {  
2   "err": 0,  
3   "msg": "Cập nhật thành công",  
4   "response": {  
5     "username": "demo"  
6   }  
7 }
```

Hình 6.5. Postman trả về status 200 khi cập nhật người dùng thành công

- + Có thể thấy ở trường hợp hợp lệ về xác thực, phân quyền và tham số thì Postman trả về status 200 OK, mã lỗi là 0, thông báo cập nhật thành công và trường được cập nhật dựa vào request.
- **API admin xóa một chuỗi rạp khỏi hệ thống:**
 - + Route: `http://localhost:5000/api/v1/cinemachains/:chain_id`
 - + Method: **DELETE**.
 - + Tham số: `chain_id` – là mã định danh (UUID) của chuỗi rạp cần xóa.
 - + Yêu cầu xác thực & phân quyền: Cần JWT token đính kèm trong header theo định dạng Bearer Token. Middleware sẽ xác minh token và kiểm tra quyền vai trò (role) của người dùng từ token đó. Chỉ người dùng có role là admin mới được phép thực hiện thao tác xóa chuỗi rạp.

XÂY DỰNG HỆ THỐNG ĐẶT VÉ XEM PHIM TRỰC TUYẾN



Hình 6.6. Postman trả về status 200 khi xóa một chuỗi rạp thành công

- + Khi một chuỗi rạp được xóa thì các cụm rạp và rạp được liên kết qua các khóa ngoại của nó cũng sẽ bị xóa theo do thuộc tính **onDelete**: “**CASCADE**” trong cơ sở dữ liệu nhưng nếu một vé đã được đặt có thông tin của chuỗi rạp này thì thông tin đó sẽ không bị xóa nhò nhàng trường snapshot trong bảng Ticket.

CHƯƠNG 7: ĐÁNH GIÁ VÀ KẾT LUẬN

7.1 Kết quả đạt được

- Xây dựng hoàn chỉnh một website đặt vé xem phim với các chức năng chính như: đăng ký, đăng nhập, xem danh sách phim và suất chiếu, chọn ghế, đặt vé, thanh toán và theo dõi lịch sử đặt vé. Bên cạnh đó, hệ thống còn hỗ trợ quản lý người dùng, quản lý phim, suất chiếu, rạp chiếu và thống kê doanh thu theo từng thời điểm...
- Ứng dụng sử dụng cron job để tự động xóa các suất chiếu đã quá thời gian cho phép đặt vé. Điều này giúp hệ thống luôn cập nhật dữ liệu hiển thị, giảm tải truy vấn không cần thiết và cải thiện hiệu năng tổng thể.
- Thiết kế UI/UX bằng Figma, bám sát trải nghiệm người dùng thực tế; triển khai thành công trên frontend bằng React và Tailwind CSS.
- Xây dựng và triển khai kiến trúc RESTful API rõ ràng giữa client và server. Các API được kiểm thử hiệu quả bằng Postman và được tự động sinh tài liệu với Swagger, hỗ trợ mở rộng và bảo trì hệ thống sau này.
- Áp dụng mô hình phát triển phần mềm Agile/Scrum, chia công việc theo các sprint rõ ràng, sử dụng Jira để phân công, theo dõi tiến độ và cập nhật trạng thái từng nhiệm vụ một cách minh bạch giữa các thành viên trong nhóm.
- Nâng cao kỹ năng làm việc nhóm và sử dụng công cụ hỗ trợ phát triển phần mềm, như Git/GitHub để quản lý mã nguồn, Jira để quản lý công việc, Postman để kiểm thử API, Swagger để viết tài liệu kỹ thuật và Figma cho thiết kế giao diện.

7.2 Khó khăn gặp phải

- Tích hợp Docker và tối ưu hiệu suất chưa hoàn chỉnh, chưa triển khai được CI/CD trên GitHub Action hoặc cloud như AWS.
- Chưa có nhiều kinh nghiệm xây dựng một hệ thống đặt vé hoàn chỉnh cũng như lập trình full-stack, giao tiếp giữa lập trình viên front-end và back-end.
- Việc áp dụng Agile, Scrum, sử dụng Jira trong quá trình phát triển còn chưa thực sự hiệu quả tối ưu, chưa quá đúng với tiến độ thực tế.

7.3 Bài học kinh nghiệm được rút ra

- Nâng cao kỹ năng teamwork, quản lý dự án và sử dụng các công cụ như GitHub, Jira, Postman.
- Thành thạo quy trình phát triển phần mềm theo mô hình Agile/Scrum, bao gồm lập kế hoạch sprint, daily meeting và review định kỳ.
- Rèn luyện kỹ năng thiết kế hệ thống theo hướng fullstack (frontend - backend - database).
- Hiểu rõ hơn quy trình phát triển một hệ thống phần mềm thực tế thông qua dự án website đặt vé xem phim, bao gồm các giai đoạn: phân tích, thiết kế, lập trình, kiểm thử và triển khai.
- Thành thạo hơn trong việc thiết kế API RESTful, kiểm thử bằng Postman, tài liệu hóa bằng Swagger và phối hợp hiệu quả giữa frontend – backend.

7.4 Đề xuất cải thiện

- Tích hợp thêm thanh toán thực tế (VNPay, Momo...), chức năng bình luận, đánh giá, độ yêu thích của một phim. Triển khai ứng dụng di động (React Native).
- Áp dụng các kiến trúc nâng cao như **Clean Architecture**, **CQRS**, phân tách module rõ ràng hơn. Tự động hóa quy trình triển khai (CI/CD) bằng Docker Compose và GitHub Actions nhằm tăng tính chuyên nghiệp trong vận hành hệ thống.
- Bổ sung các API nâng cao cho việc quản lý hệ thống như thông kê chuyên sâu, áp dụng các API chức năng quản lý đơn hàng, cấp lại mật khẩu, quên mật khẩu được thực hiện hóa lên giao diện.
- Cải thiện cơ sở dữ liệu: Chuẩn hoá dữ liệu bằng cách tách các thuộc tính dạng enum thành bảng riêng như: trang_thai_ve, vai_tro_nguo_dung, hinh_thuc_thanh_toan để dễ mở rộng. Thêm bảng về tỉnh thành hỗ trợ triển khai đa khu vực. Có thể thay thế bằng hệ quản trị NoSQL như MongoDB.
- Chuyên nghiệp hóa quy trình Scrum bằng cách chia nhỏ các user story thành task rõ ràng, cụ thể hơn. Jira được sử dụng hiệu quả để lập sprint, phân công và theo dõi tiến độ từng ngày qua burndown chart. Điều này giúp dự đoán, đo lường chất lượng và tốc độ làm việc nhóm một cách sát thực tế hơn.

TÀI LIỆU THAM KHẢO

[1]	M. Richards và N. Ford, Fundamentals of Software Architecture: An Engineering Approach, O'Reilly Media, 2020.
[2]	R. C. Martin, Clean Architecture: A Craftsman's Guide to Software Structure and Design, Prentice Hall, 2017.
[3]	Moveek. "Moveek." [Trực tuyến]. Có tại: https://moveek.com/ . [Truy cập: 15/04/2025].
[4]	CGV Vietnam. "CGV Vietnam." [Trực tuyến]. Có tại: https://www.cgv.vn/default/ . [Truy cập: 30/03/2025].
[5]	[3] Mozilla, “Promise”, MDN Web Docs, [Trực tuyến]. Có tại: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise/Promise . [Truy cập: 15/04/2025].
[6]	Nguyen Van Truong, “Sequelize - JavaScript ORM cho NodeJS”, Viblo, 28/11/2015. [Trực tuyến]. Có tại: https://viblo.asia/p/sequelize-javascript-orm-cho-nodejs-l0rvmmJDvyqA . [Truy cập: 20/04/2025].
[7]	Axios, “Interceptors”, Axios Documentation, [Trực tuyến]. Có tại: https://axios-http.com/docs/interceptors . [Truy cập: 17/04/2025].
[8]	Nguyễn Thu Hué, “Agile là gì? Phương pháp Agile và cách áp dụng vào quản trị doanh nghiệp”, Base Blog, 16/02/2024. [Trực tuyến]. Có tại: https://base.vn/blog/agile-la-gi/ . [Truy cập: 01/05/2025].
[9]	"createStore - Redux API." [Trực tuyến]. Có tại: https://redux.js.org/api/createstore . [Truy cập: 20/04/2025].
[10]	S. Unlu, “How to wait for a container to be ready,” Medium, 04/08/2019. [Trực tuyến]. Có tại: https://selahattinunlu.medium.com/how-to-wait-for-a-container-to-be-ready-757bc1a86468 . [Truy cập: 10/07/2025].