**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC BÀ RỊA –VŨNG TÀU**

**KHOA KỸ THUẬT CÔNG NGHỆ**

**---🙠**🕮**🙢---**

A blue and yellow logo

Description automatically generated

**BÁO CÁO TIỂU LUẬN MÔN HỌC**

**LẬP TRÌNH .NET**

**HỌC KỲ 3 NĂM HỌC 2023-2024**

**ĐỀ TÀI: QUẢN LÝ RẠP PHIM**

**Trình độ đào tạo : Đại học**

**Chuyên ngành : Công nghệ thông tin**

**Khóa học : 2022-2026**

**Lớp học phần : DH22CT1**

**Thành viên nhóm: Phạm Thái Bảo – DH22CT1 - 22030413**

**Nguyễn Minh Cường – DH22PM - 22030126**

**GVHD : ThS. Lê Thị Vĩnh Thanh**

**BÀ RỊA – VŨNG TÀU, NĂM 2024**

**ĐÁNH GIÁ CỦA GIẢNG VIÊN**

**……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………**

**GIẢNG VIÊN XÁC NHẬN**

# **LỜI MỞ ĐẦU**

Hiện nay, ngành công nghiệp chiếu phim đang trở thành một phần không thể thiếu trong văn hóa giải trí của Việt Nam cũng như các nước trên toàn cầu, và đây cũng là một lĩnh vực có sự phát triển rất mạnh trong khoảng thời gian dịch Covid-19 năm 2020 cho đến nay.

Sự phát triển của web bán vé xem phim online đã mở ra nhiều cơ hội cho các studio làm phim, thương hiệu, doanh nghiệp, … có cơ hội để hợp tác để chiếu phim, từ đó tạo ra doanh thu lớn cho cả 2 bên.

Có nhiều yếu tố có thể tạo nên sự khác biệt, bao gồm giao diện trang web, dịch vụ khách hàng, chất lượng phim, thương hiệu và cách tiếp cận với khán giả. Thương hiệu trong ngành web, app bán vé phim online không chỉ là một cái tên, mà còn mang đến một cái nhìn, một cảm giác riêng biệt mà khán giả có thể trải nghiệm khi sử dụng dịch vụ.

Do đó, việc xây dựng và phát triển thương hiệu trong ngành web, app bán vé phim online đóng vai trò quan trọng trong việc thu hút và duy trì người xem. Và trong bài báo cáo này, nhóm em sẽ đề xuất các chiến lược và ý tưởng về phát triển thương hiệu cho một nền tảng chiếu phim trực tuyến cụ thể, nhằm tạo ra sự khác biệt và thu hút sự quan tâm của khán giả.

# **LỜI CẢM ƠN**

Báo cáo này đã được thực hiện trong khoảng thời gian 6 tuần, và trong quá trình phát triển, có thể sẽ có những thiếu sót không tránh khỏi. Và nếu như có cơ hội được áp dụng vào thực tiễn thì chắc chắn cần rất nhiều sự hợp tác của rất nhiều người để hoàn thành dự án. Sau cùng nhóm đã cố gắng hết sức để hoàn thiện bài báo cáo chất lượng nhất có thể. Sau buổi báo cáo này, nhóm em mong muốn nhận được sự đóng góp ý kiến từ giảng viên để giúp hoàn thiện hơn nữa và mở rộng kiến thức của từng cá nhân.

**MỤC LỤC**

[**LỜI MỞ ĐẦU** 3](#_Toc174680300)

[**LỜI CẢM ƠN** 3](#_Toc174680301)

[**CHƯƠNG 1: MỤC TIÊU NGHIÊN CỨU VÀ PHẠM VI ĐỀ TÀI** 7](#_Toc174680302)

[1.1 Mục tiêu nghiên cứu 7](#_Toc174680303)

[1.2 Đối tượng và phạm vi nghiên cứu 7](#_Toc174680304)

[1.3 Tính thực tiễn của đề tài 7](#_Toc174680305)

[**CHƯƠNG 2: KHẢO SÁT HỆ THỐNG** 8](#_Toc174680306)

[2.1 Giới thiệu 8](#_Toc174680307)

[2.2 Quy trình 8](#_Toc174680308)

[1.1.1 Giới thiệu đối tượng 8](#_Toc174680309)

[2.2.2 Giới thiệu các tính năng 8](#_Toc174680310)

[2.3 Xác định yêu cầu chức năng 9](#_Toc174680311)

[2.4 Kết luận 9](#_Toc174680312)

[**CHƯƠNG 3: PHÂN TÍCH HỆ THỐNG** 10](#_Toc174680313)

[3.1 Giới thiệu hệ thống 10](#_Toc174680314)

[3.2Mô tả chi tiết 11](#_Toc174680315)

[3.3 Yêu cầu chức năng 12](#_Toc174680316)

[3.4. Sơ đồ UseCase 13](#_Toc174680317)

[3.5. Bảng đặc tả 15](#_Toc174680318)

[3.6. Sơ đồ tuần tự 40](#_Toc174680319)

[3.7. Kết luận 62](#_Toc174680320)

[**CHƯƠNG 4: CƠ SỞ DỮ LIỆU** 63](#_Toc174680321)

[1. Sơ đồ ERD Diagram 63](#_Toc174680322)

[2. Sơ đồ lớp Class Diagram 64](#_Toc174680323)

[3. Các Thực Thể Và Các Thuộc Tính 65](#_Toc174680324)

[4. Sơ đồ quan hệ 69](#_Toc174680325)

[**CHƯƠNG 5: THIẾT KẾ FORM** 72](#_Toc174680326)

[1. Form đăng nhập 72](#_Toc174680327)

[2. Form đăng ký 75](#_Toc174680328)

[3. Form quên mật khẩu 78](#_Toc174680329)

[4. Form trang chủ 80](#_Toc174680330)

[5. Form đổi mật khẩu 91](#_Toc174680331)

[6. Form xem lịch sử chiếu phim 94](#_Toc174680332)

[7. Form đặt vé 94](#_Toc174680333)

[8. Form lịch sử mua vé 100](#_Toc174680334)

[9. Form khiếu nại 111](#_Toc174680335)

[10. Form quản lý khiếu nại 112](#_Toc174680336)

[11. Form quản lý phòng chiếu 112](#_Toc174680337)

[12. Form quản lý ghế ngồi 113](#_Toc174680338)

[13. Form quản lý phim 113](#_Toc174680339)

[14. Form quản lý lịch chiếu 114](#_Toc174680340)

[15. Form quản lý nhân viên 114](#_Toc174680341)

[16. Form quản lý người dùng 115](#_Toc174680342)

[17. Form quản lý thống kê 115](#_Toc174680343)

[**CHƯƠNG 6: XỬ LÝ FORM** 116](#_Toc174680344)

[1. Xây dựng Class Hỗ Trợ (SqlHook) 116](#_Toc174680345)

[2. Class SessionManager 118](#_Toc174680346)

[3. Class MovementController 119](#_Toc174680347)

[4. Class KeyMessageFilter 120](#_Toc174680348)

[5. Class DoubleBufferedEnhancer 121](#_Toc174680349)

[6. Class FormStateManager 122](#_Toc174680350)

[7. Class CenterController 123](#_Toc174680351)

[**CHƯƠNG 7: HƯỚNG DẪN SỬ DỤNG CHƯƠNG TRÌNH** 124](#_Toc174680352)

[**CHƯƠNG 8: TỔNG KẾT & HƯỚNG PHÁT TRIỂN** 125](#_Toc174680353)

[1. Tổng kết 125](#_Toc174680354)

[2. Hướng phát triển 125](#_Toc174680355)

**DANH MỤC HÌNH ẢNH**

[Hình 1. Các tác nhân 10](#_Toc174679488)

[Hình 2. Sơ đồ UseCase 13](#_Toc174679489)

[Hình 3. Sơ đồ UseCase Nhân Viên 14](#_Toc174679490)

[Hình 4. Sơ đồ tuần tự đăng nhập 40](#_Toc174679491)

[Hình 5. Sơ đồ tuần tự chức năng đăng ký 41](#_Toc174679492)

[Hình 6. Sơ đồ tuần tự chức năng quên mật khẩu 42](#_Toc174679493)

[Hình 7. Sơ đồ tuần tự chức năng quên mật khẩu 43](#_Toc174679494)

[Hình 8. Sơ đồ tuần tự chức năng đăng xuất 44](#_Toc174679495)

[Hình 9. Sơ đồ tuần tự chức năng đặt vé 45](#_Toc174679496)

[Hình 10. Sơ đồ tuần tự chức năng gửi phản hồi khiếu nại 46](#_Toc174679497)

[Hình 11. Sơ đồ tuần tự chức năng quản lý doanh thu 47](#_Toc174679498)

[Hình 12. Sơ đồ tuần tự chức năng quản lý ghế ngồi 48](#_Toc174679499)

[Hình 13. Sơ đồ tuần tự chức năng quản lý khách hàng 49](#_Toc174679500)

[Hình 14. Sơ đồ tuần tự chức năng quản lý lịch chiếu phim 50](#_Toc174679501)

[Hình 15. Sơ đồ tuần tự chức năng quản lý lịch làm việc 51](#_Toc174679502)

[Hình 16. Sơ đồ tuần tự chức năng quản lý nhân viên 52](#_Toc174679503)

[Hình 17. Sơ đồ tuần tự chức năng quản lý phim 53](#_Toc174679504)

[Hình 18. Sơ đồ tuần tự chức năng quản lý phòng 54](#_Toc174679505)

[Hình 19. Sơ đồ tuần tự chức năng quản lý sản phẩm 55](#_Toc174679506)

[Hình 20. Sơ đồ tuần tự chức năng xem lịch chiếu 56](#_Toc174679507)

[Hình 21. Sơ đồ tuần tự chức năng xem lịch làm việc 57](#_Toc174679508)

[Hình 22. Sơ đồ tuần tự chức năng xem lịch sử thanh toán 58](#_Toc174679509)

[Hình 23. Sơ đồ tuần tự chức năng xem tình trạng ghế ngồi 59](#_Toc174679510)

[Hình 24. Sơ đồ tuần tự chức năng xem và cập nhật thông tin cá nhân 60](#_Toc174679511)

[Hình 25. Sơ đồ tuần tự chức năng xem và trả lời phản hồi khiếu nại 61](#_Toc174679512)

# **CHƯƠNG 1: MỤC TIÊU NGHIÊN CỨU VÀ PHẠM VI ĐỀ TÀI**

* 1. Mục tiêu nghiên cứu

"Xây dựng hệ thống quản lý và phân loại khách hàng mua vé xem phim tại rạp ABC" được thực hiện dựa trên các chuẩn mực quản lý bán vé xem phim của rạp ABC.

Mục tiêu chính là giúp công việc quản lý các giao dịch hằng ngày của rạp được đơn giản, nhanh chóng và chính xác hơn.

Để có thể thực hiện được mục tiêu đề ra, phải xác định được các nhiệm vụ:

* Quản lý thông tin khách hàng.
* Quản lý lịch sử giao dịch.
* Quản lý lịch trình chiếu phim.
* Quản lý nhân viên.
* Quản lý phòng, ghế và phim.
* Quản lý báo cáo thống kê.
  1. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu của đề tài là quản lý và phân loại khách hàng mua vé xem phim tại rạp ABC. Phạm vi nghiên cứu của đề tài tập trung vào việc quản lý mua vé xem phim của khách hàng và phân loại khách hàng của rạp phim ABC dựa trên cơ sở lý thuyết nghiên cứu để thiết kế mô hình và tổ chức dữ liệu.

* 1. Tính thực tiễn của đề tài

Ngày nay, các rạp chiếu phim ngày càng mở ra nhiều hơn. Các rạp muốn có một phần mềm quản lý tốt hơn, giá thành rẻ hơn và dễ sử dụng hơn nên việc tìm kiếm và phát triển các phần mềm quản lý là rất cần thiết. Hiện vẫn còn tồn tại những rạp chiếu phim chưa có phần mềm quản lý. Điều này cho thấy việc tìm hiểu và phát triển phần mềm này là rất cần thiết cho đời sống hiện nay.

# **CHƯƠNG 2: KHẢO SÁT HỆ THỐNG**

* 1. Giới thiệu

Nhu cầu giải trí của con người, đặc biệt là việc xem phim tại các rạp chiếu, ngày càng tăng cao. Điều này đòi hỏi một hệ thống quản lý rạp chiếu phim hiệu quả để đáp ứng nhu cầu của khách hàng và nâng cao hiệu suất hoạt động của rạp. Ứng dụng quản lý rạp chiếu phim ra đời nhằm cung cấp giải pháp toàn diện cho việc bán vé xem phim online, quản lý lịch chiếu, phòng chiếu, ghế ngồi, phim, nhân viên, lịch làm việc và thống kê.

* 1. Quy trình
     1. Giới thiệu đối tượng

Ứng dụng quản lý rạp chiếu phim hướng đến ba tác nhân chính:

* Quản lý: Người chịu trách nhiệm quản lý toàn bộ hoạt động của rạp chiếu phim.
* Nhân viên: Người thực hiện các nhiệm vụ hàng ngày như kiểm tra vé, hướng dẫn khách hàng, và hỗ trợ quản lý.
* Khách hàng: Người mua vé và đến xem phim tại rạp.
  + 1. Giới thiệu các tính năng

Ứng dụng quản lý rạp chiếu phim cung cấp các tính năng chính như sau:

* **Bán vé xem phim online:** Khách hàng có thể mua vé trực tuyến, chọn suất chiếu và ghế ngồi theo ý thích mà không cần đến trực tiếp rạp.
* **Quản lý lịch chiếu:** Quản lý các suất chiếu của từng bộ phim, cập nhật lịch chiếu mới và thông báo cho khách hàng về các suất chiếu sắp tới.
* **Quản lý phòng chiếu và ghế ngồi:** Hệ thống quản lý phòng chiếu, số lượng ghế ngồi và tình trạng ghế (đã đặt hay còn trống) để đảm bảo khách hàng có thể chọn lựa ghế ngồi một cách thuận tiện.
* **Quản lý phim:** Quản lý thông tin về các bộ phim đang chiếu, sắp chiếu và các thông tin liên quan như tên phim, thể loại, đạo diễn, diễn viên, thời lượng, mô tả phim và hình ảnh.
* **Quản lý nhân viên:** Quản lý thông tin nhân viên, phân công công việc và theo dõi lịch làm việc của từng nhân viên.
* **Thống kê và báo cáo:** Cung cấp các báo cáo thống kê về doanh thu bán vé, số lượng khách hàng, hiệu suất làm việc của nhân viên và các thông tin quan trọng khác để hỗ trợ quản lý rạp chiếu phim đưa ra quyết định chính xác.
  1. Xác định yêu cầu chức năng

Quản lý:

* Quản lý tất cả các hoạt động của rạp chiếu phim.
* Lên kế hoạch và tổ chức thực hiện nhiệm vụ.
* Theo dõi và đánh giá hiệu quả làm việc của nhân viên.
* Quản lý lịch chiếu phim và phòng chiếu.

Nhân viên:

* Hỗ trợ khách hàng mua vé và chọn ghế.
* Kiểm tra vé và hướng dẫn khách hàng vào phòng chiếu.
* Bảo trì và vệ sinh phòng chiếu.
* Xem lịch đi làm.

Khách hàng:

* Mua vé xem phim trực tuyến hoặc tại rạp.
* Chọn suất chiếu và ghế ngồi.
* Nhận thông tin về các suất chiếu và các bộ phim đang chiếu.
  1. Kết luận

Hệ thống quản lý rạp chiếu phim cần phải xử lý nhiều thông tin hàng ngày, do đó cần một hệ thống hiệu quả để đáp ứng nhu cầu hiện tại và tương lai của rạp. Việc sử dụng phần mềm quản lý rạp chiếu phim, được xây dựng với ngôn ngữ C#, sẽ giúp quản lý khách hàng, lịch chiếu và các hoạt động của rạp một cách trực quan và dễ sử dụng. Phần mềm này sẽ hỗ trợ nhân viên và quản lý trong việc cải thiện hiệu suất làm việc và cung cấp dịch vụ tốt hơn cho khách hàng.

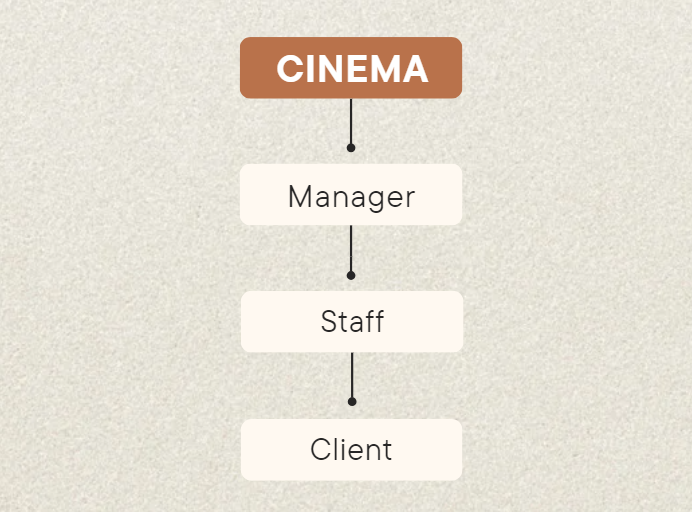
# **CHƯƠNG 3: PHÂN TÍCH HỆ THỐNG**

## 3.1 Giới thiệu hệ thống

Phần mềm "Xây dựng hệ thống quản lý rạp chiếu phim" được thiết kế để cung cấp các chức năng sau:

|  |  |
| --- | --- |
| STT | Tên Tác Nhân |
| 1 | Quản Lý |
| 2 | Nhân Viên |
| 3 | Khách Hàng |

* Quản lý lịch chiếu phim
* Quản lý phòng chiếu và ghế ngồi
* Quản lý thông tin phim
* Quản lý nhân viên
* Thống kê và báo cáo



Hình 1. Các tác nhân

## 3.2Mô tả chi tiết

|  |  |  |  |
| --- | --- | --- | --- |
| Nhóm | Tên UserCase | | Tác Nhân |
| Cơ bản | Đăng nhập | | Quản lý,  Nhân viên,  Khách hàng |
| Đăng xuất | |
| Quên mật khẩu | |
| Đổi mật khẩu | |
| Ghi nhận các phản hồi, khiếu nại | |
| Xem và cập nhật thông tin cá nhân | |
| Xem lịch chiếu | |
| Đăng ký | | Khách hàng |
| Xem lịch sử thanh toán | |
| Đặt vé | |
| Quản lý hệ thống |  | | Quản lý |
| Xem, thêm, cập nhật hoặc  chỉnh sửa | Doanh thu (Chỉ xem) |
| Các nhân viên |
| Các khách hàng |
| Lịch làm việc nhân viên |
| Phim |
| Lịch chiếu |
| Phòng |
| Ghế ngồi |
| Xem và trả lời các phản hồi, khiếu nại | |
| Quản lý kinh doanh |  | | Quản lý  Nhân viên |
| Xem tình trạng phòng, ghế ngồi | |
|  | |

**Hệ thống: Đăng nhập, đăng xuất, đăng ký user**

* **Đăng nhập:** Yêu cầu người dùng nhập thông tin tài khoản để sử dụng hệ thống.
* **Đăng xuất:** Thoát khỏi hệ thống và kết thúc phiên làm việc.
* **Đăng ký:** Thêm tài khoản mới cho người dùng.

**Quản lý lịch chiếu phim**

* **Danh sách lịch chiếu:** Quản lý thông tin các suất chiếu phim.
* **Thêm/sửa/xóa lịch chiếu**: Cung cấp chức năng để thêm mới, chỉnh sửa hoặc xóa lịch chiếu phim.

**Quản lý phòng chiếu và ghế ngồi**

* **Danh sách phòng chiếu:** Quản lý thông tin về các phòng chiếu.
* **Thêm/sửa/xóa phòng chiếu:** Cho phép thêm mới, chỉnh sửa hoặc xóa thông tin phòng chiếu.
* **Quản lý ghế ngồi:** Theo dõi và cập nhật trạng thái ghế ngồi trong các phòng chiếu (đã đặt hoặc còn trống hoặc khóa ghế).

**Quản lý thông tin phim**

* **Danh sách phim:** Quản lý thông tin các bộ phim đang chiếu và sắp chiếu.
* **Thêm/sửa/xóa phim:** Thêm mới, chỉnh sửa hoặc xóa thông tin về phim.

**Quản lý nhân viên hoặc khách hàng**

* **Danh sách người dùng:** Quản lý thông tin người dùng.
* **Thêm/sửa/xóa người dùng:** Thêm mới, chỉnh sửa hoặc xóa thông tin người dùng.

**Thống kê và báo cáo**

* **Thống kê doanh thu:** Thống kê doanh thu bán vé theo các khoảng thời gian (hôm nay, tháng trước, tháng này).
* **In báo cáo:** Cung cấp chức năng in các báo cáo doanh thu, số lượng khách hàng, nhân viên.

## 3.3 Yêu cầu chức năng

**Chức năng quản lý**

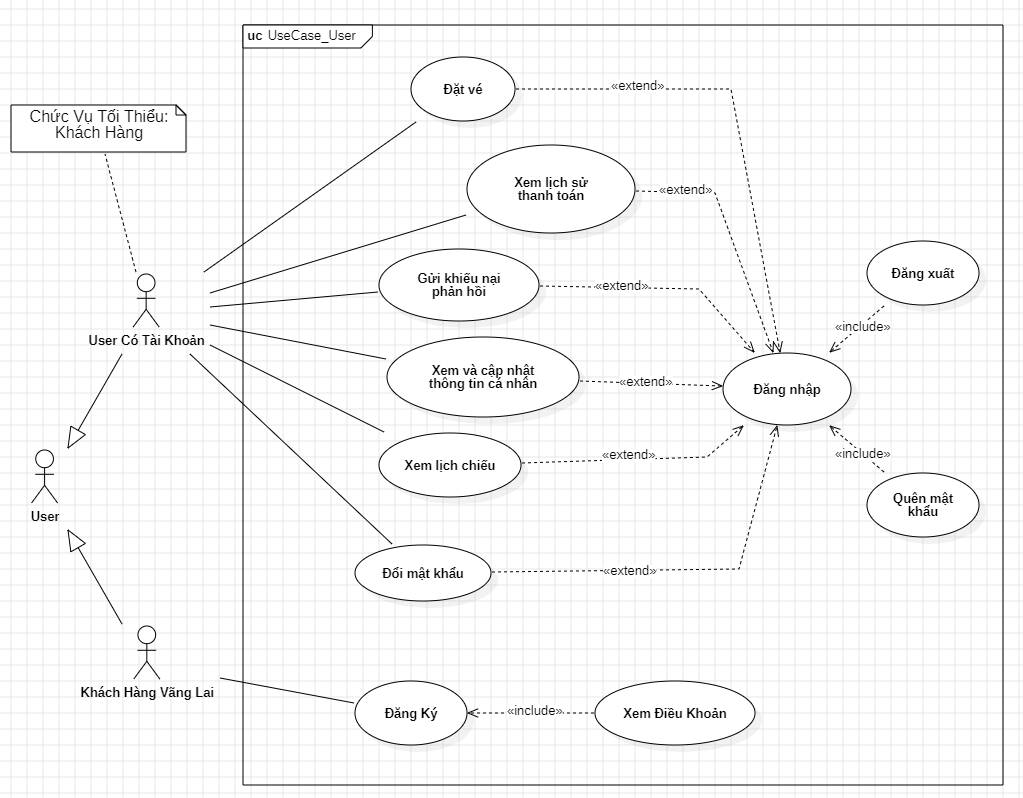
* Quản lý toàn bộ hoạt động của rạp chiếu phim.
* Theo dõi và đánh giá hiệu quả làm việc của nhân viên.
* Quản lý lịch chiếu phim và phòng chiếu.

**Chức năng của nhân viên**

* Hỗ trợ khách hàng mua vé và chọn ghế.
* Kiểm tra vé và hướng dẫn khách hàng vào phòng chiếu.
* Bảo trì và vệ sinh phòng chiếu.
* Xem lịch đi làm.

**Chức năng của khách hàng**

* Mua vé xem phim trực tuyến hoặc tại rạp.
* Chọn suất chiếu và ghế ngồi.
* Nhận thông tin về các suất chiếu và các bộ phim đang chiếu.

1. 

Hình 2. Sơ đồ UseCase

* 1. Sơ đồ UseCase

A diagram of a diagram

Description automatically generated

Hình 3. Sơ đồ UseCase Nhân Viên

* 1. Bảng đặc tả
     1. **Bảng đặc tả đăng nhập:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC001** | **Tên UseCase** | **Đăng Nhập** |
| Tác Nhân | Quản lý, nhân viên, khách hàng | | |
| Mô Tả | Tác nhân đăng nhập vào hệ thống để sử dụng các chức năng hệ thống | | |
| Sự kiện kích hoạt | Click vào nút đăng nhập | | |
| Tiền điều kiện | Tác nhân đã có tài khoản trên hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng đăng nhập |
| 2 | Hệ thống | Hiển thị giao diện đăng nhập |
| 3 | Người dùng | Nhập username và mật khẩu |
| 4 | Người dùng | Yêu cầu đăng nhập |
| 5 | Hệ thống | Kiểm tra xem khách đã nhập các trường bắt buộc  nhập hay chưa |
| 6 | Hệ thống | Kiểm tra username và mật khẩu có hợp lệ hay không |
| 7 | Hệ thống | Kiểm tra username và mật khẩu có tồn tạitrong hệ thống hay không |
| 8 | Hệ thống | Thông báo đăng nhập thành công, hiển thị chức năng tương ứng đối với Người dùng |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 5a | Hệ thống | Thông báo lỗi: Cần nhập các trường bắt buộc nhập  nếu khách nhập thiếu |
| 6a | Hệ thống | Thông báo lỗi: username và/hoặc mật khẩu chưa hợp lệ, nếu nhập sai cú pháp |
|  | 7a | Hệ thống | nếu không tìm thấy username hoặc mật khẩu trong hệ  thống thì thông báo không tồn tại |
| Hậu điều kiện | Tác nhân đăng nhập được vào hệ thống | | |

* + 1. **Bảng đặc tả đăng ký:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC002** | **Tên UseCase** | **Đăng Ký** |
| Tác Nhân | Khách hàng (người dùng chưa có tài khoản) | | |
| Mô Tả | Tác nhân đăng ký tài khoản mới để sử dụng các chức năng hệ thống | | |
| Sự kiện kích hoạt | Click vào đăng ký ở trên form đăng nhập | | |
| Tiền điều kiện | Tác nhân chưa có tài khoản trên hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng đăng ký |
| 2 | Hệ thống | Hiển thị giao diện đăng ký |
| 3 | Người dùng | Nhập thông tin cá nhân và thông tin đăng nhập |
| 4 | Người dùng | Yêu cầu đăng ký |
| 5 | Hệ thống | Kiểm tra xem đã nhập các trường bắt buộc  nhập hay chưa |
| 6 | Hệ thống | Kiểm tra thông tin nhập vào có hợp lệ hay không |
| 7 | Hệ thống | Kiểm tra xem thông tin đăng nhập nhâp vào đã tồn tại trong hệ thống hay chưa |
| 8 | Hệ thống | Thông báo đăng ký thành công và quay trở về form đăng nhập |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 5a | Hệ thống | Thông báo lỗi: Cần nhập các trường bắt buộc nhập  nếu nhập thiếu |
| 6a | Hệ thống | Thông báo lỗi: thông tin nhập vào chưa hợp lệ, nếu nhập sai cú pháp |
| 7a | Hệ thống | Thông báo lỗi: thông tin đăng ký đã tồn tại trong hệ thống, nếu dùng username, email hoặc số điện thoại đã được đăng ký trước đây |
| Hậu điều kiện | Tác nhân đăng ký tài khoản mới thành công và có thể đăng nhập vào hệ thống | | |

* + 1. **Bảng đặc tả quên mật khẩu:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC003** | **Tên UseCase** | **Quên mật khẩu** |
| Tác Nhân | Quản lý, nhân viên, khách hàng | | |
| Mô Tả | Tác nhân quên mật khẩu và yêu cầu đặt lại mật khẩu mới | | |
| Sự kiện kích hoạt | Click vào "Quên mật khẩu" ở trên form đăng nhập | | |
| Tiền điều kiện | Tác nhân đã có tài khoản trên hệ thống và quên mật khẩu | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | |  | | --- | |  |   Chọn “quên mật khẩu” |
| 2 | Hệ thống | Hiển thị form quên mật khẩu |
| 3 | Người dùng | Chọn hình thức nhận mã OTP (email hoặc số điện thoại) |
| 4 | Người dùng | Nhập thông tin email hoặc số điện thoại |
| 5 | Người dùng | Yêu cầu xác minh |
| 6 | Hệ thống | Kiểm tra xem đã nhập các trường bắt buộc  nhập hay chưa |
| 7 | Hệ thống | Kiểm tra thông tin nhập vào có hợp lệ hay không |
| 8 | Hệ thống | Kiểm tra thông tin nhập vào có tồn tại hay không |
| 9 | Hệ thống | Gửi mã OTP đến email hoặc số điện thoại đã nhập |
| 10 | Người dùng | Nhập mã OTP và chọn xác minh |
| 11 | Hệ thống | Xác thực mã OTP |
| 12 | Người dùng | Nhập mật khẩu mới và xác nhận mật khẩu mới |
| 13 | Hệ thống | Kiểm tra cú pháp có hợp lệ không, mật khẩu nhập lại có giống mật khẩu hay không |
| 14 | Người dùng | Yêu cầu đổi mật khẩu |
| 15 | Hệ thống | Thông báo thay đổi mật khẩu thành công và quay trở về form đăng nhập |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 6a | Hệ thống | Thông báo lỗi: Cần nhập email hoặc số điện thoại nếu nhập thiếu |
| 7a | Hệ thống | Thông báo lỗi: email hoặc số điện thoại không đúng cú pháp nếu nhập sai |
| 8a | Hệ thống | Thông báo lỗi: email hoặc số điện thoại không tồn tại trong hệ thống nếu nó chưa được đăng ký |
| 11a | Hệ thống | Thông báo lỗi: mã OTP không hợp lệ nếu nhập sai |
| 13a | Hệ thống | Thông báo lỗi: mật khẩu nhập lại không khớp nếu mật khẩu nhập lại không giống với mật khẩu |
| Hậu điều kiện | Tác nhân đặt lại mật khẩu mới thành công và có thể đăng nhập vào hệ thống | | |

* + 1. **Bảng đặc tả đổi mật khẩu**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC004** | **Tên UseCase** | **Đổi mật khẩu** |
| Tác Nhân | Quản lý, nhân viên, khách hàng | | |
| Mô Tả | Tác nhân chọn đổi mật khẩu và thực hiện thay đổi mật khẩu | | |
| Sự kiện kích hoạt | Chọn "Cài đặt" trên trang chính, sau đó chọn "Đổi mật khẩu" | | |
| Tiền điều kiện | Tác nhân đã đăng nhập vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng đổi mật khẩu |
| 2 | Hệ thống | Hiển thị giao diện đổi mật khẩu |
| 3 | Người dùng | Nhập mật khẩu cũ, mật khẩu mới và xác nhận mật khẩu mới |
| 4 | Người dùng | Yêu cầu đổi mật khẩu |
| 5 | Hệ thống | Kiểm tra xem đã nhập các trường bắt buộc  nhập hay chưa |
| 6 | Hệ thống | Kiểm tra thông tin nhập vào có hợp lệ hay không |
| 7 | Hệ thống | Kiểm tra mật khẩu mới có khớp với xác nhận mật khẩu không |
| 8 | Hệ thống | Kiểm tra xem mật khẩu cũ có đúng không |
| 9 | Hệ thống | Thông báo đổi mật khẩu thành công và quay trở về form chính |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 5a | Hệ thống | Thông báo lỗi: Cần nhập các trường bắt buộc nhập  nếu nhập thiếu |
| 6a | Hệ thống | Thông báo lỗi: thông tin nhập vào chưa hợp lệ, nếu nhập sai cú pháp |
| 7a | Hệ thống | Thông báo lỗi: mật khẩu mới nhập lại không trùng với mật khẩu mới nếu nhập không trùng nhau |
|  | 8a | Hệ thống | Thông báo lỗi: Sai mật khẩu nếu nhập vào mật khẩu cũ không đúng |
| Hậu Điều Kiện | Tác nhân đã thay đổi mật khẩu thành công và có thể đăng nhập với mật khẩu mới | | |

* + 1. **Bảng đặc tả xem và cập nhật thông tin cá nhân:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC005** | **Tên UseCase** | **Xem và cập nhật thông tin cá nhân** |
| Tác Nhân | Quản lý, nhân viên, khách hàng | | |
| Mô Tả | Tác nhân xem và cập nhật thông tin cá nhân. Các thông tin có thể chỉnh sửa và sẽ kiểm tra tính hợp lệ, đảm bảo thông tin độc nhất không bị trùng lặp. | | |
| Sự kiện kích hoạt | Chọn "Hồ sơ" trên trang chính | | |
| Tiền điều kiện | Tác nhân đã đăng nhập vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | |  | | --- | |  |   Chọn "Hồ sơ" trên trang chính |
| 2 | Hệ thống | Hiển thị form thông tin cá nhân |
| 3 | Người dùng | Chỉnh sửa các thông tin có thể chỉnh sửa |
| 4 | Người dùng | Chọn "Chỉnh sửa" để cập nhật thông tin |
| 6 | Hệ thống | Kiểm tra tính hợp lệ của các thông tin đã chỉnh sửa |
| 7 | Hệ thống | Kiểm tra thông tin độc nhất (username, số điện thoại, email) không bị trùng lặp trong hệ thống |
| 8 | Hệ thống | Thông báo thành công |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 6a | Hệ thống | Thông báo lỗi nếu có thông tin không hợp lệ |
| 7a | Hệ thống | Thông báo lỗi nếu thông tin độc nhất (username, số điện thoại, email) bị trùng trong cơ sở dữ liệu |
| Hậu điều kiện | Tác nhân đã cập nhật thông tin cá nhân thành công và sử dụng các thông tin đã được thay đổi mới | | |

* + 1. **Bảng đặc tả đăng xuất:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC006** | **Tên UseCase** | **Đăng Xuất** |
| Tác Nhân | Quản lý, nhân viên, khách hàng | | |
| Mô Tả | Tác nhân đăng xuất khỏi hệ thống quay về trang đăng nhập | | |
| Sự kiện kích hoạt | Chọn "Cài đặt" trên trang chính, sau đó chọn "Đăng xuất" | | |
| Tiền điều kiện | Tác nhân đã đăng nhập vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng đăng xuất |
| 2 | Hệ thống | Hiển thị thông báo xác nhận đăng xuất |
| 3 | Người dùng | Xác nhận đăng xuất |
| 4 | Hệ thống | |  | | --- | |  |   Đóng phiên đăng nhập hiện tại |
| 5 | Hệ thống | Chuyển hướng về trang đăng nhập |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 3a | Người dùng | Hủy bỏ xác nhận đăng xuất |
| 3b | Hệ thống | Không đóng phiên đăng nhập nếu người dùng hủy xác nhận đăng xuất |
| Hậu điều kiện | Tác nhân đã đăng xuất khỏi hệ thống và quay lại trang đăng nhập | | |

* + 1. **Bảng đặc tả xem lịch sử thanh toán:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC007** | **Tên UseCase** | **Xem lịch sử thanh toán** |
| Tác Nhân | Quản lý, nhân viên, khách hàng | | |
| Mô Tả | Tác nhân xem lịch sử thanh toán của mình | | |
| Sự kiện kích hoạt | Chọn "Hồ sơ" trên trang chính và sau đó chọn "Xem lịch sử thanh toán" ở góc dưới form thông tin cá nhân | | |
| Tiền điều kiện | Tác nhân đã đăng nhập vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng "Xem lịch sử thanh toán" |
| 2 | Hệ thống | Hiển thị form lịch sử thanh toán |
| 3 | Hệ thống | Lấy dữ liệu lịch sử thanh toán của người dùng từ cơ sở dữ liệu |
| 4 | Hệ thống | Hiển thị danh sách các giao dịch đã thực hiện của người dùng |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 4a | Người dùng | Hiển thị trống nếu không có giao dịch nào trong lịch sử |
| Hậu điều kiện | Tác nhân đã xem được lịch sử thanh toán của mình | | |

* + 1. **Bảng đặc tả xem lịch chiếu:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC008** | **Tên UseCase** | **Xem lịch chiếu** |
| Tác Nhân | Quản lý, nhân viên, khách hàng | | |
| Mô Tả | Tác nhân xem lịch chiếu của các bộ phim do quản lý sắp xếp | | |
| Sự kiện kích hoạt | Chọn "Trang chủ" trên trang chính và sau đó chọn "Xem lịch chiếu" | | |
| Tiền điều kiện | Tác nhân đã đăng nhập vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng "Xem lịch chiếu" |
| 2 | Hệ thống | Hiển thị form lịch chiếu |
| 3 | Hệ thống | Lấy dữ liệu lịch chiếu từ cơ sở dữ liệu |
| 4 | Hệ thống | Hiển thị danh sách các suất chiếu và thông tin liên quan |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 4a | Người dùng | Hiển thị trống nếu không có suất chiếu nào được tìm thấy |
| Hậu điều kiện | Tác nhân đã xem được lịch chiếu của các bộ phim | | |

* + 1. **Bảng đặc tả đặt vé:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC009** | **Tên UseCase** | **Đặt vé** |
| Tác Nhân | Quản lý, nhân viên, khách hàng | | |
| Mô Tả | Tác nhân chọn đặt vé trên trang chủ để mở form, sau đó chọn phim muốn xem, lựa chọn ghế ngồi, chọn thanh toán hoặc mua thêm đồ ăn, nước uống, và hoàn tất giao dịch. | | |
| Sự kiện kích hoạt | Chọn trang chủ sau đó chọn “Đặt vé” trên form chính | | |
| Tiền điều kiện | Tác nhân phải có tài khoản và đăng nhập vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | |  | | --- | |  |   Chọn “đặt vé” |
| 2 | Hệ thống | Hiển thị form đặt vé. |
| 3 | Người dùng | Chọn phim |
| 4 | Hệ thống | Hiển thị thông tin phim và các ghế ngồi còn trống. |
| 5 | Người dùng | Lựa chọn ghế ngồi (hiển thị tổng tiền phải trả). |
| 6 | Người dùng | Chọn "Thanh toán ngay" |
| 7 | Người dùng | Thanh toán theo quy định điều khoản (với số tiền tương đương với tất cả dịch vụ đã chọn) |
| 8 | Hệ thống | Xác nhận thông tin thanh toán. |
| 9 | Hệ thống | Kiểm tra thông tin thanh toán và thông báo thành công. |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
|
|
| 8a | Hệ thống | Thông báo lỗi nếu thanh toán không hợp lệ. |
| 9a | Người dùng | Nếu là nhân viên đặt cho khách thì sẽ có thêm nút in hóa đơn và hóa đơn sẽ được in ra, đồng thời lịch sử giao dịch sẽ được lưu vào lịch sử của nhân viên thực hiện |
| Hậu điều kiện | Tác nhân đã đặt vé thành công và được lưu lịch sử giao dịch tương ứng. | | |

* + 1. **Bảng đặc tả quản lý doanh thu:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC010** | **Tên UseCase** | **Quản lý doanh thu** |
| Tác Nhân | Quản lý | | |
| Mô Tả | Quản lý mở form thống kê doanh thu thông qua form chính, sau đó chọn thống kê doanh thu theo nhu cầu. | | |
| Sự kiện kích hoạt | Chọn "Quản lý" trên form chính, sau đó chọn "Thống kê". | | |
| Tiền điều kiện | Quản lý phải đăng nhập vào hệ thống. | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng “Thống kê” |
| 2 | Hệ thống | Hiển thị form thống kê với các tùy chọn "Thống kê doanh thu tháng trước", "Thống kê doanh thu hôm nay", “Thống kê doanh thu tháng này”. |
| 3 | Người dùng | Chọn "Thống kê" |
| 4 | Hệ thống | Hiện thông tin doanh thu theo yêu cầu vào hệ thống và thông báo thành công. |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 3a | Hệ thống | Hiển thị thông tin các mã đơn hàng (bill) mà khách hàng đã khách hàng đã thanh toán. |
| 3b | Người dùng | Chỉ xem thông tin doanh thu. |
| 3c | Hệ thống | |  | | --- | |  |   Kiểm tra tính hợp lệ của thông tin doanh thu.   |  | | --- | |  | |
| 3d | Hệ thống | Thông báo lỗi nếu thông tin doanh thu không hợp lệ. |
| Hậu điều kiện | Thông tin doanh thu và đã được hiện trên form. | | |

* + 1. **Bảng đặc tả quản lý nhân viên:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC011** | **Tên UseCase** | **Quản lý nhân viên** |
| Tác Nhân | Quản lý | | |
| Mô Tả | Quản lý mở form quản lý nhân viên, thêm nhân viên mới, chỉnh sửa thông tin nhân viên, và xóa nhân viên khỏi hệ thống. | | |
| Sự kiện kích hoạt | Chọn "Quản lý" trên form chính, sau đó chọn "Nhân Viên". | | |
| Tiền điều kiện | Quản lý phải đăng nhập vào hệ thống. | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng quản lý “Nhân Viên” |
| 2 | Hệ thống | Hiển thị form quản lý nhân viên với danh sách tất cả nhân viên trong hệ thống |
| 3 | Người dùng | Chọn nhân viên muốn điều chỉnh |
| 4 | Hệ thống | Thông báo cập nhật thông tin thành công |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 3a | Hệ thống | Nhân viên sẽ được xóa khỏi hệ thống nếu người dùng chọn xóa |
| 3b1 | Hệ thống | Sẽ kiểm tra thông tin người dùng nhập vào có hợp lệ hay không, nếu người dùng chỉnh sửa thông tin nhân viên đã chọn và nhấn chỉnh sửa |
| 3b2 | Hệ thống | Kiểm tra thông tin độc nhất (username, số điện thoại, email) không bị trùng lặp trong hệ thống |
| 3b3 | Hệ thống | Thông báo thông tin nhập vào đã được sử dụng nếu nó đã tồn tại trong hệ thống |
| 3c1 | Người dùng | Nhập thông tin vào tất cả các trường nếu người dùng muốn thêm mới 1 tài khoản nhân viên và nhấn “Thêm” |
| 3c2 | Hệ thống | Kiểm tra thông tin người dùng nhập vào có hợp lệ hay không, nếu không thì sẽ thông báo lỗi không hợp lệ |
| 3c3 | Hệ thống | Kiểm tra thông tin độc nhất (username, số điện thoại, email) không bị trùng lặp trong hệ thống |
| 3c4 | Hệ thống | Thông báo thông tin nhập vào đã được sử dụng nếu nó đã tồn tại trong hệ thống |
| Hậu điều kiện | Thông tin nhân viên đã được thêm, chỉnh sửa hoặc xóa thành công trong hệ thống. | | |

* + 1. **Bảng đặc tả quản lý khách hàng:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC012** | **Tên UseCase** | **Quản lý khách hàng:** |
| Tác Nhân | Quản lý | | |
| Mô Tả | Quản lý mở form quản lý khách hàng, thêm khách hàng mới, chỉnh sửa thông tin khách hàng, và xóa khách hàng khỏi hệ thống. | | |
| Sự kiện kích hoạt | Chọn "Quản lý" trên form chính, sau đó chọn "Khách hàng". | | |
| Tiền điều kiện | Quản lý phải đăng nhập vào hệ thống. | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng quản lý “Khách hàng” |
| 2 | Hệ thống | Hiển thị form quản lý khách hàng với danh sách tất cả khách hàng trong hệ thống |
| 3 | Người dùng | Chọn khách hàng muốn điều chỉnh |
| 4 | Hệ thống | Thông báo cập nhật thông tin thành công |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 3a | Hệ thống | Khách hàng sẽ được xóa khỏi hệ thống nếu người dùng chọn xóa |
| 3b1 | Hệ thống | Sẽ kiểm tra thông tin người dùng nhập vào có hợp lệ hay không, nếu người dùng chỉnh sửa thông tin khách hàng đã chọn và nhấn chỉnh sửa |
| 3b2 | Hệ thống | Kiểm tra thông tin độc nhất (username, số điện thoại, email) không bị trùng lặp trong hệ thống |
| 3b3 | Hệ thống | Thông báo thông tin nhập vào đã được sử dụng nếu nó đã tồn tại trong hệ thống |
| 3c1 | Người dùng | Nhập thông tin vào tất cả các trường nếu người dùng muốn thêm mới 1 tài khoản khách hàng và nhấn “Thêm” |
| 3c2 | Hệ thống | Kiểm tra thông tin người dùng nhập vào có hợp lệ hay không, nếu không thì sẽ thông báo lỗi không hợp lệ |
| 3c3 | Hệ thống | Kiểm tra thông tin độc nhất (username, số điện thoại, email) không bị trùng lặp trong hệ thống |
| 3c4 | Hệ thống | Thông báo thông tin nhập vào đã được sử dụng nếu nó đã tồn tại trong hệ thống |
| Hậu điều kiện | Thông tin khách hàng đã được thêm, chỉnh sửa hoặc xóa thành công trong hệ thống. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC013** | **Tên UseCase** | **Lịch làm việc của nhân viên** |
| Tác Nhân | Quản lý | | |
| Mô Tả | Quản lý mở form quản lý lịch làm việc, thêm, sửa, xóa lịch làm việc của từng ngày, thay đổi giờ làm, thay đổi nhân viên, xóa ngày và thêm ngày mới. | | |
| Sự kiện kích hoạt | Chọn "Quản lý" trên form chính, sau đó chọn "Lịch". | | |
| Tiền điều kiện | Quản lý phải đăng nhập vào hệ thống. | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng "Lịch". |
| 2 | Hệ thống | Hiển thị form chứa lịch làm việc và lịch chiếu phim đã được lưu trong sơ sở dữ liệu. |
| 3 | Người dùng | Chọn "Lịch làm việc". |
| 4 | Hệ thống | Hiển thị form lịch làm việc. |
| 5 | Người dùng | Thêm, sửa, xóa lịch làm việc của từng ngày, thay đổi giờ làm, thay đổi nhân viên, xóa ngày hoặc thêm ngày mới. |
| 6 | Hệ thống | Kiểm tra tính hợp lệ của các thay đổi. |
| 7 | Hệ thống | Lưu thông tin lịch làm việc vào hệ thống và thông báo thành công. |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 6a | Hệ thống | Thông báo lỗi nếu thông tin lịch làm việc không hợp lệ |
| 7a | Hệ thống | Thông báo lỗi nếu không thể lưu lịch làm việc vào hệ thống. |
| Hậu điều kiện | Thông tin lịch làm việc đã được cập nhật mới trong hệ thống thành công | | |

* + 1. **Bảng đặc tả quản lý lịch làm việc của nhân viên:**
    2. **Bảng đặc tả quản lý lịch chiếu phim**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC014** | **Tên UseCase** | **Quản lý lịch chiếu phim** |
| Tác Nhân | Quản lý | | |
| Mô Tả | Quản lý mở form quản lý lịch làm việc, thêm, sửa, xóa lịch làm việc của từng ngày, thay đổi giờ làm, thay đổi nhân viên, xóa ngày và thêm ngày mới. | | |
| Sự kiện kích hoạt | Chọn "Quản lý" trên form chính, sau đó chọn "Lịch". | | |
| Tiền điều kiện | Quản lý phải đăng nhập vào hệ thống. | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng "Lịch". |
| 2 | Hệ thống | Hiển thị form chứa lịch làm việc và lịch chiếu phim. |
| 3 | Người dùng | Chọn "Lịch chiếu phim". |
| 4 | Hệ thống | Hiển thị form lịch chiếu phim đã được lưu trong sơ sở dữ liệu. |
| 5 | Người dùng | Thêm, sửa, xóa lịch chiếu ngày, thay đổi giờ chiếu, thay đổi phim, xóa ngày hoặc thêm ngày mới. |
| 6 | Hệ thống | Kiểm tra tính hợp lệ của các thay đổi. |
| 7 | Hệ thống | Lưu thông tin lịch chiếu vào hệ thống và thông báo thành công. |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 6a | Hệ thống | Thông báo lỗi nếu thông tin lịch chiếu không hợp lệ |
| 7a | Hệ thống | Thông báo lỗi nếu không thể lưu lịch chiếu vào hệ thống. |
| Hậu điều kiện | Thông tin lịch chiếu đã được cập nhật mới trong hệ thống thành công | | |

* + 1. **Bảng đặc tả quản lý phim:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC015** | **Tên UseCase** | **Quản lý phim** |
| Tác Nhân | Quản lý | | |
| Mô Tả | Quản lý mở form quản lý phim, thêm, sửa, xóa thông tin phim của rạp phim. | | |
| Sự kiện kích hoạt | Chọn "Quản lý" trên form chính, sau đó chọn "Phim". | | |
| Tiền điều kiện | Quản lý phải đăng nhập vào hệ thống. | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng "Phim". |
| 2 | Hệ thống | Hiển thị form chứa thông tin phim đã được lưu trong sơ sở dữ liệu. |
| 3 | Người dùng | Thêm, sửa, hoặc xóa thông tin phim (nếu sửa hoặc xóa người dùng phải chọn ít nhất 1 phim trên form) |
| 4 | Hệ thống | Khi người dùng thêm hoặc sửa thông tin phim:kiểm tra tính hợp lệ của các thay đổi. |
| 5 | Hệ thống | Lưu các sự thay đổi về phim vào hệ thống và thông báo thành công. |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 3a | Hệ thống | Thông báo lỗi chưa chọn phim cần xóa hoặc sửa nếu người dùng chưa chọn |
| 4a | Hệ thống | Thông báo lỗi nếu thông tin cập nhật vào không hợp lệ |
| 5a | Hệ thống | Thông báo lỗi nếu không thể lưu thông tin phim vào hệ thống. |
| Hậu điều kiện | Thông tin phim được cập nhật mới trong hệ thống thành công. | | |

* + 1. **Bảng đặc tả quản lý phòng:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC016** | **Tên UseCase** | **Quản lý phòng** |
| Tác Nhân | Quản lý | | |
| Mô Tả | Quản lý mở form quản lý phòng chiếu, thêm, sửa, xóa thông tin phòng chiếu của rạp phim. | | |
| Sự kiện kích hoạt | Chọn "Quản lý" trên form chính, sau đó chọn "Phòng". | | |
| Tiền điều kiện | Quản lý phải đăng nhập vào hệ thống. | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng "Phòng". |
| 2 | Hệ thống | Hiển thị form chứa thông tin phòng đã được lưu trong sơ sở dữ liệu. |
| 3 | Người dùng | Thêm, sửa, hoặc xóa thông tin phòng (nếu sửa hoặc xóa người dùng phải chọn ít nhất 1 phòng trên form) |
| 4 | Hệ thống | Khi người dùng thêm hoặc sửa thông tin phòng:kiểm tra tính hợp lệ của các thay đổi. |
| 5 | Hệ thống | Lưu các sự thay đổi về phòng vào hệ thống và thông báo thành công. |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 3a | Hệ thống | Thông báo lỗi chưa chọn phim cần xóa hoặc sửa nếu người dùng chưa chọn |
| 4a | Hệ thống | Thông báo lỗi nếu thông tin cập nhật vào không hợp lệ |
| 5a | Hệ thống | Thông báo lỗi nếu không thể lưu thông tin phòng vào hệ thống. |
| Hậu điều kiện | Thông tin phòng chiếu được cập nhật mới trong hệ thống thành công. | | |

* + 1. **Bảng đặc tả quản lý ghế ngồi:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC017** | **Tên UseCase** | **Quản lý ghế** |
| Tác Nhân | Quản lý | | |
| Mô Tả | Quản lý mở form quản lý ghế ngồi, thêm, sửa, xóa thông tin ghế ngồi của phòng chiếu. | | |
| Sự kiện kích hoạt | Chọn "Quản lý" trên form chính, sau đó chọn "Ghế". | | |
| Tiền điều kiện | Quản lý phải đăng nhập vào hệ thống. | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng "Ghế". |
| 2 | Hệ thống | Hiển thị form chứa thông tin phòng đã được lưu trong sơ sở dữ liệu. |
| 3 | Người dùng | Chọn một phòng chiếu cần cập nhật mới ghế ngồi. |
| 4 | Hệ thống | Hiển thị thông tin ghế ngồi của phòng đã chọn từ cơ sở dữ liệu. |
| 5 | Người dùng | Thêm, sửa, hoặc xóa thông tin ghế ngồi (nếu sửa hoặc xóa người dùng phải chọn ít nhất 1 ghế ngồi trên form). |
| 6 | Hệ thống | Khi người dùng thêm hoặc sửa thông tin ghế ngồi: kiểm tra tính hợp lệ của các thay đổi. |
| 7 | Hệ thống | Lưu các sự thay đổi về ghế của phòng được chọn vào hệ thống và thông báo thành công. |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 5a | Hệ thống | Thông báo lỗi chưa chọn ghế cần xóa hoặc sửa nếu người dùng chưa chọn |
| 6a | Hệ thống | Thông báo lỗi nếu thông tin cập nhật vào không hợp lệ |
| 7a | Hệ thống | Thông báo lỗi nếu không thể lưu thông tin phòng vào hệ thống. |
| Hậu điều kiện | Thông tin ghế ngồi được cập nhật mới trong hệ thống thành công. | | |

* + 1. **Bảng đặc tả xem và trả lời khiếu nại, phản hồi:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC018** | **Tên UseCase** | **Xem và trả lời phản hồi khiếu nại** |
| Tác Nhân | Quản lý | | |
| Mô Tả | Quản lý mở form xem và trả lời phản hồi khiếu nại của khách hàng. | | |
| Sự kiện kích hoạt | Chọn "Quản lý" trên form chính, sau đó chọn "Khiếu nại". | | |
| Tiền điều kiện | Quản lý phải đăng nhập vào hệ thống. | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng quản lý "Khiếu nại". |
| 2 | Hệ thống | Hiển thị form chứa thông tin các phản hồi khiếu nại. |
| 3 | Người dùng | Chọn một phản hồi khiếu nại để xem chi tiết. |
| 4 | Hệ thống | Hiển thị chi tiết phản hồi khiếu nại đã được lưu trong cơ sở dữ liệu. |
| 5 | Người dùng | Nhập nội dung trả lời phản hồi khiếu nại, sau đó gửi đi. |
| 6 | Hệ thống | Lưu nội dung trả lời vào hệ thống và thông báo thành công. |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 6a | Hệ thống | Thông báo lỗi nếu không thể lưu nội dung trả lời vào hệ thống. |
| Hậu điều kiện | Khiếu nại được chọn sẽ được trả lời và lưu vào hệ thống | | |

* + 1. **Bảng đặc tả gửi phản hồi khiếu nại:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC019** | **Tên UseCase** | **Gửi phản hồi khiếu nại** |
| Tác Nhân | Quản lý, Khách hàng, Nhân viên | | |
| Mô Tả | Ghi nhận các thông tin phản hồi khiếu nại người dùng | | |
| Sự kiện kích hoạt | Chọn "Cài đặt" trên form chính, sau đó chọn "Khiếu nại". | | |
| Tiền điều kiện | Người dùng phải đăng nhập vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng "Khiếu nại". |
| 2 | Hệ thống | Hiển thị form để nhập các thông tin phản hồi khiếu nại. |
| 3 | Người dùng | Chọn loại khiếu nại và nhập đầy đủ nội dung bắt buộc, sau đó gửi đi |
| 4 | Hệ thống | Kiểm tra tất cả các trường đã được nhập đầy đủ chưa |
| 5 | Hệ thống | Lưu nội dung phản hồi khiếu nại vào hệ thống và thông báo thành công. |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 4a | Hệ thống | Thông báo lỗi: chưa nhập đầy đủ bản khiếu nại nếu người dùng chưa điền đầy đủ thông tin |
| 5a | Hệ thống | Thông báo lỗi nếu không thể lưu nội dung phản hồi khiếu nại vào hệ thống. |
| Hậu điều kiện | Nội dung phản hồi khiếu nại được ghi nhận trong hệ thống thành công, quản lý có thể xem được | | |

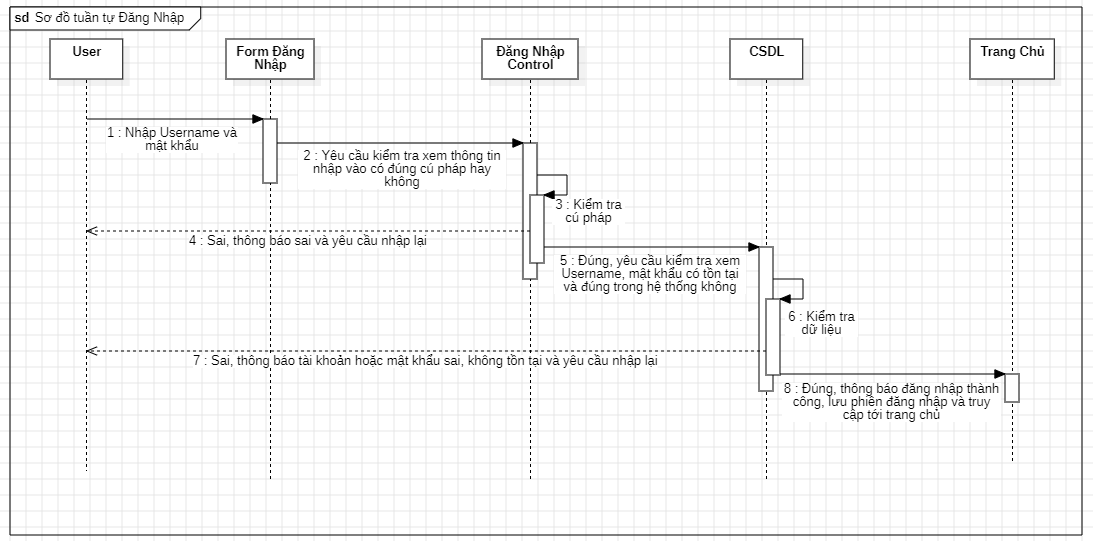
* + 1. **Bảng đặc tả xem lịch làm việc:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC020** | **Tên UseCase** | **Gửi phản hồi khiếu nại** |
| Tác Nhân | Nhân viên | | |
| Mô Tả | Nhân viên mở form xem lịch làm việc do quản lý cung cấp. | | |
| Sự kiện kích hoạt | Chọn "Hồ sơ" trên form chính, sau đó chọn "Xem lịch làm việc". | | |
| Tiền điều kiện | Nhân viên phải đăng nhập vào hệ thống. | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng "Hồ sơ". |
| 2 | Hệ thống | Hiển thị form hồ sơ cá nhân của nhân viên |
| 3 | Người dùng | Chọn "Xem lịch làm việc |
| 4 | Hệ thống | Hiển thị lịch làm việc của nhân viên đã được lưu trong cơ sở dữ liệu |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 4a | Hệ thống | Hiển thị trống nếu nhân viên chưa được quản lý cập nhật lịch làm việc |
| Hậu điều kiện | Lịch làm việc của nhân viên được hiển thị thành công | | |

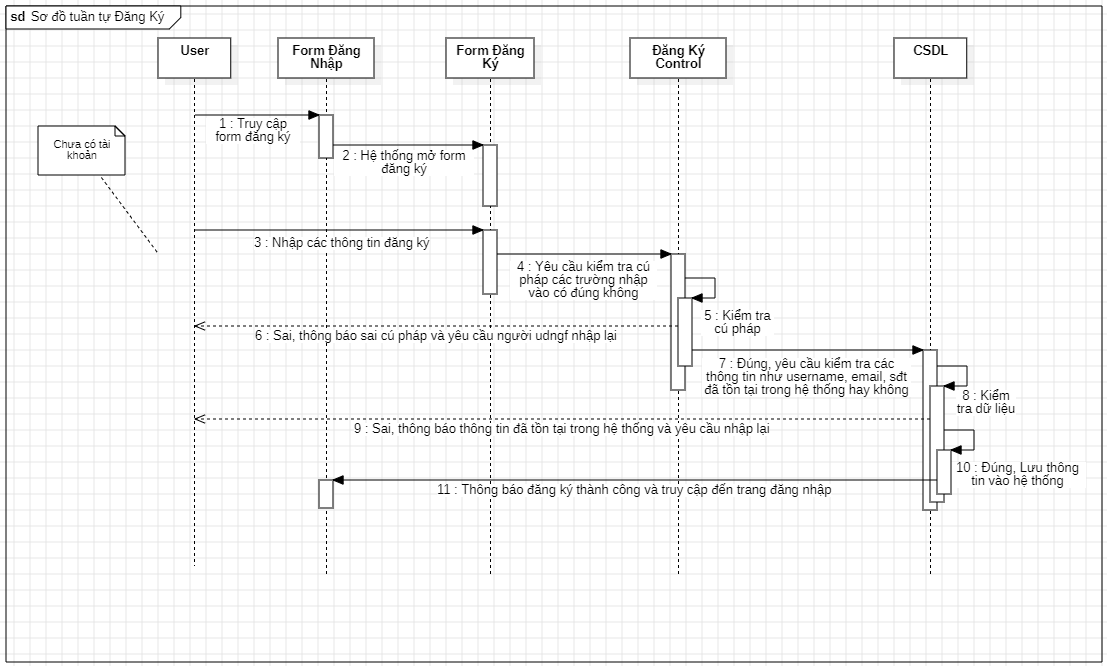
* + 1. **Bảng đặc tả xem tình trạng phòng và ghế ngồi:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Mã Usecase** | **UC021** | **Tên UseCase** | **Xem tình trạng phòng và ghế ngồi** |
| Tác Nhân | Quản lý, Nhân viên | | |
| Mô Tả | Xem tình trạng phòng và ghế ngồi của rạp phim. | | |
| Sự kiện kích hoạt | Chọn "Trang chủ" trên form chính, sau đó chọn "Tình trạng phòng". | | |
| Tiền điều kiện | Người dùng phải đăng nhập vào hệ thống. | | |
| Luồng sự kiện chính | STT | Thực hiện bởi | Hành động |
| 1 | Người dùng | Chọn chức năng " Tình trạng phòng ". |
| 2 | Hệ thống | Hiển thị danh sách các phòng và tình trạng của từng phòng. |
| 3 | Người dùng | Chọn một phòng để xem tình trạng ghế ngồi. |
| 4 | Hệ thống | Hiển thị tình trạng ghế ngồi của phòng đã chọn |
| Luồng sự kiện thay thế | STT | Thực hiện bởi | Hành động |
| 4a | Hệ thống | Sẽ không hiển thị nếu phòng đó chưa được quản lý thêm ghế vào |
| Hậu điều kiện | Tình trạng phòng và ghế ngồi được hiển thị thành công | | |

* 1. Sơ đồ tuần tự

1. ****Sơ đồ tuần tự chức năng đăng nhập (tác nhân: tất cả người dùng có tài khoản)

Hình 4. Sơ đồ tuần tự đăng nhập

1. ****Sơ đồ tuần tự chức năng đăng ký (tác nhân: tất cả người dùng)

Hình 5. Sơ đồ tuần tự chức năng đăng ký

1. Sơ đồ tuần tự chức năng quên mật khẩu (tác nhân: tất cả người dùng đã có tài **A diagram of a project

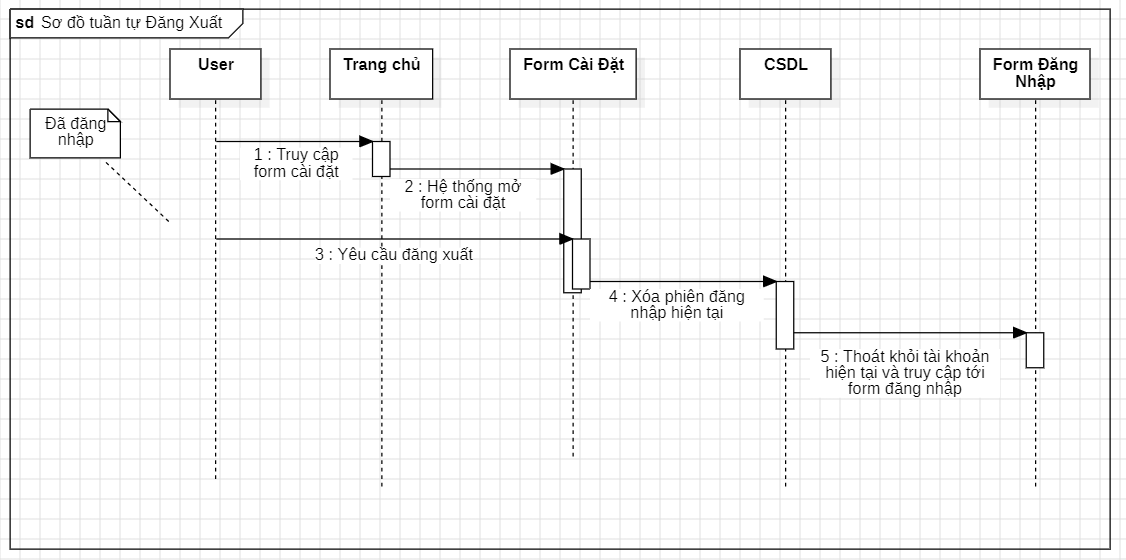
   Description automatically generated**khoản**)**

Hình 6. Sơ đồ tuần tự chức năng quên mật khẩu

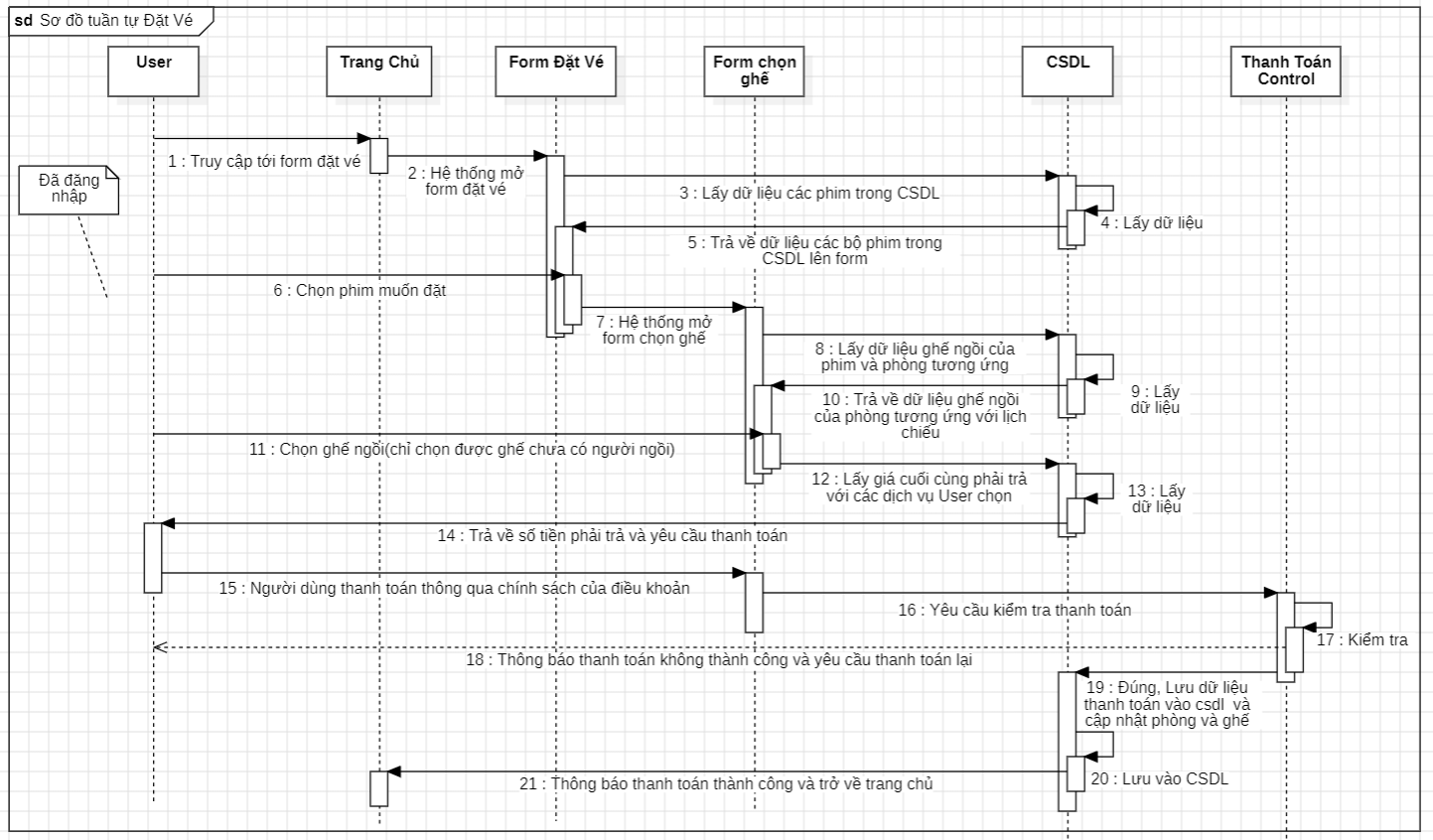
1. **A screenshot of a computer screen

   Description automatically generated**Sơ đồ tuần tự chức năng đổi mật khẩu (tác nhân: tất cả người dùng đã đăng nhập**)**

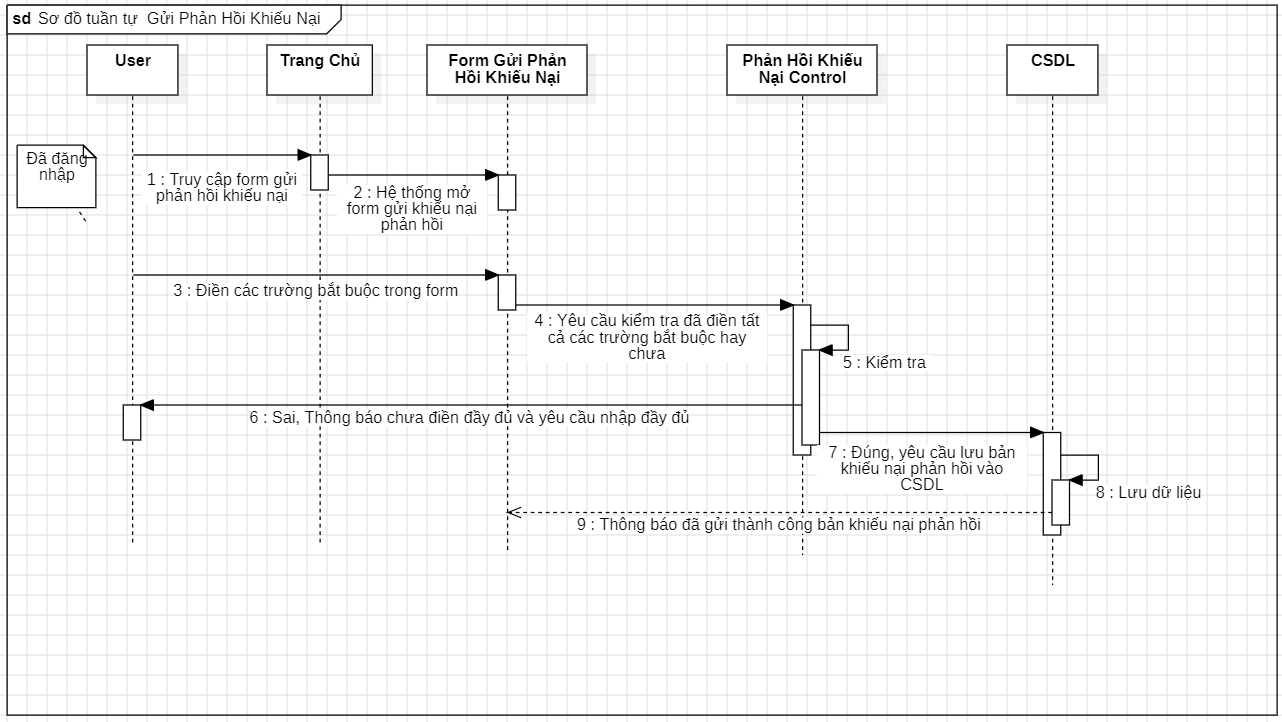
Hình 7. Sơ đồ tuần tự chức năng quên mật khẩu

1. ****Sơ đồ tuần tự chức năng đăng xuất (tác nhân: tất cả người dùng đã đăng nhập**)**

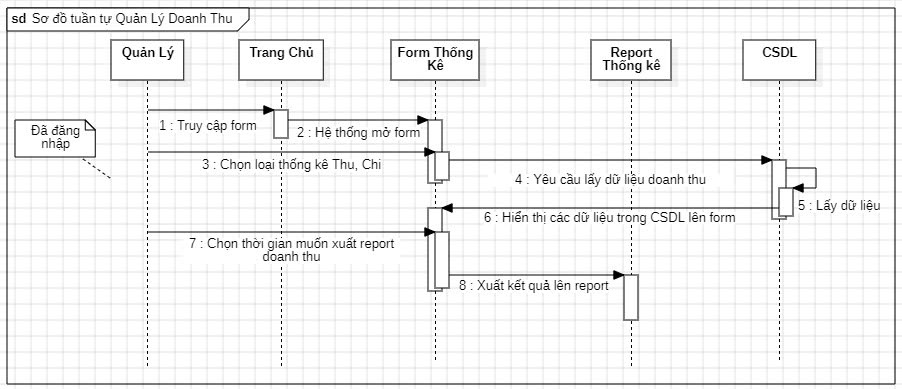
Hình 8. Sơ đồ tuần tự chức năng đăng xuất

1. ****Sơ đồ tuần tự chức năng đặt vé (tác nhân: tất cả người dùng đã đăng nhập**)**

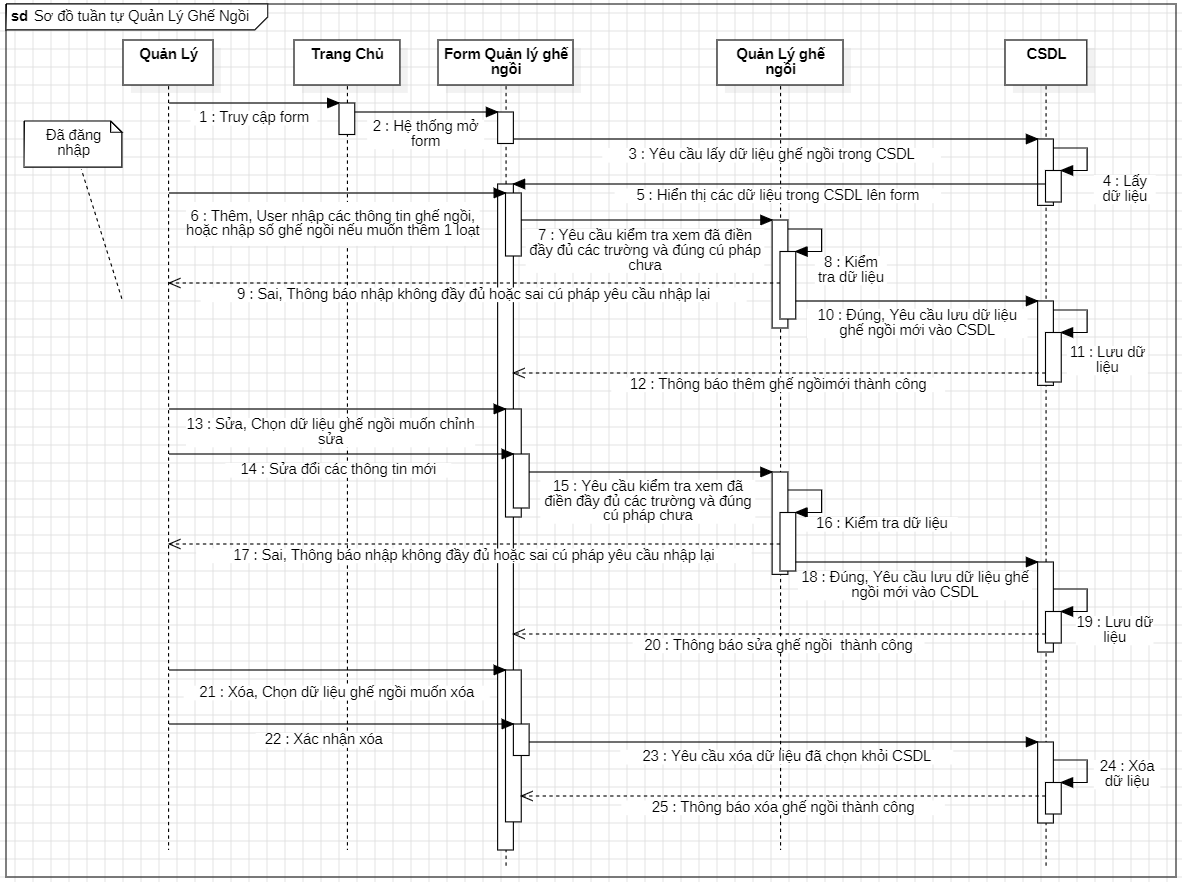
Hình 9. Sơ đồ tuần tự chức năng đặt vé

1. Sơ đồ tuần tự chức năng gửi phản hồi khiếu nại (tác nhân: tất cả người dùng đã đăng nhập**)**

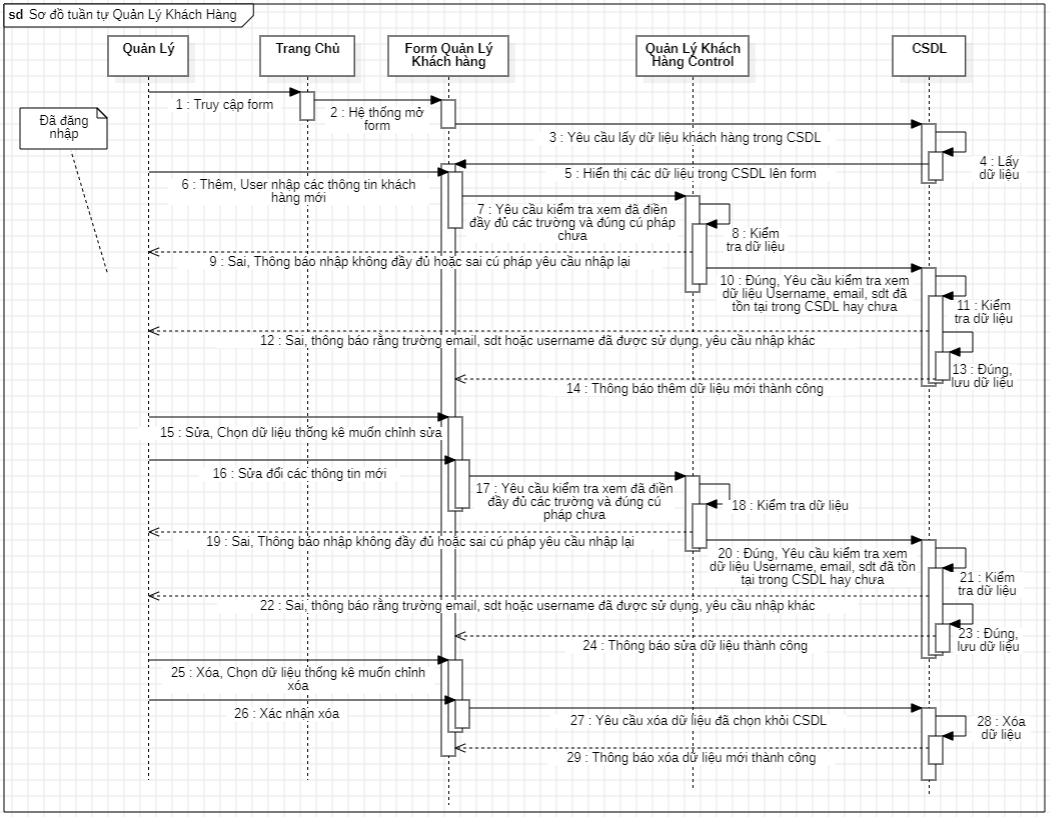
Hình 10. Sơ đồ tuần tự chức năng gửi phản hồi khiếu nại

1. Sơ đồ tuần tự chức năng quản lý doanh thu (tác nhân: chỉ người dùng quản lý ****đã đăng nhập**)**

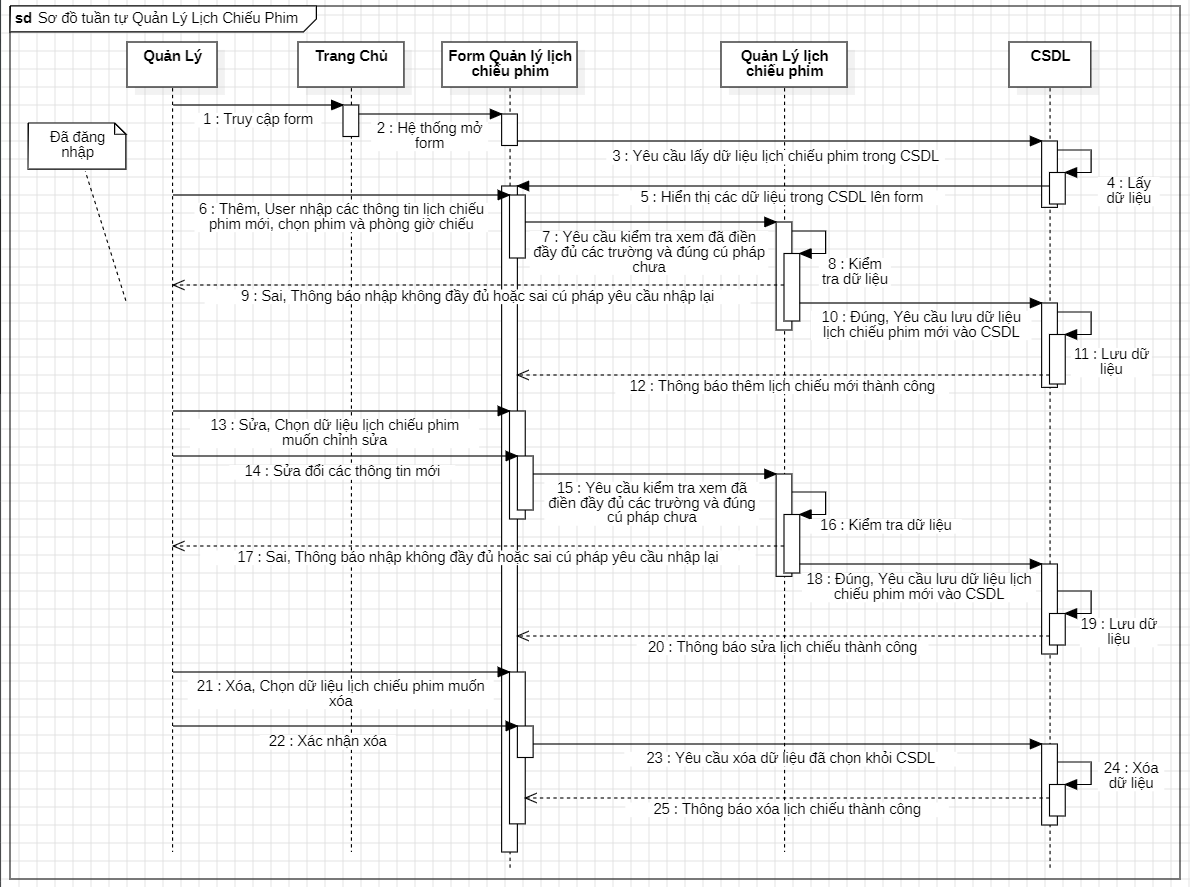
Hình 11. Sơ đồ tuần tự chức năng quản lý doanh thu

1. Sơ đồ tuần tự chức năng quản lý ghế ngồi (tác nhân: chỉ người dùng quản lý đã đăng nhập**)**

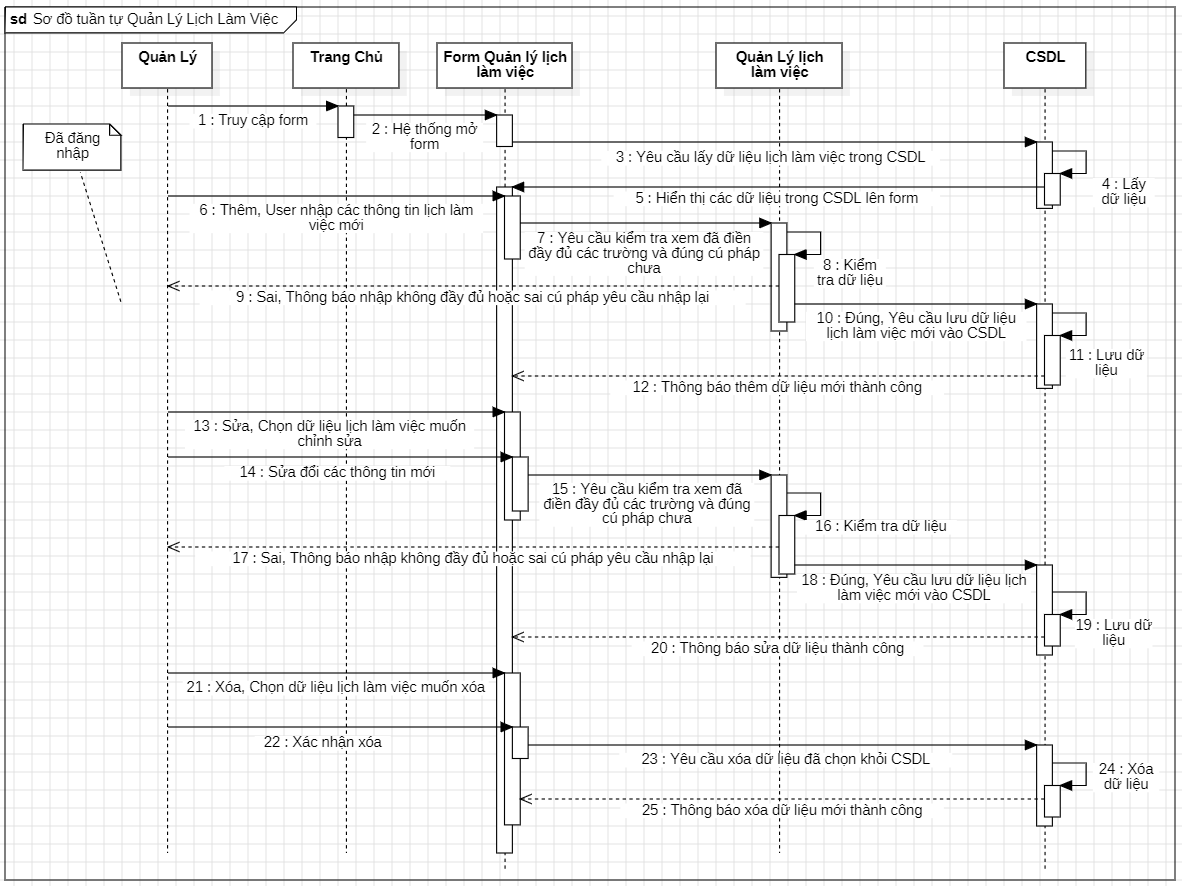
Hình 12. Sơ đồ tuần tự chức năng quản lý ghế ngồi

1. ****Sơ đồ tuần tự chức năng quản lý khách hàng (tác nhân: chỉ người dùng quản lý đã đăng nhập**)**

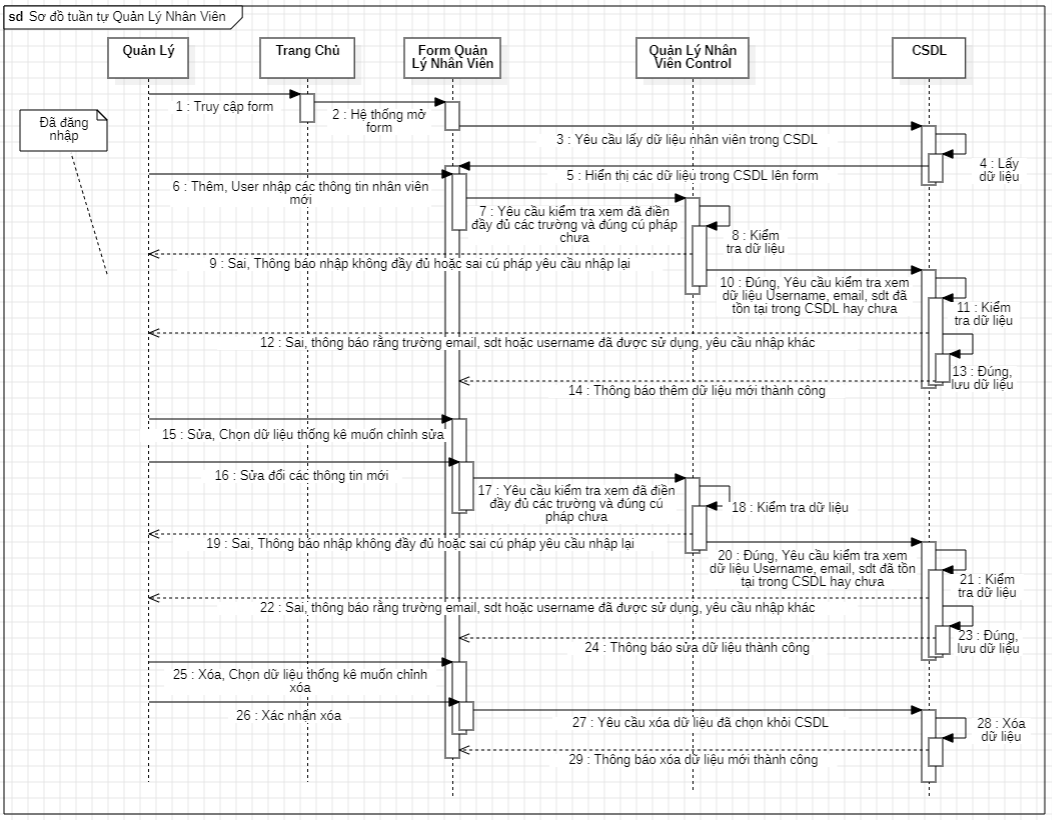
Hình 13. Sơ đồ tuần tự chức năng quản lý khách hàng

1. Sơ đồ tuần tự chức năng quản lý lịch chiếu phim (tác nhân: chỉ người dùng quản lý đã đăng nhập**)**

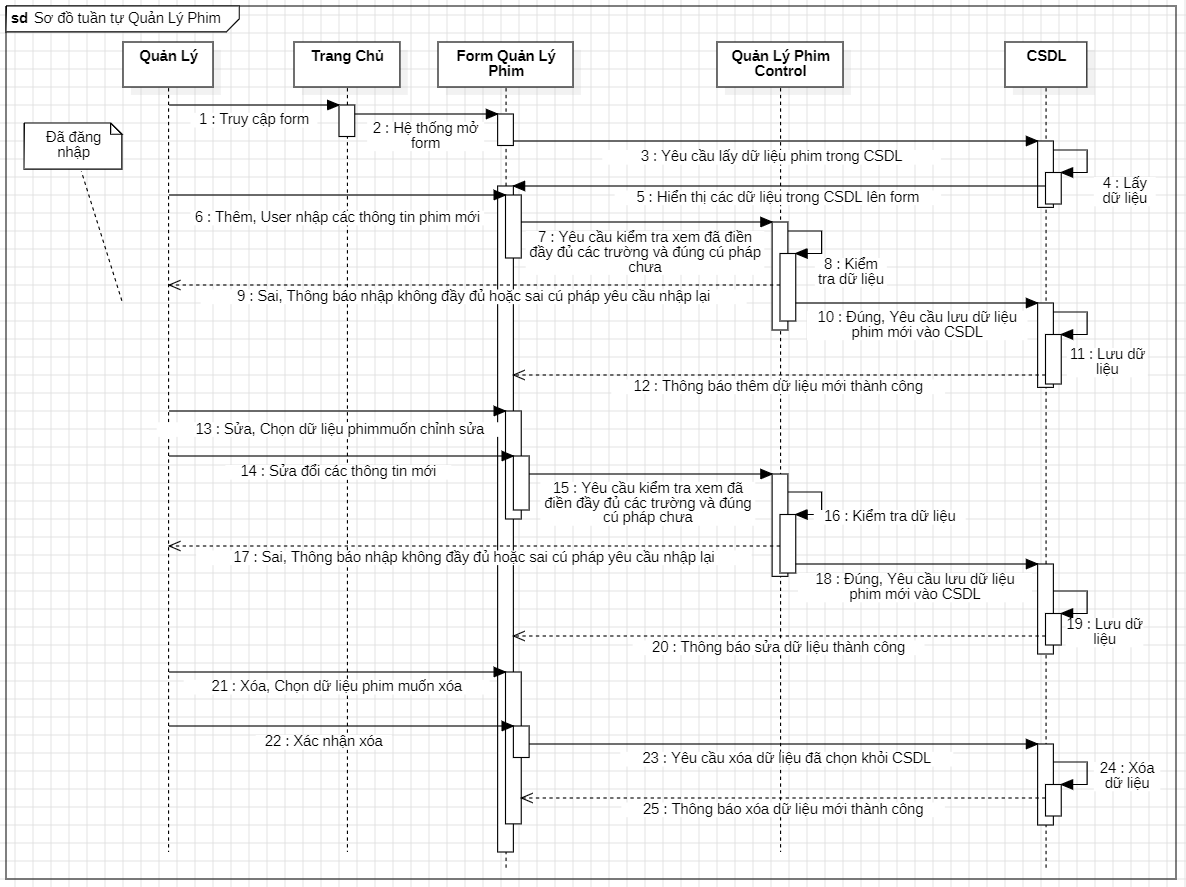
Hình 14. Sơ đồ tuần tự chức năng quản lý lịch chiếu phim

1. Sơ đồ tuần tự chức năng quản lý lịch làm việc (tác nhân: chỉ người dùng ****quản lý đã đăng nhập**)**

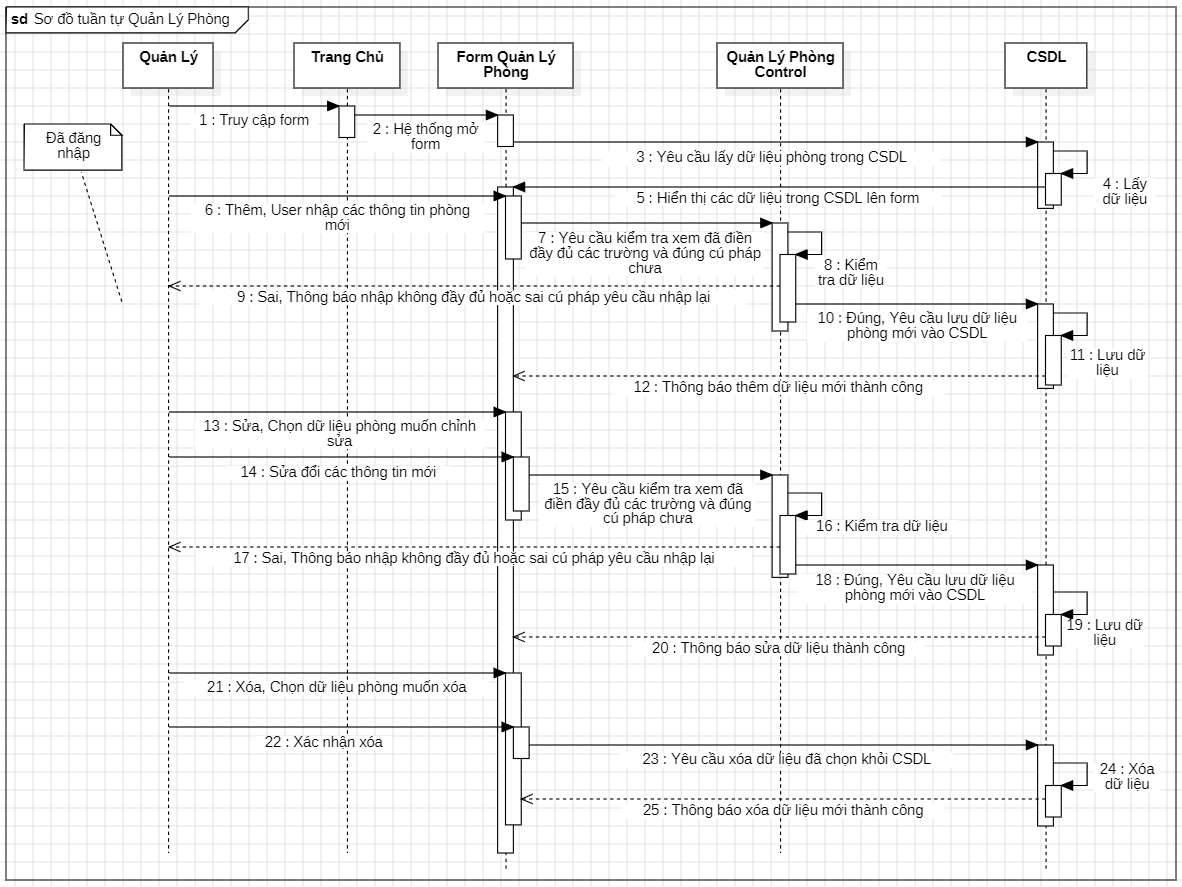
Hình 15. Sơ đồ tuần tự chức năng quản lý lịch làm việc

1. ****Sơ đồ tuần tự chức năng quản lý nhân viên (tác nhân: chỉ người dùng quản lý đã đăng nhập**)**

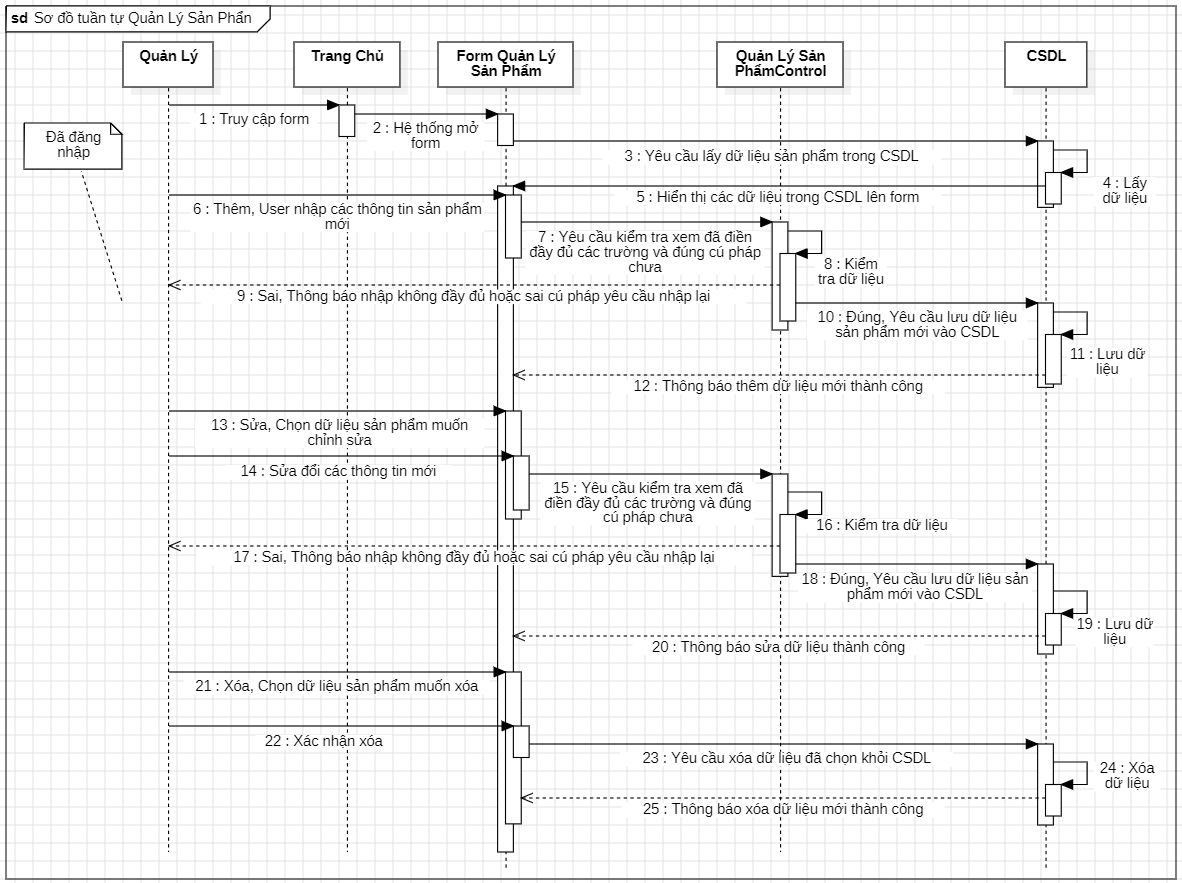
Hình 16. Sơ đồ tuần tự chức năng quản lý nhân viên

1. ****Sơ đồ tuần tự chức năng quản lý phim (tác nhân: chỉ người dùng quản lý đã đăng nhập**)**

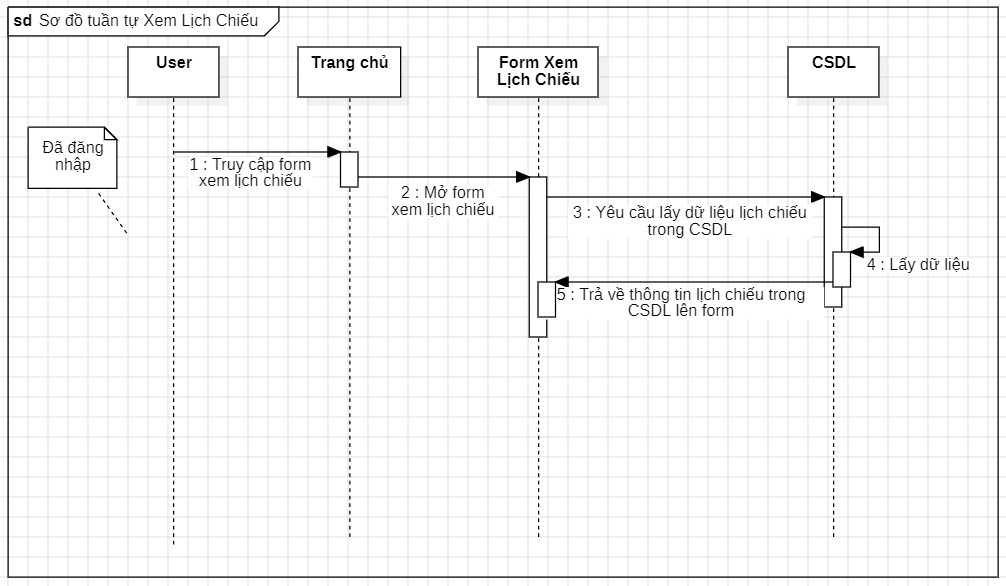
Hình 17. Sơ đồ tuần tự chức năng quản lý phim

1. ****Sơ đồ tuần tự chức năng quản lý phòng (tác nhân: chỉ người dùng quản lý đã đăng nhập**)**

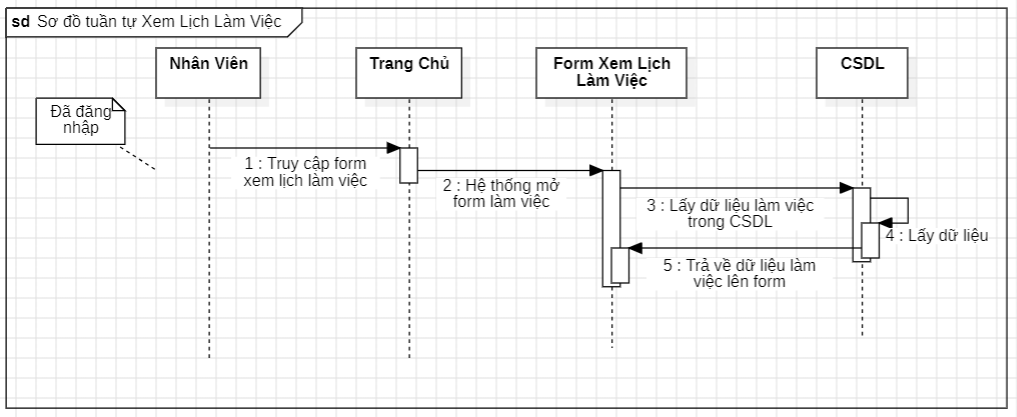
Hình 18. Sơ đồ tuần tự chức năng quản lý phòng

1. ****Sơ đồ tuần tự chức năng quản lý sản phẩm (tác nhân: chỉ người dùng quản lý đã đăng nhập**)**

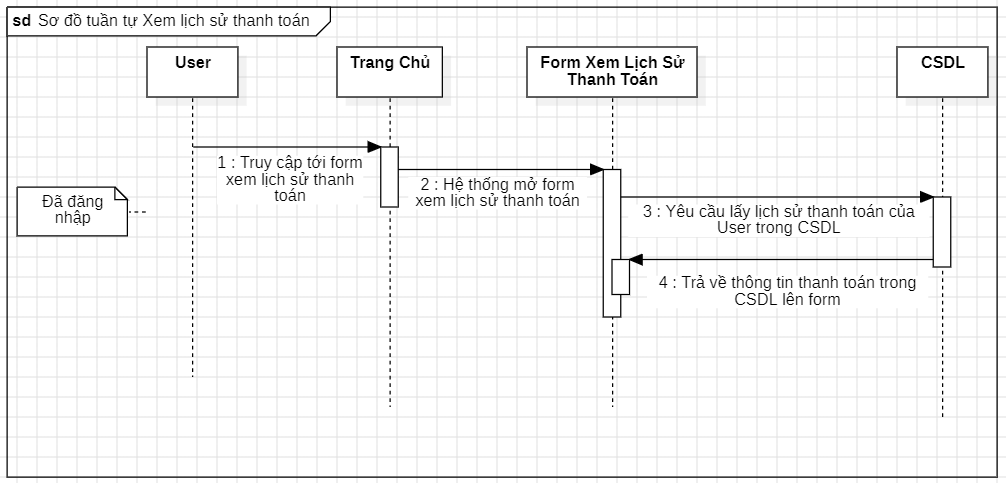
Hình 19. Sơ đồ tuần tự chức năng quản lý sản phẩm

1. ****Sơ đồ tuần tự chức năng xem lịch chiếu (tác nhân: người dùng đã đăng nhập**)**

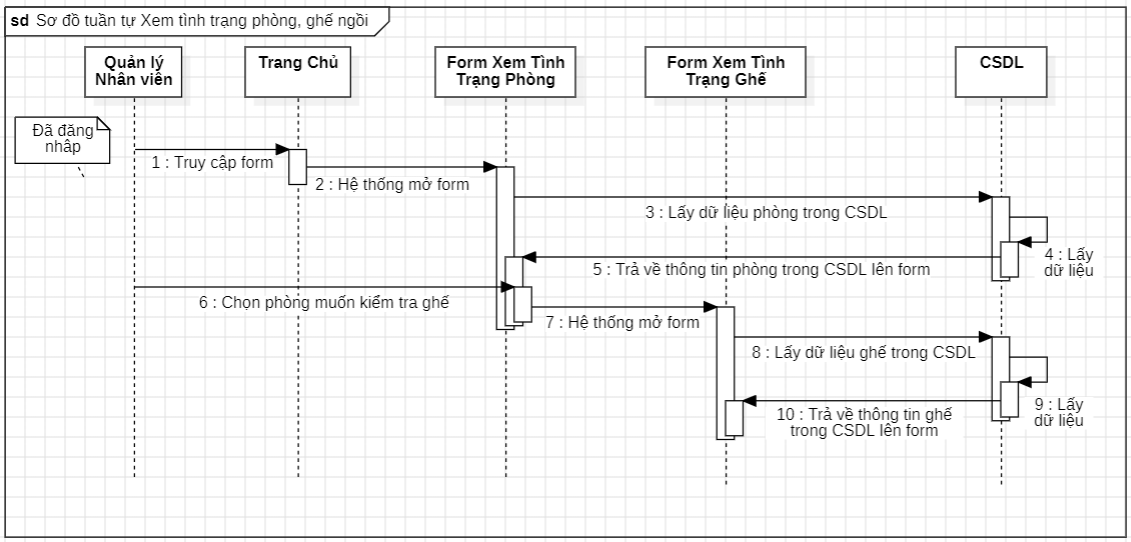
Hình 20. Sơ đồ tuần tự chức năng xem lịch chiếu

1. ****Sơ đồ tuần tự xem lịch làm việc (tác nhân: chỉ nhân viên đã đăng nhập)

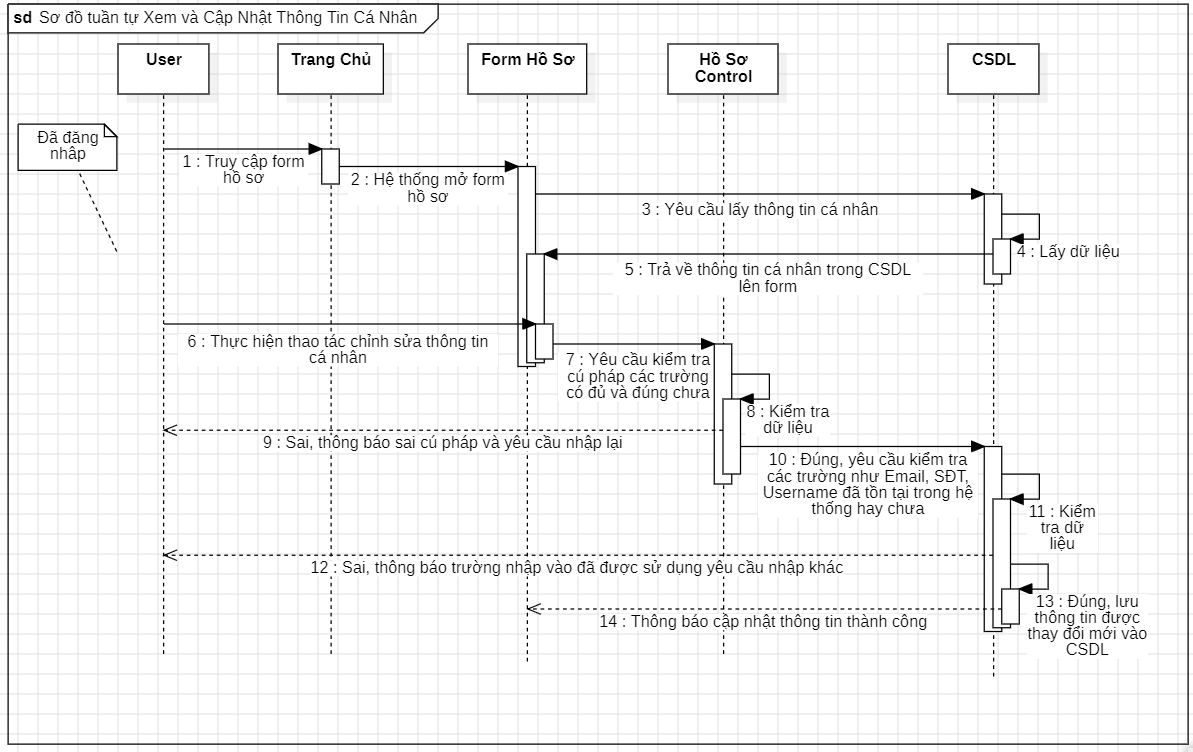
Hình 21. Sơ đồ tuần tự chức năng xem lịch làm việc

1. ****Sơ đồ tuần tự xem lịch sử thanh toán (tác nhân: người dùng đã đăng nhập)

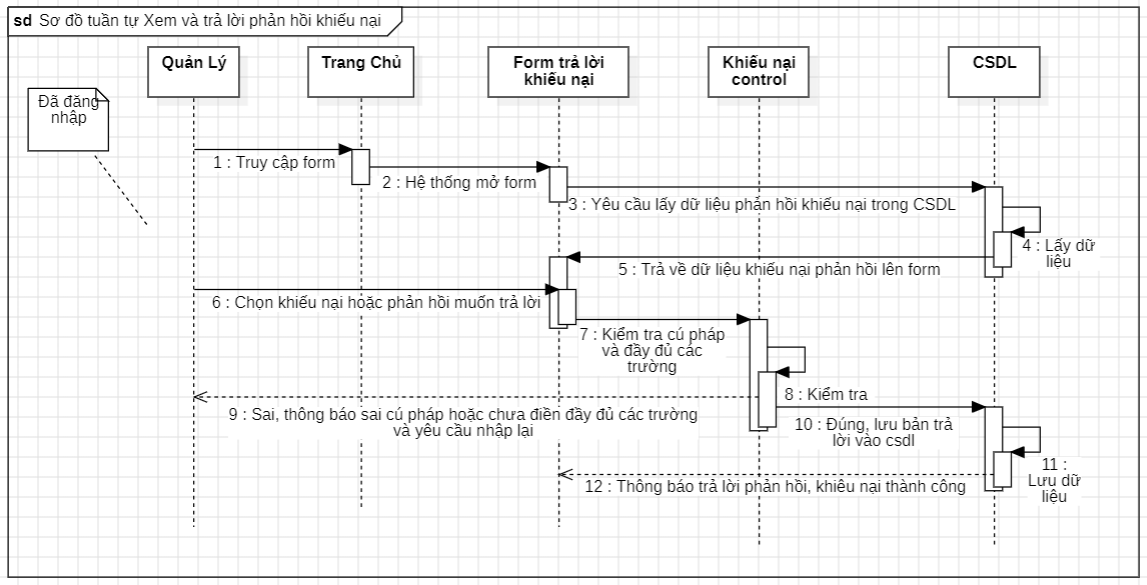
Hình 22. Sơ đồ tuần tự chức năng xem lịch sử thanh toán

1. ****Sơ đồ tuần tự xem tình trạng phòng ghế ngồi (tác nhân: chỉ người dùng quản lý đã đăng nhập)

Hình 23. Sơ đồ tuần tự chức năng xem tình trạng ghế ngồi

1. ****Sơ đồ tuần tự xem và cập nhật thông tin cá nhân (tác nhân: chỉ người dùng đã đăng nhập)

Hình 24. Sơ đồ tuần tự chức năng xem và cập nhật thông tin cá nhân

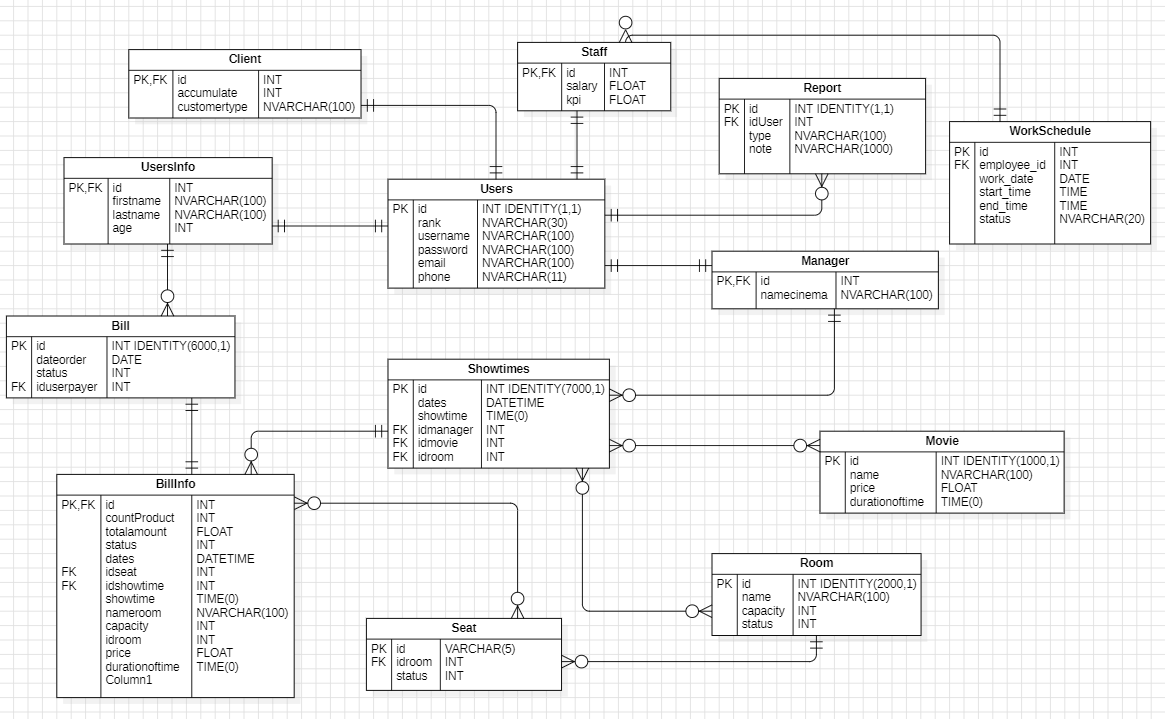
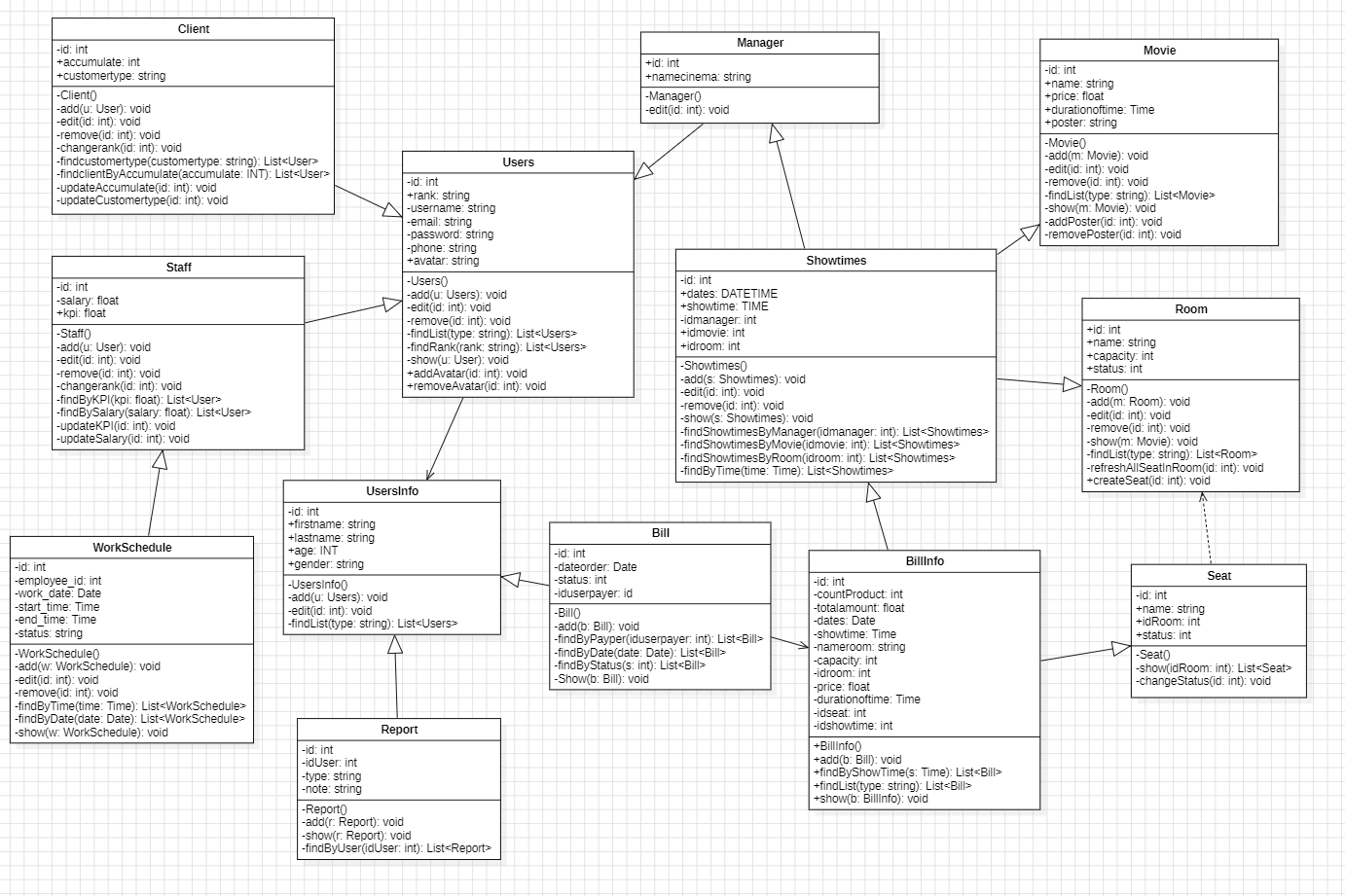
1. ****Sơ đồ tuần tự xem và trả lời phản hồi khiếu nại (tác nhân: chỉ người dùng quản lý đã đăng nhập)

Hình 25. Sơ đồ tuần tự chức năng xem và trả lời phản hồi khiếu nại

* 1. Kết luận

Phần mềm quản lý rạp chiếu phim được xây dựng nhằm đáp ứng nhu cầu quản lý toàn diện các hoạt động của rạp. Hệ thống này sẽ giúp tối ưu hóa quy trình làm việc, nâng cao hiệu suất và chất lượng dịch vụ. Đồng thời, phần mềm cũng cung cấp các báo cáo thống kê chi tiết để hỗ trợ quản lý đưa ra các quyết định chính xác.

# **CHƯƠNG 4: CƠ SỞ DỮ LIỆU**

* 1. Sơ đồ ERD Diagram
  2. Sơ đồ lớp Class Diagram
  3. Các Thực Thể Và Các Thuộc Tính

Bảng Users

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| id | INT IDENTITY(1,1) | PRIMARY KEY | Mã tài khoản |
| rank | NVARCHAR(30) | Not Null | Chức vụ |
| username | NVARCHAR(100) | Not Null | Tên đăng nhập |
| password | NVARCHAR(100) | Not Null | Mật khẩu |
| email | NVARCHAR(100) | Not Null | Email |
| phone | NVARCHAR(11) | Not Null | Số điện thoại |

Bảng UsersInfo

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| id | INT | PRIMARY KEY | Mã người dùng |
| firstname | NVARCHAR(100) | NOT NULL | Tên |
| lastname | NVARCHAR(100) | NOT NULL | Họ |
| age | INT | NOT NULL | Tuổi |
| gender | NVARCHAR(3) | NOT NULL | Giới tính |
| avatar | VARBINARY(MAX) | NULL | Ảnh đại diện |

Bảng Manager

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| Id | INT | PRIMARY KEY | Mã người dùng |
| namecinema | NVARCHAR(100) | NOT NULL | Tên rạp phim sở hữu |

Bảng Client

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| id | INT | PRIMARY KEY | Mã người dùng |
| accumulate | INT | NOT NULL | Điểm tích lũy |
| customertype | NVARCHAR(100) | NOT NULL | Loại khách hàng |

Bảng Staff

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| id | INT | PRIMARY KEY | Mã nhân viên |
| salary | FLOAT | NOT NULL | Lương |
| kpi | FLOAT | NOT NULL | Đánh giá hiệu suất |

Bảng Movie

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| id | INT IDENTITY(1000,1) | PRIMARY KEY | Mã phim |
| name | NVARCHAR(100) | NOT NULL | Tên phim |
| price | FLOAT | NOT NULL | Giá phim |
| durationoftime | INT | NOT NULL | Thời lượng |
| poster | VARBINARY(MAX) | NULL | Ảnh poster phim |

Bảng Room

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| id | INT IDENTITY(2000,1) | PRIMARY KEY | Mã phòng |
| name | NVARCHAR(100) | NOT NULL | Tên phòng |
| capacity | INT | NOT NULL | Sức chứa |
| status | INT | NOT NULL | Trạng thái (0: Trống, 1: Bận) |

Bảng Seat

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| id | INT IDENTITY(3000,1) | PRIMARY KEY | Mã ghế |
| name | NVARCHAR(100) | NOT NULL | Tên ghế |
| idroom | INT | FOREIGN KEY | Mã phòng |
| status | INT | NOT NULL | Trạng thái (0: Trống, 1: Đã đặt, 2: Bị khóa) |

Bảng Showtimes

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| id | INT IDENTITY(7000,1) | PRIMARY KEY | Mã lịch chiếu |
| dates | DATETIME | NOT NULL | Ngày |
| showtime | TIME(0) | NOT NULL | Giờ chiếu |
| idmanager | INT | FOREIGN KEY | Mã người tạo lịch |
| idmovie | INT | FOREIGN KEY | Mã phim |
| idroom | INT | FOREIGN KEY | Mã phòng |

Bảng Bill

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| id | INT IDENTITY(5000,1) | PRIMARY KEY | Mã hóa đơn |
| dateorder | DATE | NOT NULL | Ngày thanh toán |
| status | INT | NOT NULL | Trạng thái (0: chưa thanh toán, 1: đã thanh toán) |
| iduserpayer | INT | FOREIGN KEY | Mã người dùng thanh toán |

Bảng BillInfo

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| id | INT | PRIMARY KEY, FOREIGN KEY | Mã thông tin hóa đơn |
| countProduct | INT | NOT NULL | Số lượng ghế đặt |
| totalamount | FLOAT | NOT NULL | Tổng tiền |
| status | INT | NOT NULL | Trạng thái (0: chưa thanh toán, 1: đã thanh toán) |
| dates | DATETIME | NULL | Ngày tháng năm của phim |
| showtime | TIME(0) | NULL | Giờ chiếu phim |
| nameroom | NVARCHAR(100) | NULL | Tên phòng chiếu |
| capacity | INT | NULL | Sức chứa |
| idroom | INT | NULL | Mã sản phẩm |
| price | FLOAT | NULL | Giá ghế ngồi |
| durationoftime | TIME(0) | NULL | Mã lịch chiếu |
| idseat | VARCHAR(100) | NULL, FOREIGN KEY | Mã ghế ngồi |
| idshowtime | INT | NULL, FOREIGN KEY | Mã lịch chiếu |
| movieName | NVARCHAR(100) | NULL | Tên phim |

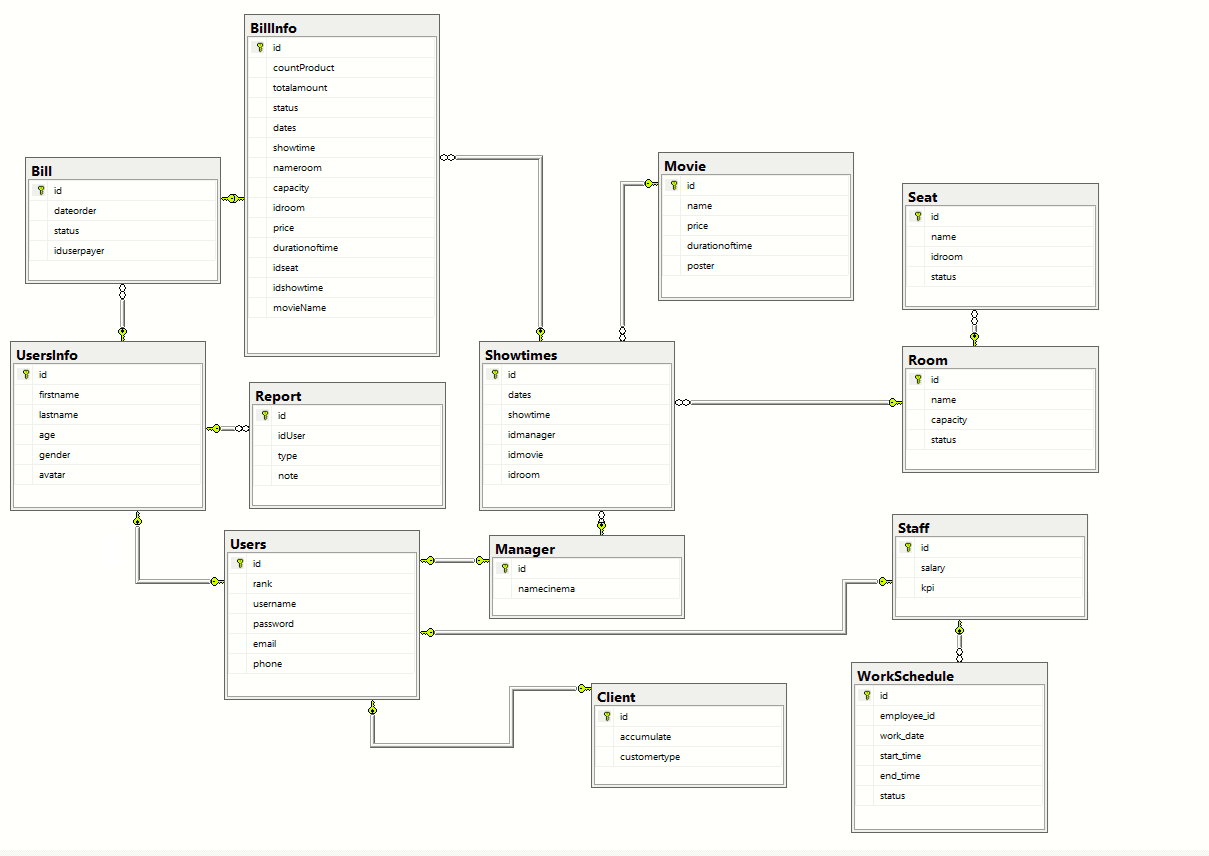
Bảng Report

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| id | INT IDENTITY(1,1) | PRIMARY KEY | Mã khiếu nại |
| idUser | INT | NOT NULL | Mã người gửi khiếu nại |
| type | NVARCHAR(100) | NOT NULL | Loại khiếu nại |
| note | NVARCHAR(1000) | FOREIGN KEY | Ghi chú |

Bảng WorkSchedule

|  |  |  |  |
| --- | --- | --- | --- |
| **Thuộc tính** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| id | INT IDENTITY(1,1) | PRIMARY KEY | Mã số |
| employee\_id | INT | NOT NULL | Mã nhân viên |
| work\_date | DATE | NOT NULL | Ngày làm việc |
| start\_time | TIME(0) | NOT NULL | Giờ làm việc |
| end\_time | TIME(0) | NOT NULL | Giờ kết thúc làm việc |
| status | NVARCHAR(20) | NOT NULL | Kích hoạt (0: Không hoạt động, 1: Hoạt động) |

1. Sơ đồ quan hệ



* 1. Cập nhật ràng buộc

ALTER TABLE [dbo].[Bill] WITH CHECK ADD FOREIGN KEY([iduserpayer])

REFERENCES [dbo].[UsersInfo] ([id])

ON DELETE SET NULL

GO

ALTER TABLE [dbo].[BillInfo] WITH CHECK ADD FOREIGN KEY([id])

REFERENCES [dbo].[Bill] ([id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[BillInfo] WITH CHECK ADD FOREIGN KEY([idshowtime])

REFERENCES [dbo].[Showtimes] ([id])

ON DELETE SET NULL

GO

ALTER TABLE [dbo].[Client] WITH CHECK ADD FOREIGN KEY([id])

REFERENCES [dbo].[Users] ([id])

GO

ALTER TABLE [dbo].[Manager] WITH CHECK ADD CONSTRAINT [FK\_\_Manager\_\_id\_\_7C4F7684] FOREIGN KEY([id])

REFERENCES [dbo].[Users] ([id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Manager] CHECK CONSTRAINT [FK\_\_Manager\_\_id\_\_7C4F7684]

GO

ALTER TABLE [dbo].[Report] WITH CHECK ADD CONSTRAINT [FK\_\_Report\_\_idUser\_\_7E37BEF6] FOREIGN KEY([idUser])

REFERENCES [dbo].[UsersInfo] ([id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Report] CHECK CONSTRAINT [FK\_\_Report\_\_idUser\_\_7E37BEF6]

GO

ALTER TABLE [dbo].[Seat] WITH CHECK ADD FOREIGN KEY([idroom])

REFERENCES [dbo].[Room] ([id])

ON DELETE SET NULL

GO

ALTER TABLE [dbo].[Showtimes] WITH CHECK ADD FOREIGN KEY([idmanager])

REFERENCES [dbo].[Manager] ([id])

ON DELETE SET NULL

GO

ALTER TABLE [dbo].[Showtimes] WITH CHECK ADD FOREIGN KEY([idmovie])

REFERENCES [dbo].[Movie] ([id])

ON DELETE SET NULL

GO

ALTER TABLE [dbo].[Showtimes] WITH CHECK ADD FOREIGN KEY([idroom])

REFERENCES [dbo].[Room] ([id])

ON DELETE SET NULL

GO

ALTER TABLE [dbo].[Staff] WITH CHECK ADD FOREIGN KEY([id])

REFERENCES [dbo].[Users] ([id])

GO

ALTER TABLE [dbo].[Staff] WITH CHECK ADD CONSTRAINT [FK\_Staff\_Users] FOREIGN KEY([id])

REFERENCES [dbo].[Users] ([id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Staff] CHECK CONSTRAINT [FK\_Staff\_Users]

GO

ALTER TABLE [dbo].[UsersInfo] WITH CHECK ADD FOREIGN KEY([id])

REFERENCES [dbo].[Users] ([id])

GO

ALTER TABLE [dbo].[UsersInfo] WITH CHECK ADD CONSTRAINT [FK\_UsersInfo\_Users] FOREIGN KEY([id])

REFERENCES [dbo].[Users] ([id])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[UsersInfo] CHECK CONSTRAINT [FK\_UsersInfo\_Users]

GO

ALTER TABLE [dbo].[WorkSchedule] WITH CHECK ADD FOREIGN KEY([employee\_id])

REFERENCES [dbo].[Staff] ([id])

GO

ALTER TABLE [dbo].[WorkSchedule] WITH CHECK ADD CONSTRAINT [FK\_WorkSchedule\_Staff] FOREIGN KEY([employee\_id])

REFERENCES [dbo].[Staff] ([id])

ON DELETE CASCADE

# **CHƯƠNG 5: THIẾT KẾ FORM**

* 1. Form đăng nhập
* Tên form: User\_LoginForm
* Chức năng: Form đăng nhập cho phép người dùng đăng nhập vào chương trình dưới 3 dạng quyền Manager, Staff, Client.
* Mục tiêu: Thực hiện việc đăng nhập của người dùng bằng cách xác thực tên đăng nhập và mật khẩu với cơ sở dữ liệu. Nếu thông tin hợp lệ, nó tạo một phiên đăng nhập, chuyển đến Form chính của ứng dụng và ẩn Form đăng nhập. Nếu thông tin không hợp lệ hoặc có lỗi xảy ra, nó hiển thị thông báo lỗi cho người dùng.

A screenshot of a computer

Description automatically generated

private void btn\_Dangnhap\_Click(object sender, EventArgs e)

{

string username = txtbox\_username.Text;

string password = txtbox\_password.Text;

SqlConnection conn = Hook.Connection; // Kết nối SQL

try {

conn.Open();

string query = @"

SELECT Users.id, Users.username, Users.rank, UsersInfo.firstname, UsersInfo.lastname, Users.email, Users.phone, UsersInfo.age, UsersInfo.gender, UsersInfo.avatar

FROM Users

INNER JOIN UsersInfo ON Users.id = UsersInfo.id

WHERE Users.username = @username AND Users.password = @password";

using (SqlCommand cmd = new SqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@username", username);

cmd.Parameters.AddWithValue("@password", password);

using (SqlDataReader reader = cmd.ExecuteReader())

{

if (reader.HasRows)

{

reader.Read();

int userId = reader.GetInt32(0);

string user = reader.GetString(1);

string rank = reader.GetString(2);

string firstName = reader.GetString(3);

string lastName = reader.GetString(4);

string email = reader.GetString(5);

string phone = reader.GetString(6);

int age = reader.GetInt32(7);

string gender = reader.GetString(8);

byte[] avatar = reader.IsDBNull(9) ? null : (byte[])reader[9]; // Avatar dạng byte[]

MessageBox.Show("Đăng nhập thành công!", "Thành công", MessageBoxButtons.OK, MessageBoxIcon.Information);

ClearInput();

//Gán session cho chương trình

SessionManager.CreateSession(userId, user, firstName, lastName, rank, email, phone, age, gender, avatar);

this.Hide();

User\_Home home = new User\_Home();

home.Show();

home.FormClosed += (s, args) => this.Show();

}

else

{

CheckError();

}

}

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

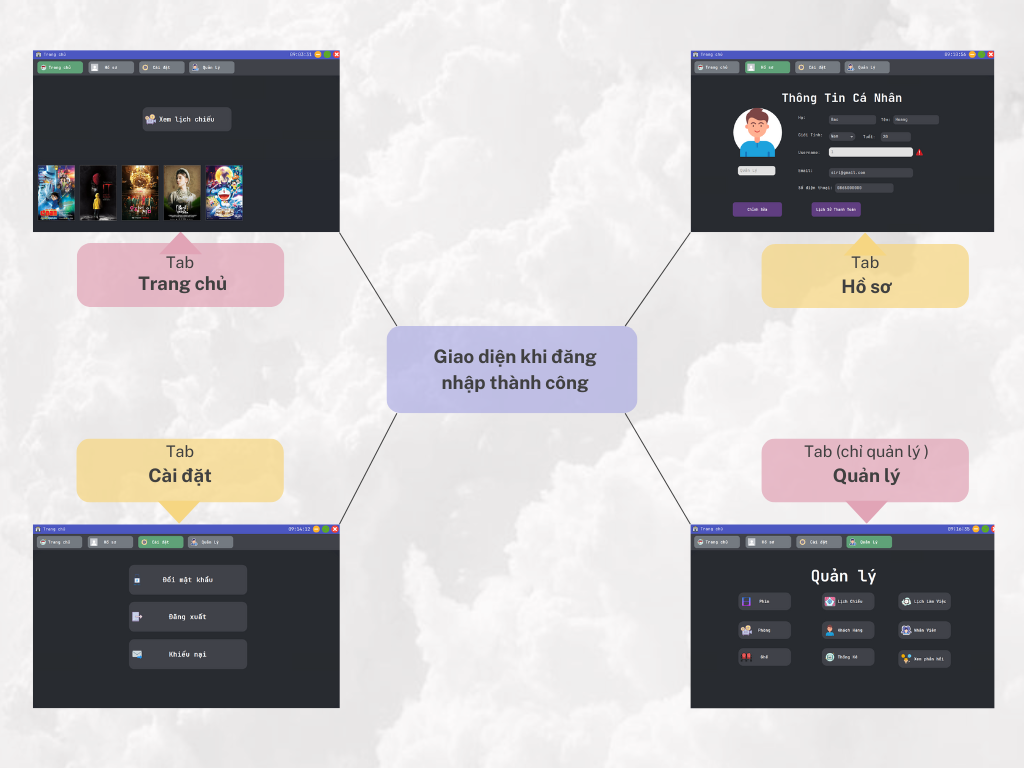
finally

{

conn.Close();

}

}

Giao diện khi đăng nhập thành công:

* **Tab "Trang chủ":**  
  Tab này cung cấp truy cập nhanh đến form xem lịch chiếu phim và chọn ghế để mua vé.
* **Tab "Hồ sơ":**  
  Tab Hồ sơ cho phép người dùng quản lý thông tin cá nhân. Người dùng có thể xem và cập nhật chi tiết cá nhân, kiểm tra lịch sử thanh toán, và nếu là nhân viên, họ có thể xem lịch làm việc của mình.
* **Tab "Cài đặt":**  
  Trong tab Cài đặt, người dùng có ba tùy chọn chính:
  + **Đổi mật khẩu:** Cập nhật thông tin đăng nhập.
  + **Đăng xuất:** Đăng xuất khỏi ứng dụng.
  + **Khiếu nại:** Gửi phản hồi hoặc khiếu nại.
* **Tab "Quản lý":**  
  Tab này dành riêng cho việc quản lý toàn bộ hoạt động của rạp chiếu phim. Nó bao gồm quản lý cơ sở vật chất, ghế ngồi, lịch chiếu phim, lịch làm việc của nhân viên, xem doanh thu, phản hồi khiếu nại của khách hàng, và quản lý hồ sơ khách hàng cùng nhân viên.
  1. Form đăng ký
* Tên form: User\_RegisterForm
* Chức năng: Form cho phép người dùng đăng ký tài khoản vào chương trình.
* Mục tiêu: Thực hiện việc đăng ký của người dùng bằng cách đọc dữ liệu từ các control. Nếu thông tin hợp lệ, chương trình chạy lệnh đăng ký tài khoản và thông báo nội dung, sau đó chuyển đến Form đăng nhập của ứng dụng và ẩn Form đăng ký. Nếu thông tin không hợp lệ hoặc có lỗi xảy ra, nó hiển thị thông báo lỗi cho người dùng. Tại đây còn có UserControl “Điều khoản”.

A screenshot of a computer

Description automatically generated

private void btn\_dangky\_Click(object sender, EventArgs e)

{

SqlConnection conn = Hook.Connection; // Kết nối SQL

{

if (!ValidateInputs()) //Hàm kiểm tra validator

{

return;

}

try

{

conn.Open();

// Kiểm tra email, username, phone nếu có tồn tại

SqlCommand checkCmd = new SqlCommand(@"SELECT COUNT(\*) FROM [dbo].[Users]

WHERE email = @email OR username = @username OR phone = @phone", conn);

checkCmd.Parameters.AddWithValue("@email", email);

checkCmd.Parameters.AddWithValue("@username", username);

checkCmd.Parameters.AddWithValue("@phone", phone);

int exists = (int)checkCmd.ExecuteScalar();

if (exists > 0)

{

MessageBox.Show("Email, tên đăng nhập hoặc số điện thoại đã tồn tại trong hệ thống. Vui lòng kiểm tra lại.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

// Thực hiện tạo tài khoản

using (SqlTransaction register = conn.BeginTransaction())

{

// Nhét thông tin vào Users

SqlCommand cmdUsers = new SqlCommand(@"INSERT INTO [dbo].[Users]

([username], [password], [email], [phone]) VALUES

(@username, @password, @email, @phone); SELECT SCOPE\_IDENTITY();", conn, register);

cmdUsers.Parameters.AddWithValue("@username", username);

cmdUsers.Parameters.AddWithValue("@password", password);

cmdUsers.Parameters.AddWithValue("@email", email);

cmdUsers.Parameters.AddWithValue("@phone", phone);

int userId = Convert.ToInt32(cmdUsers.ExecuteScalar());

// Nhét thông tin vào UsersInfo

SqlCommand cmdUsersInfo = new SqlCommand(@"INSERT INTO [dbo].[UsersInfo]

([id], [firstname], [lastname], [age], [gender]) VALUES

(@id, @firstname, @lastname, @age, @gender)", conn, register);

cmdUsersInfo.Parameters.AddWithValue("@id", userId);

cmdUsersInfo.Parameters.AddWithValue("@firstname", firstname);

cmdUsersInfo.Parameters.AddWithValue("@lastname", lastname);

cmdUsersInfo.Parameters.AddWithValue("@age", age);

cmdUsersInfo.Parameters.AddWithValue("@gender", gender);

cmdUsersInfo.ExecuteNonQuery();

register.Commit();

}

if (MessageBox.Show("Đăng ký tài khoản thành công!\nChọn Yes để quay về đăng nhập.", "Hoàn tất đăng ký", MessageBoxButtons.YesNo, MessageBoxIcon.Information) == DialogResult.Yes)

{

Close();

}

}

catch (SqlException ex) when (ex.Number == 2627)

{

MessageBox.Show("Tên đăng nhập đã tồn tại. Vui lòng chọn tên đăng nhập khác.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

catch (Exception ex)

{

MessageBox.Show("Đã xảy ra lỗi: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

finally

{

conn.Close();

}

}

}

* 1. Form quên mật khẩu
* Tên form: UC\_ForgetPasswordForm
* Chức năng: Form cho phép người dùng đã đăng ký mà quên mật khẩu thì có thể sử dụng chức năng này để đặt lại mật khẩu.
* Mục tiêu: Thực hiện việc của người dùng bằng cách đọc dữ liệu từ các control. Nếu thông tin hợp lệ, chương trình chạy lệnh yêu cầu nhập mã OTP theo yêu cầu, nếu nhập đúng thì chuyển đến Form đặt lại mật khẩu.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

// Hàm đổi mật khẩu này nằm trong Form Quên mật khẩu

**private** **void** **doimatkhau\_Click**(**object** sender, EventArgs e)

{

**if** (!ValidatePassword())

**return**;

**string** newPassword = txt\_password.Text;

**string** infoUser = txt\_info.Text;

**string** columnName = GetColumnName();

**if** (columnName == **null**)

{

ShowError("Vui lòng chọn phương thức xác minh!");

**return**;

}

**try**

{

**using** (SqlConnection conn = **new** SqlConnection(Hook.connectionString))

{

conn.Open();

**string** query = $"UPDATE Users SET password = @NewPassword WHERE {columnName} = @InfoUser";

**using** (SqlCommand cmd = **new** SqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@NewPassword", newPassword);

cmd.Parameters.AddWithValue("@InfoUser", infoUser);

**int** rowsAffected = cmd.ExecuteNonQuery();

**if** (rowsAffected > **0**)

{

MessageBox.Show("Đổi mật khẩu thành công!", "Thành công", MessageBoxButtons.OK, MessageBoxIcon.Information);

Close();

}

**else**

{

ShowError("Có lỗi xảy ra khi đổi mật khẩu. Có thể không tìm thấy người dùng với thông tin cung cấp.");

}

}

}

}

**catch** (Exception ex)

{

ShowError($"Lỗi: {ex.Message}");

}

}

* 1. Form trang chủ
* Tên form: User\_Home
* Chức năng: Hiển thị các UserControl theo tùy chọn.
* Nếu ở “Trang chủ” => hiển thị UC\_Home vào panel background.
* Nếu tài khoản có Rank = “Quản lý” thì hiển thị nút chức năng Quản Lý.
* A screenshot of a computer

  Description automatically generatedNếu có **nhiều** poster phim thì ảnh poster sẽ di chuyển từ trái sang phải đến cuối và chạy ngược lại.

**private** **void** **Check\_QuanLy**()

{

btn\_management.Visible = SessionManager.IsManager();

}

// Lưu ý: Hàm dưới đây ở Class SessionManager.cs

**public** **static** **bool** **IsManager**()

{

**return** Rank == "Quản Lý";

}

**private** Timer scrollTimer;

**private** **int** scrollSpeed = **1**; // Tốc độ cuộn

**private** **bool** scrollingRight = **true**; // Biến để theo dõi hướng cuộn

**public** **User\_Home**()

{

// Khởi tạo Timer cho cuộn hình ảnh

scrollTimer = **new** Timer

{

Interval = **30** // Thay đổi thời gian giữa các lần cuộn

};

scrollTimer.Tick += ScrollTimer\_Tick;

DisplayMoviePosters();

isHomeSelected = **true**;

}

**private** **void** **InitializePosterPanel**()

{

**if** (posterPanel == **null**)

{

posterPanel = **new** FlowLayoutPanel

{

AutoScroll = **true**,

FlowDirection = FlowDirection.LeftToRight,

WrapContents = **false**,

Padding = **new** Padding(**10**),

Visible = **false**,

Height = **300**, // Chiều cao của panel hình ảnh

Dock = DockStyle.Bottom // Đặt panel ở dưới cùng

};

**this**.Controls.Add(posterPanel);

}

}

**private** **void** **ShowPosters**()

{

**if** (posterPanel != **null**)

{

posterPanel.Visible = **true**;

DisplayMoviePosters();

**if** (scrollTimer != **null**)

{

scrollTimer.Start();

}

}

}

**private** **void** **HidePosters**()

{

**if** (posterPanel != **null**)

{

posterPanel.Visible = **false**;

**if** (scrollTimer != **null**)

{

scrollTimer.Stop();

}

}

}

**private** **void** **DisplayMoviePosters**()

{

**if** (posterPanel == **null**)

{

InitializePosterPanel();

}

DataTable dtPosters = GetMoviePosters();

**if** (dtPosters == **null** || dtPosters.Rows.Count == **0**)

{

//MessageBox.Show("Không có poster nào cả.");

**return**;

}

// Xóa các poster cũ để làm mới

posterPanel.Controls.Clear();

**int** repeatCount = **1**; // Số lần lặp lại hình ảnh

**int** posterWidth = **150**; // Chiều rộng của hình ảnh

**int** posterMargin = **10**; // Khoảng cách giữa các hình ảnh

// Tạo một danh sách chứa các PictureBox

List<PictureBox> allPosters = **new** List<PictureBox>();

// Thêm hình ảnh lặp lại

**foreach** (DataRow row **in** dtPosters.Rows)

{

**byte**[] posterData = row["poster"] **as** **byte**[];

**if** (posterData != **null**)

{

**try**

{

**using** (MemoryStream ms = **new** MemoryStream(posterData))

{

Image posterImage = Image.FromStream(ms);

**for** (**int** i = **0**; i < repeatCount; i++)

{

PictureBox picPoster = **new** PictureBox

{

Image = **new** Bitmap(posterImage), // Sao chép hình ảnh

Size = **new** Size(posterWidth, **225**),

SizeMode = PictureBoxSizeMode.StretchImage,

Margin = **new** Padding(posterMargin),

BorderStyle = BorderStyle.FixedSingle

};

allPosters.Add(picPoster);

}

}

}

**catch** (Exception ex)

{

MessageBox.Show("Lỗi khi load hình ảnh: " + ex.Message);

}

}

}

// Thêm tất cả các PictureBox vào posterPanel

**foreach** (**var** picPoster **in** allPosters)

{

posterPanel.Controls.Add(picPoster);

}

// Cập nhật chiều rộng của posterPanel để bao phủ tất cả các hình ảnh

**int** totalWidth = allPosters.Count \* (posterWidth + posterMargin \* **2**);

posterPanel.Width = totalWidth;

// Khởi động timer cuộn

**if** (scrollTimer == **null**)

{

scrollTimer = **new** Timer

{

Interval = **30** // Thay đổi thời gian giữa các lần cuộn

};

scrollTimer.Tick += ScrollTimer\_Tick;

}

scrollTimer.Start();

}

**private** DataTable **GetMoviePosters**()

{

DataTable dtPosters = **new** DataTable();

**string** query = "SELECT poster FROM Movie"; // Thay đổi nếu cần, vd chỉ hiện các poster đang có lịch chiếu, mặc định hiện tại là có poster phim gì show ra hết không quan tâm có lịch chiếu hay chưa

**using** (SqlConnection conn = **new** SqlConnection(connectionString))

{

SqlDataAdapter da = **new** SqlDataAdapter(query, conn);

**try**

{

da.Fill(dtPosters);

}

**catch** (Exception ex)

{

MessageBox.Show("Lỗi khi cố gắng get hình ảnh: " + ex.Message);

}

}

**return** dtPosters;

}

**private** **void** **ScrollTimer\_Tick**(**object** sender, EventArgs e)

{

**if** (posterPanel != **null**)

{

// Nếu cuộn phải

**if** (scrollingRight)

{

// Nếu cuộn đến cuối của panel

**if** (posterPanel.HorizontalScroll.Value + scrollSpeed >= posterPanel.HorizontalScroll.Maximum - posterPanel.HorizontalScroll.LargeChange)

{

// Đổi hướng cuộn sang trái

scrollingRight = **false**;

posterPanel.HorizontalScroll.Value = posterPanel.HorizontalScroll.Maximum; // Đặt về cuối cùng

}

**else**

{

posterPanel.HorizontalScroll.Value += scrollSpeed;

}

}

// Nếu cuộn trái

**else**

{

// Nếu cuộn đến đầu của panel

**if** (posterPanel.HorizontalScroll.Value - scrollSpeed <= **0**)

{

// Đổi hướng cuộn sang phải

scrollingRight = **true**;

posterPanel.HorizontalScroll.Value = **0**; // Đặt về đầu

}

**else**

{

posterPanel.HorizontalScroll.Value -= scrollSpeed;

}

}

posterPanel.PerformLayout(); // Cập nhật lại layout

}

}

+ Nếu ở “Hồ sơ” => hiển thị UC\_Info vào panel background.

* Nếu tài khoản có Rank = “Staff” thì hiển thị nút xem lịch làm việc kế bên nút Lịch sử thanh toán.
* A screenshot of a video chat

  Description automatically generatedTại đây người dùng có thể thay đổi thông tin cá nhân, trừ Username.

**private** **void** **LoadUserData**()

{

txt\_firstname.Text = SessionManager.FirstName;

txt\_lastname.Text = SessionManager.LastName;

txt\_username.Text = SessionManager.Username;

txt\_email.Text = SessionManager.Email;

txt\_sdt.Text = SessionManager.Phone;

txt\_age.Text = SessionManager.Age.ToString();

cbo\_gender.Text = SessionManager.Gender;

txt\_rank.Text = SessionManager.Rank;

// Hiển thị avatar

**byte**[] avatarBytes = SessionManager.Avatar;

**if** (avatarBytes != **null** && avatarBytes.Length > **0**)

{

**using** (MemoryStream ms = **new** MemoryStream(avatarBytes))

{

btn\_image.Image = Image.FromStream(ms);

btn\_image.FillColor = Color.Transparent; // Chỉnh Fill Color thành Transparent nếu như biến avatar của người dùng có tồn tại (Mục đích để ẩn vòng tròn màu trắng ấy)

}

btn\_deleteAvatar.Visible = **true**; // Hiển thị nút xóa avatar

}

**else**

{

// Hiển thị hình ảnh từ resources

btn\_image.Image = Properties.Resources.default\_avatar; // Lấy ảnh mặc định từ Resources nếu như biến avatar của người dùng đang là "NULL"

btn\_image.FillColor = Color.White; // Chỉnh Fill Color thành White nếu như biến avatar của người dùng không tồn tại

btn\_deleteAvatar.Visible = **false**; // Ẩn nút xóa avatar

}

}

**private** **void** **btn\_chinhsua\_Click**(**object** sender, EventArgs e)

{

**int** userId = SessionManager.UserId; // Lấy dữ liệu ID hiện tại của user

**string** firstName = txt\_firstname.Text;

**string** lastName = txt\_lastname.Text;

**string** email = txt\_email.Text;

**string** phone = txt\_sdt.Text;

**int** age = **int**.Parse(txt\_age.Text);

**string** gender = cbo\_gender.Text;

**if**(!Vadilator())**return**;

SqlConnection conn = Hook.Connection; // Kết nối SQL

**try**

{

conn.Open();

{

// Kiểm tra nếu số điện thoại đã tồn tại trong hệ thống

**string** checkPhoneQuery = "SELECT COUNT(\*) FROM Users WHERE phone = @phone AND id != @userId";

**using** (SqlCommand cmdCheckPhone = **new** SqlCommand(checkPhoneQuery, conn))

{

cmdCheckPhone.Parameters.AddWithValue("@phone", phone);

cmdCheckPhone.Parameters.AddWithValue("@userId", userId);

**int** count = (**int**)cmdCheckPhone.ExecuteScalar();

**if** (count > **0**)

{

MessageBox.Show("Số điện thoại đã tồn tại trong hệ thống. Vui lòng nhập số điện thoại khác.", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

txt\_sdt.SelectAll();

txt\_sdt.Focus();

**return**;

}

}

// Cập nhật bảng UsersInfo

**string** queryUsersInfo = @"

UPDATE UsersInfo

SET firstname = @firstname, lastname = @lastname, age = @age, gender = @gender

WHERE id = @userId";

// Cập nhật bảng Users

**string** queryUsers = @"

UPDATE Users

SET email = @email, phone = @phone

WHERE id = @userId";

**using** (SqlCommand cmdUsersInfo = **new** SqlCommand(queryUsersInfo, conn))

{

cmdUsersInfo.Parameters.AddWithValue("@firstname", firstName);

cmdUsersInfo.Parameters.AddWithValue("@lastname", lastName);

cmdUsersInfo.Parameters.AddWithValue("@age", age);

cmdUsersInfo.Parameters.AddWithValue("@gender", gender);

cmdUsersInfo.Parameters.AddWithValue("@userId", userId);

cmdUsersInfo.ExecuteNonQuery();

}

**using** (SqlCommand cmdUsers = **new** SqlCommand(queryUsers, conn))

{

cmdUsers.Parameters.AddWithValue("@email", email);

cmdUsers.Parameters.AddWithValue("@phone", phone);

cmdUsers.Parameters.AddWithValue("@userId", userId);

cmdUsers.ExecuteNonQuery();

}

MessageBox.Show("Thông tin đã được cập nhật!", "Thành công", MessageBoxButtons.OK, MessageBoxIcon.Information);

SessionManager.UpdateSession(firstName, lastName, email, phone, age, gender, SessionManager.Avatar); // Reload lại dữ liệu

}

}

**catch** (Exception ex)

{

MessageBox.Show("Lỗi: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

**finally**

{

conn.Close();

}

}

**private** **void** **btn\_uploadimage\_Click**(**object** sender, EventArgs e)

{

**if** (InvokeRequired)

{

Invoke(**new** Action(() => btn\_uploadimage\_Click(sender, e)));

**return**;

}

// Bật select file

Thread thread = **new** Thread(() =>

{

**using** (OpenFileDialog openFileDialog = **new** OpenFileDialog())

{

openFileDialog.Filter = "Image Files|\*.jpg;\*.jpeg;\*.png";

**if** (openFileDialog.ShowDialog() == DialogResult.OK)

{

**string** filePath = openFileDialog.FileName;

**byte**[] avatar = **null**;

**try**

{

avatar = File.ReadAllBytes(filePath);

// Cập nhật hình ảnh trong hệ thống

UpdateAvatarInDatabase(SessionManager.UserId, avatar);

Invoke(**new** Action(LoadUserData)); // Reload lại để hiện ảnh Avatar mới

}

**catch** (Exception ex)

{

MessageBox.Show("Lỗi: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

});

thread.SetApartmentState(ApartmentState.STA);

thread.Start();

}

**private** **void** **btn\_deleteAvatar\_Click**(**object** sender, EventArgs e)

{

**if** (MessageBox.Show("Bạn chắc chắn muốn xóa avatar không?", "Xác nhận", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)

{

**if** (InvokeRequired)

{

Invoke(**new** Action(() => btn\_deleteAvatar\_Click(sender, e)));

**return**;

}

**try**

{

// Cập nhật avatar = null trong hệ thống

UpdateAvatarInDatabase(SessionManager.UserId, **null**);

// Cập nhật lại Session

SessionManager.UpdateSession(SessionManager.FirstName, SessionManager.LastName, SessionManager.Email, SessionManager.Phone, SessionManager.Age, SessionManager.Gender, **null**);

// Cập nhật lại dữ liệu để hiển thị avatar mặc định

LoadUserData();

}

**catch** (Exception ex)

{

MessageBox.Show("Lỗi: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

**private** **void** **UpdateAvatarInDatabase**(**int** userId, **byte**[] avatar)

{

**string** query = avatar == **null**

? "UPDATE UsersInfo SET avatar = NULL WHERE id = @userId"

: "UPDATE UsersInfo SET avatar = @avatar WHERE id = @userId";

SqlConnection conn = Hook.Connection; // Kết nối SQL

**try**

{

conn.Open();

**using** (SqlCommand cmd = **new** SqlCommand(query, conn))

{

**if** (avatar != **null**)

{

cmd.Parameters.Add("@avatar", SqlDbType.VarBinary).Value = avatar;

}

cmd.Parameters.AddWithValue("@userId", userId);

cmd.ExecuteNonQuery();

SessionManager.UpdateSession(SessionManager.FirstName, SessionManager.LastName, SessionManager.Email, SessionManager.Phone, SessionManager.Age, SessionManager.Gender, avatar);

}

}

**catch** (Exception ex)

{

MessageBox.Show("Lỗi: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

**finally**

{

conn.Close();

}

}

+ Nếu ở “Cài đặt” => hiển thị UC\_Setting vào panel background.

A screenshot of a computer

Description automatically generated

A screenshot of a video chat

Description automatically generated+ “Quản lý” => hiển thị UC\_QuanLy vào panel background.

* 1. Form đổi mật khẩu
* Tên form: User\_ChangePasswordForm
* Chức năng: Đổi mật khẩu cho user.
* Yêu cầu: Phải đăng nhập thì mới thực hiện được lệnh, mật khẩu mới không thể trùng với mật khẩu cũ, mật khẩu mới và mật khẩu xác nhận phải khớp, mật khẩu phải ít nhất 8 ký tự.

A screenshot of a computer

Description automatically generated

**private** **void** **btn\_changePassword\_Click**(**object** sender, EventArgs e)

{

**string** oldPassword = txt\_OldPass.Text.Trim();

**string** newPassword = txt\_NewPass.Text.Trim();

**string** confirmNewPassword = txt\_ConfirmNewPass.Text.Trim();

**if** (**string**.IsNullOrWhiteSpace(oldPassword) || **string**.IsNullOrWhiteSpace(newPassword) || **string**.IsNullOrWhiteSpace(confirmNewPassword))

{

ShowError("Vui lòng điền đầy đủ thông tin.");

**return**;

}

**if** (newPassword.Length < **8**)

{

ShowError("Mật khẩu mới phải có ít nhất 8 ký tự.");

**return**;

}

**if** (newPassword != confirmNewPassword)

{

ShowError("Mật khẩu mới và mật khẩu xác nhận không khớp.");

**return**;

}

**if** (oldPassword == newPassword)

{

ShowError("Mật khẩu mới không thể trùng với mật khẩu cũ.");

**return**;

}

SqlConnection conn = **null**;

**try**

{

conn = Hook.Connection;

conn.Open();

**if** (!IsOldPasswordCorrect(conn, oldPassword))

{

ShowError("Mật khẩu cũ không đúng.");

**return**;

}

UpdatePassword(conn, newPassword);

SessionManager.UpdatePassword(newPassword);

MessageBox.Show("Đổi mật khẩu thành công.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

**this**.Refresh();

Close();

}

**catch** (Exception ex)

{

ShowError("Có lỗi xảy ra: " + ex.Message);

}

**finally**

{

**if** (conn != **null** && conn.State == System.Data.ConnectionState.Open)

{

conn.Close();

}

}

}

**private** **bool** **IsOldPasswordCorrect**(SqlConnection conn, **string** oldPassword)

{

**string** query = "SELECT COUNT(\*) FROM Users WHERE username = @username AND password = @oldPassword";

**using** (SqlCommand cmd = **new** SqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@username", SessionManager.Username);

cmd.Parameters.AddWithValue("@oldPassword", oldPassword);

**return** (**int**)cmd.ExecuteScalar() > **0**;

}

}

**private** **void** **UpdatePassword**(SqlConnection conn, **string** newPassword)

{

**string** query = "UPDATE Users SET password = @newPassword WHERE username = @username";

**using** (SqlCommand cmd = **new** SqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@newPassword", newPassword);

cmd.Parameters.AddWithValue("@username", SessionManager.Username);

cmd.ExecuteNonQuery();

}

}

* 1. Form xem lịch sử chiếu phim
* Tên form: User\_XemLichChieuPhim
* Chức năng: Hiển thị lịch chiếu trong tuần, người dùng chỉ cần ấn nút thời gian để đặt vé.
* Nếu trong ngày có quá nhiều phim đang chiếu thì người dùng có thể kéo xuống dưới để xem.

A screenshot of a computer

Description automatically generated

**private** **const** **int** ButtonWidth = **100**;

**private** **const** **int** ButtonHeight = **60**;

**private** **const** **int** ButtonMargin = **13**;

**private** **const** **int** PosterWidth = **200**;

**private** **const** **int** PosterHeight = **300**;

**private** **const** **int** ShowTimeButtonWidth = **80**;

**private** **const** **int** ShowTimeButtonHeight = **30**;

**private** **const** **int** PosterPadding = **10**;

**private** Button currentDayButton; // Thêm thuộc tính lưu trữ button của ngày hiện tại

**private** **void** **User\_XemLichChieuPhim\_Load**(**object** sender, EventArgs e)

{

// Hiển thị các button ngày

DisplayDateButtons();

// Hiển thị các poster phim cho ngày hiện tại

**if** (currentDayButton != **null**)

{

// Kích hoạt sự kiện Click của button ngày hiện tại để hiển thị lịch chiếu phim

DisplayMoviePosters((DateTime)currentDayButton.Tag);

}

}

**private** **void** **DisplayDateButtons**()

{

DateTime today = DateTime.Today;

**int** numberOfDays = **7**;

grp\_Lich.Controls.Clear();

**int** buttonWidth = grp\_Lich.ClientSize.Width / numberOfDays;

**int** buttonHeight = ButtonHeight;

**for** (**int** i = **0**; i < numberOfDays; i++)

{

DateTime currentDay = today.AddDays(i);

Button btnDay = **new** Button

{

Text = $"{GetDayOfWeek(currentDay)}\n{currentDay:dd/MM/yyyy}",

Width = buttonWidth,

Height = buttonHeight,

Location = **new** Point(i \* buttonWidth, ButtonMargin),

FlatStyle = FlatStyle.Flat,

BackColor = Color.LightSkyBlue, // Màu nền mặc định

ForeColor = Color.White,

TextAlign = ContentAlignment.TopCenter,

Tag = currentDay // Gán ngày hiện tại vào Tag của button

};

btnDay.Click += (s, e) => OnDateButtonClick(btnDay, currentDay);

grp\_Lich.Controls.Add(btnDay);

// Kiểm tra và lưu button của ngày hiện tại

**if** (currentDay.Date == DateTime.Today.Date)

{

currentDayButton = btnDay;

}

}

// Đánh dấu nút ngày hiện tại là màu đỏ

**if** (currentDayButton != **null**)

{

currentDayButton.BackColor = Color.Red;

}

}

**private** **void** **OnDateButtonClick**(Button clickedButton, DateTime selectedDate)

{

// Đổi màu cho nút ngày đã chọn

**if** (currentDayButton != **null**)

{

currentDayButton.BackColor = Color.LightSkyBlue; // Màu nền mặc định

}

clickedButton.BackColor = Color.Red; // Màu nền của nút đã chọn

currentDayButton = clickedButton; // Cập nhật nút ngày đã chọn

// Hiển thị các poster phim cho ngày được chọn

DisplayMoviePosters(selectedDate);

}

// Hàm để lấy tên thứ trong tuần

**private** **string** **GetDayOfWeek**(DateTime date)

{

**switch** (date.DayOfWeek)

{

**case** DayOfWeek.Monday:

**return** "Thứ 2";

**case** DayOfWeek.Tuesday:

**return** "Thứ 3";

**case** DayOfWeek.Wednesday:

**return** "Thứ 4";

**case** DayOfWeek.Thursday:

**return** "Thứ 5";

**case** DayOfWeek.Friday:

**return** "Thứ 6";

**case** DayOfWeek.Saturday:

**return** "Thứ 7";

**case** DayOfWeek.Sunday:

**return** "Chủ nhật";

**default**:

**return** "";

}

}

**private** **void** **DisplayMoviePosters**(DateTime selectedDate)

{

DataTable dtPosters = GetMoviePosters(selectedDate);

panel\_Poster.Controls.Clear();

**var** moviePosters = **new** Dictionary<**string**, List<Tuple<**string**, **string**>>>();

**foreach** (DataRow row **in** dtPosters.Rows)

{

**string** movieName = row["name"].ToString();

**byte**[] posterData = row["poster"] **as** **byte**[];

TimeSpan showtimeSpan = (TimeSpan)row["showtime"];

**string** showtime = showtimeSpan.ToString(@"hh\:mm\:ss");

**string** roomName = row["roomName"].ToString();

**if** (!moviePosters.ContainsKey(movieName))

{

moviePosters[movieName] = **new** List<Tuple<**string**, **string**>>();

}

**var** existingShowtimes = moviePosters[movieName].Select(s => s.Item1).ToHashSet();

**if** (!existingShowtimes.Contains(showtime))

{

moviePosters[movieName].Add(Tuple.Create(showtime, roomName));

}

**if** (posterData != **null** && !panel\_Poster.Controls.OfType<PictureBox>().Any(pb => pb.Tag.ToString() == movieName))

{

**using** (MemoryStream ms = **new** MemoryStream(posterData))

{

PictureBox picPoster = **new** PictureBox

{

Size = **new** Size(PosterWidth, PosterHeight),

Location = **new** Point(**10**, panel\_Poster.Controls.OfType<PictureBox>().Count() \* (PosterHeight + PosterPadding)),

SizeMode = PictureBoxSizeMode.StretchImage,

BorderStyle = BorderStyle.FixedSingle,

Tag = movieName

};

picPoster.Image = Image.FromStream(ms);

panel\_Poster.Controls.Add(picPoster);

}

}

}

**foreach** (**var** movie **in** moviePosters)

{

**string** movieName = movie.Key;

List<Tuple<**string**, **string**>> showtimes = movie.Value;

PictureBox picPoster = panel\_Poster.Controls.OfType<PictureBox>().FirstOrDefault(pb => pb.Tag.ToString() == movieName);

**if** (picPoster != **null**)

{

**int** yOffset = picPoster.Top;

**int** xOffset = picPoster.Right + PosterPadding;

Panel moviePanel = **new** Panel

{

Location = **new** Point(picPoster.Right + PosterPadding, picPoster.Top),

AutoSize = **true**

};

Label lblMovieName = **new** Label

{

Text = movieName,

Font = **new** Font("JetBrains Mono", **12**, FontStyle.Bold),

AutoSize = **true**

};

moviePanel.Controls.Add(lblMovieName);

FlowLayoutPanel flpShowtimes = **new** FlowLayoutPanel

{

FlowDirection = FlowDirection.LeftToRight,

AutoSize = **true**,

Location = **new** Point(**0**, lblMovieName.Bottom + **5**)

};

**foreach** (**var** showtime **in** showtimes)

{

Button btnShowtime = **new** Button

{

Text = $"{showtime.Item1} (Phòng: {showtime.Item2})",

Width = ShowTimeButtonWidth,

Height = ShowTimeButtonHeight,

FlatStyle = FlatStyle.Flat,

BackColor = Color.LightGray,

ForeColor = Color.Black,

Margin = **new** Padding(**5**)

};

btnShowtime.Click += (s, e) => OpenTicketBookingForm(movieName, showtime.Item1, showtime.Item2, selectedDate);

flpShowtimes.Controls.Add(btnShowtime);

}

moviePanel.Controls.Add(flpShowtimes);

panel\_Poster.Controls.Add(moviePanel);

}

}

panel\_Poster.AutoScroll = **true**;

}

**private** **void** **OpenTicketBookingForm**(**string** movieName, **string** showtime, **string** roomName, DateTime dates)

{

**float** price = GetTicketPrice(movieName);

Form parentForm = **this**.FindForm();

parentForm.Hide();

FormStateManager.UpdateFormStateOnClose(parentForm);

User\_DatVe datve = **new** User\_DatVe(movieName, showtime, price, roomName,dates.ToString("dd/MM/yyyy"));

datve.Show();

datve.FormClosed += (s, args) => parentForm.Show();

}

**private** **float** **GetTicketPrice**(**string** movieName)

{

**float** price = **0**;

**string** query = @"

SELECT price

FROM Movie

WHERE name = @movieName";

**using** (SqlConnection conn = **new** SqlConnection(Hook.connectionString))

{

**using** (SqlCommand cmd = **new** SqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@movieName", movieName);

conn.Open();

**object** result = cmd.ExecuteScalar();

**if** (result != **null**)

{

price = Convert.ToSingle(result);

}

}

}

**return** price;

}

**private** DataTable **GetMoviePosters**(DateTime selectedDate)

{

DataTable dtPosters = **new** DataTable();

**string** query = @"

SELECT m.name, m.poster, s.showtime, r.name AS roomName

FROM Movie m

JOIN Showtimes s ON m.id = s.idmovie

JOIN Room r ON s.idroom = r.id

WHERE s.dates = @selectedDate";

**using** (SqlConnection conn = **new** SqlConnection(Hook.connectionString))

{

**using** (SqlCommand cmd = **new** SqlCommand(query, conn))

{

cmd.Parameters.AddWithValue("@selectedDate", selectedDate.Date);

**using** (SqlDataAdapter adapter = **new** SqlDataAdapter(cmd))

{

adapter.Fill(dtPosters);

}

}

}

**return** dtPosters;

}

* 1. Form đặt vé
* Tên form: User\_DatVe
* Chức năng: Đặt vé.
* Người dùng sử dụng chuột trái để chọn ghế ngồi, chuột phải để hủy chọn ghế.
* Ghế trống có màu xám, ghế bị khóa có màu đỏ, ghế đã được đặt có màu xanh biển, ghế đang chọn có màu tím.
* Sau khi chọn ghế ngồi thì hiện nút thanh toán hoặc nút hủy.
* Trên form phải hiện ngày chiếu mình đã chọn trước đó, giờ chiếu, giá vé, tên phim.

A screenshot of a computer

Description automatically generated

**string** connectionString = Hook.connectionString;

Panel seatPanel = **new** Panel();

**private** **string** movieName;

**private** **string** showtime;

**private** **float** price;

**private** **string** roomName;

**private** **string** date;

**private** **int** selectedSeatCount = **0**;

**private** **float** totalAmount = **0f**;

**private** HashSet<**string**> selectedSeats = **new** HashSet<**string**>();

**public** **User\_DatVe**(**string** movieName, **string** showtime, **float** price, **string** roomName,**string** date) : **this**()

{

**this**.movieName = movieName;

**this**.showtime = showtime;

**this**.price = price;

**this**.roomName = roomName;

**this**.date=date;

}

**private** **void** **User\_DatVe\_Load**(**object** sender, EventArgs e)

{

lbl\_namemovie.Text = "Phim bạn chọn: " + movieName;

lbl\_showtime.Text = "Giờ chiếu: " + Convert.ToDateTime(showtime).ToString("HH:mm:ss");

lbl\_giave.Text = "Giá vé: " + price.ToString("#,0") + " VNĐ";

lbl\_date.Text = "Ngày chiếu: " + date;

LoadRoomNames();

SelectFirstAvailableRoom(); // chọn phòng đầu tiên khi load form

}

**private** **void** **LoadRoomNames**()

{

**string** query = @"

SELECT r.name

FROM Room r

WHERE r.status = 0

AND r.id IN (

SELECT s.idroom

FROM Showtimes s

WHERE s.dates = @date

AND s.showtime = @showtime

AND s.idmovie = (SELECT id FROM Movie WHERE name = @movieName)

)";

**using** (SqlConnection conn = **new** SqlConnection(connectionString))

{

SqlCommand cmd = **new** SqlCommand(query, conn);

// Use the correct format for date and showtime

cmd.Parameters.AddWithValue("@date", DateTime.Parse(date).ToString("yyyy-MM-dd")); // Only date part

cmd.Parameters.AddWithValue("@showtime", TimeSpan.Parse(showtime).ToString()); // Use TimeSpan format

cmd.Parameters.AddWithValue("@movieName", movieName);

**try**

{

conn.Open();

SqlDataReader reader = cmd.ExecuteReader();

cmb\_selectRoom.Items.Clear();

**while** (reader.Read())

{

cmb\_selectRoom.Items.Add(reader["name"].ToString());

}

reader.Close();

}

**catch** (Exception ex)

{

MessageBox.Show("Lỗi: " + ex.Message);

}

}

}

**private** **void** **LoadRoomCapacity**(**string** roomName)

{

**string** query = "SELECT capacity FROM Room WHERE name = @name";

**using** (SqlConnection conn = **new** SqlConnection(connectionString))

{

SqlCommand cmd = **new** SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@name", roomName);

**try**

{

conn.Open();

**object** result = cmd.ExecuteScalar();

**if** (result != **null**)

{

**int** capacity = Convert.ToInt32(result);

txt\_capacity.Text = capacity.ToString();

DisplaySeats(capacity);

}

**else**

{

txt\_capacity.Text = "Không có thông tin";

groupBox\_panel.Controls.Clear();

}

}

**catch** (Exception ex)

{

MessageBox.Show("Lỗi: " + ex.Message);

}

}

}

**private** **void** **DisplaySeats**(**int** capacity)

{

groupBox\_panel.Controls.Clear();

**int** buttonWidth = **50**;

**int** buttonHeight = **30**;

**int** spacing = **5**;

**int** seatsPerRow = **10**; // Số ghế mỗi hàng

**int** rows = (**int**)Math.Ceiling((**double**)capacity / seatsPerRow);

**int** gridWidth = (seatsPerRow + **1**) \* (buttonWidth + spacing); // +1 cho tiêu đề cột

**int** gridHeight = rows \* (buttonHeight + spacing) + buttonHeight; // +1 cho tiêu đề hàng

**int** startX = (groupBox\_panel.ClientSize.Width - gridWidth) / **2**;

**int** startY = (groupBox\_panel.ClientSize.Height - gridHeight) / **2**;

// Thêm tiêu đề cột

**for** (**int** col = **0**; col < seatsPerRow; col++)

{

Label colLabel = **new** Label

{

Text = col.ToString(),

Width = buttonWidth,

Height = buttonHeight,

Left = startX + col \* (buttonWidth + spacing) + buttonWidth / **2**,

Top = startY - buttonHeight / **2**,

TextAlign = ContentAlignment.MiddleCenter

};

groupBox\_panel.Controls.Add(colLabel);

}

// Thêm các nút ghế

**for** (**int** row = **0**; row < rows; row++)

{

**char** rowLetter = (**char**)('A' + row);

// Thêm tiêu đề hàng

Label rowLabel = **new** Label

{

Text = rowLetter.ToString(),

Width = buttonWidth,

Height = buttonHeight,

Left = startX - buttonWidth / **2** - spacing,

Top = startY + row \* (buttonHeight + spacing) + buttonHeight / **2**,

TextAlign = ContentAlignment.MiddleCenter

};

groupBox\_panel.Controls.Add(rowLabel);

**for** (**int** col = **0**; col < seatsPerRow; col++)

{

**int** seatNumber = row \* seatsPerRow + col;

**if** (seatNumber >= capacity) **break**;

**string** seatId = $"{rowLetter}{col}"; // ID ghế như A0, A1, v.v.

Button button = **new** Button

{

Text = seatId,

Width = buttonWidth,

Height = buttonHeight,

Left = startX + col \* (buttonWidth + spacing) + buttonWidth / **2**,

Top = startY + row \* (buttonHeight + spacing) + buttonHeight / **2**,

Tag = seatId,

BackColor = Color.LightGray, // Mặc định

Enabled = **true**

};

SetButtonColor(seatId, button); // Cập nhật màu sắc của ghế

button.MouseDown += SeatButton\_MouseDown;

groupBox\_panel.Controls.Add(button);

}

}

}

**private** **void** **SetButtonColor**(**string** seatId, Button button)

{

**string** query = @"

SELECT status

FROM Seat

WHERE name = @seatId AND idroom = (SELECT id FROM Room WHERE name = @roomName)";

**using** (SqlConnection conn = **new** SqlConnection(connectionString))

{

SqlCommand cmd = **new** SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@seatId", seatId);

cmd.Parameters.AddWithValue("@roomName", cmb\_selectRoom.SelectedItem?.ToString());

**try**

{

conn.Open();

**object** result = cmd.ExecuteScalar();

**if** (result != **null**)

{

**int** status = Convert.ToInt32(result);

**if** (status == **2**)

{

button.BackColor = Color.Red; // Đã đặt

button.Enabled = **false**;

}

**else** **if** (status == **1**)

{

button.BackColor = Color.FromArgb(**76**, **87**, **193**); // Đã chọn

button.Enabled = **false**;

}

**else**

{

button.BackColor = Color.LightGray; // Mặc định

button.Enabled = **true**;

}

}

**else**

{

button.BackColor = Color.LightGray;

button.Enabled = **true**;

}

}

**catch** (Exception ex)

{

MessageBox.Show("Lỗi: " + ex.Message);

}

}

}

**private** **void** **SeatButton\_MouseDown**(**object** sender, MouseEventArgs e)

{

Button seatButton = sender **as** Button;

**if** (seatButton != **null**)

{

**string** seatId = seatButton.Tag.ToString();

**if** (e.Button == MouseButtons.Right)

{

// Nếu ghế đã chọn, bỏ chọn nó

**if** (seatButton.BackColor == Color.FromArgb(**96**, **62**, **129**)) // Ghế đã chọn

{

seatButton.BackColor = Color.LightGray; // Bỏ chọn ghế

selectedSeats.Remove(seatId);

selectedSeatCount--;

}

}

**else** **if** (e.Button == MouseButtons.Left)

{

// Nếu ghế có sẵn, chọn nó

**if** (seatButton.BackColor == Color.LightGray) // Ghế có sẵn

{

seatButton.BackColor = Color.FromArgb(**96**, **62**, **129**); // Đánh dấu là đã chọn

selectedSeats.Add(seatId);

selectedSeatCount++;

}

// Nếu ghế đã đặt, không làm gì

**else** **if** (seatButton.BackColor == Color.FromArgb(**76**, **87**, **193**)) // Ghế đã đặt

{

// Có thể hiển thị thông báo hoặc bỏ qua nhấp chuột

}

}

UpdateTicketCountAndAmount();

UpdatePaymentAndCancelButtons(); // Cập nhật trạng thái của nút thanh toán và hủy

}

}

**private** **void** **UpdateTicketCountAndAmount**()

{

lbl\_soLuongVe.Text = "Số lượng vé: " + selectedSeatCount.ToString();

totalAmount = selectedSeatCount \* price;

lbl\_thanhTien.Text = "Thành tiền: " + totalAmount.ToString("#,0") + " VNĐ";

}

**private** **void** **cmb\_selectRoom\_SelectedIndexChanged**(**object** sender, EventArgs e)

{

**string** selectedRoom = cmb\_selectRoom.SelectedItem?.ToString();

// Kiểm tra xem nút thanh toán và nút hủy có nên hiển thị không

// Nút thanh toán sẽ hiển thị khi có phòng được chọn và số lượng vé đã chọn lớn hơn hoặc bằng 1

btn\_thanhToan.Visible = !**string**.IsNullOrEmpty(selectedRoom) && selectedSeatCount >= **1**;

btn\_cancel.Visible = !**string**.IsNullOrEmpty(selectedRoom) && selectedSeatCount >= **1**;

**if** (!**string**.IsNullOrEmpty(selectedRoom))

{

LoadRoomCapacity(selectedRoom);

}

}

**private** **void** **UpdateSeatStatus**(**string** seatId, **int** status)

{

**string** query = @"

UPDATE Seat

SET status = 1

WHERE name = @seatId AND idroom = (SELECT id FROM Room WHERE name = @roomName)";

**using** (SqlConnection conn = **new** SqlConnection(connectionString))

{

SqlCommand cmd = **new** SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@seatId", seatId);

cmd.Parameters.AddWithValue("@roomName", cmb\_selectRoom.SelectedItem?.ToString());

**try**

{

conn.Open();

**int** rowsAffected = cmd.ExecuteNonQuery();

}

**catch** (Exception ex)

{

MessageBox.Show("Lỗi: " + ex.Message);

}

}

}

**private** **int** **GetRoomId**(**string** roomName)

{

**int** roomId = -**1**;

**string** query = "SELECT id FROM Room WHERE name = @roomName";

**using** (SqlConnection conn = **new** SqlConnection(connectionString))

{

SqlCommand cmd = **new** SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@roomName", roomName);

**try**

{

conn.Open();

**object** result = cmd.ExecuteScalar();

**if** (result != **null**)

{

roomId = Convert.ToInt32(result);

}

}

**catch** (Exception ex)

{

MessageBox.Show("Lỗi: " + ex.Message);

}

}

**return** roomId;

}

**private** **void** **btn\_thanhToan\_Click**(**object** sender, EventArgs e)

{

**if** (selectedSeats.Count > **0**)

{

**string** selectedRoom = cmb\_selectRoom.SelectedItem?.ToString();

**if** (!**string**.IsNullOrEmpty(selectedRoom))

{

**int** roomId = GetRoomId(selectedRoom);

**if** (roomId != -**1**)

{

**foreach** (**string** seatId **in** selectedSeats)

{

UpdateSeatStatus(seatId, **1**); // chỉnh status = 1

}

DateTime billDate = DateTime.Parse(date); // Sử dụng biến date đã lưu trữ

**int** billId = CreateNewBill();

InsertBillInfo(billId, roomId, billDate, TimeSpan.Parse(showtime), roomName, GetRoomCapacity(roomId), price, movieName);

MessageBox.Show("Thanh toán thành công!");

Close();

}

**else**

{

MessageBox.Show("Không tìm thấy thông tin phòng.");

}

}

}

**else**

{

MessageBox.Show("Chưa chọn ghế.");

}

}

**private** **int** **GetRoomCapacity**(**int** roomId)

{

**int** capacity = **0**;

**string** query = "SELECT capacity FROM Room WHERE id = @roomId";

**using** (SqlConnection conn = **new** SqlConnection(connectionString))

{

SqlCommand cmd = **new** SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@roomId", roomId);

**try**

{

conn.Open();

**object** result = cmd.ExecuteScalar();

capacity = result != **null** ? Convert.ToInt32(result) : **0**;

}

**catch** (Exception ex)

{

MessageBox.Show("Lỗi khi lấy sức chứa phòng: " + ex.Message);

}

}

**return** capacity;

}

**private** **float** **GetSeatPrice**()

{

// Trả về giá vé hiện tại đã được lưu trữ

**return** price;

}

**private** **int** **CreateNewBill**()

{

**int** billId = -**1**;

**string** query = "INSERT INTO Bill (dateorder, status, iduserpayer) OUTPUT INSERTED.id VALUES (@dateorder, @status, @iduserpayer)";

**using** (SqlConnection conn = **new** SqlConnection(connectionString))

{

SqlCommand cmd = **new** SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@dateorder", DateTime.Now);

cmd.Parameters.AddWithValue("@status", **1**);

cmd.Parameters.AddWithValue("@iduserpayer", SessionManager.UserId);

**try**

{

conn.Open();

**object** result = cmd.ExecuteScalar();

**if** (result != **null**)

{

billId = Convert.ToInt32(result);

}

}

**catch** (Exception ex)

{

MessageBox.Show("Lỗi khi tạo hóa đơn: " + ex.Message);

}

}

**return** billId;

}

**private** **void** **InsertBillInfo**(**int** billId, **int** roomId, DateTime date, TimeSpan showtime, **string** roomName, **int** capacity, **float** price, **string** movieName)

{

**string** seats = **string**.Join(",", selectedSeats);

**string** query = @"

INSERT INTO BillInfo (id, countProduct, totalamount, status, dates, showtime, nameroom, capacity, idroom, price, idseat, movieName)

VALUES (@billId, @countProduct, @totalAmount, @status, @dates, @showtime, @nameroom, @capacity, @idroom, @price, @idseat, @movieName)";

**using** (SqlConnection conn = **new** SqlConnection(connectionString))

{

SqlCommand cmd = **new** SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@billId", billId);

cmd.Parameters.AddWithValue("@countProduct", selectedSeats.Count);

cmd.Parameters.AddWithValue("@totalAmount", totalAmount);

cmd.Parameters.AddWithValue("@status", **1**);

cmd.Parameters.AddWithValue("@dates", date);

cmd.Parameters.AddWithValue("@showtime", showtime);

cmd.Parameters.AddWithValue("@nameroom", roomName);

cmd.Parameters.AddWithValue("@capacity", capacity);

cmd.Parameters.AddWithValue("@idroom", roomId);

cmd.Parameters.AddWithValue("@price", price);

cmd.Parameters.AddWithValue("@idseat", seats);

cmd.Parameters.AddWithValue("@movieName", movieName);

**try**

{

conn.Open();

cmd.ExecuteNonQuery();

}

**catch** (Exception ex)

{

MessageBox.Show("Lỗi khi lưu thông tin hóa đơn: " + ex.Message);

}

}

}

**private** **void** **btn\_cancel\_Click**(**object** sender, EventArgs e)

{

// Đặt lại màu

**foreach** (**string** seatId **in** selectedSeats)

{

**foreach** (Control control **in** groupBox\_panel.Controls)

{

**if** (control **is** Button button && button.Tag.ToString() == seatId)

{

// Đặt lại màu sắc của ghế dựa trên trạng thái của nó từ cơ sở dữ liệu

SetButtonColor(seatId, button);

**break**;

}

}

}

// Xóa tất cả các ghế đã chọn khỏi danh sách selectedSeats

selectedSeats.Clear();

// Đặt lại số lượng vé và tổng tiền về giá trị ban đầu

selectedSeatCount = **0**;

totalAmount = **0f**;

UpdateTicketCountAndAmount();

// Cập nhật lại nút thanh toán và hủy

UpdatePaymentAndCancelButtons();

}

**private** **void** **UpdatePaymentAndCancelButtons**()

{

// Kiểm tra nếu có ít nhất một ghế được chọn và phòng đã được chọn

**bool** shouldShowButtons = selectedSeatCount > **0** && cmb\_selectRoom.SelectedItem != **null**;

btn\_thanhToan.Visible = shouldShowButtons;

btn\_cancel.Visible = shouldShowButtons;

}

**private** **void** **SelectFirstAvailableRoom**()

{

**if** (cmb\_selectRoom.Items.Count > **0**)

{

cmb\_selectRoom.SelectedIndex = **0**;

}

}

* 1. Form lịch sử mua vé
* Tên form: User\_HistoryBuy
* Chức năng: Hiển thị lịch sử đặt vé.

A screenshot of a computer

Description automatically generated

* 1. Form khiếu nại
* Tên form: User\_Report
* Chức năng: Tạo phản hồi khiếu nại, có thể chọn loại khiếu nại và ghi mô tả.

A screenshot of a computer

Description automatically generated

* 1. Form quản lý khiếu nại
* Tên form: Manage\_Report
* Chức năng: Xem toàn bộ khiếu nại, có thể xuất báo cáo hoặc trả lời.

A screenshot of a computer

Description automatically generated

* 1. Form quản lý phòng chiếu
* Tên form: Manage\_Room
* Chức năng: Tạo, sửa, xóa, hủy các ghế đã đặt phòng chiếu.

A screenshot of a computer

Description automatically generated

* 1. Form quản lý ghế ngồi
* Tên form: Manage\_Seat
* Chức năng: Khóa hoặc mở ghế.

A screenshot of a computer

Description automatically generated

* 1. Form quản lý phim
* Tên form: Manage\_Film
* Chức năng: Thêm, sửa, xóa phim

A screenshot of a computer

Description automatically generated

* 1. Form quản lý lịch chiếu
* Tên form: Manage\_ShowTimes
* Chức năng: Thêm, sửa, xóa lịch chiếu.

A screenshot of a computer

Description automatically generated

* 1. Form quản lý nhân viên
* Tên form: Manage\_Staff
* Chức năng: Xuất danh sách, sửa, chỉnh thành khách hàng.

A screenshot of a computer

Description automatically generated

* 1. Form quản lý người dùng
* Tên form: Manage\_Users
* Chức năng: Xuất danh sách, sửa, xóa, chỉnh thành nhân viên.

A screenshot of a computer

Description automatically generated

* 1. Form quản lý thống kê
* Tên form: Manage\_ThongKe
* Chức năng: Xem thống kê toàn bộ lịch sử đặt vé, có thể xem doanh thu tháng trước, doanh thu hôm nay, doanh thu tháng này.

A screenshot of a computer

Description automatically generated

# **CHƯƠNG 6: XỬ LÝ FORM**

* 1. Xây dựng Class Hỗ Trợ (SqlHook)
* Là cầu nối giữa SQL đến với các Form.

public static class Hook

{

public static readonly string connectionString = @"Data Source=[SỬA CHỔ NÀY];Initial Catalog=QuanLyRapPhim;Integrated Security=True";

private static readonly Lazy<SqlConnection> \_connection = new Lazy<SqlConnection>(() => new SqlConnection(connectionString));

public static SqlConnection Connection => \_connection.Value;

}

Lớp Hook cung cấp một cách quản lý kết nối đến cơ sở dữ liệu bằng cách sử dụng Lazy<> để trì hoãn việc tạo đối tượng SqlConnection cho đến khi nó thực sự cần thiết. Mục đích: Giúp tối ưu hóa hiệu suất bằng cách chỉ tạo kết nối khi cần và đảm bảo rằng chỉ có một kết nối duy nhất được sử dụng trong suốt quá trình ứng dụng chạy.

public static void OpenConnection()

{

if (Connection.State == ConnectionState.Closed)

{

Connection.Open();

}

}

public static void CloseConnection()

{

if (Connection.State == ConnectionState.Open)

{

Connection.Close();

}

}

Hai phương thức tĩnh để mở và đóng kết nối đến cơ sở dữ liệu.

public static DataTable LoadCSDL(string query)

{

DataTable dt = new DataTable();

try

{

OpenConnection();

using (SqlCommand command = new SqlCommand(query, Connection))

{

using (SqlDataAdapter da = new SqlDataAdapter(command))

{

da.Fill(dt);

}

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi: " + ex.Message);

}

finally

{

CloseConnection();

}

return dt;

}

Phương thức LoadCSDL thực thi một câu lệnh SQL và trả về kết quả dưới dạng DataTable. Nó mở kết nối cơ sở dữ liệu, thực thi câu lệnh, và đảm bảo rằng kết nối được đóng sau khi hoàn tất. Được áp dụng khi sử dụng Report (RDLC).

* 1. Class SessionManager
* Mục đích: Lưu trữ phiên đăng nhập của người dùng.

internal class SessionManager

{

public static string Username { get; private set; }

public static string FirstName { get; private set; }

public static string LastName { get; private set; }

public static int UserId { get; private set; }

public static string Rank { get; private set; }

public static string Email { get; private set; }

public static string Phone { get; private set; }

public static int Age { get; private set; }

public static string Gender { get; private set; }

public static byte[] Avatar { get; private set; }

public static void CreateSession(int userId, string username, string firstName, string lastName, string rank, string email, string phone, int age, string gender, byte[] avatar)

{

UserId = userId; Username = username; FirstName = firstName;

LastName = lastName; Rank = rank; Email = email; Phone = phone;

Age = age; Gender = gender; Avatar = avatar;

}

public static void UpdateSession(string firstName, string lastName, string email, string phone, int age, string gender, byte[] avatar)

{

FirstName = firstName; LastName = lastName; Email = email;

Phone = phone; Age = age; Gender = gender; Avatar = avatar;

}

public static void ClearSession()

{

UserId = 0; Username = null; FirstName = null; LastName = null;

Rank = null; Email = null; Phone = null; Age = 0; Gender = null;

Avatar = null;

}

}

* CreateSession: Tạo phiên đăng nhập của người dùng. (Sử dụng khi đăng nhập)
* UpdateSession: Cập nhật phiên đăng nhập. (Sử dụng khi đổi thông tin)
* ClearSession: Xóa phiên đăng nhập hiện tại của người dùng. (Sử dụng khi đăng xuất)
  1. Class MovementController
* Mục đích: Làm vài control được chỉ định có thể thay đổi vị trí window của giao diện chương trình.

public static class MovementController

{

public static void FormMovement(Form form, Panel panelRed, PictureBox picLogo, Label lblTitle)

{

panelRed.MouseDown += (sender, e) => {

mouseDown = true;

lastLocation = e.Location;

};

panelRed.MouseMove += (sender, e) => {

if (mouseDown)

{

form.Location = new System.Drawing.Point(

form.Location.X - lastLocation.X + e.X, form.Location.Y - lastLocation.Y + e.Y);

form.Update();

}

};

panelRed.MouseUp += (sender, e) => {

mouseDown = false;

};

// Áp dụng di chuyển form bằng các controls đã được đưa vào

ApplyControlMovement(picLogo, form);

ApplyControlMovement(lblTitle, form);

}

private static bool mouseDown;

private static Point lastLocation;

private static void ApplyControlMovement(Control control, Form form)

{

control.MouseDown += (sender, e) => {

mouseDown = true;

lastLocation = e.Location;

};

control.MouseMove += (sender, e) => {

if (mouseDown)

{

form.Location = new System.Drawing.Point(

form.Location.X - lastLocation.X + e.X, form.Location.Y - lastLocation.Y + e.Y);

form.Update();

}

};

control.MouseUp += (sender, e) => {

mouseDown = false;

};

}

}

* 1. Class KeyMessageFilter
* Mục đích: Được sử dụng để xử lý khi hệ thống nhận tổ hợp phím.
* Alt + F4 => Thoát nhanh chương trình.
* F12 => Chụp ảnh màn hình chương trình.

**public** **class** **KeyMessageFilter** : IMessageFilter

{

**public** **bool** **PreFilterMessage**(**ref** Message m)

{

**const** **int** WM\_SYSKEYDOWN = **0**x0104;

**const** **int** WM\_KEYDOWN = **0**x0100;

**if** (m.Msg == WM\_SYSKEYDOWN) // Alt + F4 để đóng chương trình

{

Keys keyCode = (Keys)(m.WParam.ToInt32() & (**int**)Keys.KeyCode);

**if** (Control.ModifierKeys == Keys.Alt && keyCode == Keys.F4)

{

Application.Exit();

**return** **true**;

}

}

**if** (m.Msg == WM\_KEYDOWN) // F12 để chụp ảnh màn hình

{

Keys keyCode = (Keys)(m.WParam.ToInt32() & (**int**)Keys.KeyCode);

**if** (keyCode == Keys.F12)

{

CaptureProgramWindowToClipboard();

**return** **true**;

}

}

**return** **false**;

}

**private** **void** **CaptureProgramWindowToClipboard**()

{

Form activeForm = Form.ActiveForm;

**if** (activeForm != **null**)

{

Rectangle bounds = activeForm.Bounds;

**using** (Bitmap bitmap = **new** Bitmap(bounds.Width, bounds.Height))

{

**using** (Graphics g = Graphics.FromImage(bitmap))

{

g.CopyFromScreen(activeForm.Location, Point.Empty, bounds.Size);

}

Clipboard.SetImage(bitmap); // Copy vào clipboard

}

}

}

}

* 1. Class DoubleBufferedEnhancer
* Mục đích: Cung cấp các phương thức để bật tính năng Double Buffering cho control và tất cả các control con bên trong một control cha. Giúp cải thiện hiệu suất 3D và giảm hiện tượng nhấp nháy trong ứng dụng.

internal class DoubleBufferedEnhancer

{

public static void SetDoubleBuffered(Control control)

{

typeof(Control).InvokeMember("DoubleBuffered",

BindingFlags.SetProperty | BindingFlags.Instance | BindingFlags.NonPublic,

null, control, new object[] { true });

}

public static void EnableDoubleBuffering(Control parentControl)

{

foreach (Control control in parentControl.Controls)

{

SetDoubleBuffered(control);

if (control.HasChildren)

{

EnableDoubleBuffering(control);

}

}

SetDoubleBuffered(parentControl); // Bật DoubleBuffered cho Control

}

}

* 1. Class FormStateManager
* Mục đích: Giúp ghi nhớ trạng thái của Form khi nó được đóng hoặc ẩn và khôi phục trạng thái này khi biểu mẫu được mở hoặc hiển thị lại. Cung cấp trải nghiệm người dùng liền mạch bằng cách giữ trạng thái của biểu mẫu giữa các lần mở ứng dụng.

public static class FormStateManager

{

private static bool wasPreviousFormMaximized = false;

public static void UpdateFormStateOnClose(Form form)

{

wasPreviousFormMaximized = (form.WindowState == FormWindowState.Maximized);

}

public static void SetFormStateOnLoad(Form form)

{

if (wasPreviousFormMaximized)

{

form.WindowState = FormWindowState.Maximized;

}

else

{

form.WindowState = FormWindowState.Normal;

}

}

public static void UpdateFormStateOnCloseOrHide(Form form)

{

if (form.WindowState == FormWindowState.Maximized)

{

wasPreviousFormMaximized = true;

}

else

{

wasPreviousFormMaximized = false;

}

}

public static void SetFormStateOnLoadOrShow(Form form)

{

if (wasPreviousFormMaximized)

{

form.WindowState = FormWindowState.Maximized;

}

else

{

form.WindowState = FormWindowState.Normal;

}

}

}

* 1. Class CenterController
* Mục đích: Phương thức Center căn giữa một control (panel) trong một control cha (parent). Tính toán tọa độ mới cho control panel để đảm bảo rằng nó nằm ở giữa control parent.

internal static class CenterController

{

public static void Center(Control panel, Control parent)

{

if (panel == null)

throw new ArgumentNullException(nameof(panel), "Panel không thể trống.");

if (parent == null)

throw new ArgumentNullException(nameof(parent), "Parent control không thể trống.");

// Tính toán vị trí để căn giữa

int x = (parent.ClientSize.Width - panel.Width) / 2;

int y = (parent.ClientSize.Height - panel.Height) / 2;

// Đặt vị trí mới cho panel hoặc UserControl

panel.Location = new Point(x, y);

}

}

# **CHƯƠNG 7: HƯỚNG DẪN SỬ DỤNG CHƯƠNG TRÌNH**

Tại giao diện đăng nhập, nếu chưa có tài khoản đăng nhập, hãy đăng ký tài khoản, nếu đã có tài khoản mà quên mật khẩu, hãy sử dụng tính năng quên mật khẩu.

Sau khi đăng nhập, ở trang chủ tại đây người dùng có thể thấy poster phim đang có sẵn, nếu có nhu cầu đặt vé xem phim, hãy ấn “Xem lịch chiếu”.

Nếu có nhu cầu thay đổi thông tin của người dùng, hãy ấn “Hồ sơ”, tại đây người dùng có thể thay đổi các thông tin, ngoại trừ Username. Ngoài ra còn có thể thay đổi ảnh đại diện hoặc xóa ảnh đại diện. Tại đây còn có nút “Lịch sử thanh toán” và người dùng có thể xuất hóa đơn tất cả hóa đơn theo nhu cầu.

Nút “Cài đặt” bao gồm 3 chức năng

* Đổi mật khẩu
* Đăng xuất
* Khiếu nại => Chọn loại khiếu nại & viết miêu tả và ấn gửi.

**Đối với người dùng nhân viên**, chỉ có thể “Xem lịch làm việc” trong “Hồ sơ”.

**Đối với người dùng quản lý**, có thể truy cập vào danh mục “Quản lý”.

* Phim: Quản lý phim (thêm, sửa, xóa, thay đổi ảnh poster, tìm kiếm).
* Lịch chiếu: Quản lý lịch chiếu (thêm, sửa, xóa, tìm kiếm).
* Lịch làm việc: Quản lý lịch làm việc của nhân viên (thêm, sửa, xóa, tìm kiếm)
* Phòng: Quản lý phòng (thêm, sửa, xóa, tìm kiếm, hủy các ghế đã đặt).
* Khách hàng: Quản lý thông tin khách hàng (sửa, xóa, chỉnh thành nhân viên, xuất danh sách).
* Nhân viên: Quản lý thông tin nhân viên (sửa, chỉnh thành khách hàng, xuất danh sách).
* Ghế: Quản lý ghế (chọn phòng, tìm kiếm, khóa ghế).
* Thống kê: Xem thống kê doanh thu (tháng này, tháng trước, hôm nay).
* Xem phản hồi: Xuất danh sách khiếu nại, trả lời.

# **CHƯƠNG 8: TỔNG KẾT & HƯỚNG PHÁT TRIỂN**

* 1. Tổng kết
* Chương trình giúp quản lý & lưu trữ tiện lợi, rành mạch.
* Giao diện hiện đại thân thiện với người dùng.
* Tính tương thích và bảo mật cao.
  1. Hướng phát triển
* Cần phát triển thêm tính năng “Quản lý lịch làm việc” & “Xem lịch làm việc”.
* Tối ưu lại SQL.
* Cải thiện tốc độ load hình ảnh bằng cách sử dụng GPU (nếu có).
* Cần thêm nhiều chức năng hơn cho “Nhân viên”.
* Áp dụng kỹ thuật Caching (Bộ nhớ đệm để giảm thời gian tải).