**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG THÁI NGUYÊN**

**KHOA CÔNG NGHỆ ĐIỆN TỬ & TRUYỀN THÔNG**

…………..o0o…………..



**Báo cáo**

**THỰC TẬP CHUYÊN NGÀNH**

**Đề tài : Xây dựng hệ thống quản lý bán vé xem phim trực tuyến**

**Giảng viên hướng dẫn**: Ts. Ngô Thị Vinh

**Bộ môn :** Điện tử máy tính

**Sinh viên**: Hoàng Văn Xuân

**Mã sinh viên**: DTC1954801080008 **Lớp**: KTMT-K18A

**LỜI CẢM ƠN**

🙠🕮🙢

Lời đầu tiên em xin chân thành cảm ơn đến tất cả các thầy cô giáo trong khoa Công Nghệ Điện Tử & Truyền Thông – Trường Đại học CNTT&TT Thái Nguyên, những người đã trực tiếp giảng dạy, truyền đạt những kiến thức bổ ích trong suốt những năm học qua, đã cung cấp cho em rất nhiều những kiến thức cơ bản, là những kiến thức vô cùng quý giá, là nền tảng phục vụ cho công việc sau này. Cùng với thời gian thực tập chuyên ngành, em đã học được những bài học kinh nghiệm quý giá cho bản thân, để em có thể triển khai đề tài xây dựng hệ thống quản lý bán vé xem phim trực tuyến.Đặc biệt nhất, em xin tỏ lòng biết ơn đến cô Ngô Thị Vinh, người đã trực tiếp hướng dẫn tận tình và giúp đỡ em trong quá trình nghiên cứu thực hiện để hoàn thành đề tài này.

Em xin trân trọng cảm ơn!

*Thái Nguyên, 02 tháng 4 năm 2023*

**Người thực hiện**

Xuân

Hoàng Văn Xuân

MỤC LỤC

[CHƯƠNG I: BÀI TOÁN QUẢN LÝ BÁN VÉ XEM PHIM TRỰC TUYẾN 5](#_Toc131441034)

[1.1 Đặt vấn đề 5](#_Toc131441035)

[1.2 Các yêu cầu chính của bài toán 6](#_Toc131441036)

[1.3 Mô tả hệ thống bằng UML 6](#_Toc131441037)

[1.3.1 Biểu đồ ca sử dụng 6](#_Toc131441038)

[1.3.2 Chi tiết các chức năng 7](#_Toc131441039)

[1.3.3 Biểu đồ trình tự 9](#_Toc131441040)

[1.3.4 Biểu đồ hoạt động 15](#_Toc131441041)

[1.4 Thiết kế cơ sở dữ liệu 21](#_Toc131441042)

[1.4.1 Các bảng lưu trữ 21](#_Toc131441043)

[1.4.2 Biểu đồ mối quan hệ thực thể 23](#_Toc131441044)

[CHƯƠNG II : KHÁI QUÁT VỀ NGÔN NGỮ LẬP TRÌNH JAVA CHO WEBSITE 25](#_Toc131441045)

[2.1 Tổng quan về ngôn ngữ Java 25](#_Toc131441046)

[2.2. Tổng quan về ngôn ngữ HTML 27](#_Toc131441047)

[2.2 Tổng quan về ngôn ngữ CSS 29](#_Toc131441048)

[2.3 Tổng quan về ngôn ngữ JavaScript 30](#_Toc131441049)

[2.4 Tổng quan về cơ sở dữ liệu 31](#_Toc131441050)

[CHƯƠNG III :CHƯƠNG TRÌNH QUẢN LÝ BÁN VÉ XEM PHIM TRỰC TUYẾN 32](#_Toc131441051)

[3.1 Giao diện và chức năng 32](#_Toc131441052)

[- Trang chủ 32](#_Toc131441053)

[- Giao diện đăng nhập 34](#_Toc131441054)

[- Giao diện đăng ký 36](#_Toc131441055)

[- Giao diện chọn phòng và chọn ghế 40](#_Toc131441056)

[- Giao diện kiểm tra thông tin vé 43](#_Toc131441057)

[- Giao diện trang quản lý vé của khách hàng 45](#_Toc131441058)

[- Giao diện trang quản trị 48](#_Toc131441059)

[- Giao diện trang quản lý phim 49](#_Toc131441060)

[- Giao diện trang quản lý phòng chiếu 52](#_Toc131441061)

[- Giao diện trang quản lý tài khoản 55](#_Toc131441062)

[CHƯƠNG IV:KẾT LUẬN 58](#_Toc131441063)

[TÀI LIỆU THAM KHẢO 59](#_Toc131441064)

# CHƯƠNG I: BÀI TOÁN QUẢN LÝ BÁN VÉ XEM PHIM TRỰC TUYẾN

## 1.1 Đặt vấn đề

Thời đại công nghệ đã và đang tiếp tục phát triển mạnh mẽ,để theo để theo kịp xu hướng toàn cầu thì việc cải tiến quản lý dữ liệu trên nền tảng điện tử số đối với các ngành dịch vụ là cần thiết ,công nghệ thông tin được áp dụng trong nhiều lĩnh vực khác nhau, với khả năng ứng dụng rộng rãi và chính sách thúc đẩy phát triển ngàng công nghệ của Nhà Nước thì việc ứng dụng công nghệ thông tin vào hỗ trợ các liên quan đến quản lý bán vé xem phim cho ngành dịch vụ là cần thiết để nâng cao sự phát triển của ngành công nghệ dịch vụ ở Việt Nam

Trên thực tế,các hệ thống quản lý bán vé xem phim đã xuất hiện ở Việt Nam vài năm trở lại đây khi mà kỹ thuật công nghệ phát triển nhanh như vậy.hệ thống quản lý bán vé phim trực tuyến đã và đang mang lại lợi ích không nhỏ cho các chuỗi rạp phim trên đất nước Việt Nam nói riêng và toàn thế giới nói chung.

Được sự góp ý từ người dùng về trải nghiệm các tiện ích của hệ thống bán vé xem phim từ các chuỗi rạp đã có mặt từ rất sớm ở Việt Nam như CGV,Beta Cinema, BHD, việc đặt được vé chiếu sớm đã dễ dàng hơn,Tiết kiệm chi phí di chuyển đến rạp,biết sớm thông tin về phim mà người dùng mong muốn xem ,nhận các ưu đãi và khuyến mại từ việc đặt vé trực tuyến và còn rất nhiều tiện ích khác mà người dùng đã phản hồi tại website của hãng đó.

C:\Users\Admin\Pictures\cgvlogo.png  

**Hình 1.1 Các hệ thống rạp chiếu phim lớn tại Việt Nam**

Hệ thống sẽ dễ dàng quản lý lượng khách của hãng rạp chiếu,các thống kê báo cáo sẽ rõ ràng và chính xác hơn khi được quản lý bằng tài liệu điện tử ,giảm thiểu đi việc của nhân viên bán vé tại quầy. Xem và phản hồi các ý kiến từ phía khách hàng nhanh chóng.

Tuy nhiên các hệ thống rạp chiếu chỉ khởi sắc tại các thành phố lớn tại các tỉnh thành lân cận trung tâm còn xa lạ với khái niệm rạp chiếu phim.Các bộ phim hoành tráng sở hữu công nghệ 3D,4D còn chưa tiếp cận được với đông đảo quần chúng.

Xuất phát từ những khảo sát đã tìm hiểu và nhằm nâng cao chất lượng quản lý bán vé tại các rạp,tránh những sai sót khi vẫn vận hàng bằng phương pháp thủ công,đề tài cá nhân của em sẽ đặt trọng tâm vào vấn đề phân tích,mô phỏng hệ thống và thiết kế các chức năng cho người dùng sẽ sử dụng,người đọc dễ hiểu.

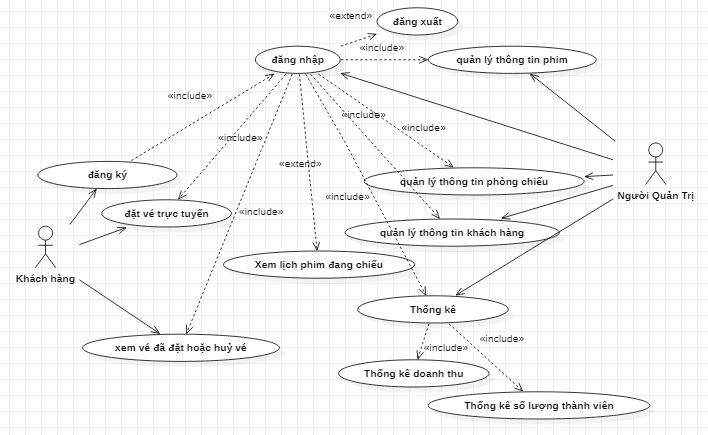
## 1.2 Các yêu cầu chính của bài toán

Qua việc tìm hiểu các Website trên thực tế, hệ thống quản lý bán vé xem phim gồm có các chức năng như sau :

* Quản lý người dùng, hệ thống cho phép người quản trị phân quyền cho người dùng
* Quản lý phim,quản lý phòng chiếu,hệ thống cho phép người quản trị thao tác với dữ liệu trong cơ sở dữ liệu
* Người dùng đăng ký tài khoản và thực hiện lên lịch xem phim,huỷ lịch xem phim,xem những vé đã được lên lịch

## 1.3 Mô tả hệ thống bằng UML

### 1.3.1 Biểu đồ ca sử dụng

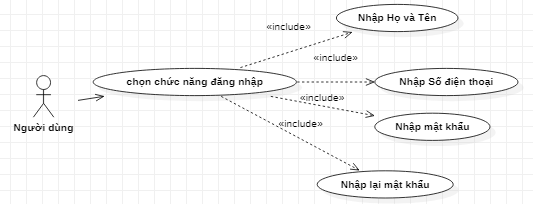


**Hình 1.2 Biểu đồ ca sử dụng tổng quát hệ thống**

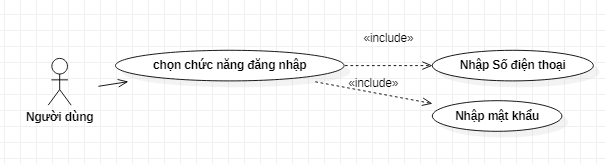
Từ hình 1.2 cho thấy các tác nhân của hệ thống gồm có: khách hàng,người quản trị tất cả sự tương tác giữa tác nhân và hệ thống thì phải đăng nhập,còn xem lịch chiếu là không cần đăng nhập

### 1.3.2 Chi tiết các chức năng

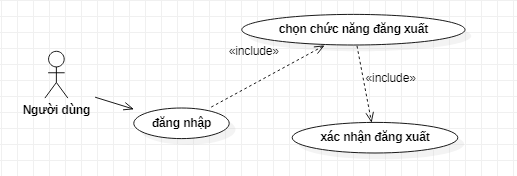
* Phân rã chức năng **đăng ký**



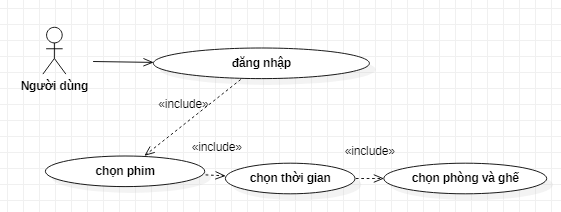
* Phân rã chức năng **đăng nhập**



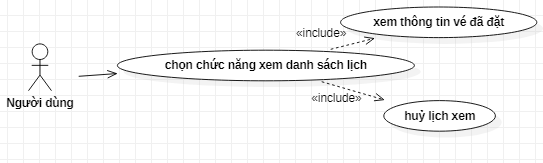
* Phân rã chức năng **đăng xuất**



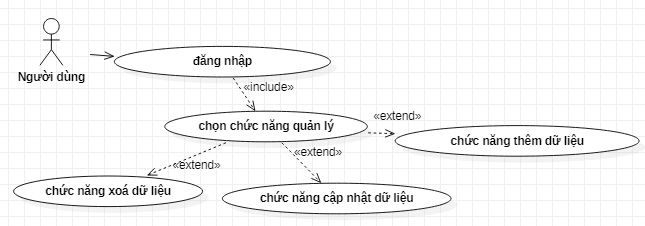
* phân rã chức năng **đặt vé trực tuyến**



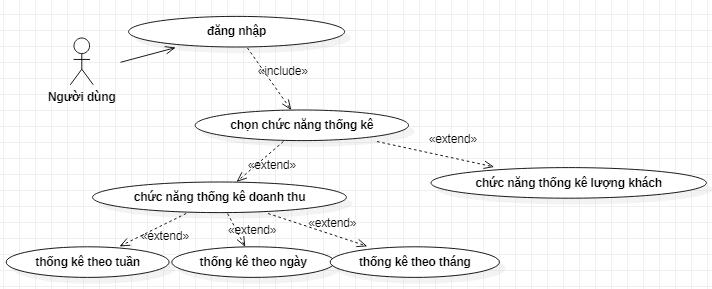
* phân rã chức năng **xem vé đã đặt và huỷ vé**



* Phân rã chức năng **quản lý** của người quản trị

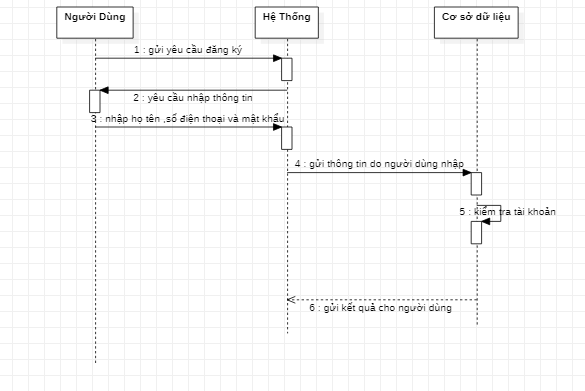


* Phân rã chức năng **Thống kê**



### 1.3.3 Biểu đồ trình tự

* Biểu đồ trình tự đăng ký

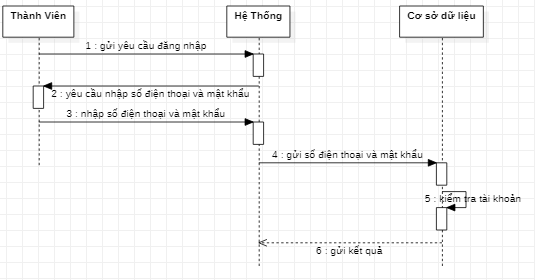


**Hình 1.3. Biểu đồ trình tự đăng ký**

Hình 1.3, có 3 đối tượng là: người dùng, hệ thống và cơ sở dữ liệu. Luồng xử lí của chức năng đăng ký có thể diễn giải như sau :

Người dùng đăng ký dựa trên thông tin mà người dùng cung cấp cho hệ thống,xảy ra lỗi khi người dùng bỏ trống một hoặc nhiều dữ liệu cần nhập và nhập tài khoản đã thuộc sở hữu của người khác

* Biểu đồ trình tự đăng nhập

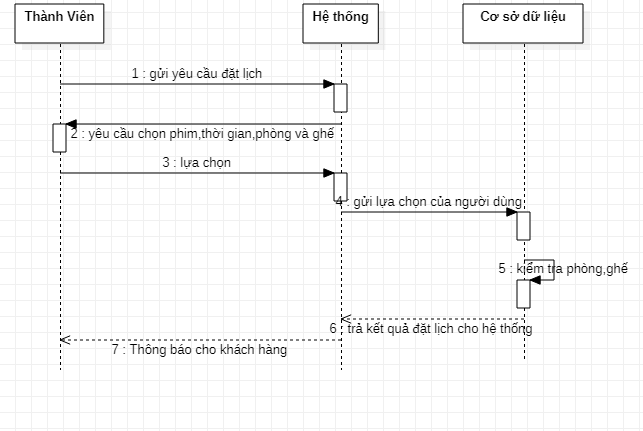


Hình 1.4 Biểu đồ trình tự đăng nhập

Hình 1.4 có 3 đối tượng là: Thành viên, hệ thống và cơ sở dữ liệu. Luồng xử lí của chức năng đăng nhập có thể diễn giải như sau:

Người dùng sau khi đăng ký sẽ được ghi nhận là Thành viên của hệ thống,khi đăng nhập người dùng cần nhập tài khoản và mật khẩu,Hệ thống kiểm tra dựa trên tài khoản mà người dùng đã đăng ký và xảy ra ngoại lệ khi người dùng bỏ trống dữ liệu cần nhập,nhập sai một trong hai thông tin đã đăng ký,người dùng chưa đăng ký.

* Biểu đồ trình tự đặt vé trực tuyến

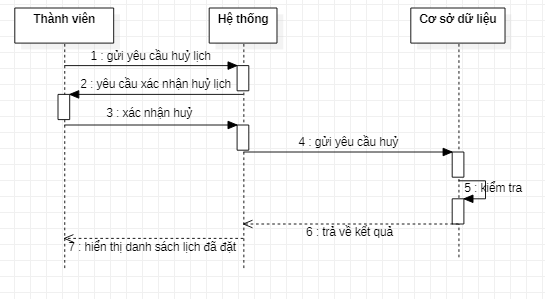


Hình 1.5 Biểu đồ trình tự đặt lịch xem phim

Hình 1.5 có 3 đối tượng là: Thành viên, hệ thống và cơ sở dữ liệu. Luồng xử lí của chức năng đặt vé trực tuyến có thể diễn giải như sau.

Người dùng sau khi đăng nhập sẽ được phép chọn phim ,chọn thời gian chiếu và chọn phòng. Xảy ra ngoại lệ khi người dùng chọn vào ghế đã có người khác đặt trước

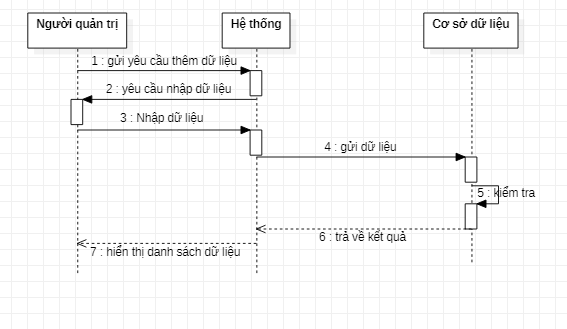
* Biểu đồ trình tự huỷ lịch xem phim



**Hình 1.6 Biểu đồ trình tự huỷ lịch xem phim**

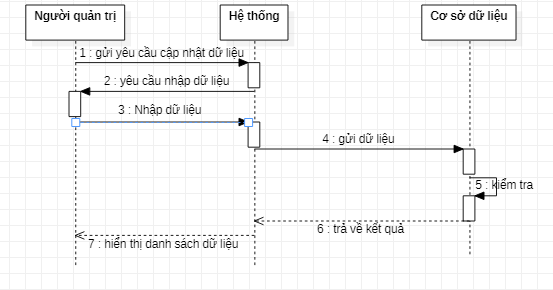
Hình 1.6 Thành viên gửi yêu cầu đến hệ thống về việc huỷ lịch xem phim ,hệ thống xác nhận lại thành viên có huỷ vé hay không,nếu huỷ thì hệ thống sẽ gửi yêu cầu đó cho cơ sở dữ liệu,xảy ra ngoại lệ khi thành viên hoàn tác việc huỷ hoặc lịch đó đã quá hạn huỷ

* Biểu đồ trình tự quản lý dữ liệu



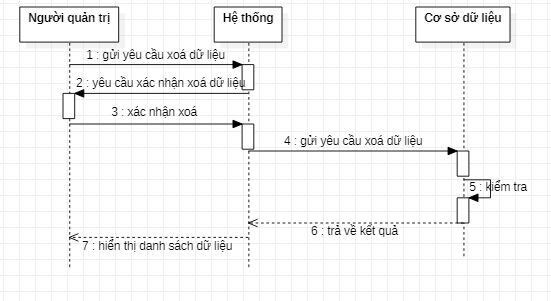
**Hình 1.7 Biểu đồ trình tự thêm dữ liệu**

Hình 1.7, có 3 đối tượng là Người quản trị,Hệ thống và Cơ sở dữ liệu, thao tác từ người quản trị được áp dụng cho việc quản lý phim, quản lý phòng chiếu phim và quản lý tài khoản người dùng, người quản trị sẽ yêu cầu thêm dữ liệu vào cơ sở dữ liệu,xảy ra ngoại lệ khi người quản trị nhập không đủ thông tin ,người quản trị bị mất quyền quản trị đột ngột



**Hình 1.8 Biểu đồ trình tự cập nhật dữ liệu**

Hình 1.8 , có 3 đối tượng là Người quản trị,Hệ thống và Cơ sở dữ liệu, thao tác từ người quản trị được áp dụng cho việc quản lý phim, quản lý phòng chiếu phim và quản lý tài khoản người dùng, người quản trị sẽ yêu cầu cập nhật dữ liệu vào cơ sở dữ liệu,xảy ra ngoại lệ khi người quản trị nhập không đủ thông tin ,người quản trị bị mất quyền quản trị đột ngột

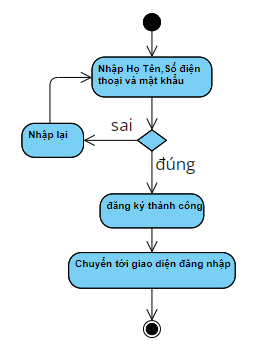


**Hình 1.9 Biểu đồ trình tự xoá dữ liệu**

Hình 1.9, có 3 đối tượng là Người quản trị, Hệ thống và Cơ sở dữ liệu, thao tác từ người quản trị được áp dụng cho việc quản lý phim, quản lý phòng chiếu phim và quản lý tài khoản người dùng, người quản trị sẽ thao tác xoá vào cơ sở dữ liệu,xảy ra ngoại lệ khi người quản trị huỷ bỏ việc xoá,hoặc bị mất quyền truy cập quản trị

### 1.3.4 Biểu đồ hoạt động

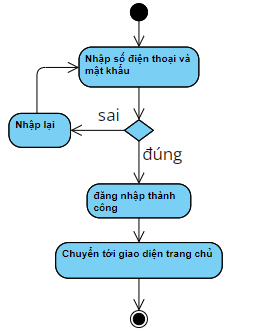
* Biểu đồ hoạt động đăng ký



**Hình 1.10 Biểu đồ hoạt động đăng ký**

1. Người dùng chuyển tới giao diện đăng ký
2. Hệ thống yêu cầu người dùng nhập đủ 3 thông tin đó là Họ Và tên,số điện thoại di động và mật khẩu
3. Người dùng nhập thông tin
4. Hệ thống kiểm tra người dùng đã nhập đủ hay chưa?
5. Nhập đủ và số điện thoại người dùng nhập vào chưa có ai đăng ký thì sẽ được thông báo đăng ký thành công
6. Nếu vi phạm sẽ trả lại bước nhập
7. Chuyển tới giao diện đăng nhập và kết thúc

* Hoạt động đăng nhập

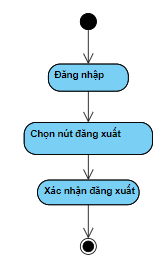


**Hình 1.11 Biểu đồ hoạt động đăng nhập**

Người dùng chuyển tới giao diện đăng nhập

1. Hệ thống yêu cầu người dùng nhập đủ số điện thoại di động và mật khẩu
2. Người dùng nhập thông tin
3. Hệ thống kiểm tra người dùng đã nhập đủ hay chưa?
4. Nhập đủ và số điện thoại và mật khẩu được kiểm tra sẽ được chuyển về giao diện trang chủ kèm lời chào đến khách hàng
5. Nếu vi phạm sẽ trả lại bước nhập
6. Chuyển tới giao diện trang chủ và kết thúc

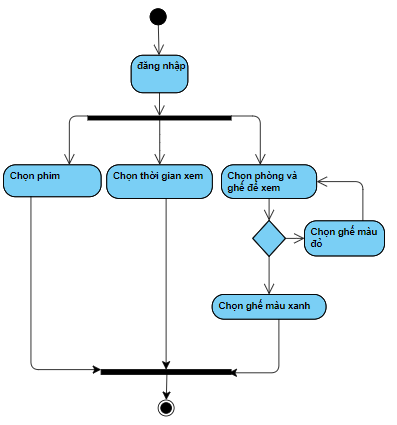
* Hoạt động đăng xuất



**Hình 1.12 Biểu đồ hoạt động đăng xuất**

Người dùng chọn vào nút và thao tác như biểu đồ rồi kết thúc

* Hoạt động đặt lịch xem phim

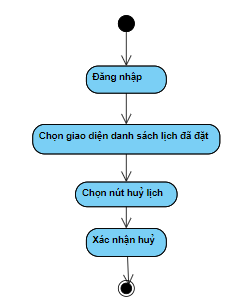


**Hình 1.13 Biểu đồ hoạt động đặt lịch xem phim**

Biểu đồ hoạt động đặt lịch xem phim được diễn tả qua các bước sau:

1. Người dùng chuyển tới giao diện chứa thông tin phim
2. Hệ thống yêu cầu chọn phim,chọn ngày giờ và phòng ghế
3. Người dùng lựa chọn
4. Hệ thống kiểm tra ghế này đã được thuê hay chưa?
5. Nếu ghế đã được thuê bởi người khác thì sẽ được chọn lại ghế khác
6. Chuyển tới giao diện kiểm tra lại thông tin và kết thúc

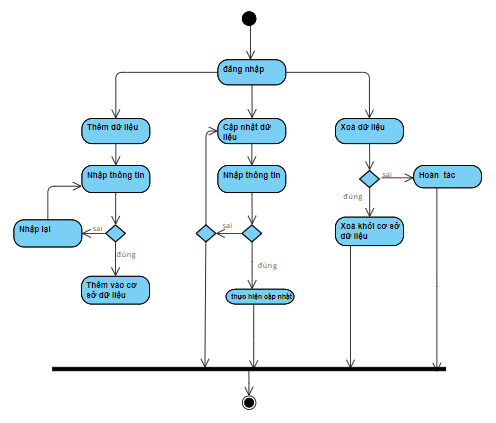
* Hoạt động huỷ lịch xem



**Hình 1.14 Biểu đồ hoạt động huỷ lịch xem phim**

Khách hàng chọn vào lịch đã đặt và thao tác huỷ và xác nhận lại việc muốn huỷ và kết thúc

* Hoạt động quản lý của người quản trị



**Hình 1.14 Biểu đồ hoạt động quản lý của người quản trị**

Biểu đồ hoạt động quản lý của người quản trị được diễn tả như sau

1. Người quản trị đăng nhập và được chuyển tới trang quản lý
2. Thao tác thêm dữ liệu

2. 1 Nhập dữ liệu

2. 2 Nếu dữ liệu hợp lệ thì sẽ được thêm vào cơ sở dữ liệu

1. Thao tác cập nhật

3.1 Nhập dữ liệu

3.2 Hệ thống kiểm tra hợp lệ

3.2 Nếu thông tin hợp lệ thì sẽ được thêm vào cơ sở dữ liệu

3.2 Nếu chưa hợp lệ thì yêu cầu cập nhật lại hoặc kết thúc

1. Thao tác xoá

4.1 Người quản trị chọn chức năng xoá tại giao diện

4.2 Hệ thống thực hiện xoá tại cơ sở dữ liệu

1. Kết thúc hoạt động

## 1.4 Thiết kế cơ sở dữ liệu

Bản thiết kế cơ sở dữ liệu được cài đặt và sử dụng bởi phần mềm MySQL ,dữ liệu ngắn gọn, tránh gây lặp lại dữ liệu,tiết kiệm bộ nhớ hiệu quả

### 1.4.1 Các bảng lưu trữ

* Bảng Tài khoản (Account) {

Mã tài khoản (id) kiểu số nguyên là khoá chính ,tự động tăng

Tên Người dùng (fullname) kiểu chuỗi

Mật Khẩu (password) kiểu chuỗi

Phân quyền người dùng (roleid) kiểu số nguyên }

Bảng Phim (Movie) {

Mã phim (id) kiểu số nguyên là khoá chính,tự động tăng

Tên phim(movie\_name) kiểu chuỗi

ảnh bộ phim (image) kiểu chuỗi

Mô tả (decription) kiểu chuỗi

Diễn viên (actors) kiểu chuỗi

Đạo diễn (director) kiểu chuỗi

Thời lượng bộ phim (total\_time) kiểu số nguyên

Ngôn ngữ (language) kiểu chuỗi

Năm phát hành (durian) kiểu date }

* Bảng Phòng (Room) {

Mã Phòng (id) kiểu số nguyên là khoá chính và tự động đăng

Tên phòng (room\_name) kiểu chuỗi

ảnh phòng (room\_image) kiểu chuỗi

Sức chúa (capacity) kiểu số nguyên

}

* Bảng Ghế (Seat) {

Mã ghế (id) kiểu số nguyên là khoá chính và tự động tăng

Tên ghế(name\_seat) là kiểu chuỗi

Trạng thái (status) kiểu số nguyên

Mã phòng (id\_room) kiểu số nguyên là khoá ngoại liên kết với bảng Phòng }

* Bảng Ngày (Schedu){

Mã ngày (id) kiểu số nguyên là khoá chính và tự động tăng

Ngày (date\_in\_movie) kiểu ngày

}

Bảng giờ (schedule\_time) {

Mã giờ (id) kiểu số nguyên là khoá chính và tự động tăng

Giờ (time) kiểu giờ

Mã ngày(id\_date) kiểu số nguyên là khoá ngoại liên kết với bảng Ngày

}

Bảng Vé(Ticket) {

Mã vé (id) kiểu số nguyên là khoá chính và tự động tăng

Tên phim(name\_movie) kiểu chuỗi

Tên phòng (room\_name) kiểu chuỗi

Tên Ghế (seat\_name) kiểu chuỗi

Ngày ,giờ chiếu(date ,time) kiểu chuỗi

Thời gian lập vé (setime) kiểu chuỗi

Giá vé (price) kiểu số nguyên

Mã tài khoản (account\_id) kiểu số nguyên là khoá ngoại liên kết với bảng Tài khoản

Mã ghế(seat\_id) kiểu số nguyên là khoá ngoại liên kết với bảng Ghế

Mã ghế(seat\_id) kiểu số nguyên là khoá ngoại liên kết với bảng Ghế

Mã phim(movie\_id) kiểu số nguyên là khoá ngoại liên kết với bảng Phim

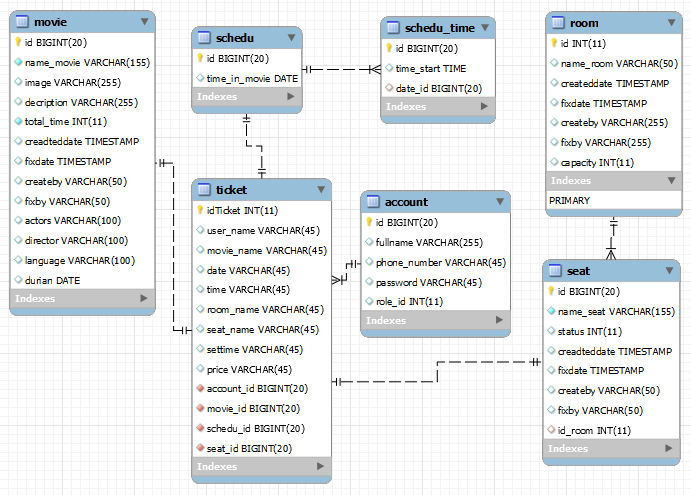
Mã ngày(date\_id) kiểu số nguyên là khoá ngoại liên kết với bảng Ngày

}

Sử dụng dữ liệu đã thiết kế để tạo bảng với mã lệnh sau :

|  |
| --- |
| CREATE TABLE Tên bảng (  Cột 1 tên dữ liệu kiểu dữ liệu,  Cột 2 tên dữ liệu kiểu dữ liệu,  Cột 3 tên dữ liệu kiểu dữ liệu,  ....  );  Đối với mã định danh đối tượng của bảng sau kiểu dữ liệu cần thêm các mã lệnh sau:  Tạo khoá chính : primary key  Tự động tăng : auto increment not null  Các câu lệnh thêm,sửa,xoá dữ liệu sẽ được viết khi thực hiện viết mã chương trình Java |

### 1.4.2 Biểu đồ mối quan hệ thực thể



# CHƯƠNG II : KHÁI QUÁT VỀ NGÔN NGỮ LẬP TRÌNH JAVA CHO WEBSITE

## 2.1 Tổng quan về ngôn ngữ Java

**Java** là một trong những ngôn ngữ lập trình hướng đối tượng được sử dụng trong phát triển phần mềm, trang web, game hay ứng dụng trên các thiết bị di động.Ngôn ngữ này được giới thiệu lần đầu tiên bởi James Gosling và cộng sự của ông vào năm 1991

Đặc trưng của ngôn ngữ lập trình hướng đối tượng được thể hiện trong ngôn ngữ này là kế thừa tính đóng gói,tính đa hình và trừu tượng hoá, nền tảng độc lập khi được thông dịch bởi máy ảo JVM(Java Vituarl Machine) trong chính nền tảng công cụ mà lập trình viên sử dụng, Ngoài ra những đặc điểm nổi trội hơn các ngôn ngữ khác có thể kể đến là mức độ bảo mật,sửa lỗi,hiệu năng cao…,Chính vì vậy đây là ngôn ngữ đáng để lựa chọn để lập trình Website

**Java Servlet** là một công nghệ tạo ra ứng dụng web nằm ở phía máy chủ và tạo ra trang web động và cũng được biết đến là API(Application Programming Interface) cung cấp các interface và lớp tài liệu phục vụ việc tạo trang web

Công nghệ này nhận dữ liệu do trình duyệt từ khách hàng gửi đến và lấy dữ liệu đó gửi đến tầng phục vụ phía máy chủ, nhận dữ liệu từ máy chủ và hiển thị cho khách hàng,dữ liệu trả về có thể là HTML hay định dạng nhị phân như ảnh,File tài liệu…

Nhược điểm duy nhất của công nghệ này là viết mã HTML và mã Java cùng một file cho cảm giác phức tạp đối với lập trình viên

Sử dụng công nghệ này bằng cách các gói javax.servlet và javax.servlet.http đây là một phần của Java, một phiên bản mở rộng của thư viện lớp Java hỗ trợ các dự án phát triển quy mô lớn

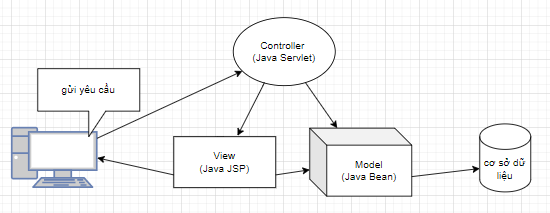
Để sử dụng hiệu quả công nghệ này hãy viết mã cho các lớp chứa Controller trong môn hình **MVC**(**Model**-**View**-**Controller**).**Model** sẽ là các lớp chứa đối tượng cần đến trong bài toán cần thiết trong đề tài.

Còn công nghệ giúp xử lý phần hiển thị mà khách hàng có thể thao tác và nhìn thấy được chính là **JSP**

**JSP(Java Server Page)** là công nghệ giúp lập trình viên tạo ra các phần hiển thị hay xử lý các mã HTML ngay trên file định dạng .jsp, công nghệ này cũng có thể nhúng các mã Java,Truy cập cơ sở dữ liệu,hay các truy vấn động trên đó nhưng để tránh việc phức tạp hoá vấn đề thì các lập trình viên thường sử dụng để thiết kế các phần hiển thị cho trang web

Công nghệ này còn cho phép lấy dữ liệu đầu vào từ người dùng thông qua các thẻ nhập của HTML và thực hiện đưa dữ liệu tới các Servlet tương ứng,Hiển thị các dữ liệu ra từ phía máy chủ,thực thi mã lệnh nhanh hơn các ngôn nhữ khác có thể kể đến là Common Gateway Interface(CGI)

Hai công nghệ này luôn được hoạt động cùng lúc vì JSP sẽ xử lý phần hiển thị nhanh chóng rồi chuyển đổi thành các Servlet để biên dịch



**Hình 2.1 Cách thức hoạt động của mô hình MVC**

* Công nghệ và công cụ lập trình :

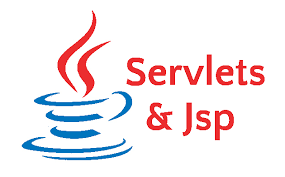
**Java Development Kit** (JDK) là một trong ba gói công nghệ cốt lõi được sử dụng trong lập trình Java, cùng với JVM (Máy ảo Java - Java Virtual Machine) và JRE (Java Runtime Environment - Môi trường Java Runtime).

**Eclipse** là một công cụ hỗ trợ lập trình mã nguồn mở được phát triển bởi IBM.

Công cụ này hỗ trợ cho lập trình viên viết mã theo các mô hình phát triển như MVC, tạo thêm các thư viện hỗ trợ phát triển phần mềm.

**Apache Tomcat** là một loại web server HTTP được phát triển bởi Apache Software Foundation, nó có khả năng hỗ trợ mạnh cho các ứng dụng Java thay vì các website tĩnh khác,có khả năng chạy trên nhiều bản Java chuyên biệt như: Java Servlet, JavaServer Pages (JSP), Java EL và WebSocket.Có tính ổn định cao và sử dụng được hầu hết trên các nền tảng hệ điều hành và các ngôn ngữ như PHP,Java,sở hữu các tính năng như một web thương mại và là lựa chọn đáng tin cậy cho việc kiểm tử ứng dụng web khi chưa đăng ký được bản quyền tên miền

**Maven** là công cụ quản lý và thiết lập tự động một dự án phần mềm. Chủ yếu dùng cho các lập trình viên Java, nhưng nó cũng có thể được dùng để xây dựng và quản lý các dự án dùng C#, Ruby, Scala hay ngôn ngữ khác

[](https://tomcat.apache.org/download-80.cgi)  [C:\Users\Admin\Pictures\javaee.png](https://www.eclipse.org/downloads/packages/release/kepler/sr2/eclipse-ide-java-ee-developers)

[C:\Users\Admin\Pictures\apache-maven-project.png](https://maven.apache.org/download.cgi)

**Hình 2.2 Các công cụ và công nghệ sử dụng trong đề tài**

## 2.2. Tổng quan về ngôn ngữ HTML

**HTML** (Hypertext Markup Language, hay là “Ngôn ngữ đánh dấu siêu văn bản”) là một ngôn ngữ đánh dấu được thiết kế ra để tạo nên các trang web trên World Wide Web. Nó có thể được trợ giúp bởi các công nghệ như CSS và các ngôn ngữ kịch bản giống như JavaScript.

Cấu trúc chung :

|  |
| --- |
| <!DOCTYPE html>  <html>  <head>  <title>Tiêu đề trang web</title>  </head>  <body>  Nội dung được viết ở đây  </body>  </html> |

Ngôn ngữ này có các ưu điểm sau:

* Là ngôn ngữ có thể chạy trên nhiều nền tảng phần cứng khác nên được sử dụng rộng rãi trên thế giới.
* Có quá trình học đơn giản
* Tiêu chuẩn chính của web được vận hành bởi World Wide Web Consortium (W3C).
* Dễ dàng tích hợp với các ngôn ngữ khác như PHP,Java và Node.js.
* Mã nguồn mở và hoàn toàn miễn phí.

Ngoài những ưu điểm thì ngôn ngữ này còn có những khuyết điểm sau :

* Khó kiểm soát cảnh thực thi của trình duyệt (ví dụ như những trình duyệt cũ không thể cập nhật được những thẻ mới ).
* Được dùng chủ yếu cho web tĩnh. Đối với các tính năng động, bạn cần sử dụng JavaScript hoặc ngôn ngữ xử lý tác vụ bên thứ 3 như PHP vậy, hầu hết các trang đều cần được tạo riêng biệt, kể cả khi nó sử dụng cùng các yếu tố, như là headers hay footers.
* Một số trình duyệt chậm hỗ trợ tính năng mới.

## 2.2 Tổng quan về ngôn ngữ CSS

**CSS (Cascading Style Sheets)** là một ngôn ngữ được sử dụng để tìmvà định dạng lại các phần tử được tạo ra bởi các ngôn ngữ đánh dấu (ví dụ như HTML).

Cấu trúc chung :

|  |
| --- |
| Selector{  properties: value;  properties: value;  properties: value;  …..  } |
|  |

Trong đó:

* + Selector là các đối tượng được tác động vào. Ví dụ như các thẻ div, p hay các class, id trong HTML.
  + properties là các thuộc tính trong css như: backgorund, color,...
  + value là các giá chị của thuộc tính .

Những ưu điểm của ngôn ngữ này là :

* + - Hạn chế tối thiểu việc làm rối mã HTML của trang Web bằng các thẻ quy định kiểu dáng (chữ đậm, chữ in nghiêng, chữ có gạch chân, chữ màu), khiến mã nguồn của trang Web được gọn gàng hơn, tách nội dung của trang Web và định dạng hiển thị, dễ dàng cho việc cập nhật nội dung.
    - Tạo ra các kiểu dáng có thể áp dụng cho nhiều trang Web, giúp tránh phải lặp lại việc định dạng cho các trang Web giống nhau.
    - Ngoài những ưu điểm thì ngôn ngữ này còn hạn chế sau :
    - Với những thay đổi ban đầu của CSS trên một website rất dễ dàng. Tuy nhiên, khi thay đổi đã được thực hiện đòi hỏi chúng ta phải xác nhận được tính tương thích khi CSS hiển thị hiệu ứng thay đổi tương tự cho từng trình duyệt.

## 2.3 Tổng quan về ngôn ngữ JavaScript

**Javascript** chính là một ngôn ngữ lập trình web rất phổ biến ngày nay. **Javascript** được tích hợp đồng thời nhúng vào HTML để hỗ trợ cho website trở nên sống động hơn. Chúng cũng đóng vai trò tương tự như một phần của website, cho phép Client-side Script từ người dùng tương tự máy chủ (Nodejs) để tạo ra những website động.

**Brendan Eich** chính là người đã phát triển Javascript tại Netscape với tiền thân là **Mocha**. Sau đó, Mocha được đổi thành LiveScript và cuối cùng mới đổi thành **JavaScript**. Năm 1998, JavaScript với phiên bản mới nhất là ECMAScript 2 phát hành và đến năm 1999 thì ECMAScript 3 được ra mắt

* Cấu trúc chung :

|  |
| --- |
| <script language=”javascript”>  alert(“Hello World!”);  </script> |

Ngôn ngữ này có các đặc điểm sau

* Chương trình rất dễ học.
  + Những lỗi Javascript rất dễ để phát hiện, từ đó giúp bạn sửa lỗi một cách nhanh chóng hơn.
  + Những trình duyệt web có thể dịch thông qua HTML mà không cần sử dụng đến một compiler.
  + JS có thể hoạt động ở trên nhiều nền tảng và các trình duyệt web khác nhau.

Ngoài các ưu điểm trên ngôn ngữ này còn hạn chế sau

* + JS dễ bị các tin tặc khai thác hơn.
  + JS cũng không có khả năng đa luồng hoặc đa dạng xử lý.
  + Có thể được dùng để thực thi những mã độc ở trên máy tính của người sử dụng.
  + Những thiết bị khác nhau có thể sẽ thực hiện JS khác nhau, từ đó dẫn đến sự không đồng nhất.
  + Vì tính bảo mật và an toàn nên các Client-Side Javascript sẽ không cho phép đọc hoặc ghi các file.
  + JS không được hỗ trợ khi bạn sử dụng ở trong tình trạng thiết bị được kết nối mạng.

## 2.4 Tổng quan về cơ sở dữ liệu

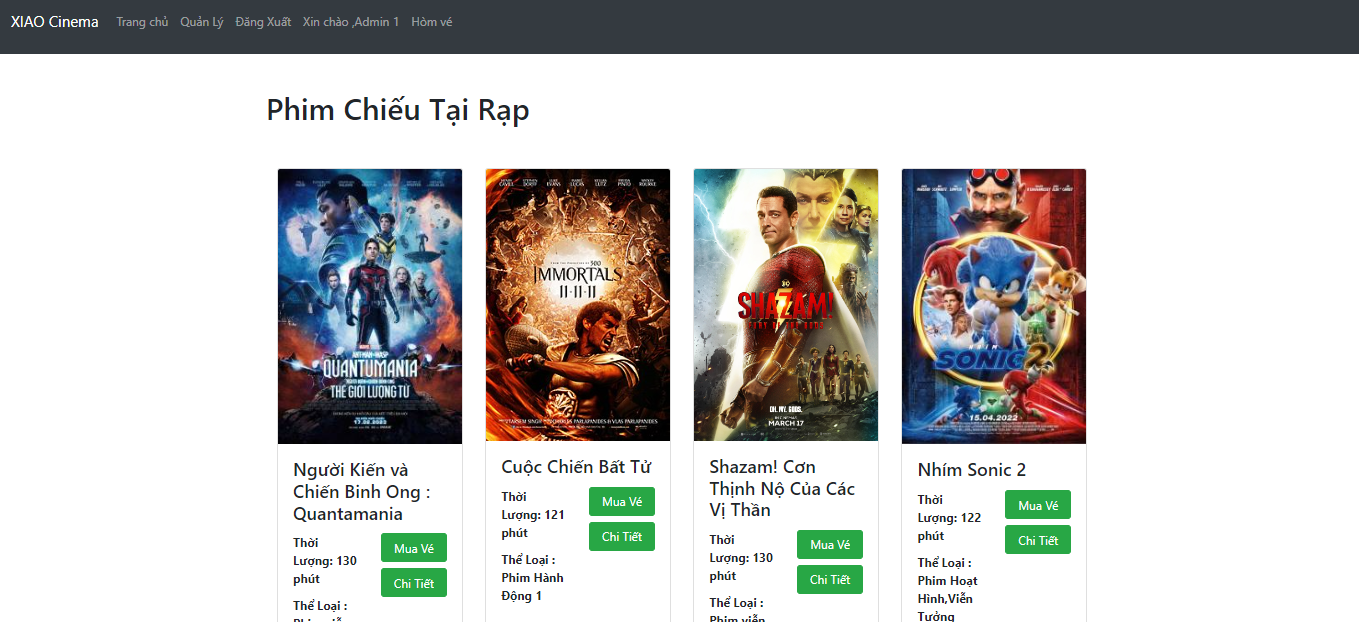
**Cơ sở dữ liệu** là một bộ sưu tập dữ liệu có hệ thống, được lưu trữ bằng điện tử và có thể chứa bất kỳ loại dữ liệu nào, bao gồm từ, số, hình ảnh, video và tệp. Cơ sở dữ liệu hiệu năng cao có ý nghĩa quan trọng đối với bất kỳ tổ chức nào. Cơ sở dữ liệu hỗ trợ các hoạt động nội bộ trong công ty và lưu trữ hoạt động tương tác với khách hàng cũng như nhà cung cấp.

**MySQL** là một hệ thống quản lý cơ sở dữ liệu quan hệ mã nguồn mở dựa trên ngôn ngữ truy vấn có cấu trúc SQL(Structor Query Language) được phát triển, phân phối và hỗ trợ bởi tập đoàn Oracle. Phần mềm này chạy trên hầu hết tất cả các nền tảng, bao gồm cả Linux UNIX và Windows. MySQL thường được kết hợp với các ứng dụng web.Hệ thống này hoạt động dựa trên mô hình khách – máy chủ,các câu lệnh truy vấn được gửi đến máy chủ thông qua máy khách đã cài đặt MySQL.

# CHƯƠNG III :CHƯƠNG TRÌNH QUẢN LÝ BÁN VÉ XEM PHIM TRỰC TUYẾN

## 3.1 Giao diện và chức năng

### Trang chủ



**Hình 3.1 Giao diện trang chủ**

Mã lệnh java :

@Override

**protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

resp.setContentType("text/html;charset=UTF-8");

MovieDAO mvdao = **new** MovieDAO();

**try** {

List<Movies> listmv = mvdao.getListMovie();

req.setAttribute("listmovie", listmv);

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

}

RequestDispatcher rd = req.getRequestDispatcher("/views/client/home.jsp");

rd.forward(req, resp);

}

Nguyên lý hoạt động

Khi người dùng truy cập vào đường dẫn url http://localhost:8080/Website\_Booking\_Movie\_Online/trangchu

DispatcherServlet sẽ nhận yêu cầu này từ người dùng và gọi đến Handler Mapping để chỉ dẫn cho DispatcherServlet gọi đến phương thức doGet của lớp ClientController rồi chuyển tiếp yêu cầu đến tầng phục vụ và lấy dữ liệu về các bộ phim thông qua lớp Movies thuộc gói Model

**public** List<Movies> getListMovie() **throws** ClassNotFoundException {

List<Movies> listmovie = **new** ArrayList<>();

String sql = "SELECT \* FROM movie";

**try** {

conn = **new** ConnectDB().getConnection();

statement = conn.prepareStatement(sql);

rs = statement.executeQuery();

**while** (rs.next()) {

listmovie.add(**new** Movies(rs.getLong(1),

rs.getString(2),

rs.getString(4),

rs.getString(3),

rs.getInt(5),

rs.getString(10),

rs.getString(11),

rs.getString(12),

rs.getDate(13),

rs.getString(14)));

}

} **catch** (SQLException e) {

e.printStackTrace();

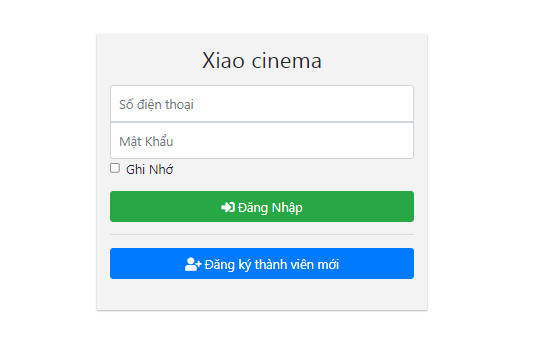
}

**return** listmovie;

}

Phương thức getListMovie() thuộc lớp MoviesDAO trong gói dao sẽ có nhiệm vụ giao tiếp với cơ sở dữ liệu để lấy ra toàn bộ danh sách các bộ phim chiếu tại rạp,phương thức này còn xử lý ngoại lệ về cơ sở dữ liệu và ngoại lệ không tìm thấy lớp kết nối với cơ sở dữ liệu các gói tin thuộc chương trình

### Giao diện đăng nhập



**Hình 3.2 Giao diện đăng nhập**

Mã lệnh Java:

* Phương thức service thuộc lớp LoginController

**protected** **void** service(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

resp.setContentType("text/html;charset=UTF-8");//dữ liệu HTML hiển thị tiếng việt

String number\_phoner=req.getParameter("number\_phone");//dữ liệu lấy từ HTML

String password=req.getParameter("pass");

String remember=req.getParameter("remember");

AccountS a=**null**;

AccountDAO Adao=**new** AccountDAO();

**try** {

Cookie number\_phone\_inCookie=**new** Cookie("number\_phone1", number\_phoner);//lưu trữ tài khoản

Cookie password\_inCookie=**new** Cookie("password", password);

a =Adao.Login(number\_phoner, password);

**if**(a == **null**) {

**if**(a == **null** &&number\_phoner !=**null**) {

req.setAttribute("mess","Tài khoản chưa đăng ký!");

}

req.getRequestDispatcher("/views/client/Login.jsp").forward(req, resp);

}

**else** {

**if**(remember !=**null**) {

number\_phone\_inCookie.setMaxAge(60\*60\*24\*365);

password\_inCookie.setMaxAge(60\*60\*24\*365);

}

**else** {

number\_phone\_inCookie.setMaxAge(0);

password\_inCookie.setMaxAge(0);

}

resp.addCookie(number\_phone\_inCookie);

resp.addCookie(password\_inCookie);

HttpSession ses=req.getSession();

ses.setAttribute("acc", a);

resp.sendRedirect("trangchu");

}

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

}

}

* Phương thức login() thuộc lớp AccountDAO

**public** AccountS Login(String numberphone, String password) **throws** ClassNotFoundException {

String sql = "SELECT \* FROM account where phone\_number =? and password=?";

**try** {

conn = **new** ConnectDB().getConnection();

statement = conn.prepareStatement(sql);

statement.setString(1, numberphone);

statement.setString(2, password);

rs = statement.executeQuery();

**while** (rs.next()) {

**return** **new** AccountS(

rs.getLong(1),

rs.getString(2),

rs.getString(3),

rs.getString(4),

rs.getLong(5));

}

} **catch** (SQLException e) {

e.printStackTrace();

}

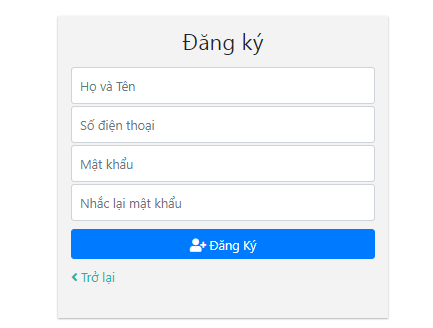
**return** **null**;

}

Nguyên lý hoạt động

Khi người dùng muốn thao tác mua vé trước hết hệ thống sẽ yêu cầu người dùng đăng nhập vào hệ thống, yêu cầu của người dùng được DispatcherServet nhận và cần sự trợ giúp của Handler Mapping để gọi đến lớp LoginController và thực hiện kiểm tra bởi tầng phục vụ nơi mà lớp AccountDAO sẽ được xử lý bởi phương thức Login(),đối chiếu dữ liệu người dùng nhập vào gồm tài khoản và mật khẩu tới cơ sở dữ liệu.Nếu tài khoản này đã được đăng ký thì được chuyển về trang chủ và dùng Session ghi nhận hành vi đăng nhập của người dùng,nếu người dùng được xác định là người quản trị thì sẽ được phép truy cập trang quản trị.Cookie là dữ liệu được lưu trên trình duyệt của người dùng để tiện ghi nhớ lần đăng nhập tiếp theo cho người dùng.nếu dữ liệu người dùng nhập vào chưa tồn tại trong cơ sở dữ liệu thì người dùng được yêu cầu đăng ký tài khoản

### Giao diện đăng ký



**Hình 3.3 Giao diện đăng ký**

Mã lệnh Java:

**protected** **void** service(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

resp.setContentType("text/html;charset=UTF-8");

req.setCharacterEncoding("UTF-8");

String user\_name = req.getParameter("user\_name");

String number\_phoner = req.getParameter("phone\_number");

String password = req.getParameter("pass");

String repass = req.getParameter("repass");

AccountDAO Adao = **new** AccountDAO();

**try** {

AccountS a = Adao.CheckAccountSingup(number\_phoner);

**if** (a == **null**) {

**if** (password.equals(repass)) {

Adao.SignUp(user\_name, number\_phoner,password);

req.setAttribute("mess3", "Đăng kí thành công!");

req.getRequestDispatcher("/views/client/Login.jsp").forward(req, resp);

} **else** {

req.setAttribute("mess1", "Mật khẩu nhập lại không khớp!");

req.getRequestDispatcher("/views/client/Login.jsp").forward(req, resp);}}

**else**{

req.setAttribute("mess2", "tài khoản này đã có người khác đăng kí rồi!!") req.getRequestDispatcher("/views/client/Login.jsp").forward(req, resp); }

System.***out***.println(a);

} **catch** (ClassNotFoundException e) {

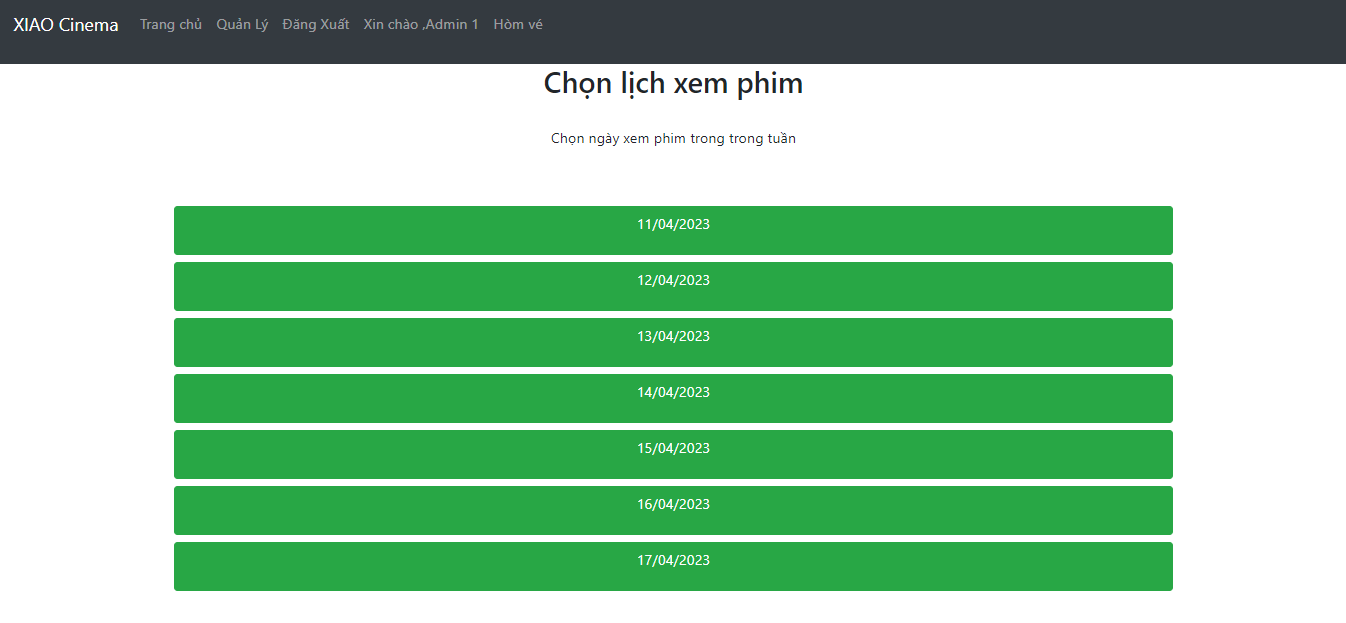
e.printStackTrace();

}}

Nguyên lý hoạt động:

Người dùng yêu cầu đăng ký tài khoản sẽ được DispatcherServlet nhận và đồng thời với sự trợ giúp của Handler Mapping để gọi đến lớp SignupController,trước hết với dữ liệu người dùng nhập vào thì thực hiện so sánh số điện thoại trong cơ sở dữ liệu ,nếu tài khoản này đã tồn tại thì yêu cầu người dùng nhập vào số điện thoại khác,nếu tài khoản chưa có trong cơ sở dữ liệu và người dùng nhập mật khẩu và nhập lại mật khẩu có sự trùng khớp thì cho phép đăng ký và chuyển về trang đăng nhập để tiếp tục đăng nhập.

* **Giao diện đặt lịch**





**Hình 3.5 Giao diện chọn lịch ngày và giờ**

* Phương thức doGet thuộc lớp ScheduController

**protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp)

**throws** ServletException, IOException {

resp.setContentType("text/html;charset=UTF-8");

ScheduleDAO schedao = **new** ScheduleDAO();

Movies movie;

MovieDAO dao = **new** MovieDAO();

String id\_movie=req.getParameter("movie\_id");

HttpSession ses=req.getSession();

**try** {

movie=dao.getMovieByID(id\_movie);

List<Schedule> listdate = schedao.getDate();

req.setAttribute("listDates", listdate);

ses.setAttribute("tenphim", movie);

} **catch** (Exception e) {

e.printStackTrace();

}

req.getRequestDispatcher("/views/client/schedule.jsp").forward(req, resp);

}

* Phương thức doget thuộc lớp ChoosetimeController

**protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp)

**throws** ServletException, IOException {

resp.setContentType("text/html;charset=UTF-8");

ScheduleDAO schedao = **new** ScheduleDAO();

Schedule schedule;

String DateID = req.getParameter("Date\_id");

HttpSession ses = req.getSession();

**try** {

List<Schedule> listdate = schedao.getDate();

req.setAttribute("listDates", listdate);

List<Time> listtime\_on\_movie = schedao.getListTimeByID(DateID);

schedule=schedao.getDateByID(DateID);

req.setAttribute("listStartTimes", listtime\_on\_movie);

ses.setAttribute("date", schedule);

} **catch** (Exception e) {

e.printStackTrace();

}

req.getRequestDispatcher("/views/client/choose\_time.jsp").forward(req , resp);

}

* Phương thức lấy ra danh sách thời gian

**public** List<Schedule> getDate() **throws** ClassNotFoundException {

List<Schedule> listdate = **new** ArrayList<>();

String sql = "SELECT \* FROM schedu";//câu lệnh truy vấn **try** {

conn = **new** ConnectDB().getConnection();//kết nối cơ sở dữ liệu

statement = conn.prepareStatement(sql);//truyền truy vấn đến cơ sở dữ liệu

rs = statement.executeQuery();//thực hiện truy vấn

**while** (rs.next()) {

listdate.add(**new** Schedule(

rs.getLong(1),

rs.getDate(2)));

}//lưu trữ kết quả lấy từ cơ sở dữ liệu

} **catch** (SQLException e) {

e.printStackTrace();

}

**return** listdate;}

**public** List<Time> getListTimeByID(String Date\_id) **throws** ClassNotFoundException {

List<Time> listtime = **new** ArrayList<>();

String sql="SELECT \* FROM schedu\_time where date\_id=?";

**try** {

conn=**new** ConnectDB().getConnection();

statement=conn.prepareStatement(sql);

statement.setString(1, Date\_id);

rs=statement.executeQuery();

**while**(rs.next()) {

listtime.add(**new** Time(

rs.getLong(1),

rs.getDate(2),

rs.getLong(3)));

}

} **catch** (SQLException e) {

e.printStackTrace();

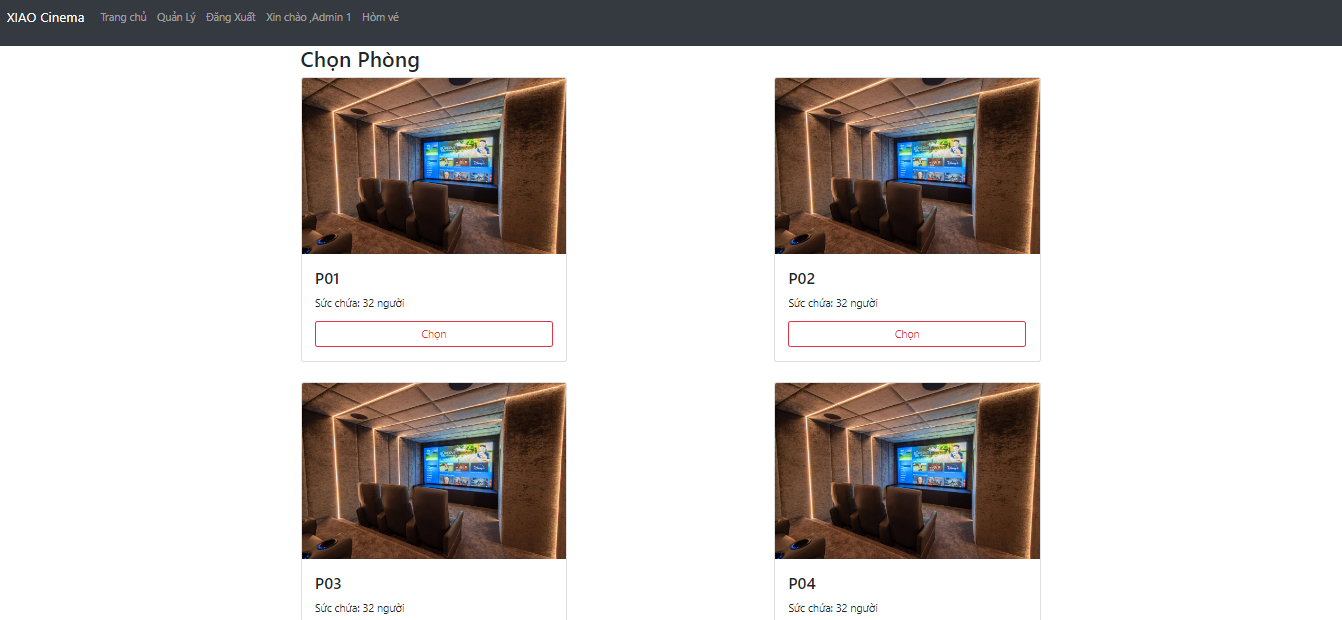
}

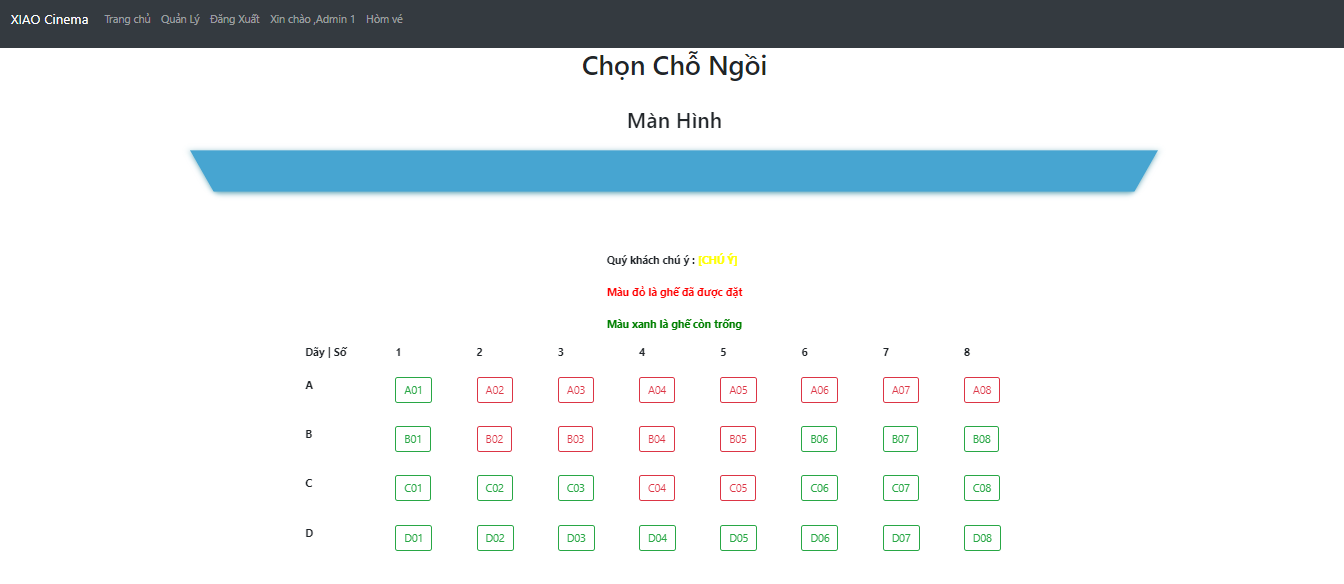
**return** listtime;

}

Sau khi người dùng chọn nút mua vé hệ thống sẽ trả về kết quả là trang chọn lịch ngày và giờ dựa trên cơ sở dữ liệu cố định mà người quản trị đã thêm vào,Hệ thống sẽ dùng session để ghi nhớ hành hi của người dùng đã chọn thông qua id của lịch ngày và giờ,sau khi chọn xong ngày và giờ hệ thống sẽ chuyển tiếp người dùng đến giao diện chọn phòng và ghế

### Giao diện chọn phòng và chọn ghế





**Hình 3.6 Giao diện chọn phòng và chọn**

Mã lệnh java :

* Phương thức doget thuộc lớp Roomcontroller

**protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp)

**throws** ServletException, IOException {

resp.setContentType("text/html;charset=UTF-8");

Time time;

String time\_id=req.getParameter("Time\_id");

ScheduleDAO Sdao=**new** ScheduleDAO();

RoomDAO Rdao = **new** RoomDAO();

HttpSession ses=req.getSession();

**try** {

List<Rooms> listroom = Rdao.getListRoom();

time=Sdao.getTimeByID(time\_id);

req.setAttribute("listRooms", listroom);

ses.setAttribute("time", time);

} **catch** (ClassNotFoundException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

RequestDispatcher rd = req.getRequestDispatcher("/views/client/room.jsp");

rd.forward(req, resp);

}

* Phương thức doget thuộc lớp Seatcontroller

**protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

resp.setContentType("text/html;charset=UTF-8");

SeatDAO Sdao=**new** SeatDAO();

String RoomID=req.getParameter("Rid");

Rooms room;

RoomDAO Rdao=**new** RoomDAO();

HttpSession ses1=req.getSession();

**try** {

room=Rdao.getRoomBYID(RoomID);

List<Seats> listseat=Sdao.getListSeatsBYID(RoomID);

req.setAttribute("listSeat", listseat);

ses1.setAttribute("room", room);

**int** a=(**int**) ses1.getAttribute("singleticket");

ses1.setAttribute("ok", a);

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

}

RequestDispatcher rd=req.getRequestDispatcher("/views/client/seats.jsp");

rd.forward(req, resp);

}

* Phương thức lấy danh sách phòng thuộc lớp RoomDAO

**public** List<Rooms> getListRoom() **throws** ClassNotFoundException {

List<Rooms> listroom = **new** ArrayList<>();

String sql = "SELECT \* FROM room";

**try** {

conn = **new** ConnectDB().getConnection();

statement = conn.prepareStatement(sql);

rs = statement.executeQuery();

**while** (rs.next()) {

listroom.add(**new** Rooms(

rs.getLong(1),

rs.getString(2),

rs.getInt(7)

));

}

} **catch** (SQLException e) {

e.printStackTrace();

}

**return** listroom;

}

* Phương thức lấy danh sách ghế thuộc lớp SeatDAO

**public** List<Seats> getListSeatsBYID(String Rid) **throws** ClassNotFoundException {

List<Seats> listseat = **new** ArrayList<>();

String sql = "SELECT \* FROM db\_bookticketmovie.seat\r\n" +

"where id\_room =?";

**try** {

conn = **new** ConnectDB().getConnection();

statement = conn.prepareStatement(sql);

statement.setString(1, Rid);

rs = statement.executeQuery();

**while** (rs.next()) {

listseat.add(**new** Seats(rs.getLong(1),

rs.getString(2),

rs.getInt(3),

rs.getInt(4)

));

}

} **catch** (SQLException e) {

e.printStackTrace();

}

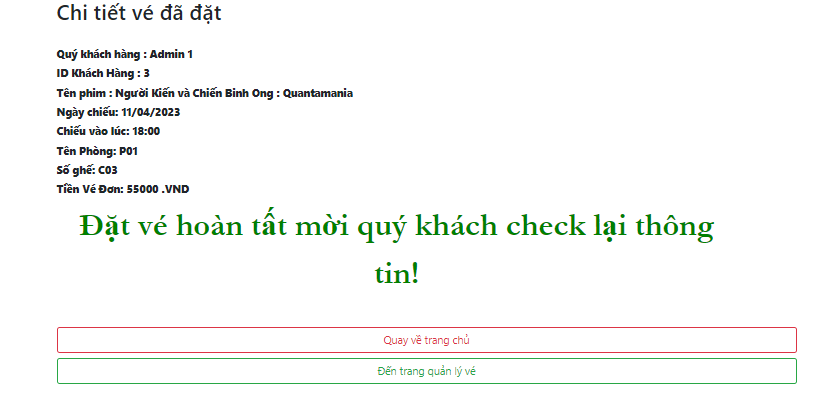
**return** listseat;

}

Cách thức hoạt động :

Người dùng thực hiện thao tác chọn phòng chiếu tại địa điểm chiếu thông qua mô phỏng của hệ thống,các phòng hiển thị thông tin và trạng thái của phòng đó,sau khi chọn được phòng hệ thống chuyển tiếp người dùng đến trang chọn ghế đã được đánh số và mô phỏng cho người dùng thấy được số hàng và dãy ghế đã được đánh dấu,sau khi chọn hoàn tất người dùng sẽ được kiểm tra lại thông tin về vé mình đã lên lịch kèm theo mệnh giá vé mà khách hàng phải tri trả khi đến nhận ghế để xem phim.

### Giao diện kiểm tra thông tin vé



**Hình 3.7 Giao diện kiểm tra lại thông tin vé**

* Phương thức doget thuộc lớp AddTicketController

**protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

String Cseats=req.getParameter("seats1");

Seats seat;

SeatDAO Sedao=**new** SeatDAO();

HttpSession sess=req.getSession();

TicketDAO ticketdao=**new** TicketDAO();

**try** {

seat=Sedao.getSeatsBYID(Cseats);

sess.setAttribute("seat", seat);

DateTimeFormatter dtf = DateTimeFormatter.*ofPattern*("dd/MM/yyyy");

DateTimeFormatter dtftime = DateTimeFormatter.*ofPattern*("HH:mm:ss");

LocalDateTime now = LocalDateTime.*now*();

AccountS user= (AccountS) sess.getAttribute("acc");//lấy dữ liệu về khi session đã ghi nhận thông tin từ hành vi của người dùng

Movies movie=(Movies) sess.getAttribute("tenphim");

Seats seat1=(Seats) sess.getAttribute("seat");

Rooms room=(Rooms) sess.getAttribute("room");

Schedule sche=(Schedule) sess.getAttribute("date");

Time time=(Time) sess.getAttribute("time");

**int** price=55000;

req.setAttribute("money", price);

TicketDAO tdao=**new** TicketDAO();

**if**(sess.getAttribute("seat")!= **null**) {

ticketdao.Getdata(user.getId(), user.getFullname(), movie.getTittle(), sche.getDate(),

time.getTime(), room.getRoom\_name(), seat1.getSeat\_name(), dtf.format(now),price,seat1.getId(),movie.getId(),dtftime.format(now));

}

**if**(seat1!=**null**) {

tdao.SetstatusChoese(Cseats);//ghế đã chọn thì được đánh dấu trạng thái đã được đặt

}

Ticket ticket\_unchoose=(Ticket) sess.getAttribute("ticketdel");

**if**(ticket\_unchoose!= **null**) {

tdao.SetstatusUnchose(Cseats);

};

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

} **catch** (SQLException e) {

}

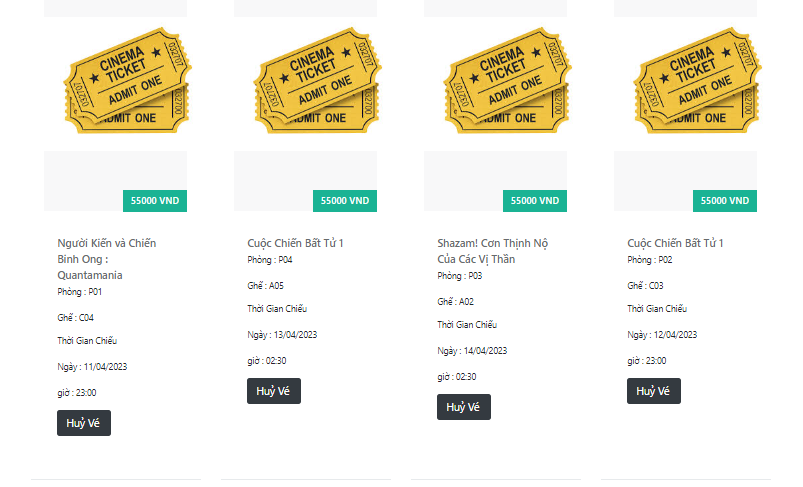
RequestDispatcher rd=req.getRequestDispatcher("views/client/checkinforticket.jsp");

rd.forward(req, resp);

}

Người dùng xác nhận lại thông tin mình đã yêu cầu ,hệ thống sẽ ghi nhận yêu cầu của người dùng và thêm dữ liệu vào bảng vé trong cơ sở dữ liệu được xử lý bởi phương thức Getdata() ,người dùng lựa chọn quay về trang chủ hoặc là đến trang quản lý vé mà mình đã đặt

### Giao diện trang quản lý vé của khách hàng



**Hình 3.8 Giao diện trang quản lý vé**

**protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

TicketDAO dao=**new** TicketDAO();

HttpSession ses=req.getSession();

AccountS acc=(AccountS) ses.getAttribute("acc");

Long acc\_id=acc.getId();

**int** toltalmony=0;

**int** singleticket=0;

**try** {

List<Ticket> listicket=dao.getListTicketBYID(acc\_id);

req.setAttribute("listticket", listicket);

ses.setAttribute("user", listicket);

Seats seat=(Seats) ses.getAttribute("seat");

**for**(Ticket ticket: listicket) {

singleticket=Integer.*parseInt*(ticket.getPrice());

toltalmony+=singleticket++;

String.*valueOf*(toltalmony);

}

**int** moneysingleticket=singleticket-1;

ses.setAttribute("toltal", toltalmony);

ses.setAttribute("singleticket", moneysingleticket);

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

}

req.getRequestDispatcher("/views/client/Bill.jsp").forward(req, resp);

}

* Phương thức lấy ra danh sách vé thông qua account\_id thuộc lớp TicketDAOs

**public** List<Ticket> getListTicketBYID(Long a) **throws** ClassNotFoundException {

List<Ticket> listticket = **new** ArrayList<>();

String sql = "SELECT \* FROM db\_bookticketmovie.ticket\r\n" +

"where account\_id =?";

**try** {

conn = **new** ConnectDB().getConnection();

statement = conn.prepareStatement(sql);

statement.setLong(1, a);

rs = statement.executeQuery();

**while** (rs.next()) {

listticket.add(**new** Ticket(

rs.getLong(1),

rs.getLong(2),

rs.getString(3),

rs.getString(4),

rs.getString(5),

rs.getString(6),

rs.getString(7),

rs.getString(8),

rs.getString(9),

rs.getString(10),

rs.getString(14)

));

}

} **catch** (SQLException e) {

e.printStackTrace();

}

**return** listticket;

}

* Phương thức deleteTicket() thông qua ticket\_id thuộc lớp TicketDAOs

**public** Ticket deleleTicket(String ticket\_id) **throws** SQLException {

String sql="delete FROM db\_bookticketmovie.ticket where idTicket=?";

**try** {

conn=**new** ConnectDB().getConnection();

statement=conn.prepareStatement(sql);

statement.setString(1, ticket\_id);

statement.executeUpdate();

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

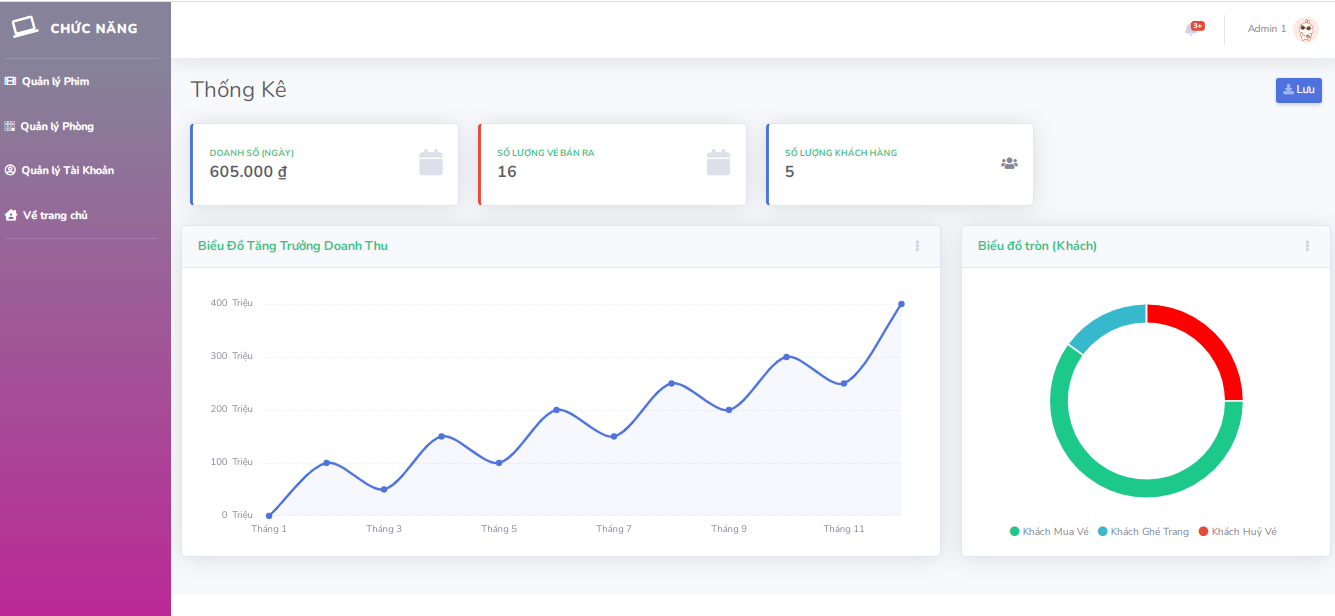
}

**return** ticket;

}

Người dùng xem những vé mình đã đặt tại trang quản lý vé .Lớp Billcontroller sẽ gọi đến phương thức lấy ra danh sách vé bởi id của tài khoản đang đăng nhập và có thể thao tác huỷ vé bởi phương thức deletetickit() khi không muốn xem bộ phim đó,Thông tin vé được lưu tại hình 3.8 sẽ là được nhân viên kiểm tra khi khách hàng đến nhận ghế xem phim tại Rạp chiếu,đây là hình thức ghi nợ của hệ thống cho đến khi khách hàng thanh toán vé của bộ phim này

### Giao diện trang quản trị



**Hình 3.9 Giao diện trang quản trị**

* Phương thức doGet thuộc lớp HomeAdminController

**protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

resp.setContentType("text/html;charset=UTF-8");

req.getRequestDispatcher("/views/admin/home.jsp").forward(req, resp);

AccountDAO AccDao = **new** AccountDAO();

TicketDAO ticketdao = **new** TicketDAO();

List<Ticket> listicket;

List<Ticket> listPrice;

List<AccountS> Listacc;

**int** count = 0;

String date = "28/03/2023";

String getMoney = **null**;

**int** countnumber = 0;

**int** earnmoneyday = 0;

HttpSession session = req.getSession();

**try** {

listPrice = ticketdao.getpriceBydate(date);

**for** (Ticket t : listPrice) {

getMoney = t.getPrice();

countnumber = Integer.*parseInt*(getMoney);

earnmoneyday += countnumber++;

}

Locale localeVN = **new** Locale("vi", "VN");

NumberFormat currencyVN = NumberFormat.*getCurrencyInstance*(localeVN);

String moneytostring = currencyVN.format(earnmoneyday);

listicket = ticketdao.gettotalticket();

Listacc = AccDao.getListAcc();

session.setAttribute("toltalearnmoney\_day", moneytostring);

session.setAttribute("toltalticket", listicket.size());

**for** (AccountS acc : Listacc) {

**if** (acc.getRoleid() == 0) {

count++;

}

session.setAttribute("listsize", count);

}

} **catch** (ClassNotFoundException e) {

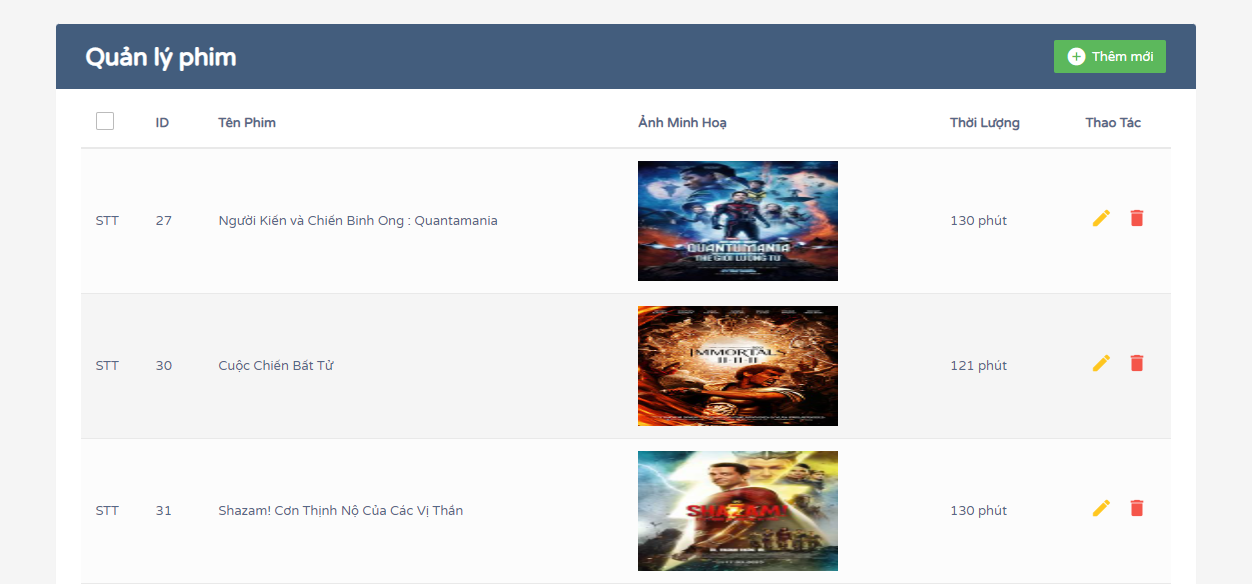
e.printStackTrace();

}

}

Người dùng được phân quyền quản lý sẽ được yêu cầu truy cập trang quản trị của hệ thống,giao diện này thống kê số lượng khách hàng,số lượng vé bán và doanh thu theo ngày mà rạp chiếu đạt được.Ngoài ra còn có chức năng quản lý phim,quản lý phòng chiếu và quản lý tài khoản người dùng

### Giao diện trang quản lý phim



**Hình 3.10 Giao diện trang quản lý phim**

* Phương thức doget thuộc lớp AdminmovieController

**protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

resp.setContentType("text/html;charset=UTF-8");

MovieDAO mvdao=**new** MovieDAO();

List<Movies> listmv;

**try** {

listmv = mvdao.getListMovie();

req.setAttribute("listmovie", listmv);

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

}

req.getRequestDispatcher("/views/admin/Movie.jsp").forward(req, resp);

}

* Phương thức doget dùng để xoá phim

**protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

String Mid=req.getParameter("mid");

MovieDAO dao=**new** MovieDAO();

dao.deleteMovie(Mid);

resp.sendRedirect("adminMovie");

}

**public** **void** deleteMovie(String id) {

String sql="delete from movie where id =?";

**try** {

conn=**new** ConnectDB().getConnection();

statement=conn.prepareStatement(sql);

statement.setString(1, id);

statement.executeUpdate();

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

} **catch** (SQLException e) {

e.printStackTrace();} }

* Phương thức cập nhật phim

**protected** **void** service(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

req.setCharacterEncoding("UTF-8");//truyền dữ liệu dạng encode

MovieDAO mdao = **new** MovieDAO();//cấp phát bộ nhớ

String mid = req.getParameter("mid");//truyền dữ liệu từ HTML

String name\_movie = req.getParameter("name\_movie");

String image = req.getParameter("image");

String decription = req.getParameter("description");

String totaltime = req.getParameter("total\_time");

String actor = req.getParameter("actors");

String director = req.getParameter("director");

String languege = req.getParameter("language");

String date = req.getParameter("date");

String long\_decription = req.getParameter("long\_decrip");

mdao.EditMovie(name\_movie, image, decription, totaltime, actor, director, languege,

date,long\_decription, mid);//truyền dữ liệu vào cơ sở dữ liệu

resp.sendRedirect("adminMovie");//về trang quản lý phim

}

* Phương thức thêm phim

**protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

resp.setContentType("text/html;charset=UTF-8");

req.setCharacterEncoding("UTF-8"); //truyền dữ liệu dạng encode UTF-8

String name\_movie=req.getParameter("name\_movie");

String image=req.getParameter("image");//truyền dữ liệu từ HTML

String decription=req.getParameter("description");

String totaltime=req.getParameter("total\_time");

String actor=req.getParameter("actors");

String director=req.getParameter("director");

String languege=req.getParameter("language");

String date=req.getParameter("date");

String long\_decription = req.getParameter("long\_decrip");

MovieDAO dao=**new** MovieDAO();

dao.AddMovie(name\_movie, image, decription, totaltime, actor, director, languege, date,

long\_decription);//truyền dữ liệu vào cơ sở dữ liệu

resp.sendRedirect("adminMovie");//về trang quản lý phim

}

**public** **void** AddMovie(String name\_movie,String image,String decription,

String totaltime,String actor,String director,String languege,String date,String long\_decription) {

String sql="insert into movie(name\_movie,image,decription"

+ ",total\_time,actors,director,language,durian) values (?,?,?,?,?,?,?,?)";//chuỗi truyền câu lệnh truy vấn

**try** {

conn=**new** ConnectDB().getConnection(); //kết nối cơ sở dữ liệu

statement=conn.prepareStatement(sql);//truyền câu lệnh truy vấn đến cơ sở dữ liệu

statement.setString(1, name\_movie);

statement.setString(2, image);

statement.setString(3, decription);

statement.setString(4, totaltime);

statement.setString(5, actor);

statement.setString(6, director);

statement.setString(7, languege);

statement.setString(8, date);

statement.setString(9, long\_decription);//thêm dữ liệu cho từng cột đến cơ sở dữ liệu

statement.executeUpdate();//thực hiện truy vấn

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

} **catch** (SQLException e) {

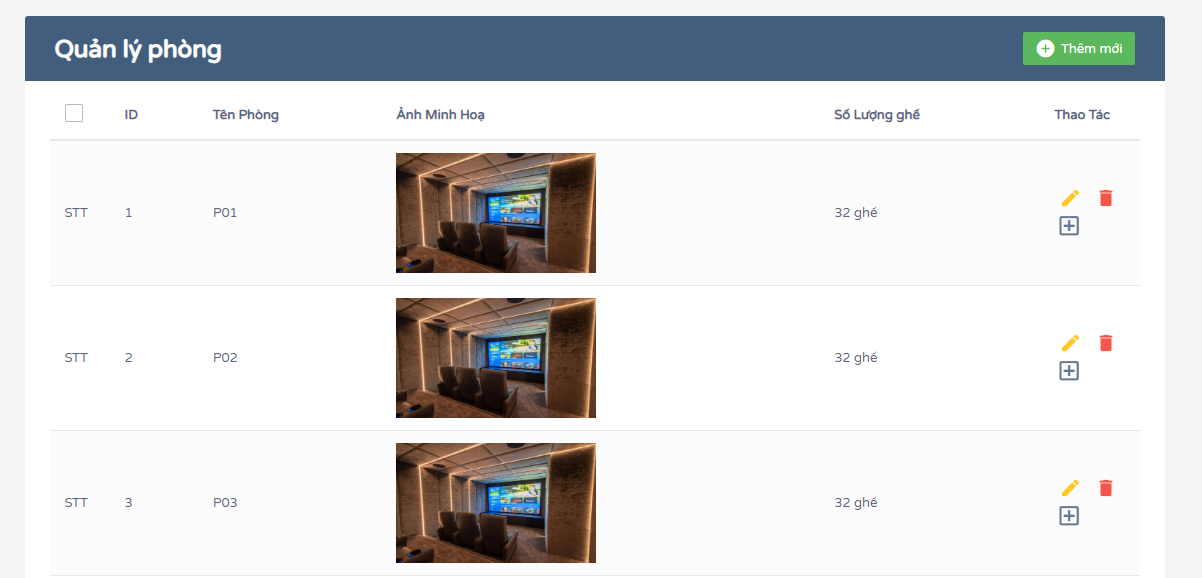
e.printStackTrace();

}

}

Người quản trị khi thực hiện chọn nút quản lý phim sẽ được hệ thống trả về trang quản lý phim,trang này gồm danh sách phim và thao tác thêm,cập nhật,xoá phim trên hệ thống

### Giao diện trang quản lý phòng chiếu



**Hình 3.11 Giao diện trang quản lý phòng chiếu**

Người quản trị yêu cầu đến trang quản lý phòng và thực hiện các thao tác .ví dụ như thêm phòng chiếu,mỗi phòng sẽ được thêm ghế,cập nhật lại thông tin phòng và xoá phòng

**protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

resp.setContentType("text/html;charset=UTF-8");

RoomDAO Rdao=**new** RoomDAO();

List<Rooms> listRoom;

**try** {

listRoom = Rdao.getListRoom();

req.setAttribute("listroom", listRoom);

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

}

req.getRequestDispatcher("/views/admin/Room.jsp").forward(req, resp);

}

* Phương thức thêm phòng

**protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

resp.setContentType("text/html;charset=UTF-8");

req.setCharacterEncoding("UTF-8");

String name\_room=req.getParameter("name\_room");

String capa=req.getParameter("capa");

RoomDAO Rdao=**new** RoomDAO();

Rdao.AddRoom(name\_room, capa);

resp.sendRedirect("adminRoom");

}

**public** **void** AddRoom(String room\_name,String capacity) {

String sql="insert into room(name\_room,capacity) values (?,?)";

**try** {

conn=**new** ConnectDB().getConnection();

statement=conn.prepareStatement(sql);

statement.setString(1, room\_name);

statement.setString(2, capacity);

statement.executeUpdate();

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

} **catch** (SQLException e) {

e.printStackTrace();

}

}

* Phương thức xoá phòng

**protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

String Mid=req.getParameter("mid");

MovieDAO dao=**new** MovieDAO();

dao.deleteMovie(Mid);

resp.sendRedirect("adminMovie");

}

**public** **void** deleteRoom(String id) {

String sql="delete from room where id =?";

**try** {

conn=**new** ConnectDB().getConnection();

statement=conn.prepareStatement(sql);

statement.setString(1, id);

statement.executeUpdate();

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

} **catch** (SQLException e) {

e.printStackTrace();

}

* Phương thức thêm ghế :

**public** **void** AddSeatFromRoom(String name\_seat,String status,String id\_room) {

String sql="insert into seat(name\_seat,status,id\_room) values (?,?,?);";

**try** {

conn=**new** ConnectDB().getConnection();

statement=conn.prepareStatement(sql);

statement.setString(1, name\_seat);

statement.setString(2, status);

statement.setString(3, id\_room);

statement.executeUpdate();

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

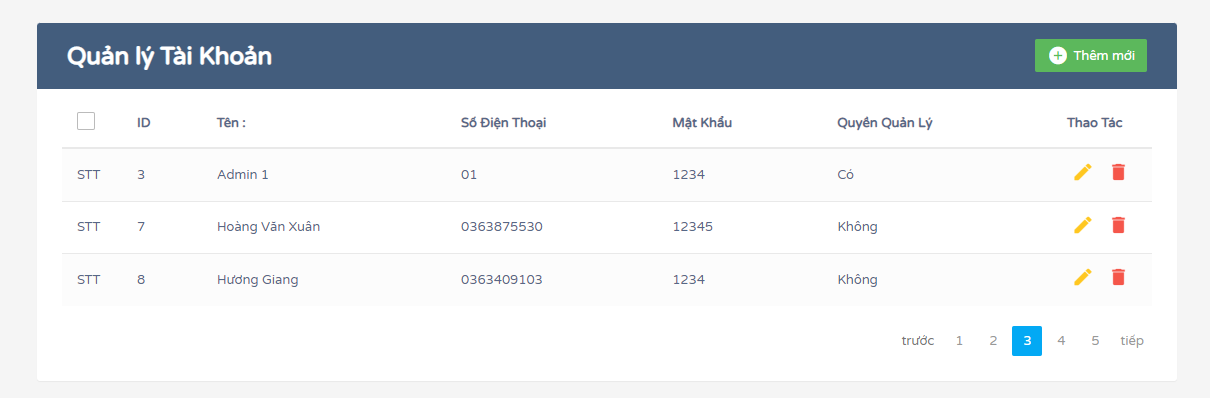
} **catch** (SQLException e) {

e.printStackTrace();

}

}

### Giao diện trang quản lý tài khoản



**Hình 3.12 Giao diện trang quản lý tài khoản**

**protected** **void** doGet(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

resp.setContentType("text/html;charset=UTF-8");

req.setCharacterEncoding("UTF-8");

AccountDAO AccDao=**new** AccountDAO();

List<AccountS> Listacc;

**try** {

Listacc = AccDao.getListAcc();

req.setAttribute("listacc", Listacc);

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

}

req.getRequestDispatcher("views/admin/account.jsp").forward(req, resp);

}

* Thêm tài khoản mới:

**protected** **void** doPost(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

resp.setContentType("text/html;charset=UTF-8");

req.setCharacterEncoding("UTF-8");

String name=req.getParameter("fullname");

String sdt=req.getParameter("phone");

String pass=req.getParameter("pass");

String roleid=req.getParameter("roleid");

AccountDAO Asdao=**new** AccountDAO();

**try** {

Asdao.AddAccount(name, sdt, pass, roleid);

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

}

resp.sendRedirect("adminAcccount");

}

**public** **void** AddAccount(String user\_name,String number\_phone, String password,String roleid) **throws** ClassNotFoundException {

String sql = "insert into account(fullname,phone\_number,password,role\_id) values(?,?,?,?)";

**try** {

conn = **new** ConnectDB().getConnection();

statement = conn.prepareStatement(sql);

statement.setString(1, user\_name);

statement.setString(2, number\_phone);

statement.setString(3, password);

statement.setString(4, roleid);

statement.executeUpdate();

} **catch** (SQLException e) {

e.printStackTrace();

}

* Xoá tài khoản :

**protected** **void** service(HttpServletRequest req, HttpServletResponse resp) **throws** ServletException, IOException {

String Aid=req.getParameter("Aid");

AccountDAO Adao=**new** AccountDAO();

Adao.DeleteAccout(Aid);

resp.sendRedirect("adminAcccount");

* }

**public** **void** DeleteAccout(String id) {

String sql="delete from account where id=?";

**try** {

conn = **new** ConnectDB().getConnection();

statement = conn.prepareStatement(sql);

statement.setString(1, id);

statement.executeUpdate();

} **catch** (SQLException e) {

e.printStackTrace();

} **catch** (ClassNotFoundException e) {

e.printStackTrace();

}

}

Người quản trị sẽ thực hiện chức năng thêm mới tài khoản bởi phương thức AddAcount ,xoá tài khoản del của người dùng tại trang này.Chức năng thêm mới và cập nhật thì người quản trị có quyền được phân quyền cho người dùng có quyền quản lý hoặc là khách hàng.

# CHƯƠNG IV: KẾT LUẬN

Trong quá trình tìm hiểu,khảo sát và thực hiện đề tài xây dựng hệ thống quản lý bán vé trực tuyến em đã tìm hiểu được về ngôn ngữ lập trình website, phân tích thiết kế hệ thống ,quản lý dữ liệu,sử dụng những công nghệ giúp lập trình website ,tích luỹ những kinh nghiệm quý báu mà giảng viên hướng dẫn đã chỉ bảo và cá nhân em đã đạt được trong quá trình xây dựng hệ thống để giúp ích cho công việc em đam mê sau khi hoàn thành chương trình học tại giảng đường .

Đối với đề tài em đã thực hiện quá trình khảo sát,đánh giá tiềm năng của việc phát triển công nghệ quản lý hệ thống rạp chiếu phim trong tương lai,em đã cài đặt thành công chương trình quản lý bán vé phim trực tuyến .Một ứng dụng website dễ sử dụng đối với người dùng và dễ quản lý đối với người quản trị hệ thống rạp chiếu.Sau khi kiểm thử và sử dụng hệ thống bởi cá nhân em và các bạn được em mời sử dùng hệ thống,Hệ thống được đánh giá về sự ổn định,tính bảo mật cao,thuận tiện và dễ dàng sử dụng,khả năng tương tác cao với người dùng,

Ngoài những gì mà em đã đạt được trong khi thực hiện đề tài thì em còn gặp những khó khăn nhất định khi chưa cài đặt được chức năng thanh toán điện tử,hộp trò chuyện để tương tác đến những đánh giá của người dùng đối với hệ thống,chức năng liên kết với các mạng xã hội được nhiều người dùng để giúp việc đăng nhập hệ thống thuận tiện hơn.

Trong thời gian tới khi đã tìm hiểu được thêm những kiến thức mới em sẽ cải thiện hệ thống và cập nhật lên phiên bản ứng dụng website 2.0 với những chức năng em chưa đạt được trong phiên bản 1.0 đó là cài đặt chức năng thanh toán trực tuyến với tài khoản ngân hàng,ví điện tử,tích hợp chức năng đăng nhập thông qua mạng xã hội,giảm thiểu khả năng xảy ra lỗi của hệ thống,và xây dựng hộp trò chuyện đối với người quản lý và người dùng

# TÀI LIỆU THAM KHẢO

1. Kurniawan , B .(2015) .*Servlet & JSP: A Tutorial ,Second Edition.*

LonDon: BrainySoftware.