**Báo Cáo**

**PHÂN TÍCH THIẾT KẾ VÀ XÂY DỰNG   
WEBSITE ĐẶT VÉ XEM PHIM**

|  |  |  |  |
| --- | --- | --- | --- |
| Giáo viên hướng dẫn |  | : | xxx |
| Sinh viên thực hiện |  | : | xxx |
| Lớp |  | : | xxx |
| Khoa |  | : | xxx |

**MỤC LỤC**

[CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI 1](#_Toc172850150)

[1.1. Lý do chọn đề tài: 1](#_Toc172850151)

[1.2. Mục đích của đề tài: 1](#_Toc172850152)

[1.3. Phương pháp nghiên cứu: 2](#_Toc172850153)

[1.4. Ý nghĩa của đề tài: 2](#_Toc172850154)

[CHƯƠNG 2. KIẾN THỨC NỀN TẢNG 4](#_Toc172850155)

[2.1. Cơ sở lý thuyết: 4](#_Toc172850156)

[2.1.1. Ngôn ngữ lập trình: 4](#_Toc172850157)

[2.1.2. Hệ quản trị cơ sở dữ liệu: 5](#_Toc172850158)

[2.2. Công cụ - công nghệ sử dụng: 6](#_Toc172850159)

[2.2.1. Ngôn ngữ lập trình ReactJS: 6](#_Toc172850160)

[2.2.2. Ngôn ngữ lập trình NodeJS: 8](#_Toc172850161)

[2.2.3. Mô hình MVC: 9](#_Toc172850162)

[2.2.4. Xử lý đăng nhập với JWT (Json Web Token): 11](#_Toc172850163)

[2.2.5. MySQL: 13](#_Toc172850164)

[2.2.6. XamPP: 14](#_Toc172850165)

[CHƯƠNG 3. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 15](#_Toc172850166)

[3.1. Use Case của hệ thống: 15](#_Toc172850167)

[3.1.1. Use Case tổng quát: 15](#_Toc172850168)

[3.1.2. Use Case hệ thống: 16](#_Toc172850169)

[3.1.3. Use Case của tác nhân NGƯỜI DÙNG: 18](#_Toc172850170)

[3.1.4. Use Case Quản lý thể loại: 21](#_Toc172850171)

[3.1.5. Use Case Quản lý lịch chiếu phim: 23](#_Toc172850172)

[3.1.6. Use Case Quản lý phim: 26](#_Toc172850173)

[3.1.7. Use Case Quản lý tài khoản: 28](#_Toc172850174)

[3.1.8. Use Case Quản lý hàng ghế - số ghế: 31](#_Toc172850175)

[3.1.9. Use Case Quản lý đặt vé: 33](#_Toc172850176)

[3.2. Biểu đồ hoạt động: 35](#_Toc172850177)

[3.2.1. Biểu đồ hoạt động Đăng ký: 35](#_Toc172850178)

[3.2.2. Biểu đồ hoạt động Đăng nhập: 36](#_Toc172850179)

[3.2.3. Biểu đồ hoạt động Thêm thể loại: 37](#_Toc172850180)

[3.2.4. Biểu đồ hoạt động Cập nhật thể loại: 38](#_Toc172850181)

[3.2.5. Biểu đồ hoạt động Xóa thể loại: 39](#_Toc172850182)

[3.2.6. Biểu đồ hoạt động Thêm thông tin phim: 40](#_Toc172850183)

[3.2.7. Biểu đồ hoạt động Cập nhật phim: 41](#_Toc172850184)

[3.2.8. Biểu đồ hoạt động Xóa phim: 42](#_Toc172850185)

[3.2.9. Biểu đồ hoạt động Đặt vé: 43](#_Toc172850186)

[3.3. Biểu đồ tuần tự: 44](#_Toc172850187)

[3.3.1. Biểu đồ tuần tự Đăng ký: 44](#_Toc172850188)

[3.3.2. Biểu đồ tuần tự Đăng nhập: 45](#_Toc172850189)

[3.3.3. Biểu đồ tuần tự Thêm thể loại: 46](#_Toc172850190)

[3.3.4. Biểu đồ tuần tự Cập nhật thể loại: 47](#_Toc172850191)

[3.3.5. Biểu đồ tuần tự Xóa thể loại: 48](#_Toc172850192)

[3.3.6. Biểu đồ tuần tự Thêm phim: 49](#_Toc172850193)

[3.3.7. Biểu đồ tuần tự Cập nhật phim: 50](#_Toc172850194)

[3.3.8. Biểu đồ tuần tự Xóa phim: 51](#_Toc172850195)

[3.3.9. Biểu đồ tuần tự Đặt vé: 52](#_Toc172850196)

[3.4. Thiết kế cơ sở dữ liệu: 53](#_Toc172850197)

[3.4.1. Lược đồ quan hệ: 53](#_Toc172850198)

[3.4.1. Xác định các thực thể: 54](#_Toc172850199)

[CHƯƠNG 4. GIAO DIỆN PHẦN MỀM 59](#_Toc172850200)

[4.1. Giao diện người dùng: 59](#_Toc172850201)

[4.1.1. Giao diện đăng ký: 59](#_Toc172850202)

[4.1.2. Giao diện đăng nhập: 59](#_Toc172850203)

[4.1.3. Giao diện xem chi tiết phim: 59](#_Toc172850204)

[4.1.4. Giao diện đặt vé (1): 59](#_Toc172850205)

[4.1.5. Giao diện đặt vé (2): 59](#_Toc172850206)

[4.1.6. Giao diện đặt vé (3): 59](#_Toc172850207)

[4.2. Giao diện quản trị: 59](#_Toc172850208)

[4.2.1. Giao diện đăng nhập: 59](#_Toc172850209)

[4.2.2. Giao diện quản lý thể loại: 59](#_Toc172850210)

[4.2.3. Giao diện quản lý phim: 60](#_Toc172850211)

[4.2.4. Giao diện quản lý đặt vé: 60](#_Toc172850212)

[CHƯƠNG 5. TỔNG KẾT 60](#_Toc172850213)

[5.1. Tổng kết: 60](#_Toc172850214)

[5.2.  Kết quả đạt được: 60](#_Toc172850215)

[5.3. Hạn chế: 60](#_Toc172850216)

[5.4. Hướng phát triển: 60](#_Toc172850217)

[TÀI LIỆU THAM KHẢO 61](#_Toc172850218)

**DANH MỤC HÌNH**

[*Hình 2. 1. Mô hình hoạt động của MySQL 6*](#_Toc172850312)

[*Hình 2. 2. JSX Code Style 7*](#_Toc172850313)

[*Hình 2. 3. So sánh Virtual DOM và Real DOM 8*](#_Toc172850314)

[*Hình 2. 4. Cấu trúc hoạt động của NodeJS 8*](#_Toc172850315)

[*Hình 2. 5. Mô hình MVC 9*](#_Toc172850316)

[*Hình 2. 6. Xử lý authentication với JWT trong NodeJS 12*](#_Toc172850317)

[*Hình 3. 1. Use Case tổng quát 15*](#_Toc172850320)

[*Hình 3. 2. Use Case hệ thống 16*](#_Toc172850321)

[*Hình 3. 3. Use Case tác nhân NGƯỜI DÙNG 18*](#_Toc172850322)

[*Hình 3. 4. Use Case Quản lý thể loại 21*](#_Toc172850323)

[*Hình 3. 5. Use Case Quản lý lịch chiếu phim 23*](#_Toc172850324)

[*Hình 3. 6. Use Case Quản lý phim 26*](#_Toc172850325)

[*Hình 3. 7. Use Case Quản lý tài khoản 28*](#_Toc172850326)

[*Hình 3. 8. Use Case Quản lý hàng ghế - số ghế 31*](#_Toc172850327)

[*Hình 3. 9. Use Case Quản lý đặt vé 33*](#_Toc172850328)

[*Hình 3. 10 Biểu đồ hoạt động Đăng ký 35*](#_Toc172850329)

[*Hình 3. 11 Biểu đồ hoạt động Đăng nhập 36*](#_Toc172850330)

[*Hình 3. 12 Biểu đồ hoạt động Thêm thể loại 37*](#_Toc172850331)

[*Hình 3. 13. Biểu đồ hoạt động Cập nhật thể loại 38*](#_Toc172850332)

[*Hình 3. 14. Biểu đồ hoạt động Xóa thể loại 39*](#_Toc172850333)

[*Hình 3. 15. Biểu đồ hoạt động Thêm thông tin phim 40*](#_Toc172850334)

[*Hình 3. 16. Biểu đồ hoạt động Cập nhật phim 41*](#_Toc172850335)

[*Hình 3. 17. Biểu đồ hoạt động Xóa phim 42*](#_Toc172850336)

[*Hình 3. 18. Biểu đồ hoạt động Đặt vé 43*](#_Toc172850337)

[*Hình 3. 19. Biểu đồ tuần tự Đăng ký 44*](#_Toc172850338)

[*Hình 3. 20. Biểu đồ tuần tự Đăng nhập 45*](#_Toc172850339)

[*Hình 3. 21. Biểu đồ tuần tự Thêm thể loại 46*](#_Toc172850340)

[*Hình 3. 22. Biểu đồ tuần tự Cập nhật thể loại 47*](#_Toc172850341)

[*Hình 3. 23. Biểu đồ tuần tự Xóa thể loại 48*](#_Toc172850342)

[*Hình 3. 24. Biểu đồ tuần tự Thêm phim 49*](#_Toc172850343)

[*Hình 3. 25. Biểu đồ tuần tự Cập nhật phim 50*](#_Toc172850344)

[*Hình 3. 26. Biểu đồ tuần tự Xóa phim 51*](#_Toc172850345)

[*Hình 3. 27. Biểu đồ tuần tự Đặt vé 52*](#_Toc172850346)

[*Hình 3. 28. Lược đồ quan hệ 53*](#_Toc172850347)

[*Hình 4. 1. Giao diện đăng ký 59*](#_Toc172850348)

[*Hình 4. 2. Giao diện đăng nhập 59*](#_Toc172850349)

[*Hình 4.3. Giao diện xem chi tiết phim 60*](#_Toc172850350)

[*Hình 4.4. Giao diện đặt vé (1) 60*](#_Toc172850351)

[*Hình 4.5. Giao diện đặt vé (2) 61*](#_Toc172850352)

[*Hình 4.6. Giao diện đặt vé (3) 61*](#_Toc172850353)

[*Hình 4.7. Giao diện đăng nhập 62*](#_Toc172850354)

[*Hình 4.8. Giao diện quản lý thể loại 62*](#_Toc172850355)

[*Hình 4. 9. Giao diện quản lý phim 63*](#_Toc172850356)

[*Hình 4. 10. Giao diện quản lý đặt vé 63*](#_Toc172850357)

**DANH MỤC BẢNG**

[*Bảng 3. 1. Bảng movie\_directors 54*](#_Toc172850373)

[*Bảng 3. 2. Bảng movie\_genre 54*](#_Toc172850374)

[*Bảng 3. 3. Bảng movie 54*](#_Toc172850375)

[*Bảng 3. 4. Bảng shown\_in 55*](#_Toc172850376)

[*Bảng 3. 5. Bảng features 55*](#_Toc172850377)

[*Bảng 3. 6. Bảng showtimes 55*](#_Toc172850378)

[*Bảng 3. 7. Bảng theatre 56*](#_Toc172850379)

[*Bảng 3. 8. Bảng hall 56*](#_Toc172850380)

[*Bảng 3. 9. Bảng hallwise\_seat 56*](#_Toc172850381)

[*Bảng 3. 10. Bảng seat 57*](#_Toc172850382)

[*Bảng 3. 11. Bảng person 57*](#_Toc172850383)

[*Bảng 3. 12. Bảng payment 57*](#_Toc172850384)

[*Bảng 3. 13. Bảng ticket 58*](#_Toc172850385)

# CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

* 1. **Lý do chọn đề tài:**

Có nhiều lý do để quyết định chọn đề tài "Phân tích, thiết kế và xây dựng website đặt vé xem phim bằng NodeJS & ReactJS" cho đồ án tốt nghiệp. Quan trọng nhất là trang web này sẽ giúp người dùng dễ dàng tìm kiếm và đặt vé xem phim theo lịch chiếu và rạp mong muốn, giúp tiết kiệm thời gian và công sức trong quá trình đặt vé.

Trang web cũng sẽ đóng vai trò liên kết giữa người dùng và các rạp chiếu phim. Nó không chỉ giúp người dùng tiếp cận các thông tin về phim và rạp mà còn tạo ra kênh giao tiếp hiệu quả giữa hai bên. Thông qua các tính năng như đánh giá phim, gửi phản hồi, và nhận thông báo khuyến mãi, website sẽ làm cầu nối và tăng cường sự tương tác giữa người dùng và rạp chiếu phim.

Bên cạnh đó, việc sử dụng NodeJS và ReactJS là lựa chọn phù hợp cho dự án này. NodeJS giúp xử lý các yêu cầu từ máy chủ nhanh chóng, cho phép xây dựng ứng dụng có khả năng mở rộng cao. ReactJS mang lại giao diện người dùng thân thiện và dễ sử dụng, giúp trải nghiệm người dùng trở nên mượt mà và trực quan. Sự kết hợp của hai công nghệ này sẽ tạo nên một website mạnh mẽ, hiệu quả.

Thêm vào đó, website sẽ cung cấp các tin tức hữu ích về các bộ phim sắp chiếu. Điều này sẽ giúp người dùng cập nhật thông tin, lựa chọn phim phù hợp, và có trải nghiệm xem phim tốt hơn.

Tóm lại, đề tài "Phân tích, thiết kế và xây dựng website đặt vé xem phim bằng NodeJS & ReactJS" nhằm tạo ra một nền tảng đặt vé thông minh và hiệu quả, giúp nâng cao chất lượng dịch vụ, hỗ trợ người dùng và các rạp chiếu phim phát triển.

* 1. **Mục đích của đề tài:**
* Khảo sát thực trạng đặt vé xem phim của các người dùng hiện nay.
* Trình bày được các vấn đề liên quan đến ngôn ngữ lập trình NodeJS, ReactJS và hệ quản trị cơ sở dữ liệu MySQL.
* Phân tích hệ thống chương trình bao gồm các giao diện, các chức năng nhằm đáp ứng nhu cầu người dùng.
* Triển khai và cài đặt chương trình thông quan phần mềm Visual Studio Code.
  1. **Phương pháp nghiên cứu:**
* Phương pháp thu thập tài liệu: Tập trung vào nghiên cứu lý thuyết từ giáo trình và các nguồn tài liệu trực tuyến đáng tin cậy. Sử dụng các bài báo và tài liệu từ nhiều nguồn để đảm bảo thông tin phong phú và chính xác.
* Phương pháp quan sát: Theo dõi và ghi nhận các hoạt động thực tế và quy trình trong quá trình nghiên cứu. Phân tích các quy luật và mô hình hoạt động để đề xuất giải pháp mới và xây dựng hệ thống hiệu quả hơn. Quan sát giúp phát hiện vấn đề thực tiễn và đưa ra giải pháp kịp thời.
* Phương pháp khảo sát chuyên sâu: Tham khảo ý kiến giảng viên và sinh viên khóa trước để thu thập thông tin chi tiết về nhu cầu và mong đợi từ hệ thống đặt vé xem phim. Cung cấp dữ liệu định tính quan trọng, bổ sung cho thông tin từ tài liệu và quan sát, giúp hiểu rõ hơn về yêu cầu và mong muốn của người dùng.
* Phương pháp phân tích và thiết kế hệ thống: Sử dụng kỹ thuật phân tích hướng đối tượng và thiết kế cơ sở dữ liệu dựa trên mô hình hóa để xây dựng hệ thống có hệ thống và hiệu quả. Dùng công cụ UML (Unified Modeling Language) để mô hình hóa các thành phần của hệ thống, đảm bảo thiết kế rõ ràng và logic.
  1. **Ý nghĩa của đề tài:**
* Ý nghĩa khoa học:Vận dụng các kiến thức đã học để có thể xây dựng website đặt vé xem phim:
* Nghiên cứu và lập trình với các ngôn ngữ NodeJS, ReactJS, cùng với hệ quản trị cơ sở dữ liệu MySQL.
* Nghiên cứu phân tích và thiết kế hệ thống.
* Nghiên cứu xây dựng giao diện website.
* Ý nghĩa thực tiễn:Website đặt vé xem phim giúp việc quản lý nhanh chóng, lưu trữ lượng dữ liệu lớn, còn có giao diện thân thiện, dễ sử dụng… Website mang lại rất nhiều lợi ích cho người dùng:
  + Giúp quản lý danh sách phim và lịch chiếu phim sắp tới với nhiều thao tác nhanh chóng.
  + Người dùng có thể theo dõi, xem chi tiết bộ phim muốn xem và thanh toán đặt vé thuận tiện.
  + Người quản lý dễ dàng trong việc tìm kiếm, lưu trữ, công khai lịch chiếu các bộ phim, cũng như thống kê số liệu theo nhu cầu.

# CHƯƠNG 2. KIẾN THỨC NỀN TẢNG

## Cơ sở lý thuyết:

### Ngôn ngữ lập trình:

*Javascript* là một ngôn ngữ lập trình của HTML và ứng dụng Web. Nó được sử dụng phổ biến nhất, như một phần của các trang web. Chúng cho phép client-server script tương tác với người sử dụng và tạo các trang web động. Nó là một ngôn ngữ chương trình thông dịch với các khả năng hướng đối tượng.

Một số ưu điểm của JavaScript:

* Tiêt kiệm băng thông máy chủ:

Javascript sử dụng trong thiết kế website có thể giúp tiết kiệm băng thông máy máy chủ vô cùng tốt bởi chúng được chạy trực tiếp trên máy của người dùng. Gia tăng hiệu suất, kéo dài tuổi thọ máy chủ và tiết kiệm diện tích băng thông.

* Linh hoạt vận hành, tương thích tốt:

Để thực thi, trình duyệt web chỉ cần tải file JavaScript về máy chủ từ một tên miền riêng biệt hoặc nhúng thẳng vào file HTML và chạy mà không cần tải thêm. Javascript có thể chạy được trên hầu hết các trình duyệt phổ biến hiện nay.

Hầu hết các trình duyệt web đều có tích hợp sẵn trình thông dịch ngôn ngữ JavaScript, đảm bảo người dùng có thể sử dụng ngay mà không mất công tải về trình duyệt, giảm tỷ lệ thực thi các chức năng của JavaScript.

* Dễ dàng xử lý vấn đề:

Với lợi thế có nhiều trình duyệt hỗ trợ nên việc kiểm tra và xử lý vấn đề dễ dàng hơn. Đa số các trình duyệt web hiện nay đều có cài sẵn các công cụ xử lý lỗi Javascript trong bảng điều khiển trình duyệt. Javascript sở hữu cấu trúc dễ đọc với các nguyên tắc rõ ràng nên việc xác định và gỡ lỗi trở nên đơn giản hơn.

* Nâng cao trải nghiệm người dùng:

Website sử dụng ngôn ngữ lập trình giúp thu hút người dùng bởi những tính năng ưu việt, hình ảnh sinh động hấp dẫn, chức năng lướt và tương tác nhanh nhạy. Nhờ những tính năng đó, người dùng cảm thấy thoải mái, thích thú và tin dùng trang web hơn.

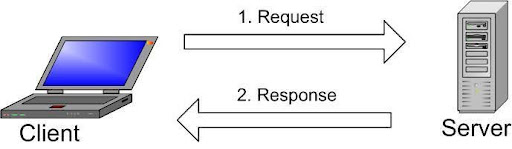
### Hệ quản trị cơ sở dữ liệu:

*Cơ sở dữ liệu (Database)* là một tập hợp các dữ liệu có tổ chức, thường được lưu trữ và truy cập điện tử từ hệ thống máy tính. Khi cơ sở dữ liệu phức tạp hơn, chúng thường được phát triển bằng cách sử dụng các kỹ thuật thiết kế và mô hình hóa chính thức.

*Hệ quản trị cơ sở dữ liệu (Database Management System)* là một phần mềm để lưu trữ và truy xuất dữ liệu của người dùng trong khi xem xét các biện pháp bảo mật thích hợp. Nó bao gồm một nhóm các chương trình thao tác cơ sở dữ liệu. Hệ quản trị cơ sở dữ liệu chấp nhận yêu cầu dữ liệu từ một ứng dụng và hướng dẫn hệ điều hành cung cấp dữ liệu cụ thể. Trong các hệ thống lớn, nó giúp người dùng và phần mềm bên thứ ba lưu trữ và truy xuất dữ liệu.

*MySQL* là một hệ thống quản trị cơ sở dữ liệu mã nguồn mở (*Relational Database Management System, viết tắt là RDBMS*) hoạt động theo mô hình client-server. MySQL là một trong số các phần mềm của RDBMS, nhưng thường được cho là một vì độ quá phổ biến của nó. Các ứng dụng web lớn nhất như Facebook, Youtube, Google... đều dùng MySQL cho mục đích lưu trữ dữ liệu.

Quy trình hoạt động khá đơn giản mỗi Client(máy khách) yêu cầu dữ liệu từ Server(máy chủ), từ Server sẽ phản hồi lại các dữ liệu đó. Ví dụ khi người dùng truy cập vào một trang web nào đó các Request sẽ gửi tới Server, lúc này Server sẽ nhận được Request và bắt đầu gửi dữ liệu quay lại máy người dùng để hiển thị thông tin trang web đó lên.



Hình 2. 1. Mô hình hoạt động của MySQL

Một số lý do chính khiến MySQL trở nên quan trọng:

* Linh hoạt và dễ dùng: Có thể sửa source code để đáp ứng nhu cầu sử dụng mà không phải thanh toán thêm chí phí nào, quá trình cài đặt cũng đơn giản.
* Hiệu năng cao: Bất kể lưu trữ dữ liệu lớn của các trang thương mại điện tử hoặc những hoạt động kinh doanh lớn liên quan đến công nghệ thông tin, MySQL cũng có thể đáp ứng.
* An toàn: An toàn dữ liệu luôn là vấn đề quan trọng nhất khi chọn phần mềm RDBMS. Với hệ thống phân quyền truy cập và quản lý tài khoản, MySQL đặt tiêu chuẩn bảo mật rất cao.

## Công cụ - công nghệ sử dụng:

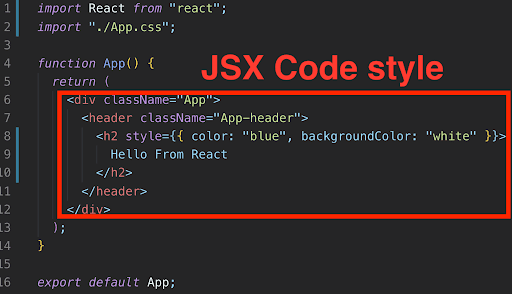
### Ngôn ngữ lập trình ReactJS:

*ReactJS* là một opensource được phát triển bởi Facebook, vào năm 2013. Đây là một thư viện Javascript được dùng để xây dựng các tương tác với các thành phần website. Một trong những điểm nổi bật của thư viện này là việc render dữ liệu không chỉ thực hiện phía Server, mà còn thực hiện được phía Client.

ReactJS là một thư hiện Javascript chuyên giúp các nhà phát triển xây dựng giao diện người dùng hay UI. Trong lập tình ứng dụng front-end, lập trình viên thường phải làm việc chính trên 2 thành phần: UI và xử lý tương tác người dùng. UI là tập hợp những thành phần nhìn thấy được trên bất kỳ một ứng dụng nào, ví dụ thanh tìm kiếm, nút nhấn… Trước khi có ReactJS, lập trình viên thường gặp khó khăn rất nhiều khi sử dụng Javascript thuần và Jquery để xây dựng UI.

Để tăng tốc quá trình phát triển và giảm thiểu rủi ro xảy ra trong khi coding, React còn cung cấp khả năng Reusable Code(tái sử dụng code) bằng cách đưa ra 2 khái niệm quan trọng: JSX và Virtual DOM.

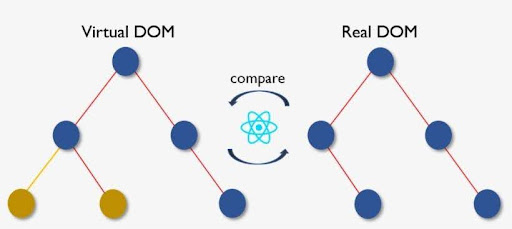
JSX(nói ngắn gọn là Javascript extension) là một React extension giúp chúng ta dễ dàng thay đổi cây DOM bằng các HTML-style code đơn giản. Và ReactJS browser hỗ trợ toàn bộ những trình duyệt web hiện đại, có thể sử dụng JSX trên bất kỳ trình duyệt nào đang làm việc.



Hình 2. 2. JSX Code Style

Nếu không sử dụng ReactJS (và JSX), website sẽ sử dụng HTML để cập nhật lại cây DOM (quá trình thay đổi diễn ra mà người dùng không cần tải lại trang). Cách này sẽ ổn với các trang web nhỏ, đơn giản, static, nhưng với các website lớn, đặc biệt thiên về xử lý tương tác, điều này sẽ làm ảnh hưởng performance website nghiêm trọng vì toàn bộ cây DOM phải reload lại mỗi lần người dùng nhấn tính năng yêu cầu tải lại trang.

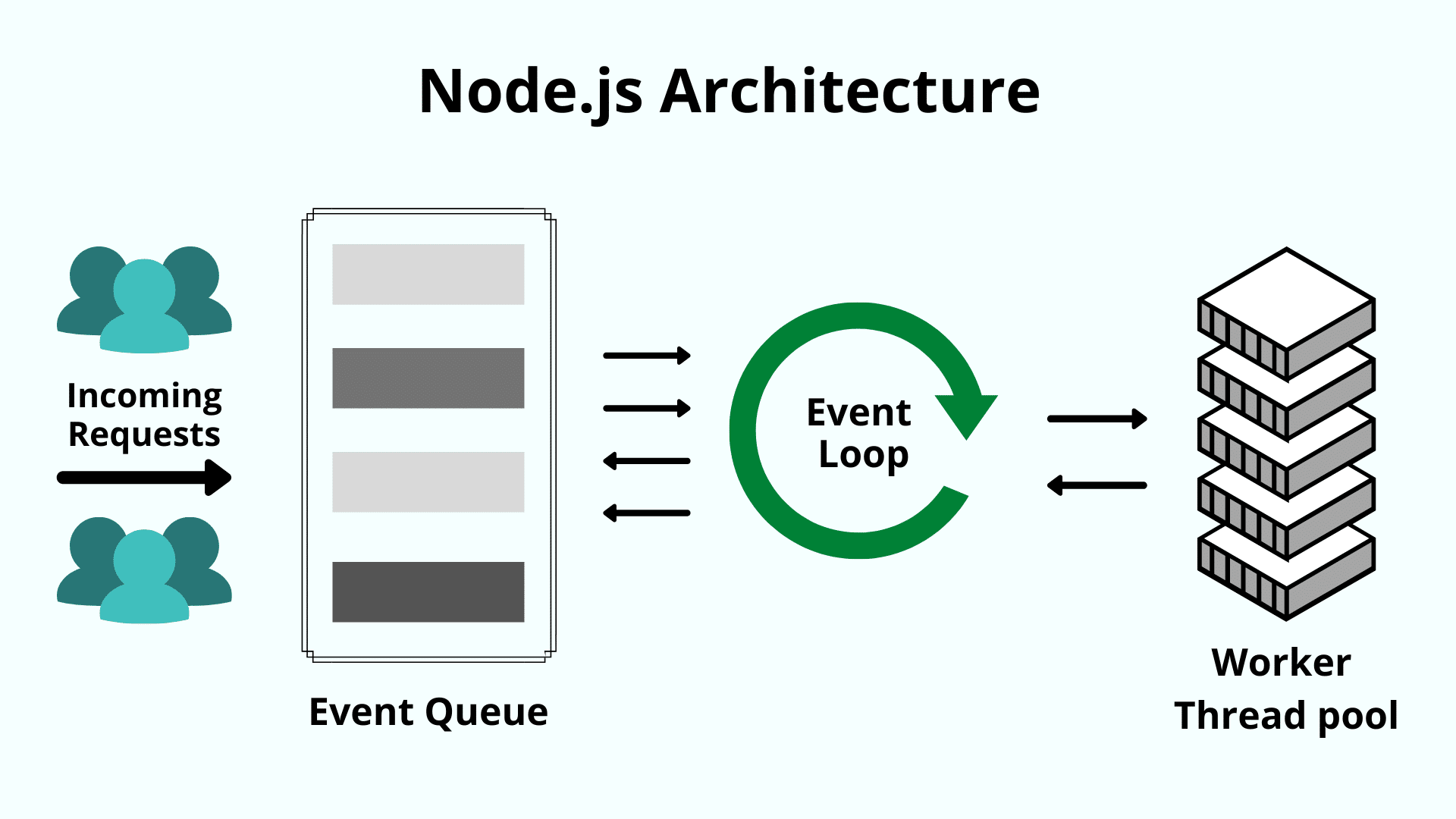
Tuy nhiên, nếu sử dụng JSX sẽ giúp cây DOM cập nhật cho chính DOM đó, ReactJS đã khởi tạo Virtual DOM (DOM ảo). Virtual DOM là bản copy của Real DOM (DOM thật) trên trang đó, và ReactJS sử dụng bản copy để tìm kiếm đúng phần mà DOM thật cần cập nhật khi bất kỳ sự kiện nào khiến thành phần trong nó thay đổi (như việc người dùng nhấn vào một nút).



Hình 2. 3. So sánh Virtual DOM và Real DOM

### Ngôn ngữ lập trình NodeJS:

*NodeJS* được tạo ra bởi Ryan Dahl năm 2009, tạo nên môi trường hỗ trợ chạy JavaScript runtime, đa nền tảng và có mã nguồn mở. Đây là môi trường hoàn hảo để chạy các ứng dụng, web ngoài trình duyệt mà người dùng đang sử dụng. Đây cũng được coi là giải pháp thiết yếu giúp ứng dụng sử dụng dữ liệu hiệu quả nhờ mô hình event driven (mô hình hướng dự kiện) một cách không đồng bộ.

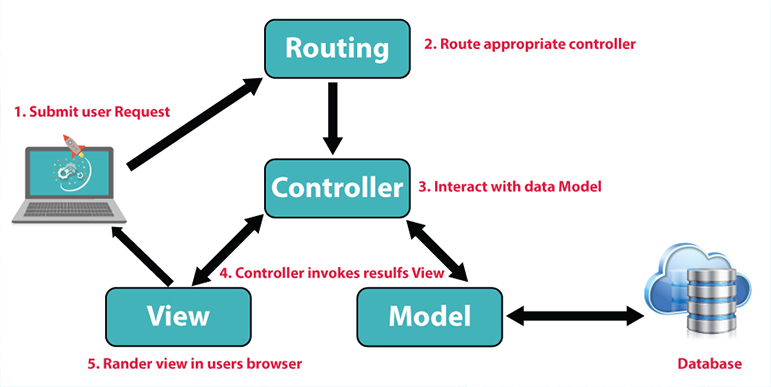


Hình 2. 4. Cấu trúc hoạt động của NodeJS

Cách hoạt động của NodeJS:

* Khi sử dụng Node JS các câu lệnh sẽ được đưa vào queue, hàng đợi. Đồng thời chạy từ trên xuống dưới.
* Lúc này Event loop lấy 1 tác vụ ở queue cho vào stack, và trước đó stack hoàn toàn trống. Điều này có thể hiểu rằng câu lệnh đầu tiên – console.log sẽ được stack xử lý và đưa ra “caulenh1”
* Sau khi stack xử lý xong, tác vụ đầu sẽ được lấy ra và tiếp tục đưa tác vụ tiếp theo vào để xử lý tiếp. Cụ thể có thể là setTimeout(function(){ console.log(“cau lenh 2”); },0);. Lúc này có thể thấy SetTimeout trở thành 1 hàm tính toán thời gian nằm trong web AIPS và đưa vào web AIPs để đợi tác vụ tiếp theo. Nó sẽ được đưa trở lại hàng đợi khi queue đã trống.
* Tương tự khi thực hiện tác vụ 1, tác vụ 3 sẽ in câu lệnh và hàng đợi ở trạng thái rỗng. Khi đó tác vụ 2 sẽ được đưa trở lại queue, tiếp đến sang stack để đi qua xử lý.

### Mô hình MVC:



Hình 2. 5. Mô hình MVC

*Mô hình MVC (Model – View – Controller)* là một kiến trúc phần mềm hay mô hình thiết kế được sử dụng trong kỹ thuật phần mềm. Nó giúp tách ứng dụng thành 3 phần: Model (xử lý dữ liệu), View (hiển thị) và Controller (điều khiển). Mỗi thành phần có một nhiệm vụ riêng biệt và độc lập với nhau.

* View:

View là một phần của ứng dụng chịu trách nhiệm cho việc trình bày dữ liệu. Thành phần này được tạo bởi dữ liệu thu thập từ dữ liệu mô hình, và giúp người dùng có cái nhìn trực quan về trang web, cũng như ứng dụng.

View cũng đại diện cho dữ liệu từ các cuộc trò chuyện, sơ đồ và bảng. Ví dụ: bất kỳ View nào cũng sẽ có bao gồm tất cả các thành phần giao diện người dùng như nút bấm, menu, khung nhập...

* Model:

Model của kiến trúc MVC là thành phần chính đảm nhiệm chức năng lưu trữ dữ liệu và các bộ phận logic liên quan của toàn bộ ứng dụng.

Model chịu trách nhiệm cho các thao tác dữ liệu giữa Controller hoặc bất kỳ logic nghiệp vụ liên quan nào khác như cho phép xem, truy xuất dữ liệu...

* Controller:

Controller xử lý tương tác người dùng của ứng dụng. Nó xử lý dữ liệu đầu vào từ bàn phím và chuột của người dùng, sau đó thông báo tới View và Model.

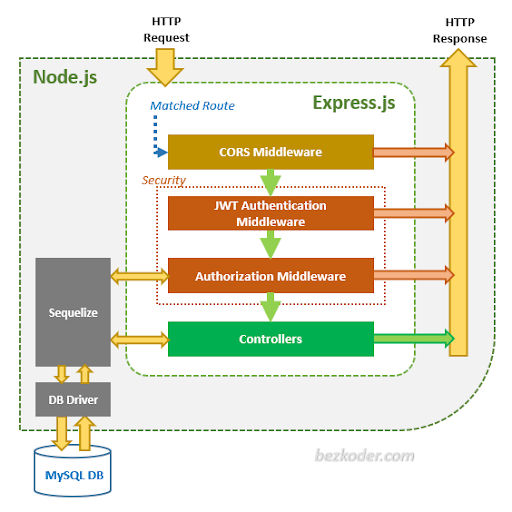
 Controller gửi các lệnh tới Model để thay đổi trạng thái của Model. Controller cũng gửi các lệnh tương tự tới View để thực hiện các thay đổi về giao diện.

Lý do nên áp dụng mô hình MVC:

* Cải thiện thời gian và quy trình lập trình: Nếu áp dụng mô hình MVC để sử dụng và lập trình trên bất kỳ ứng dụng web cụ thể nào, một lập trình viên có thể làm việc trên View, cùng thời gian với lập trình viên khác làm việc với Controller để xây dựng logic nghiệp vụ. Vì vậy, thời gian lập trình khi ứng dụng mô hình MVC có thể được hoàn thành gấp 3 lần so với các mô hình lập trình khác.
* Khả năng cung cấp nhiều chế độ View: Khi sử dụng mô hình MVC, người sử dụng có thể tạo nhiều View cho chỉ một mô hình.
* Những sửa đổi trên website không ảnh hưởng đến toàn bộ mô hình MVC: Với bất kỳ trang web nào, giao diện người dùng thường có xu hướng thay đổi theo thời gian, như bố cục, màu sắc, font chữ… Nên việc xuất hiện thêm kiểu view mới trong MVC sẽ trở nên đơn giản bởi phần Model không phụ thuộc vào phần View.
* Model trong mô hình MVC sẽ trả về dữ liệu mà không cần định dạng: MVC pattern có khả năng trả về các dữ liệu mà không cần áp dụng bất kỳ định dạng cụ thể nào.
* Nền tảng MVC thân thiện và cải thiện SEO: Nền tảng MVC giúp phát triển các trang web trở nên thân thiện hơn với các bộ tìm kiếm. Khi sử dụng nền tảng này, bạn có thể dễ dàng thay đổi và phát triển các URL để cải thiện SEO, từ đó tạo ra nhiều lượt truy cập hơn.

### Xử lý đăng nhập với JWT (Json Web Token):

[JSON Web Token (JWT)](https://jwt.io/) là một chuẩn mở (RFC 7519) định nghĩa một cách nhỏ gọn và khép kín để truyền một cách an toàn thông tin giữa các bên dưới dạng đối tượng JSON. Thông tin này có thể được xác minh và đáng tin cậy vì nó có chứa chữ ký số. JWTs có thể được ký bằng một thuật toán bí mật (với thuật toán HMAC) hoặc một public / private key sử dụng mã hoá RSA.



Hình 2. 6. Xử lý authentication với JWT trong NodeJS

Xử lý đăng nhập với JWT thường bao gồm các bước sau:

* Người dùng nhập thông tin đăng nhập (tên đăng nhập và mật khẩu) vào form đăng nhập của ứng dụng web.
* Ứng dụng web gửi thông tin đăng nhập của người dùng đến server để xác thực.
* Server kiểm tra thông tin đăng nhập của người dùng. Nếu thông tin đăng nhập chính xác, server tạo ra một JWT chứa thông tin về người dùng và trả về cho ứng dụng web.
* Ứng dụng web nhận JWT và lưu trữ nó ở phía client-side, thông thường là trong cookie hoặc local storage.
* Mỗi lần người dùng truy cập các trang được bảo vệ, ứng dụng web gửi JWT lưu trữ ở phía client-side đến server để xác thực. Server kiểm tra JWT và nếu hợp lệ, cho phép người dùng truy cập các trang được bảo vệ.

Ngoài việc xác thực người dùng, JWT cũng có thể được sử dụng để cấp quyền truy cập cho người dùng, cho phép họ truy cập vào các tài nguyên cụ thể trên server. Điều này giúp giảm thiểu việc lưu trữ thông tin đăng nhập trên server và giảm chi phí truy vấn cơ sở dữ liệu.

Để xử lý đăng nhập với JWT, có thể sử dụng các thư viện và công cụ hỗ trợ như JSON Web Token (JWT) của Node.js hoặc Django REST Framework của Python. Các thư viện này cho phép tạo, xác thực và quản lý các JWT, giúp giảm thiểu thời gian và công sức lập trình viên cần bỏ ra để xử lý đăng nhập và ủy quyền trong ứng dụng của mình.

### MySQL:

*MySQL*là hệ quản trị cơ sở dữ liệu sử dụng mã nguồn mở phổ biến nhất thế giới và được ưa chuộng trong quá trình phát triển web, ứng dụng... Với tốc độ và tính bảo mật cao, MySQL rất thích hợp cho các ứng dụng có CSDL trên Internet.

Ưu điểm của MySQL:

* Dễ sử dụng: MySQL là hệ quản trị cơ sở dữ liệu tốc độ cao, ổn định, dễ sử dụng và có thể hoạt động trên nhiều hệ điều hành cung cấp một hệ thống lớn các hàm tiện ích rất mạnh.
* Đa tính năng: MySQL hỗ trợ rất nhiều chức năng được mong chờ từ một hệ quản trị cơ sở dữ liệu quan hệ cả trực tiếp lẫn gián tiếp.
* Khả năng mở rộng và mạnh mẽ: MySQL có thể xử lý rất nhiều dữ liệu và hơn thế nữa nó có thể được mở rộng nếu cần thiết.
* Nhanh chóng: Việc đưa ra một số tiêu chuẩn cho phép MySQL để làm việc rất hiệu quả và tiết kiệm chi phí, do đó nó làm tăng tốc độ thực thi.
* Độ bảo mật cao:  MySQL rất thích hợp cho các ứng dụng có truy cập CSDL trên Internet khi sở hữu nhiều tính năng bảo mật, ngay cả ở cấp cao.

### XamPP:

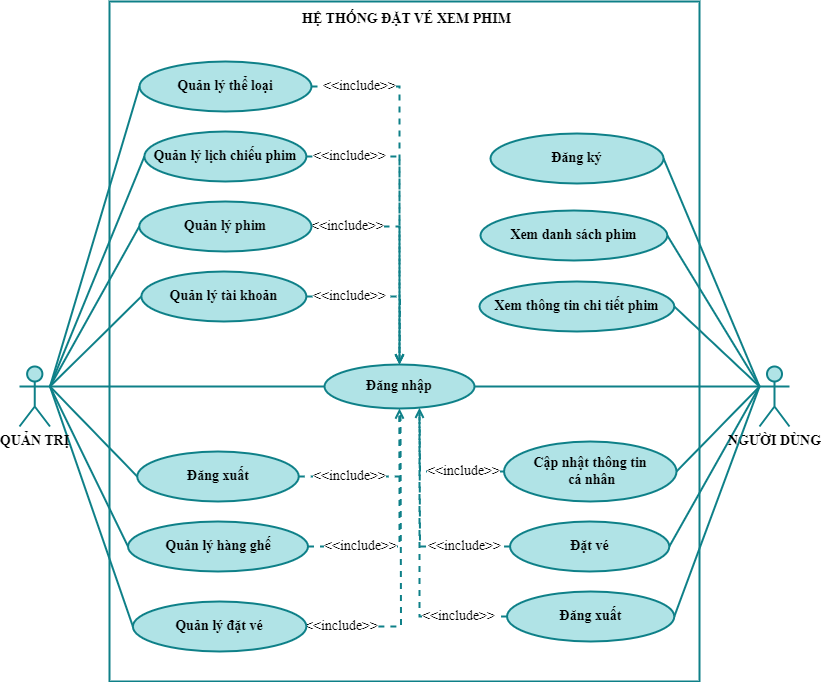
*XAMPP* là chương trình tạo web server được ứng dụng trên các hệ điều hành Linux, MacOS, Windows, Cross-platform, Solaris. XAMPP hoạt động dựa trên sự tích hợp của 5 phần mềm chính là Cross-Platform(X), Apache(A), MariaDB(M), PHP(P) và Perl(P):

* X: Viết tắt của hệ điều hành mà nó hoạt động với: Linux, Windows và Mac OS X.
* Apache: Web Server mã nguồn mở Apache là máy chủ được sử dụng rộng rãi nhất trên toàn thế giới để phân phối nội dung Web. Ứng dụng được cung cấp dưới dạng phần mềm miễn phí bởi Apache Software Foundaton.
* MySQL/MariaDB: Trong MySQL, XAMPP chứa một trong những hệ quản trị cơ sở dữ liệu quan hệ phổ biến nhất trên thế giới. Kết hợp với Web Server Apache và ngôn ngữ lập trình PHP, MySQL cung cấp khả năng lưu trữ dữ liệu cho các dịch vụ Web.
* PHP: Ngôn ngữ lập trình phía máy chủ PHP cho phép người dùng tạo các trang Web hoặc ứng dụng động.
* Perl: Ngôn ngữ kịch bản Perl được sử dụng trong hệ quản trị hệ thống, phát triển Web và lập trình mạng.

# CHƯƠNG 3. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

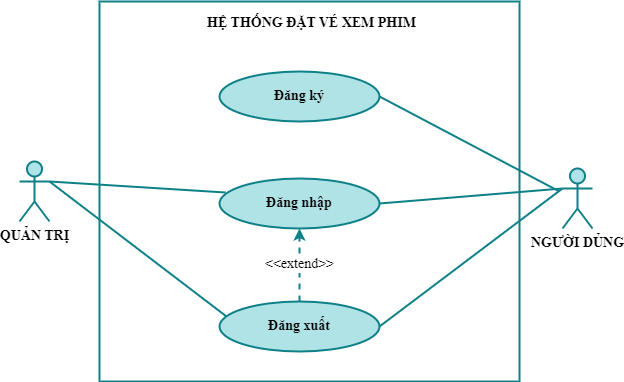
## 3.1. Use Case của hệ thống:

### 3.1.1. Use Case tổng quát:



Hình 3. 1. Use Case tổng quát

### 3.1.2. Use Case hệ thống:



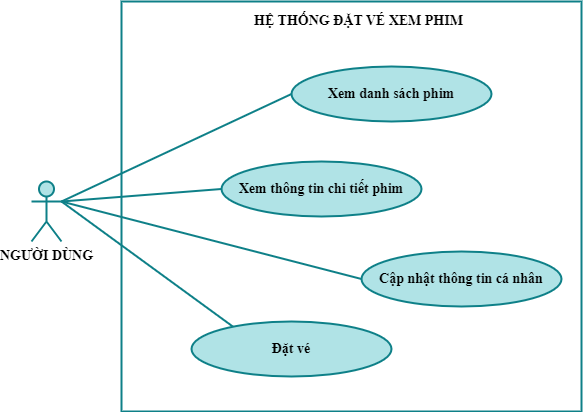
Hình 3. 2. Use Case hệ thống

|  |  |
| --- | --- |
| Use Case | Đăng ký |
| Tác nhân | Người dùng |
| Mô tả ngắn gọn | Người dùng truy cập website có thể đăng ký thông tin ứng viên hay nhà tuyển dụng |
| Tiền điều kiện | Khi người dùng truy cập và chọn đăng ký |
| Chuỗi sự kiện chính | 1. Người dùng chọn đăng ký 2. Hệ thống hiển thị trang đăng ký 3. Người dùng nhập các trường thông tin và tài khoản – mật khẩu, chọn đăng ký 4. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và quay lại bước 3 5. Hệ thống thông báo đăng ký tài khoản thành công, và cập nhật tài khoản vào cơ sở dữ liệu 6. Hiển thị giao diện trang chủ với tài khoản đã đăng ký |
| Ngoại lệ | 3. Người dùng chọn Hủy  4. Hệ thống hủy thao tác, và trả về trang chủ |
| Điều kiện đầu ra | Lưu tài khoản người dùng đã đăng ký vào cơ sở dữ liệu |

|  |  |
| --- | --- |
| Use Case | Đăng nhập |
| Tác nhân | Quản trị và người dùng |
| Mô tả ngắn gọn | Người dùng đăng nhập vào hệ thống, truy cập hệ thống với đúng quyền của tài khoản |
| Tiền điều kiện | Use Case sử dụng khi người dùng muốn đăng nhập vào hệ thống |
| Chuỗi sự kiện chính | 1. Người dùng chọn đăng nhập 2. Hệ thống yêu cầu nhập tài khoản và mật khẩu 3. Người dùng nhập tài khoản và mật khẩu 4. Hệ thống kiểm tra tên và mật khẩu vừa nhập: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và quay lại bước 3 5. Lấy quyền truy cập tài khoản 6. Hiển thị giao diện tương ứng với quyền của tài khoản |
| Ngoại lệ | 3. Người dùng chọn Hủy  4. Hệ thống hủy thao tác, và trả về trang chủ |
| Điều kiện đầu ra | Người dùng truy cập hệ thống với đúng quyền của tài khoản đã đăng nhập |

|  |  |
| --- | --- |
| Use Case | Đăng xuất |
| Tác nhân | Quản trị và người dùng |
| Mô tả ngắn gọn | Người dùng muốn đăng xuất khỏi hệ thống |
| Tiền điều kiện | Use Case sử dụng khi người dùng muốn đăng xuất hệ thống |
| Chuỗi sự kiện chính | 1. Người dùng chọn đăng xuất 2. Hệ thống hủy lưu tài khoản 3. Hệ thống hiển thị giao diện trang đăng nhập |
| Ngoại lệ | K Không có |
| Điều kiện đầu ra | Hủy lưu tài khoản người dùng, hiển thị trang đăng nhập |

### 3.1.3. Use Case của tác nhân NGƯỜI DÙNG:



Hình 3. 3. Use Case tác nhân NGƯỜI DÙNG

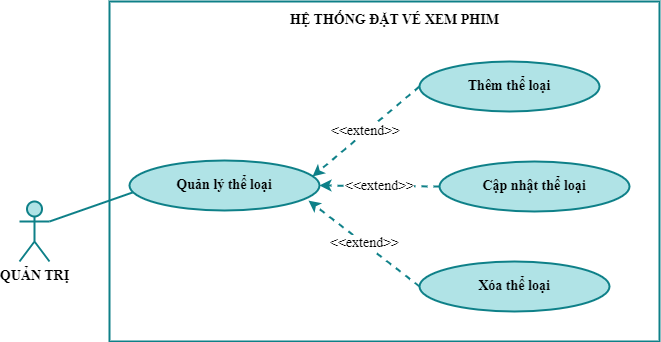
|  |  |
| --- | --- |
| Use Case | Xem danh sách phim |
| Tác nhân | Người dùng |
| Mô tả ngắn gọn | Người dùng muốn xem danh sách các bộ phim sắp chiếu |
| Tiền điều kiện | Không có |
| Chuỗi sự kiện chính | * 1. Người dùng truy cập trang chủ   2. Hệ thống lấy danh sách phim sắp chiếu   3. Hệ thống hiển thị danh sách phim |
| Ngoại lệ | Không có |
| Điều kiện đầu ra | Danh sách phim người dùng muốn xem hiển thị trên màn hình |

|  |  |
| --- | --- |
| Use Case | Xem thông tin chi tiết phim |
| Tác nhân | Người dùng |
| Mô tả ngắn gọn | Người dùng muốn xem chi tiết công việc muốn ứng tuyển |
| Tiền điều kiện | Người dùng truy cập trang web |
| Chuỗi sự kiện chính | 1. Người dùng chọn phim muốn xem chi tiết 2. Chọn xem chi tiết 3. Hệ thống lấy chi tiết phim theo ID 4. Hệ thống hiển thị trang chi tiết phim |
| Ngoại lệ | Không có |
| Điều kiện đầu ra | Chi tiết phim người dùng muốn xem hiển thị trên màn hình |

|  |  |
| --- | --- |
| Use Case | Cập nhật thông tin cá nhân |
| Tác nhân | Người dùng |
| Mô tả ngắn gọn | Người dùng thực hiện cập nhật thông tin cá nhân |
| Tiền điều kiện | Người dùng đăng nhập thành công, với đúng tài khoản |
| Chuỗi sự kiện chính | 1. Người dùng đăng nhập hệ thống, chọn cập nhật thông tin 2. Hệ thống kiểm tra tài khoản và quyền, nếu đúng thì lấy thông tin tài khoản theo ID 3. Người dùng nhập các trường dữ liệu, nhấn nút Cập nhật 4. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và quay lại bước 5. 5. Cập nhật thông tin cá nhân trong cơ sở dữ liệu 6. Hệ thống thông báo cập nhật thông tin thành công |
| Ngoại lệ | 3. Người dùng chọn Hủy  4. Hệ thống hủy thao tác, và trả về trang thông tin cá nhân |
| Điều kiện đầu ra | Các thông tin cá nhân được cập nhật vào cơ sở dữ liệu |

|  |  |
| --- | --- |
| Use Case | Đặt vé |
| Tác nhân | Người dùng |
| Mô tả ngắn gọn | Người dùng thực hiện chọn và đặt vé xem phim |
| Tiền điều kiện | UseCase sử dụng khi đăng nhập thành công hệ thống |
| Chuỗi sự kiện chính | 1. Người dùng chọn phim muốn đặt, chọn đặt vé 2. Hệ thống hiển thị form đặt vé 3. Người dùng chọn các lựa chọn với phim muốn xem và chọn đặt vé 4. Hệ thống hiển thị các lựa chọn thanh toán 5. Người dùng lựa chọn phương thức và nhập các thông tin thẻ/tài khoản, chọn thanh toán 6. Hệ thống kiểm tra các trường dữ liệu: Nếu thành công thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và quay lại bước 3. 7. Hệ thống tạo thông tin đặt vé 8. Hệ thống thông báo đặt vé thành công |
| Ngoại lệ | 3. Khách hàng chọn Hủy  4. Hệ thống hủy thao tác, và trả về trang trước đó |
| Điều kiện đầu ra | Cập nhật thông tin đặt vé của người dùng vào cơ sở dữ liệu |

### 3.1.4. Use Case Quản lý thể loại:



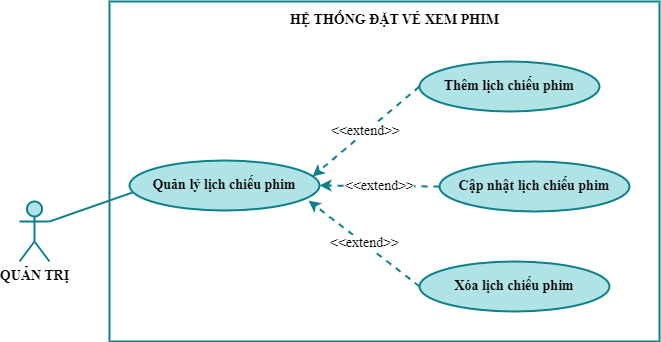
Hình 3. 4. Use Case Quản lý thể loại

|  |  |
| --- | --- |
| Use Case | Thêm thể loại |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Quản trị viên thực hiện thêm các thể loại, giúp phân loại các phim dễ dàng |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý thể loại 2. Hệ thống kiểm tra tài khoản và quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý thể loại 3. Quản lý chọn nút Thêm thể loại 4. Hệ thống hiển thị form thêm thể loại 5. Quản lý nhập các trường dữ liệu, nhấn nút Lưu 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và quay lại bước 5. 7. Hệ thống thông báo thêm thể loại thành công, và lưu thông tin thể loại trong cở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý thể loại |
| Điều kiện đầu ra | Các thông tin thể loại được lưu vào cơ sở dữ liệu |

|  |  |
| --- | --- |
| Use Case | Cập nhật thể loại |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Quản trị viên thực hiện sửa thông tin các thể loại |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý thể loại 2. Hệ thống kiểm tra tài khoản và quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý thể loại 3. Quản lý chọn thể loại muốn sửa, chọn sửa 4. Hệ thống lấy dữ liệu thể loại theo ID, và hiển thị form sửa thể loại 5. Quản lý nhập các trường dữ liệu, nhấn nút Cập nhật 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và quay lại bước 5. 7. Hệ thống thông báo sửa thể loại thành công, và cập nhật thông tin thể loại trong cơ sở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý thể loại |
| Điều kiện đầu ra | Các thông tin thể loại được cập nhật vào cơ sở dữ liệu |

|  |  |
| --- | --- |
| Use Case | Xóa thể loại |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Người quản lý thực hiện xóa các thể loại không còn quản lý |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý thể loại 2. Hệ thống kiểm tra tài khoản và quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý thể loại 3. Quản lý chọn thể loại muốn xóa, chọn xóa 4. Hệ thống hiển thị modal trước khi xóa 5. Quản lý chọn Xóa 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và hủy thao tác. 7. Hệ thống thông báo xóa thể loại thành công, và cập nhật trạng thái xóa thể loại trong cơ sở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý thể loại |
| Điều kiện đầu ra | Các thông tin thể loại được lưu vào cơ sở dữ liệu |

### 3.1.5. Use Case Quản lý lịch chiếu phim:



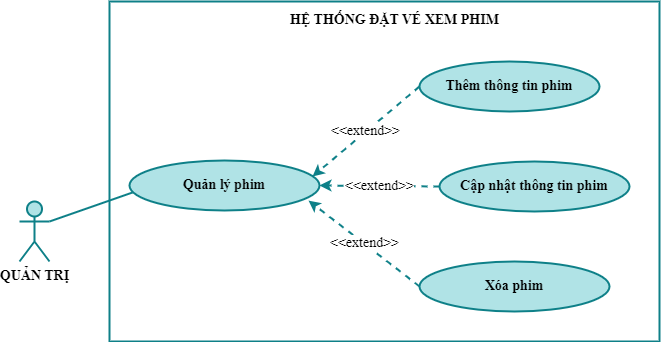
Hình 3. 5. Use Case Quản lý lịch chiếu phim

|  |  |
| --- | --- |
| Use Case | Thêm lịch chiếu phim |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Quản trị viên thực hiện thêm các lịch chiếu phim |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý lịch chiếu phim 2. Hệ thống kiểm tra tài khoản và quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý lịch chiếu phim 3. Quản lý chọn nút Thêm lịch chiếu phim 4. Hệ thống hiển thị form thêm lịch chiếu phim 5. Quản lý nhập các trường dữ liệu, nhấn nút Lưu 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và quay lại bước 5. 7. Hệ thống thông báo thêm lịch chiếu phim thành công, và lưu thông tin lịch chiếu phim trong cở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý lịch chiếu phim |
| Điều kiện đầu ra | Các thông tin lịch chiếu phim được lưu vào cơ sở dữ liệu |

|  |  |
| --- | --- |
| Use Case | Cập nhật lịch chiếu phim |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Quản trị viên thực hiện sửa thông tin các lịch chiếu phim |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý lịch chiếu phim 2. Hệ thống kiểm tra tài khoản và quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý lịch chiếu phim 3. Quản lý chọn lịch chiếu phim muốn sửa, chọn sửa 4. Hệ thống lấy dữ liệu lịch chiếu phim theo ID, và hiển thị form sửa lịch chiếu phim 5. Quản lý nhập các trường dữ liệu, nhấn nút Cập nhật 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và quay lại bước 5. 7. Hệ thống thông báo sửa lịch chiếu phim thành công, và cập nhật thông tin lịch chiếu phim trong cơ sở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý lịch chiếu phim |
| Điều kiện đầu ra | Các thông tin lịch chiếu phim được cập nhật vào cơ sở dữ liệu |

|  |  |
| --- | --- |
| Use Case | Xóa lịch chiếu phim |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Người quản lý thực hiện xóa các lịch chiếu phim không còn quản lý |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý lịch chiếu phim 2. Hệ thống kiểm tra tài khoản và quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý lịch chiếu phim 3. Quản lý chọn lịch chiếu phim muốn xóa, chọn xóa 4. Hệ thống hiển thị modal trước khi xóa 5. Quản lý chọn Xóa 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và hủy thao tác. 7. Hệ thống thông báo xóa lịch chiếu phim thành công, và cập nhật trạng thái xóa lịch chiếu phim trong cơ sở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý lịch chiếu phim |
| Điều kiện đầu ra | Các thông tin lịch chiếu phim được lưu vào cơ sở dữ liệu |

### 3.1.6. Use Case Quản lý phim:



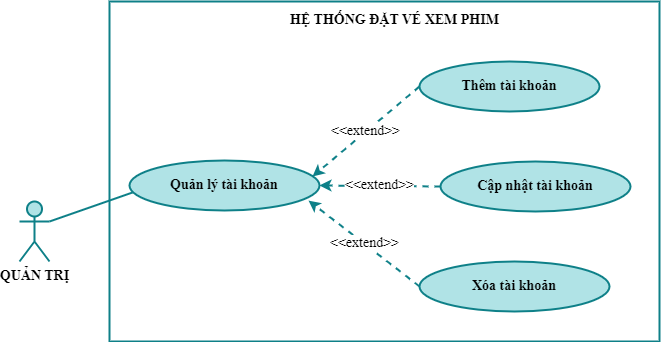
Hình 3. 6. Use Case Quản lý phim

|  |  |
| --- | --- |
| Use Case | Thêm phim |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Quản trị viên thực hiện thêm các phim |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý phim 2. Hệ thống kiểm tra tài khoản và quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý phim 3. Quản lý chọn nút Thêm phim 4. Hệ thống hiển thị form thêm phim 5. Quản lý nhập các trường dữ liệu và các lựa chọn phù hợp với phim mới tạo, nhấn nút Lưu 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và quay lại bước 5. 7. Hệ thống thông báo thêm phim thành công, và lưu thông tin phim trong cơ sở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý phim |
| Điều kiện đầu ra | Các thông tin phim được lưu vào cơ sở dữ liệu |

|  |  |
| --- | --- |
| Use Case | Cập nhật phim |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Quản trị viên thực hiện sửa thông tin các phim |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý phim 2. Hệ thống kiểm tra tài khoản và quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý phim 3. Quản lý chọn phim muốn sửa, chọn sửa 4. Hệ thống lấy dữ liệu phim theo ID, và hiển thị form sửa phim 5. Quản lý nhập các trường dữ liệu và các lựa chọn, nhấn nút Cập nhật 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và quay lại bước 5. 7. Hệ thống thông báo sửa phim thành công, và cập nhật thông tin phim trong cơ sở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý phim |
| Điều kiện đầu ra | Các thông tin phim được cập nhật vào cơ sở dữ liệu |

|  |  |
| --- | --- |
| Use Case | Xóa phim |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Người quản lý thực hiện xóa các phim không còn hoạt động hoặc hủy chiếu |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý phim 2. Hệ thống kiểm tra tài khoản và quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý phim 3. Quản lý chọn phim muốn xóa, chọn xóa 4. Hệ thống hiển thị modal trước khi xóa 5. Quản lý chọn Xóa 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và hủy thao tác. 7. Hệ thống thông báo xóa phim thành công, và cập nhật trạng thái xóa phim trong cơ sở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý phim |
| Điều kiện đầu ra | Các thông tin phim được lưu vào cơ sở dữ liệu |

### 3.1.7. Use Case Quản lý tài khoản:



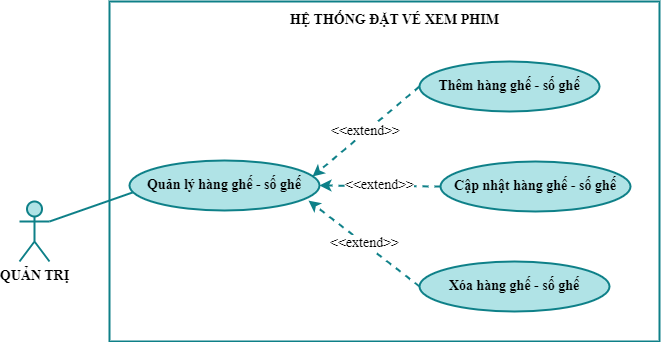
Hình 3. 7. Use Case Quản lý tài khoản

|  |  |
| --- | --- |
| Use Case | Thêm tài khoản |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Quản trị viên thực hiện thêm các tài khoản |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý tài khoản 2. Hệ thống kiểm tra tài khoản và quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý tài khoản 3. Quản lý chọn nút Thêm tài khoản 4. Hệ thống hiển thị form thêm tài khoản 5. Quản lý nhập các trường dữ liệu, nhấn nút Lưu 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và quay lại bước 5. 7. Hệ thống thông báo thêm tài khoản thành công, và lưu thông tin tài khoản trong cơ sở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý tài khoản |
| Điều kiện đầu ra | Các thông tin tài khoản được lưu vào cơ sở dữ liệu |

|  |  |
| --- | --- |
| Use Case | Cập nhật tài khoản |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Quản trị viên thực hiện sửa thông tin các tài khoản |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý tài khoản 2. Hệ thống kiểm tra tài khoản và quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý tài khoản 3. Quản lý chọn tài khoản muốn sửa, chọn sửa 4. Hệ thống lấy dữ liệu tài khoản theo ID, và hiển thị form sửa tài khoản 5. Quản lý nhập các trường dữ liệu, nhấn nút Cập nhật 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và quay lại bước 5. 7. Hệ thống thông báo sửa tài khoản thành công, và cập nhật thông tin tài khoản trong cơ sở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý tài khoản |
| Điều kiện đầu ra | Các thông tin tài khoản được cập nhật vào cơ sở dữ liệu |

|  |  |
| --- | --- |
| Use Case | Xóa tài khoản |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Người quản lý thực hiện xóa các tài khoản không còn tồn tại |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý tài khoản 2. Hệ thống kiểm tra tài khoản và quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý tài khoản 3. Quản lý chọn tài khoản muốn xóa, chọn xóa 4. Hệ thống hiển thị modal trước khi xóa 5. Quản lý chọn Xóa 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và hủy thao tác. 7. Hệ thống thông báo xóa tài khoản thành công, và cập nhật trạng thái xóa tài khoản trong cơ sở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý tài khoản |
| Điều kiện đầu ra | Các thông tin tài khoản được lưu vào cơ sở dữ liệu |

### 3.1.8. Use Case Quản lý hàng ghế - số ghế:



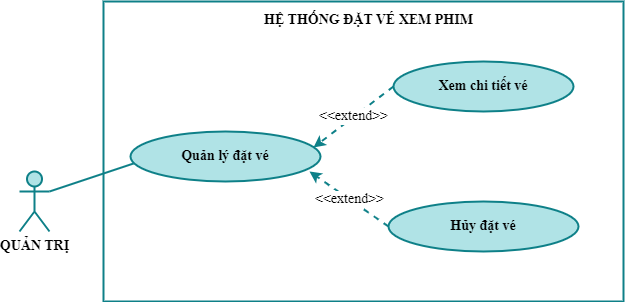
Hình 3. 8. Use Case Quản lý hàng ghế - số ghế

|  |  |
| --- | --- |
| Use Case | Thêm hàng ghế |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Quản trị viên thực hiện thêm các hàng ghế |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý hàng ghế 2. Hệ thống kiểm tra hàng ghế và quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý hàng ghế 3. Quản lý chọn nút Thêm hàng ghế 4. Hệ thống hiển thị form thêm hàng ghế 5. Quản lý nhập các thông tin hàng ghế, và bổ sung danh sách số ghế trong hàng tương ứng, nhấn nút Lưu 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và quay lại bước 5. 7. Hệ thống thông báo thêm hàng ghế thành công, và lưu thông tin hàng ghế trong cơ sở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý hàng ghế |
| Điều kiện đầu ra | Các thông tin hàng ghế được lưu vào cơ sở dữ liệu |

|  |  |
| --- | --- |
| Use Case | Cập nhật hàng ghế |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Quản trị viên thực hiện sửa thông tin các hàng ghế |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý hàng ghế 2. Hệ thống kiểm tra quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý hàng ghế 3. Quản lý chọn hàng ghế muốn sửa, chọn sửa 4. Hệ thống lấy dữ liệu hàng ghế theo ID, và hiển thị form sửa hàng ghế 5. Quản lý sửa các thông tin hàng ghế, và bổ sung danh sách số ghế trong hàng tương ứng, nhấn nút Cập nhật 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và quay lại bước 5. 7. Hệ thống thông báo sửa hàng ghế thành công, và cập nhật thông tin hàng ghế trong cơ sở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý hàng ghế |
| Điều kiện đầu ra | Các thông tin hàng ghế được cập nhật vào cơ sở dữ liệu |

|  |  |
| --- | --- |
| Use Case | Xóa hàng ghế |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Người quản lý thực hiện xóa các hàng ghế không còn tồn tại |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý hàng ghế 2. Hệ thống kiểm tra hàng ghế và quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý hàng ghế 3. Quản lý chọn hàng ghế muốn xóa, chọn xóa 4. Hệ thống hiển thị modal trước khi xóa 5. Quản lý chọn Xóa 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và hủy thao tác. 7. Hệ thống thông báo xóa hàng ghế thành công, và cập nhật trạng thái xóa hàng ghế trong cơ sở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý hàng ghế |
| Điều kiện đầu ra | Các thông tin hàng ghế được lưu vào cơ sở dữ liệu |

### 3.1.9. Use Case Quản lý đặt vé:



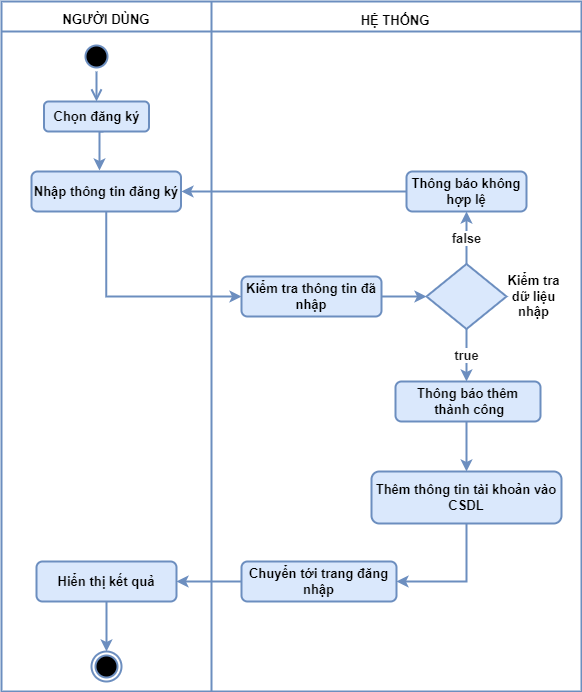
Hình 3. 9. Use Case Quản lý đặt vé

|  |  |
| --- | --- |
| Use Case | Xem chi tiết vé |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Quản trị viên muốn xem chi tiết vé của người dùng |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý chọn vé muốn xem chi tiết 2. Chọn xem chi tiết 3. Hệ thống lấy chi tiết vé theo ID 4. Hệ thống hiển thị form chi tiết thông tin vé |
| Ngoại lệ | Không có |
| Điều kiện đầu ra | Chi tiết vé của người dùng hiển thị trên màn hình quản trị |

|  |  |
| --- | --- |
| Use Case | Hủy đặt vé |
| Tác nhân | Quản trị viên |
| Mô tả ngắn gọn | Người quản lý thực hiện hủy vé |
| Tiền điều kiện | Người dùng đăng nhập thành công, với quyền quản trị |
| Chuỗi sự kiện chính | 1. Quản lý đăng nhập hệ thống, chọn quản lý đặt vé 2. Hệ thống kiểm tra quyền, nếu đúng thì truy cập hệ thống, chuyển hướng tới trang quản lý đặt vé 3. Quản lý chọn vé muốn hủy, chọn hủy 4. Hệ thống hiển thị modal trước khi hủy 5. Quản lý chọn hủy 6. Hệ thống kiểm tra dữ liệu: Nếu thỏa mãn thực hiện bước tiếp theo. Ngược lại, thông báo lỗi và hủy thao tác. 7. Hệ thống thông báo hủy đặt vé thành công, và cập nhật trạng thái hủy đặt vé trong cơ sở dữ liệu |
| Ngoại lệ | 5. Quản lý chọn Hủy  6. Hệ thống hủy thao tác, và trả về trang quản lý đặt vé |
| Điều kiện đầu ra | Các thông tin đặt vé được lưu vào cơ sở dữ liệu |

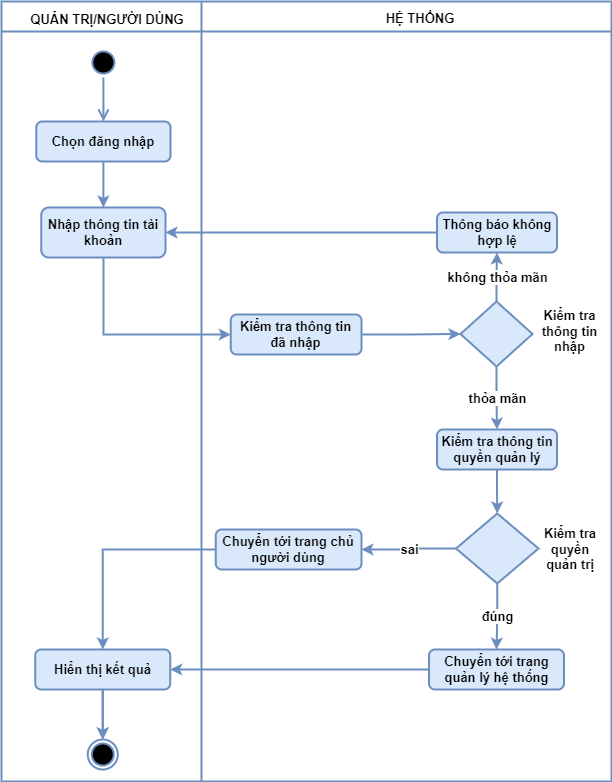
## 3.2. Biểu đồ hoạt động:

### 3.2.1. Biểu đồ hoạt động Đăng ký:



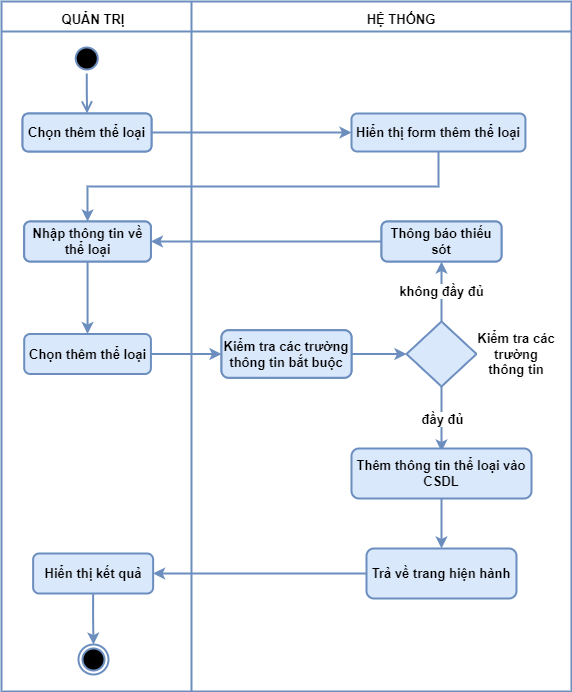
Hình 3. 10 Biểu đồ hoạt động Đăng ký

### 3.2.2. Biểu đồ hoạt động Đăng nhập:



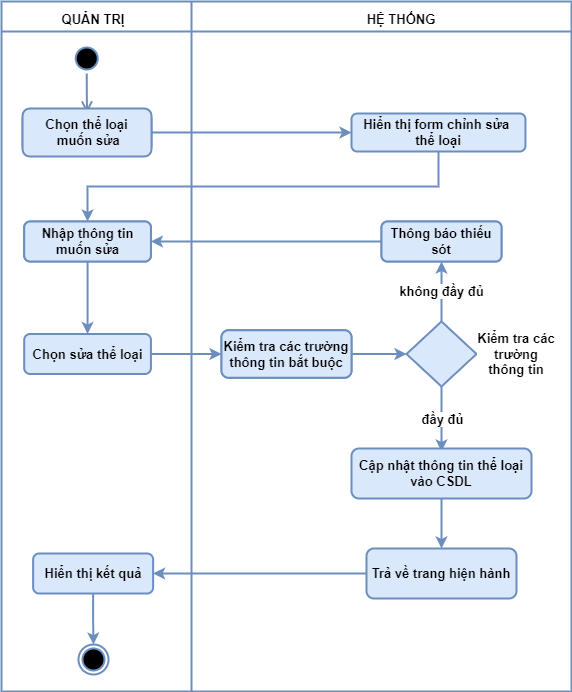
Hình 3. 11 Biểu đồ hoạt động Đăng nhập

### 3.2.3. Biểu đồ hoạt động Thêm thể loại:



Hình 3. 12 Biểu đồ hoạt động Thêm thể loại

### 3.2.4. Biểu đồ hoạt động Cập nhật thể loại:



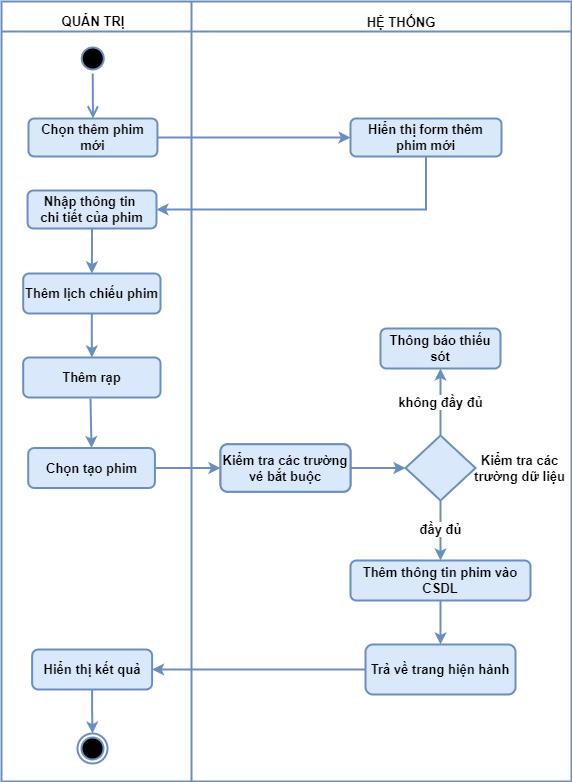
Hình 3. 13. Biểu đồ hoạt động Cập nhật thể loại

### 3.2.5. Biểu đồ hoạt động Xóa thể loại:



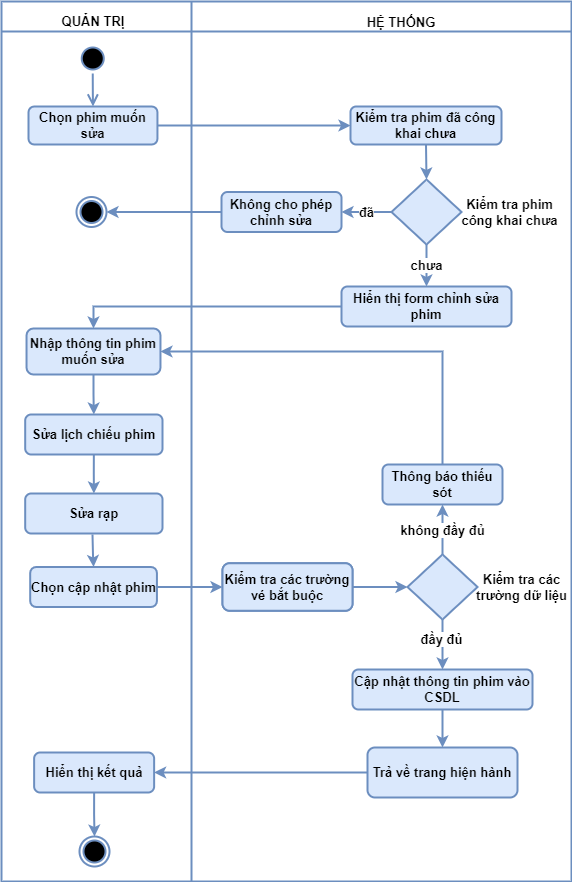
Hình 3. 14. Biểu đồ hoạt động Xóa thể loại

### 3.2.6. Biểu đồ hoạt động Thêm thông tin phim:



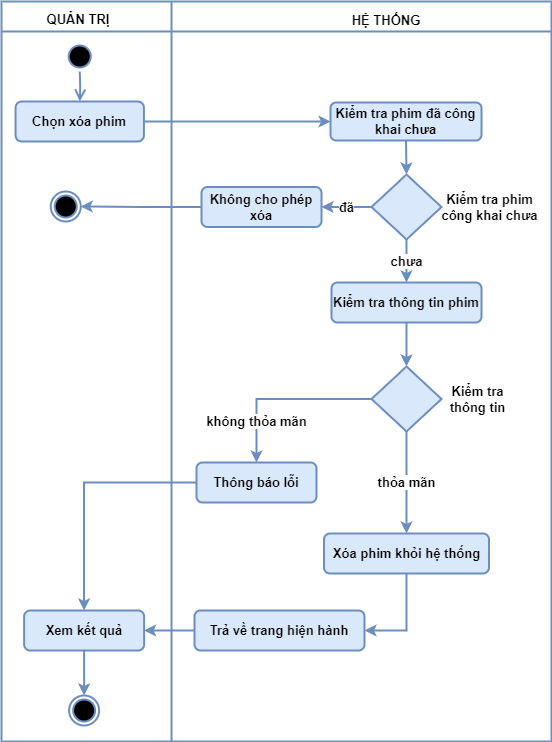
Hình 3. 15. Biểu đồ hoạt động Thêm thông tin phim

### 3.2.7. Biểu đồ hoạt động Cập nhật phim:



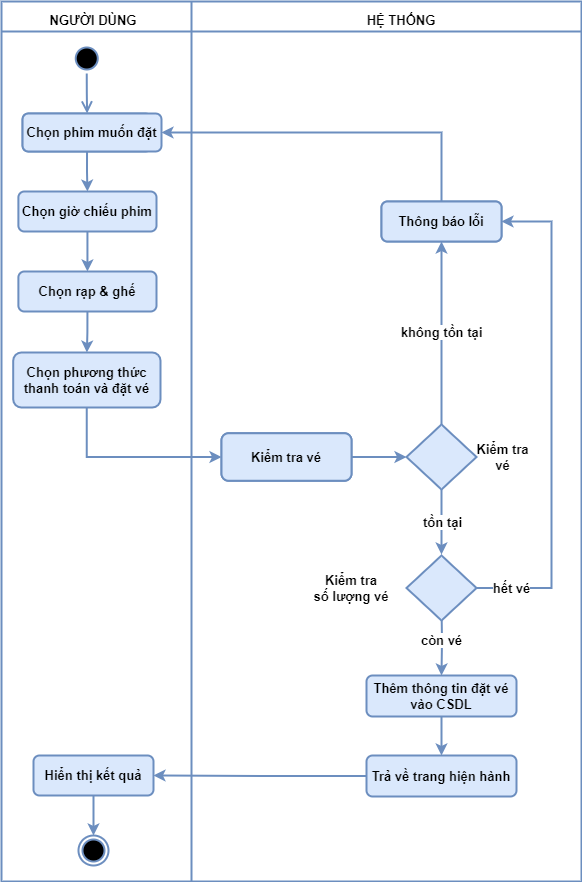
Hình 3. 16. Biểu đồ hoạt động Cập nhật phim

### 3.2.8. Biểu đồ hoạt động Xóa phim:



Hình 3. 17. Biểu đồ hoạt động Xóa phim

### 3.2.9. Biểu đồ hoạt động Đặt vé:



Hình 3. 18. Biểu đồ hoạt động Đặt vé

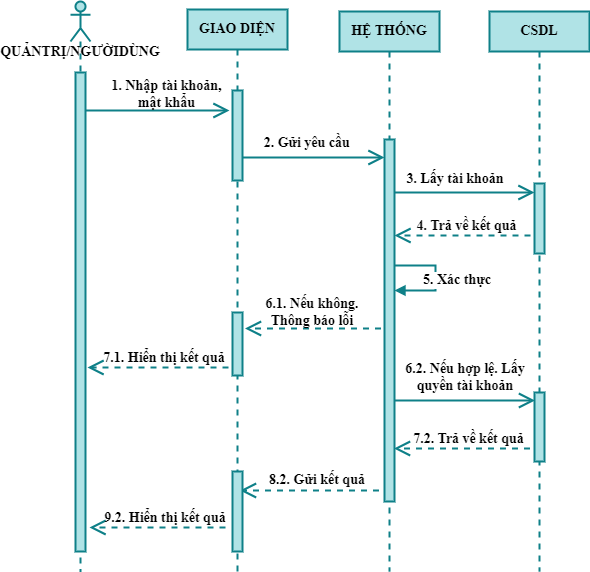
## 3.3. Biểu đồ tuần tự:

### 3.3.1. Biểu đồ tuần tự Đăng ký:



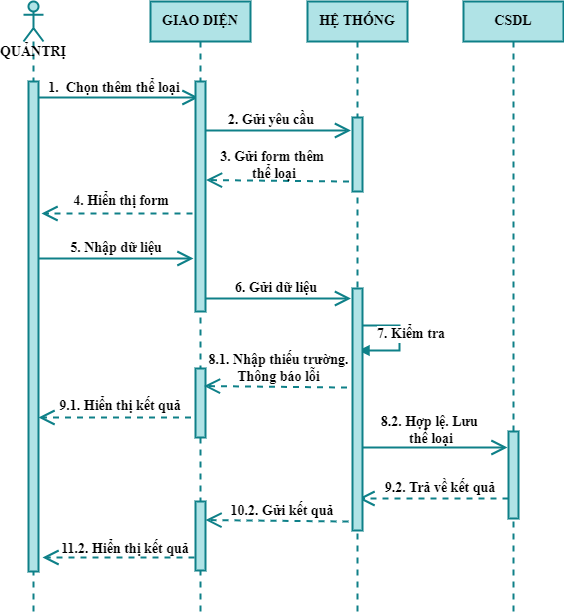
Hình 3. 19. Biểu đồ tuần tự Đăng ký

### 3.3.2. Biểu đồ tuần tự Đăng nhập:



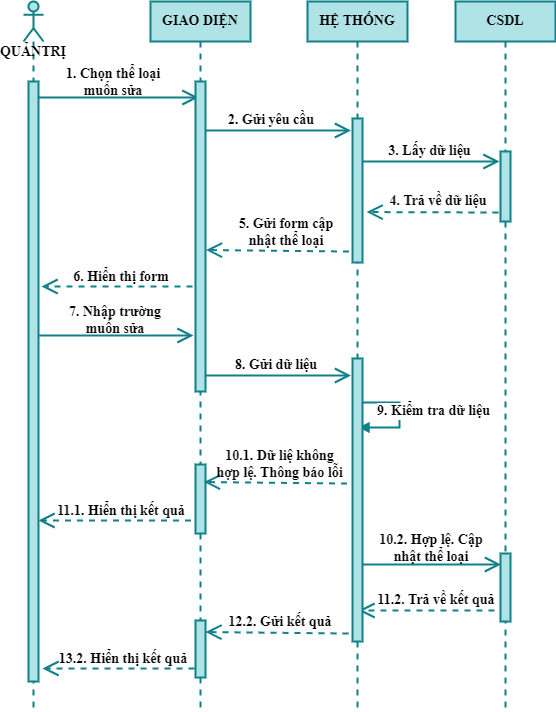
Hình 3. 20. Biểu đồ tuần tự Đăng nhập

### 3.3.3. Biểu đồ tuần tự Thêm thể loại:



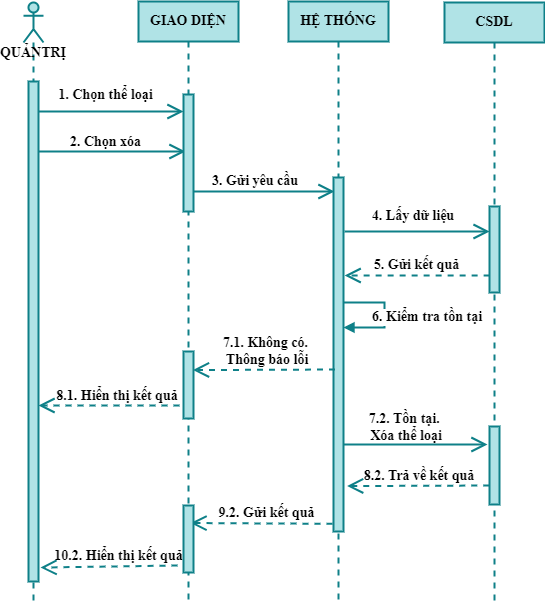
Hình 3. 21. Biểu đồ tuần tự Thêm thể loại

### 3.3.4. Biểu đồ tuần tự Cập nhật thể loại:



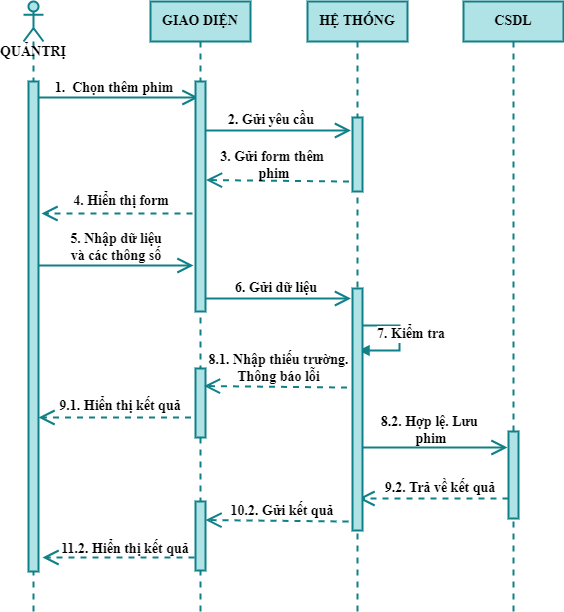
Hình 3. 22. Biểu đồ tuần tự Cập nhật thể loại

### 3.3.5. Biểu đồ tuần tự Xóa thể loại:



Hình 3. 23. Biểu đồ tuần tự Xóa thể loại

### 3.3.6. Biểu đồ tuần tự Thêm phim:



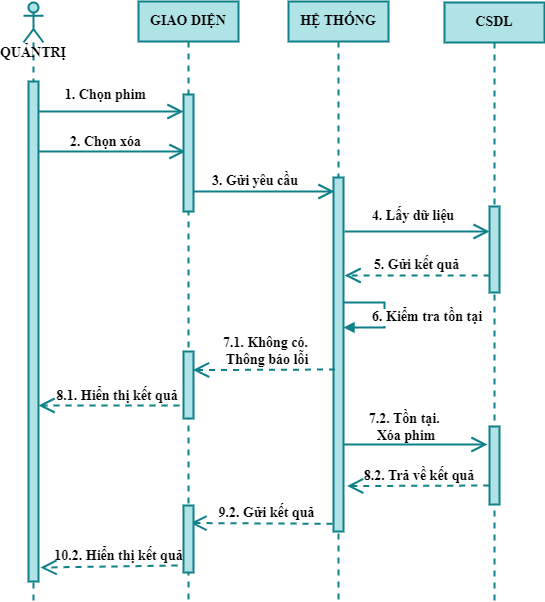
Hình 3. 24. Biểu đồ tuần tự Thêm phim

### 3.3.7. Biểu đồ tuần tự Cập nhật phim:



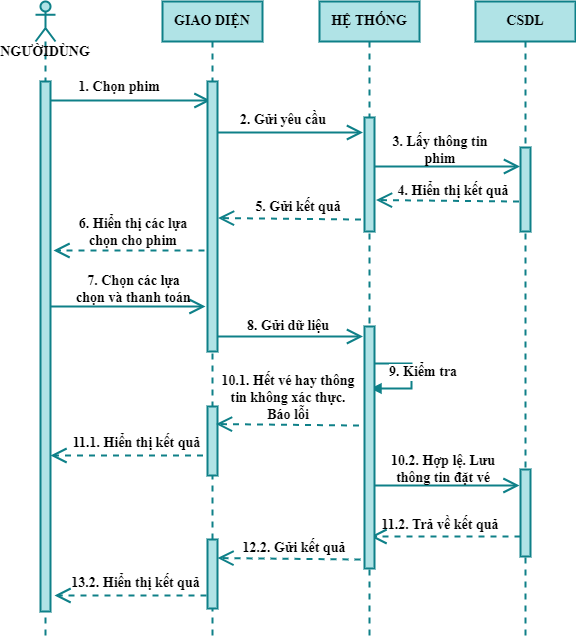
Hình 3. 25. Biểu đồ tuần tự Cập nhật phim

### 3.3.8. Biểu đồ tuần tự Xóa phim:



Hình 3. 26. Biểu đồ tuần tự Xóa phim

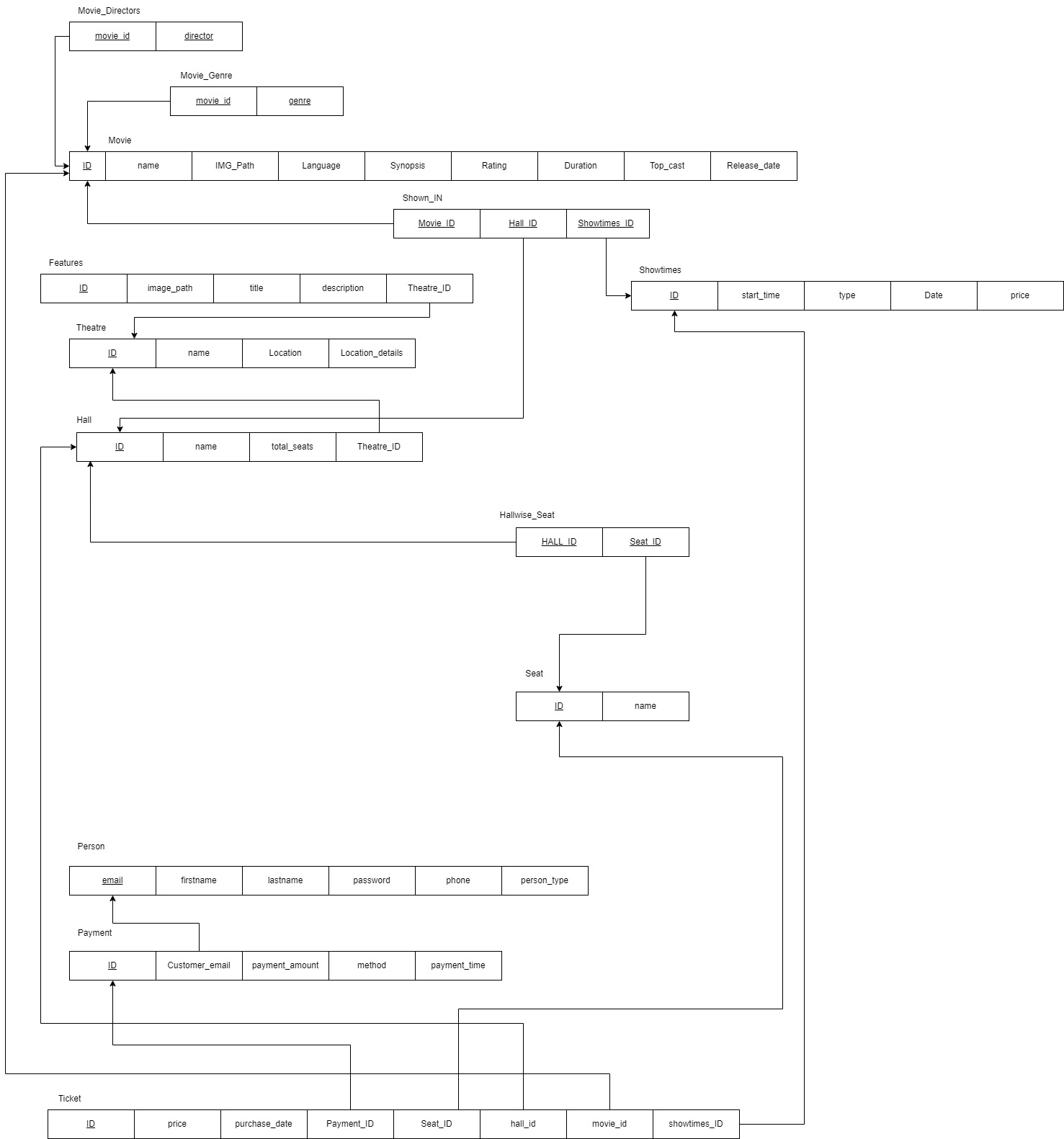
### 3.3.9. Biểu đồ tuần tự Đặt vé:



Hình 3. 27. Biểu đồ tuần tự Đặt vé

## 3.4. Thiết kế cơ sở dữ liệu:

### 3.4.1. Lược đồ quan hệ:



Hình 3. 28. Lược đồ quan hệ

### 3.4.1. Xác định các thực thể:

* Bảng movie\_directors:

Bảng 3. 1. Bảng movie\_directors

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **PK** | **FK** | **Tên trường** | **Kiểu dữ liệu** | **Diễn giải** |
| 1 | √ |  | movie\_id | bigint(20) | Mã phim |
| 2 | √ |  | director | varchar(100) | Quản lý |

* Bảng movie\_genre:

Bảng 3. 2. Bảng movie\_genre

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **PK** | **FK** | **Tên trường** | **Kiểu dữ liệu** | **Diễn giải** |
| 1 | √ |  | movie\_id | bigint(20) | Mã phim |
| 2 | √ |  | genre | varchar(100) | Thể loại |

* Bảng movie:

Bảng 3. 3. Bảng movie

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **PK** | **FK** | **Tên trường** | **Kiểu dữ liệu** | **Diễn giải** |
| 1 | √ |  | id | bigint(20) | Mã phim |
| 2 |  |  | name | varchar(100) | Thể loại |
| 3 |  |  | img\_path | varchar(255) | Ảnh |
| 4 |  |  | language | tinyint(4) | Ngôn ngữ |
| 5 |  |  | synopsis | text | Tóm tắt |
| 6 |  |  | rating | tinyint(4) | Số sao |
| 7 |  |  | duration | int(11) | Thời lượng |
| 8 |  |  | top\_cast | varchar(255) | Danh sách diễn viên hàng đầu |
| 9 |  |  | release\_date | timestamp | Ngày phát hành |

* Bảng shown\_in:

Bảng 3. 4. Bảng shown\_in

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **PK** | **FK** | **Tên trường** | **Kiểu dữ liệu** | **Diễn giải** |
| 1 | √ |  | movie\_id | bigint(20) | Mã phim |
| 2 | √ |  | hall\_id | bigint(20) | Mã sảnh |
| 3 | √ |  | showtime\_id | bigint(20) | Mã thời lượng chiếu |

* Bảng features:

Bảng 3. 5. Bảng features

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **PK** | **FK** | **Tên trường** | **Kiểu dữ liệu** | **Diễn giải** |
| 1 | √ |  | id | bigint(20) | Mã đặc trưng |
| 2 |  |  | img\_path | varchar(255) | Ảnh |
| 3 |  |  | title | varchar(100) | Tiêu đề |
| 4 |  |  | description | text | Mô tả |
| 5 |  | √ | theatre\_id | bigint(20) | Mã rạp |

* Bảng showtimes:

Bảng 3. 6. Bảng showtimes

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **PK** | **FK** | **Tên trường** | **Kiểu dữ liệu** | **Diễn giải** |
| 1 | √ |  | id | bigint(20) | Mã lịch chiếu phim |
| 2 |  |  | start\_time | timestamp | Thời gian bắt đầu |
| 3 |  |  | type | tinyint(4) | Loại |
| 4 |  |  | date | timestamp | Ngày |
| 5 |  |  | price | int(11) | Giá |

* Bảng theatre:

Bảng 3. 7. Bảng theatre

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **PK** | **FK** | **Tên trường** | **Kiểu dữ liệu** | **Diễn giải** |
| 1 | √ |  | id | bigint(20) | Mã rạp |
| 2 |  |  | name | varchar(100) | Tên |
| 3 |  |  | location | varchar(255) | Địa điểm |
| 4 |  |  | location\_detail | varchar(255) | Chi tiết địa điểm |

* Bảng hall:

Bảng 3. 8. Bảng hall

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **PK** | **FK** | **Tên trường** | **Kiểu dữ liệu** | **Diễn giải** |
| 1 | √ |  | id | bigint(20) | Mã sảnh |
| 2 |  |  | name | varchar(100) | Tên |
| 3 |  |  | total\_seat | int(11) | Tổng ghế |
| 4 |  | √ | theatre\_id | bigint(20) | Mã rạp |

* Bảng hallwise\_seat:

Bảng 3. 9. Bảng hallwise\_seat

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **PK** | **FK** | **Tên trường** | **Kiểu dữ liệu** | **Diễn giải** |
| 1 | √ |  | hall\_id | bigint(20) | Mã sảnh |
| 2 | √ |  | seat\_id | varchar(100) | Mã ghế |

* Bảng seat:

Bảng 3. 10. Bảng seat

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **PK** | **FK** | **Tên trường** | **Kiểu dữ liệu** | **Diễn giải** |
| 1 | √ |  | id | bigint(20) | Mã ghế |
| 2 |  |  | name | varchar(100) | Tên |

* Bảng person:

Bảng 3. 11. Bảng person

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **PK** | **FK** | **Tên trường** | **Kiểu dữ liệu** | **Diễn giải** |
| 1 | √ |  | id | bigint(20) | Mã tài khoản |
| 2 |  |  | email | varchar(100) | Email |
| 3 |  |  | first\_name | varchar(100) | Tên |
| 4 |  |  | last\_name | varchar(100) | Họ |
| 5 |  |  | password | varchar(100) | Mật khẩu |
| 6 |  |  | phone | varchar(12) | Số điện thoại |
| 7 |  |  | person\_type | tinyint(4) | Loại |

* Bảng payment:

Bảng 3. 12. Bảng payment

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **PK** | **FK** | **Tên trường** | **Kiểu dữ liệu** | **Diễn giải** |
| 1 | √ |  | id | bigint(20) | Mã thanh toán |
| 2 |  |  | customer\_email | varchar(100) | Email |
| 3 |  |  | payment\_amount | varchar(100) | Số tiền |
| 4 |  |  | method | tinyint(4) | Phương thức |
| 5 |  |  | payment\_time | timestamp | Thời gian |

* Bảng ticket:

Bảng 3. 13. Bảng ticket

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **PK** | **FK** | **Tên trường** | **Kiểu dữ liệu** | **Diễn giải** |
| 1 | √ |  | id | bigint(20) | Mã vé |
| 2 |  |  | price | int(11) | Giá |
| 3 |  |  | purchase\_date | timestamp | Ngày thanh toán |
| 4 |  | √ | payment\_id | bigint(20) | Mã thanh toán |
| 5 |  | √ | seat\_id | bigint(20) | Mã ghế |
| 6 |  | √ | hall\_id | bigint(20) | Mã sảnh |
| 7 |  | √ | movie\_id | bigint(20) | Mã phim |
| 8 |  | √ | showtime\_id | bigint(20) | Mã lịch chiếu phim |

# CHƯƠNG 4. GIAO DIỆN PHẦN MỀM

* 1. **Giao diện người dùng:**
     1. ***Giao diện đăng ký:***

Hình 4. 1. Giao diện đăng ký

* + 1. ***Giao diện đăng nhập:***

Hình 4. 2. Giao diện đăng nhập

* + 1. ***Giao diện xem chi tiết phim:***

Hình 4.3. Giao diện xem chi tiết phim

* + 1. ***Giao diện đặt vé (1):***

Hình 4.4. Giao diện đặt vé (1)

* + 1. ***Giao diện đặt vé (2):***

Hình 4.5. Giao diện đặt vé (2)

* + 1. ***Giao diện đặt vé (3):***

Hình 4.6. Giao diện đặt vé (3)

* 1. **Giao diện quản trị:**
     1. ***Giao diện đăng nhập:***

Hình 4.7. Giao diện đăng nhập

* + 1. ***Giao diện quản lý thể loại:***

Hình 4.8. Giao diện quản lý thể loại

* + 1. ***Giao diện quản lý phim:***

Hình 4. 9. Giao diện quản lý phim

* + 1. ***Giao diện quản lý đặt vé:***

Hình 4. 10. Giao diện quản lý đặt vé

# CHƯƠNG 5. TỔNG KẾT

* 1. **Tổng kết:**

Trong thời gian làm đồ án này đã giúp em tìm hiểu và nắm bắt được nhiều kiến thức bổ ích, đồng thời rút ra nhiều kinh nghiệm thực tế. Do hạn chế về thời gian cũng như trình độ nên chương trình không tránh khỏi nhiều sai sót. Đề tài “Phân tích thiết kế và xây dựng website đặt vé xem phim” giúp khắc phục những nhược điểm và khó khăn của việc quản lý truyền thống, nhưng cũng còn một số hạn chế.

## 5.2.  Kết quả đạt được:

* + Lưu trữ thông tin về các bộ phim trong hệ thống dài hạn, chính xác và tiết kiệm thời gian.
  + Các giao dịch tiến hành nhanh chóng.
  + Tìm kiếm thông tin nhanh chóng, dễ dàng.
  + Có thể quản lý việc đặt vé nhanh chóng.
  + Phân quyền và thực hiện đúng các chức năng tương ứng với quyền của từng tài khoản.

## 5.3. Hạn chế:

* + Một số trường thông tin chưa kiểm tra chặt chẽ.
  + Thao tác nhanh đôi khi có thể xảy ra lỗi.
  1. **Hướng phát triển:**

Phát triển, cải tiến phần mềm có thêm một số tính năng mới trong tương lai như:

* + Tăng tính bảo mật của trang web.
  + Bổ sung thêm một số chức năng nghiệp vụ.
  + ….

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| **[1]** | Trần Đình Quế - Nguyễn Mạnh Sơn, ***Giáo trình Phân tích Thiết kế hệ thống thông tin***, Học viện Công nghệ Bưu chính viễn thông, 2007 |
| **[2]** | Trịnh Thành Trung, ***Tổng quan UML***, Đại học Bách Khoa Hà Nội, <https://users.soict.hust.edu.vn/trungtt/uploads/slides/OOP_Bai13(vi).pdf> [Nguồn: Internet] |
| **[3]** | *ReactJS là gì? Tất tần tật những điều căn bản về ReactJS*, <https://stringee.com/vi/blog/post/reactJS-la-gi> (Nguồn: Internet) |
| **[4]** | ***Documents of MySQL***, <https://dev.mysql.com/doc/> [Nguồn: Internet] |
| **[5]** | ***Documents of NodeJS***, <https://nodejs.org/en/learn/getting-started/introduction-to-nodejs> [Nguồn: Internet] |