**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

# **LỜI CẢM ƠN**

Em xin chân thành cảm ơn ThS. Đoàn Phước Miền – giảng viên hướng dẫn đã tận tình hỗ trợ, định hướng chuyên môn và đưa ra những góp ý thiết thực trong suốt quá trình thực hiện đề tài *“Xây dựng website đặt vé xem phim”*. Sự hướng dẫn của thầy là nguồn động lực lớn giúp em hoàn thành tốt đề tài này.

Em cũng xin cảm ơn quý thầy cô và nhà trường đã tạo điều kiện học tập, cung cấp kiến thức nền tảng quý báu trong suốt quá trình học.

Xin cảm ơn các bạn sinh viên và người dùng đã tham gia khảo sát, đóng góp ý kiến để em có thêm dữ liệu thực tế phục vụ cho quá trình xây dựng hệ thống.

Cuối cùng, em xin gửi lời cảm ơn đến gia đình và bạn bè đã luôn ủng hộ, tạo điều kiện để em hoàn thành đề tài.

Dù đã cố gắng, đề tài vẫn có thể còn thiếu sót. Em mong nhận được những góp ý để hoàn thiện hơn trong tương lai.

**Người thực hiện**

MỤC LỤC

[LỜI CẢM ƠN iii](#_Toc209357908)

[DANH MỤC BẢNG v](#_Toc209357909)

[MỞ ĐẦU 1](#_Toc209357910)

[1. Lý do chọn đề tài 1](#_Toc209357911)

[2. Mục tiêu và phạm vi nghiên cứu 1](#_Toc209357912)

[2.1. Mục tiêu nghiên cứu 1](#_Toc209357913)

[2.2. Mục tiêu nghiên cứu 2](#_Toc209357914)

[3. Đối tượng và phạm vi nghiên cứu 2](#_Toc209357915)

[4. Phương pháp nghiên cứu 3](#_Toc209357916)

[CHƯƠNG I: TỔNG QUAN 4](#_Toc209357917)

[1.1. Giới thiệu chung 4](#_Toc209357918)

[1.2. Lý do chọn đề tài 4](#_Toc209357919)

[1.3. Mục tiêu nghiên cứu 4](#_Toc209357920)

[1.4. Đối tượng và phạm vi nghiên cứu 4](#_Toc209357921)

[CHƯƠNG II: NGHIÊN CỨU LÝ THUYẾT 6](#_Toc209357922)

[2.1. ASP.NET và MVC 6](#_Toc209357923)

[2.1.1. ASP.NET 6](#_Toc209357924)

[2.1.2. MVC 7](#_Toc209357925)

[2.2. Javascript 11](#_Toc209357926)

[2.3. Bootstrap 13](#_Toc209357927)

[2.4. Hệ cơ sở dữ liệu SQL Server 14](#_Toc209357928)

[2.5. Truy vấn LINQ 15](#_Toc209357929)

[2.5.1. LINQ là gì? 15](#_Toc209357930)

[*2.5.2.* Điểm mạnh của LINQ 15](#_Toc209357931)

[2.5.3. Các thành phần của LINQ 15](#_Toc209357932)

[CHƯƠNG III: HIỆN THỰC HÓA NGHIÊN CỨU 22](#_Toc209357933)

[3.1. Xác định thành phần và chức năng chính của hệ thống 22](#_Toc209357934)

[3.2. Phân tích vai trò người dùng trong hệ thống 23](#_Toc209357935)

[3.3. Sơ đồ cơ sở dữ liệu 24](#_Toc209357936)

[3.4. Cơ sở dữ liệu 24](#_Toc209357937)

[3.5. Sơ đồ use case 27](#_Toc209357938)

[CHƯƠNG IV: KẾT QUẢ NGHIÊN CỨU 31](#_Toc209357939)

[4.1. Màn hình giao diện chính 31](#_Toc209357940)

[4.2. Màn hình đăng nhập 31](#_Toc209357941)

[4.3. Màn hình đăng ký 31](#_Toc209357942)

[4.4. Màn hình chi tiết phim 32](#_Toc209357943)

[4.5. Màn hình danh sách phim 32](#_Toc209357944)

[4.6. Màn hình trang quản lý phim 33](#_Toc209357945)

[4.7. Màn hình trang thêm phim mới 33](#_Toc209357946)

[4.8. Màn hình quản lý lịch chiếu 34](#_Toc209357947)

[CHƯƠNG V: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 35](#_Toc209357948)

[5.1. Kết luận 35](#_Toc209357949)

[5.2. Hướng phát triển 35](#_Toc209357950)

[DANH MỤC TÀI LIỆU THAM KHẢO 36](#_Toc209357951)

# **DANH MỤC BẢNG**

[Bảng 3.1. User – Dùng để lưu thông tin người dùng 18](#_Toc203291277)

[Bảng 3.2. Movie – Dùng để quản lý thông tin về phim 18](#_Toc203291278)

[Bảng 3.3. Theater – Dùng để lưu trữ dữ liệu của rạp phim 18](#_Toc203291279)

[Bảng 3.4. Showtime – Dùng để lưu thông tin về suất chiếu 19](#_Toc203291280)

[Bảng 3.5. Ticket – Dùng để lưu dữ liệu của vé 19](#_Toc203291281)

**MỞ ĐẦU**

## **Lý do chọn đề tài**

* **Tính cấp thiết:** Trong thời đại công nghệ số, nhu cầu xem phim tại rạp ngày càng tăng cao. Tuy nhiên, việc đặt vé truyền thống gây mất thời gian và không thuận tiện trong việc chọn suất chiếu, vị trí ghế ngồi hay thanh toán.
* **Xu hướng công nghệ:** Các hệ thống đặt vé trực tuyến đang dần trở thành chuẩn mực trong ngành giải trí, tương tự như các nền tảng lớn Grab, Shopee, hay các rạp chiếu phim CGV, Galaxy đã triển khai. Việc xây dựng một hệ thống đặt vé hiện đại là cần thiết để bắt kịp xu hướng.
* **Ứng dụng thực tiễn cao:** Website đặt vé có thể được triển khai tại các rạp phim quy mô vừa và nhỏ, phục vụ khách hàng đặt vé nhanh chóng, quản lý lịch chiếu, ghế ngồi, và thanh toán dễ dàng.
* **Phù hợp chuyên môn:** Đề tài vận dụng tốt kiến thức đã học về lập trình web, cơ sở dữ liệu, mô hình MVC và kỹ thuật xử lý thông tin – giúp sinh viên rèn luyện kỹ năng phân tích, thiết kế và xây dựng hệ thống thực tế.

## **Mục tiêu** **và phạm vi nghiên cứu**

### **Mục tiêu nghiên cứu**

Dự án website đặt vé xem phim trực tuyến được thực hiện với các mục tiêu chính sau:

* **Xây dựng nền tảng đặt vé hiện đại và thân thiện**  
  Phát triển một hệ thống website đặt vé xem phim trực tuyến giúp khách hàng dễ dàng tiếp cận thông tin phim, chọn lịch chiếu, chọn ghế, và tiến hành đặt vé một cách thuận tiện nhất mà không cần phải đến trực tiếp rạp.
* **Tối ưu hóa trải nghiệm người dùng**  
  Thiết kế giao diện người dùng trực quan, dễ thao tác với bố cục hợp lý, màu sắc dễ nhìn, biểu tượng rõ ràng, hỗ trợ hiển thị tốt trên cả máy tính và thiết bị di động. Qua đó giúp người dùng tiết kiệm thời gian, tăng mức độ hài lòng và sẵn sàng quay lại sử dụng dịch vụ.
* **Tích hợp chức năng quản trị cho rạp phim**  
  Hệ thống quản trị giúp các rạp chiếu phim dễ dàng thực hiện các công việc nghiệp vụ như cập nhật lịch chiếu, quản lý phòng chiếu, theo dõi vé đã bán, phân tích doanh thu, phân quyền nhân viên và xử lý các tình huống phát sinh từ phía khách hàng.
* **Hỗ trợ vận hành và mở rộng hệ thống trong tương lai**  
  Thiết kế cấu trúc hệ thống linh hoạt, có khả năng tích hợp thêm các chức năng mở rộng như thanh toán online, mã giảm giá, hệ thống đánh giá, hoặc ứng dụng mobile để bắt kịp xu hướng công nghệ và nhu cầu thực tế.

### **Mục tiêu nghiên cứu**

* **Người dùng:** khách hàng có nhu cầu xem phim tại rạp, quản trị viên hệ thống.
* **Dữ liệu:** thông tin phim, suất chiếu, phòng chiếu, ghế ngồi, đơn đặt vé, thông tin người dùng.
* **Công nghệ sử dụng:** ASP.NET Core MVC, SQL Server, HTML/CSS, Bootstrap, JavaScript, Entity Framework..

## **Đối tượng và phạm vi nghiên cứu**

**Yêu cầu chức năng:**

* **Đăng ký và đăng nhập tài khoản người dùng**  
  Cho phép người dùng tạo tài khoản mới, đăng nhập để sử dụng các chức năng như đặt vé, lưu lịch sử giao dịch, cập nhật thông tin cá nhân.
* **Hiển thị danh sách phim và lịch chiếu**  
  Cung cấp thông tin phim chi tiết bao gồm trailer, mô tả, đạo diễn, thể loại,... cùng với danh sách lịch chiếu theo ngày, giờ và rạp.
* **Chọn suất chiếu, chọn ghế, đặt vé**  
  Người dùng có thể chọn suất chiếu mong muốn, xem sơ đồ ghế và đặt vé trực tuyến. Ghế được cập nhật trạng thái theo thời gian thực.
* **Quản trị hệ thống**  
  Quản trị viên có thể:
* Quản lý phim: thêm/sửa/xóa phim.
* Quản lý lịch chiếu: cấu hình và cập nhật suất chiếu.
* Quản lý người dùng: phân quyền, khóa/mở tài khoản.
* Thống kê doanh thu theo ngày, tuần, tháng hoặc phim cụ thể.

**Yêu cầu phi chức năng:**

* Giao diện dễ sử dụng.
* Hệ thống hoạt động ổn định, phản hồi nhanh.
* Cấu trúc hệ thống dễ nâng cấp và bảo trì.

## **Phương pháp nghiên cứu**

* **Khảo sát thực tế**: Nghiên cứu cách hoạt động của các hệ thống đặt vé hiện có (CGV, Galaxy...).
* **Phân tích hệ thống**: Xác định rõ các yêu cầu chức năng và phi chức năng cho hệ thống.
* **Thiết kế hệ thống**: Sử dụng mô hình MVC, xây dựng sơ đồ luồng dữ liệu, cơ sở dữ liệu, giao diện người dùng.
* **Lập trình và triển khai**: Xây dựng các chức năng chính và kiểm thử từng phần để đảm bảo tính ổn định.
* **Đánh giá**: Thử nghiệm với người dùng thực tế, thu thập phản hồi để cải tiến hệ thống.

**CHƯƠNG I: TỔNG QUAN**

## **Giới thiệu chung**

Trong bối cảnh công nghệ thông tin và Internet phát triển mạnh mẽ, nhu cầu giải trí trực tuyến ngày càng tăng cao. Trong đó, xem phim trực tuyến đã trở thành một xu hướng phổ biến, mang lại sự tiện lợi và trải nghiệm đa dạng cho người dùng. Các nền tảng xem phim không chỉ giúp người dùng tiếp cận nhanh chóng các bộ phim yêu thích mà còn tạo ra môi trường giải trí hiện đại, thân thiện và dễ sử dụng.

## **Lý do chọn đề tài**

Hiện nay, tại Việt Nam có nhiều trang web cung cấp dịch vụ xem phim, tuy nhiên vẫn còn tồn tại một số hạn chế như giao diện chưa tối ưu, tốc độ tải chậm, hoặc thiếu tính năng cá nhân hóa. Vì vậy, việc xây dựng một website xem phim với giao diện hiện đại, tốc độ nhanh, thân thiện với người dùng và có hệ thống quản lý phim khoa học là rất cần thiết**.**

## **Mục tiêu nghiên cứu**

* Xây dựng một website xem phim trực tuyến đáp ứng nhu cầu giải trí của người dùng.
* Thiết kế giao diện thân thiện, dễ sử dụng, phù hợp trên cả máy tính và thiết bị di động.
* Tích hợp các chức năng quản lý như: tìm kiếm phim, phân loại theo thể loại, năm phát hành, quốc gia, diễn viên.
* Cung cấp tính năng tài khoản cá nhân để lưu danh sách phim yêu thích và lịch sử xem.

## **Đối tượng và phạm vi nghiên cứu**

* Đối tượng nghiên cứu: Người dùng có nhu cầu xem phim trực tuyến.
* Phạm vi nghiên cứu:
  + Hệ thống quản lý phim: thêm, sửa, xóa phim.
  + Chức năng tìm kiếm, lọc và phân loại phim.
  + Giao diện người dùng: xem phim, đánh giá, bình luận.
  + Chưa đi sâu vào vấn đề bản quyền phim và hệ thống thanh toán trực tuyến.

**CHƯƠNG II: NGHIÊN CỨU LÝ THUYẾT**

Hiện nay, việc xây dựng và phát triển website có thể được thực hiện bằng nhiều ngôn ngữ lập trình khác nhau như JavaScript (MeteorJS, AngularJS), C#, Java, PHP,... Sau khi tham khảo ý kiến của những người đi trước cũng như dựa trên kiến thức tích lũy trong quá trình học tập, em đã lựa chọn phát triển website bằng ASP.NET với mô hình MVC 8.

Về phần giao diện, em sử dụng framework Bootstrap 5 nhằm rút ngắn thời gian thiết kế và tận dụng các ưu điểm sẵn có của framework này. Ngoài ra, em còn sử dụng JavaScript kết hợp với jQuery, Ajax để tối ưu hiệu suất xử lý trên giao diện người dùng. Về cơ sở dữ liệu, em sử dụng SQL Server, kết nối thông qua Entity Framework và sử dụng LINQ để truy xuất dữ liệu thay vì viết câu lệnh SQL thuần.

* 1. **ASP.NET và MVC**
     1. **ASP.NET**

ASP.NET, viết tắt của Active Server Pages .NET (trong đó .NET là nền tảng .NET Framework), là một công nghệ mang tính đột phá, được thiết kế để xây dựng các ứng dụng web hiện đại và cả trong tương lai. Đây không phải là một ngôn ngữ lập trình mà là một framework – hay chính xác hơn là một phương pháp tổ chức – giúp tạo ra các ứng dụng web mạnh mẽ dựa trên nền tảng CLR (Common Language Runtime). ASP.NET là một phần trong hệ sinh thái của hệ điều hành Windows, hoạt động dưới khung .NET. Nhờ vậy, ASP.NET không chỉ tái sử dụng được các đối tượng từ những ứng dụng trước đây, mà còn dễ dàng khai thác toàn bộ tài nguyên của Windows một cách hiệu quả. Các tập tin của ASP.NET có phần mở rộng là .aspx, trong khi đó tập tin của ASP cổ điển có phần mở rộng là .asp. Tập tin .aspx được xử lý ngữ pháp bởi XSPISAPI.DLL, còn với ASP thì là ASP.DLL. Một điểm khác biệt đáng kể giữa hai nền tảng là ASP.NET sử dụng mô hình lập trình hướng sự kiện (event-driven), trong khi ASP thực hiện xử lý tuần tự từ đầu đến cuối. Ngoài ra, ASP.NET sử dụng mã biên dịch (compiled code) nên có hiệu năng và tốc độ xử lý vượt trội so với ASP – vốn dựa trên mã thông dịch (interpreted code).

ASP.NET hoạt động trong môi trường biên dịch (compiled environment) và hỗ trợ gần 25 ngôn ngữ lập trình khác nhau trong nền tảng .NET, trong khi đó ASP truyền thống chỉ hỗ trợ VBScript và JavaScript, vì vậy nó bị giới hạn như một ngôn ngữ kịch bản (scripted language) trong môi trường thông dịch (interpreter environment). Không chỉ dừng lại ở đó, ASP.NET còn tích hợp tốt với XML (Extensible Markup Language), cho phép truyền tải dữ liệu qua mạng một cách linh hoạt và hiệu quả. Bên cạnh khả năng tương thích với tất cả các trình duyệt web, ASP.NET còn có ưu điểm nổi bật là hỗ trợ các thiết bị di động – một yếu tố ngày càng quan trọng trong thời đại hiện nay. Việc hỗ trợ các thiết bị di động – vốn ngày càng phổ biến – là một lợi thế lớn của ASP.NET, trong khi ASP truyền thống gặp nhiều trở ngại khi phải phục vụ thị trường mới này trong lĩnh vực phát triển web

* + 1. **MVC**

MVC là từ viết tắt của **Model – View – Controller**, một mẫu thiết kế phần mềm (design pattern) phổ biến. Mô hình này được sử dụng nhằm phân tách mã nguồn thành ba thành phần riêng biệt, mỗi phần đảm nhiệm một vai trò cụ thể trong việc xử lý các yêu cầu (request) từ người dùng. Việc áp dụng MVC giúp cho mã nguồn trở nên rõ ràng, dễ quản lý, thuận tiện cho việc bảo trì và mở rộng về sau.

Để áp dụng hiệu quả mô hình MVC, người lập trình cần có nền tảng vững chắc về lập trình hướng đối tượng (OOP). Trên thực tế, hầu hết các framework hiện nay đều dựa trên nguyên lý của mô hình MVC. Do đó, khi nắm vững MVC, bạn sẽ dễ dàng tiếp cận và làm việc với các framework khác.

* **Model**: Là thành phần phụ trách truy xuất và xử lý dữ liệu từ cơ sở dữ liệu, sau đó gửi dữ liệu đến View để hiển thị.
* **View**: Là giao diện người dùng – nơi hiển thị thông tin. View nhận dữ liệu từ Model thông qua Controller để trình bày kết quả cuối cùng cho người dùng.
* **Controller**: Là cầu nối giữa Model và View, có nhiệm vụ xử lý các yêu cầu từ phía người dùng và điều hướng chúng đến các chức năng phù hợp. Controller cũng đóng vai trò tiếp nhận và phản hồi dữ liệu giữa hệ thống và người dùng, là phần cốt lõi trong hầu hết các framework hiện đại.

**Ảnh có chứa vòng tròn

Nội dung do AI tạo ra có thể không chính xác.**

Hình 2.1. Hình ảnh cấu trúc MVC

**Xuất xứ**

Câu chuyện bắt đầu từ những năm 1970 tại phòng nghiên cứu Xerox PARC, nằm ở Palo Alto. Tại đây, hai khái niệm quan trọng – giao diện người dùng đồ họa (GUI – Graphical User Interface) và lập trình hướng đối tượng (OOP – Object-Oriented Programming) – đã được phát triển. Điều này giúp lập trình viên có thể tương tác với các thành phần đồ họa như các đối tượng có các thuộc tính và phương thức riêng biệt.

Không chỉ dừng lại ở đó, các nhà khoa học tại Xerox PARC còn tiếp tục đổi mới khi giới thiệu kiến trúc phần mềm MVC (Model – View – Controller). Kiến trúc này được phát triển bởi Trygve Reenskaug vào những năm 1970 và được biết đến lần đầu qua ngôn ngữ lập trình Smalltalk-80. Dù ban đầu không có nhiều tài liệu đề cập đến mô hình này, nhưng vào tháng 8 – 9 năm 1988, bài viết "A Cookbook for Using the Model-View-Controller User Interface Paradigm in Smalltalk-80" của Glenn Krasner và Stephen Pope đã công bố những phân tích chi tiết đầu tiên về MVC.

Khi người dùng hoặc các đối tượng trong hệ thống muốn thay đổi trạng thái của một đối tượng đồ họa, họ sẽ thực hiện thông qua Controller. Thành phần này sẽ xử lý các yêu cầu và tác động lên Model. Khi Model có bất kỳ sự thay đổi nào, nó sẽ phát thông báo đến View và Controller. Nhờ đó, View sẽ cập nhật lại nội dung hiển thị để phản ánh đúng trạng thái mới của Model, còn Controller có thể thực hiện các phản hồi cần thiết cho người dùng hoặc những đối tượng liên quan.

**Đặc điểm của mô hình MVC**

Một trong những ưu điểm nổi bật của mô hình MVC là khả năng hỗ trợ bảo trì ứng dụng một cách dễ dàng, nhờ vào việc phân chia các chức năng thành từng phần riêng biệt. Cấu trúc này giúp quá trình phát triển trở nên nhanh chóng và gọn nhẹ hơn, đồng thời tạo điều kiện thuận lợi cho việc mở rộng hoặc thay đổi các tính năng mà không gây ảnh hưởng đến toàn bộ hệ thống. Nhờ mô hình ba lớp rõ ràng gồm Model, View và Controller, các nhà phát triển và thiết kế có thể làm việc song song mà không bị chồng chéo công việc.

Điều này có được là nhờ vào khả năng tách biệt sự phụ thuộc giữa các thành phần trong một đối tượng giao diện. Sự phân tách này làm tăng tính linh hoạt và khả năng tái sử dụng của từng phần trong ứng dụng. Ví dụ, khi cần thay đổi giao diện người dùng, chỉ cần điều chỉnh phần View mà không ảnh hưởng đến phần xử lý dữ liệu (Model) hay phần điều phối luồng xử lý (Controller). Ngược lại, có thể sửa đổi logic xử lý hoặc cách lưu trữ dữ liệu mà không cần can thiệp vào giao diện.

Nhờ đặc điểm này, kiến trúc MVC đã trở thành nền tảng của nhiều framework và thư viện giao diện nổi tiếng. Trong số đó có thể kể đến thư viện giao diện của ngôn ngữ lập trình SmallTalk, do chính nhóm nghiên cứu tại Xerox PARC phát triển từ những năm 1970. Các thành phần giao diện Swing trong Java cũng được xây dựng dựa trên kiến trúc này. Đặc biệt, nền tảng ASP.NET MVC Framework là một ví dụ điển hình cho việc ứng dụng thành công mô hình MVC trong phát triển web hiện đại.

**Tìm hiểu về Controller**

Controller đóng vai trò trung tâm trong việc xử lý các yêu cầu mà người dùng gửi tới hệ thống. Mỗi khi có một request được gửi đi, nó sẽ được chuyển đến Controller để xử lý. Dựa vào các tham số đi kèm trong request, Controller sẽ xác định tác vụ tương ứng cần thực hiện trong ứng dụng. Tại đây, Controller có thể tương tác với lớp Model để truy xuất hoặc cập nhật dữ liệu, sau đó nhận lại kết quả và chuyển tiếp thông tin đó đến View.

**Tìm hiểu về Model**

Model là thành phần cốt lõi chịu trách nhiệm trong việc xử lý và tương tác với dữ liệu. Trong nhiều framework hiện đại, Model thường được triển khai theo phương pháp Active Record – một mẫu thiết kế phổ biến giúp đơn giản hóa việc thao tác với cơ sở dữ liệu. Nhờ đó, người lập trình có thể giảm thiểu đáng kể thời gian viết các truy vấn SQL thủ công bằng cách sử dụng các thư viện hỗ trợ được xây dựng sẵn.

Thông thường, Model chứa các phương thức phục vụ cho việc truy xuất và cập nhật dữ liệu như: thêm mới, sửa đổi, xóa hoặc lấy thông tin từ cơ sở dữ liệu. Dữ liệu truy xuất thường được trả về dưới dạng mảng để đảm bảo có thể truyền tải đầy đủ và linh hoạt nhiều thông tin.

Khi làm việc với Model, một nguyên tắc quan trọng là không xuất trực tiếp giá trị ra ngoài từ lớp này. Thay vào đó, toàn bộ dữ liệu cần được đóng gói dưới dạng mảng và trả về thông qua các phương thức, để lớp View tiếp nhận và hiển thị cho người dùng một cách rõ ràng và tách biệt.

**Tìm hiểu về View**

View đảm nhận vai trò hiển thị thông tin khi người dùng gửi yêu cầu và hệ thống phản hồi lại. Trước khi khái niệm View được áp dụng phổ biến, các lập trình viên thường xử lý dữ liệu và hiển thị kết quả trực tiếp trong cùng một tệp, chẳng hạn như trong các file PHP. Cách làm này khiến cho ứng dụng trở nên rối rắm, khó kiểm soát và rất phức tạp khi cần sửa đổi hoặc mở rộng về sau.

Việc cập nhật giao diện web từng là trở ngại lớn đối với các nhà thiết kế, bởi họ buộc phải can thiệp vào phần lõi của mã nguồn. Nhằm khắc phục vấn đề đó, các lập trình viên đã bắt đầu sử dụng hệ thống template để tách biệt phần giao diện và phần xử lý logic. Điều này giúp việc tùy chỉnh giao diện trở nên thuận tiện hơn, mà không ảnh hưởng tới logic xử lý của ứng dụng.

Tuy vậy, các công cụ template truyền thống vẫn tồn tại một hạn chế: chúng thường làm chậm tốc độ xử lý do cần phải thông dịch nhiều lớp kịch bản trước khi hiển thị kết quả.

**Ưu điểm của MVC:**

Mô hình MVC phân chia ứng dụng thành ba lớp riêng biệt, giúp cho cấu trúc tổng thể trở nên rõ ràng và dễ hiểu hơn. Điều này mang lại lợi ích lớn cho việc duy trì, phát triển lâu dài cũng như nâng cấp hệ thống một cách thuận tiện. MVC hiện được xem là một trong những mô hình lập trình tiên tiến nhất, mà nhiều framework hiện nay vẫn đang cố gắng áp dụng để nâng cao sự đơn giản và hiệu quả trong trải nghiệm người dùng.

**Nhược điểm của MVC:**

Mặc dù có nhiều ưu điểm, MVC yêu cầu phải tải một số lượng lớn thư viện phức tạp để xử lý dữ liệu, dẫn đến tốc độ chạy chậm hơn so với việc lập trình thủ công truyền thống. Bên cạnh đó, người học MVC cần có nền tảng vững chắc về lập trình hướng đối tượng (OOP) cùng kinh nghiệm nhất định để có thể xây dựng ứng dụng hoàn chỉnh. Nếu không am hiểu OOP, việc tiếp cận mô hình này sẽ gặp nhiều trở ngại. Ngoài ra, do sử dụng nhiều mảng đa chiều trong mô hình Active Record để truy xuất dữ liệu, lập trình viên cần hiểu sâu về cách quản lý dữ liệu phức tạp.

* 1. **Javascript**

JavaScript được tạo ra bởi Brendan Eich khi ông làm việc tại Netscape, và lần đầu tiên được tích hợp vào trình duyệt Navigator 2.0 vào năm 1995. Kể từ năm 1996, cả Netscape lẫn Microsoft đều bắt đầu tích hợp JavaScript vào trình duyệt của họ.

Hiện nay, hầu hết các trình duyệt phổ biến đều tích hợp sẵn JavaScript. Chính JavaScript cũng là một trong những yếu tố quan trọng góp phần khơi mào cho cuộc cạnh tranh giữa các trình duyệt hiện đại. Mặc dù đã tồn tại từ rất lâu, JavaScript vẫn giữ vị trí then chốt trong việc phát triển web. Do đó, việc học và nắm vững JavaScript là điều không thể thiếu đối với bất kỳ ai theo đuổi lập trình web.

**JavaScript là gì?**

**Về cách thực thi mã lệnh:**

Trong phát triển web hiện nay, mã chương trình thường được xử lý theo hai cách: phía máy chủ (Server-Side) và phía trình duyệt (Client-Side).

* **Xử lý phía trình duyệt (Client-Side):** Mã sẽ được thực thi trực tiếp trên máy tính của người truy cập thông qua trình duyệt, và chỉ kết quả mới được gửi lên máy chủ.
* **Xử lý phía máy chủ (Server-Side):** Mã sẽ được gửi đến máy chủ để xử lý, sau đó kết quả được trả về trình duyệt của người dùng.

JavaScript là một ngôn ngữ kịch bản phía trình duyệt, nghĩa là các đoạn mã JavaScript được chạy trên thiết bị của người dùng thay vì trên máy chủ. Do đó, JavaScript không thể truy cập trực tiếp vào cơ sở dữ liệu nằm trên server.

**Về phương diện phong cách lập trình**

Ban đầu, JavaScript được xem là một ngôn ngữ dựa trên đối tượng (Object-Based). Tuy nhiên, từ khi Ajax phát triển, JavaScript đã dần tiếp cận phong cách lập trình hướng đối tượng (OOP). Ngày nay, có thể coi JavaScript là một ngôn ngữ hướng đối tượng.

Cú pháp của JavaScript có nhiều điểm tương đồng với các ngôn ngữ như C, Perl, và Java – đặc biệt là ở các cấu trúc điều khiển như if, while, hay for.

**Tóm lại, JavaScript là một ngôn ngữ lập trình:**

* Chạy ở phía trình duyệt.
* Có khả năng lập trình theo hướng đối tượng.

**JavaScript được sử dụng để làm gì?**

* Tăng cường tính tương tác cho các trang HTML.
* Xử lý các hành động như sự kiện người dùng hoặc kiểm tra form đầu vào.
* Xác thực dữ liệu trư ớc khi gửi lên máy chủ.
* Làm cho trang web phản hồi linh hoạt hơn với người dùng.
* Tùy chỉnh hành vi của trình duyệt.

**Ưu điểm của JavaScript**

* Miễn phí hoàn toàn và dễ tiếp cận đối với người mới học.
* Hoạt động trên hầu hết các hệ điều hành nếu có trình duyệt hỗ trợ.
* Giúp giảm tải công việc cho server bằng cách xử lý ngay trên trình duyệt.
* Là nền tảng kiến thức vững chắc để học các công nghệ hiện đại như Ajax, Atlas,...

**Nhược điểm của JavaScript**

* Không có trình biên dịch riêng biệt; mã được trình duyệt dịch trực tiếp khi chạy. Nếu trình duyệt không hỗ trợ JavaScript hoặc bị tắt, chương trình sẽ không chạy được.
* Có thể gây nặng nề cho website nếu dùng không hợp lý.
* Mức độ bảo mật thấp, vì mã nguồn dễ bị người khác xem và chỉnh sửa.

**Nguyên tắc lập trình trong JavaScript**

* Phân biệt chữ hoa và chữ thường trong tên biến, hàm,...
* Kết thúc mỗi dòng lệnh bằng dấu ;.
* Điều kiện phải nằm trong ngoặc tròn ( ).
* Khi có nhiều câu lệnh bên trong các cấu trúc điều khiển như if, while, hoặc for, cần sử dụng ngoặc nhọn { } để nhóm lệnh.
* Dấu . được dùng để truy cập thuộc tính hoặc gọi phương thức của một đối tượng.

**Ajax là gì?**

Ajax (viết tắt của Asynchronous JavaScript and XML) là một kỹ thuật lập trình web cho phép trang web giao tiếp với máy chủ và cập nhật nội dung một phần của trang mà không cần phải tải lại toàn bộ trang web. Điều này mang lại trải nghiệm mượt mà và nhanh chóng hơn cho người dùng.

Trong lĩnh vực phát triển web hiện đại, Ajax đóng một vai trò thiết yếu trong việc xây dựng các ứng dụng web có tính tương tác cao. Vì dựa trên nền tảng JavaScript, Ajax hoạt động trực tiếp trên trình duyệt của người dùng (client-side). Do đó, mỗi người truy cập đều xử lý riêng biệt và không gây ảnh hưởng lẫn nhau.

Ngày nay, có rất nhiều thư viện và framework JavaScript như jQuery, Angular,... được phát triển để hỗ trợ việc sử dụng Ajax một cách thuận tiện và hiệu quả hơn, giúp lập trình viên thao tác nhanh chóng với ít mã lệnh hơn.

## **Bootstrap**

**Bootstrap** là một **framework giao diện người dùng phía trước (Front-end)**, được xây dựng bằng **SASS** và sau đó **biên dịch ra CSS**. Đây là một thư viện tích hợp sẵn nhiều **đoạn mã CSS tiện ích**, giúp lập trình viên **viết HTML và CSS dễ dàng, nhanh chóng hơn**.Trong khi việc **viết CSS thuần túy** thường khiến người lập trình phải **đối mặt với việc kiểm tra độ tương thích** trên nhiều trình duyệt và thiết bị, thì **Bootstrap đã giải quyết phần lớn vấn đề này**. Những người phát triển Bootstrap đã tối ưu để đảm bảo sự **tương thích cao**, giảm đáng kể công sức cho lập trình viên.

Bootstrap sử dụng hệ thống **lưới (grid system)** để chia bố cục trang web thành **12 cột dựa trên chiều rộng 960px**, giúp dễ dàng xây dựng các thiết kế phản hồi (responsive). Framework này còn đi kèm với nhiều **thành phần phổ biến** như: menu, tab, tooltip, popup,... để hỗ trợ xây dựng giao diện nhanh hơn.Ngoài CSS, **Bootstrap còn tích hợp các hiệu ứng động** thông qua **JavaScript**, cụ thể là sử dụng **jQuery**. Vì thế, để tận dụng toàn bộ tính năng của Bootstrap, cần phải thêm thư viện jQuery vào dự án.

Một số ưu điểm khi dùng Bootstrap:

* Giúp tạo giao diện nhanh và chuẩn.
* Học và sử dụng dễ dàng.
* Hỗ trợ nhiều hiệu ứng JavaScript nhờ jQuery.

## **Hệ cơ sở dữ liệu SQL Server**

**SQL (Structured Query Language)** là một **ngôn ngữ truy vấn có cấu trúc**, được sử dụng rộng rãi để **quản lý và thao tác dữ liệu** trong nhiều ngành nghề khác nhau.

Ban đầu, SQL chủ yếu được sử dụng trong **hệ thống cơ sở dữ liệu đơn lẻ**. Tuy nhiên, với sự phát triển và nhu cầu ngày càng cao về các hệ thống **Client/Server**, nhiều **hệ quản trị cơ sở dữ liệu (DBMS)** hỗ trợ SQL đã xuất hiện như: **Oracle, Microsoft SQL Server, Sybase**,...

Trong mô hình **Client/Server**, **dữ liệu được lưu trữ tập trung tại máy chủ**, còn các máy trạm chỉ đóng vai trò gửi yêu cầu hoặc nhận kết quả. Các thao tác xử lý như thêm, sửa, xóa, truy vấn dữ liệu đều được thực hiện trực tiếp trên server thông qua các lệnh SQL.

Khi Internet phát triển mạnh mẽ, **SQL trở thành công cụ then chốt** trong việc xây dựng các **trang web động** (Dynamic Web Pages). Các nội dung trên website được lấy từ cơ sở dữ liệu và hiển thị cho người dùng thông qua các **truy vấn SQL**. Bên cạnh đó, SQL còn được dùng để cập nhật dữ liệu, đảm bảo rằng thông tin luôn được đồng bộ và chính xác.

## **Truy vấn LINQ**

### **LINQ là gì?**

Để đơn giản hóa việc thao tác dữ liệu giữa nhiều ngôn ngữ lập trình khác nhau và nâng cao hiệu suất phát triển phần mềm, Microsoft đã giới thiệu **LINQ (Language Integrated Query)** – một giải pháp tích hợp truy vấn dữ liệu vào trong môi trường .NET Framework.

LINQ là một thư viện mở rộng, chủ yếu dành cho các ngôn ngữ như C# và Visual Basic.NET, nhưng vẫn có thể tùy biến để hỗ trợ các ngôn ngữ khác. LINQ cung cấp khả năng truy vấn dữ liệu một cách trực tiếp trên nhiều loại nguồn dữ liệu như đối tượng (object), cơ sở dữ liệu, tài liệu XML, với cú pháp thống nhất và dễ sử dụng.

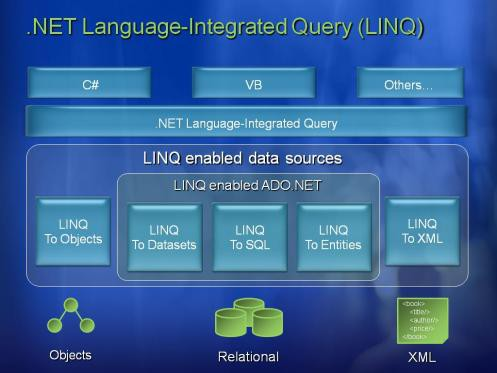
### **Điểm mạnh của LINQ**

Điểm mạnh nổi bật nhất của LINQ chính là khả năng thực hiện truy vấn trên nhiều kiểu nguồn dữ liệu khác nhau, cụ thể:

* LINQ to SQL: Dùng để truy xuất dữ liệu từ hệ quản trị cơ sở dữ liệu SQL Server.
* LINQ to XML: Cho phép xử lý và truy vấn dữ liệu từ các tệp XML.
* LINQ to Objects: Dùng để thao tác với dữ liệu trong các đối tượng hoặc mảng được tạo ra trong mã nguồn.

LINQ mang đến sự đồng nhất trong cú pháp truy vấn, giúp lập trình viên không cần học nhiều cách truy vấn riêng biệt cho từng loại dữ liệu, từ đó nâng cao tính hiệu quả và nhất quán trong lập trình

* + 1. **Các thành phần của LINQ**

****

Hình 2.2. Hình ảnh cấu trúc LINQ

**LINQ to Objects**

"LINQ to Objects" là cách áp dụng LINQ để truy vấn các tập hợp dữ liệu (collections) đã triển khai giao diện IEnumerable hoặc IEnumerable<T>. Đây là kiểu truy vấn đơn giản nhất trong LINQ, dùng để làm việc với các dữ liệu có thể liệt kê được, như danh sách (List), mảng (Array), v.v.

**LINQ to SQL**

LINQ to SQL là một công nghệ hiện thực hóa mô hình ánh xạ đối tượng-quan hệ (Object-Relational Mapping - ORM) trong .NET Framework 3.5. Với công cụ này, lập trình viên có thể biểu diễn cơ sở dữ liệu dưới dạng các lớp trong .NET. Nhờ đó, bạn có thể thực hiện truy vấn, thêm, sửa, hoặc xóa dữ liệu bằng cú pháp LINQ.

LINQ to SQL hỗ trợ các tính năng như transaction, view và stored procedure. Ngoài ra, nó còn cho phép bạn dễ dàng tích hợp các quy tắc kiểm tra dữ liệu và tính hợp lệ vào mô hình dữ liệu.

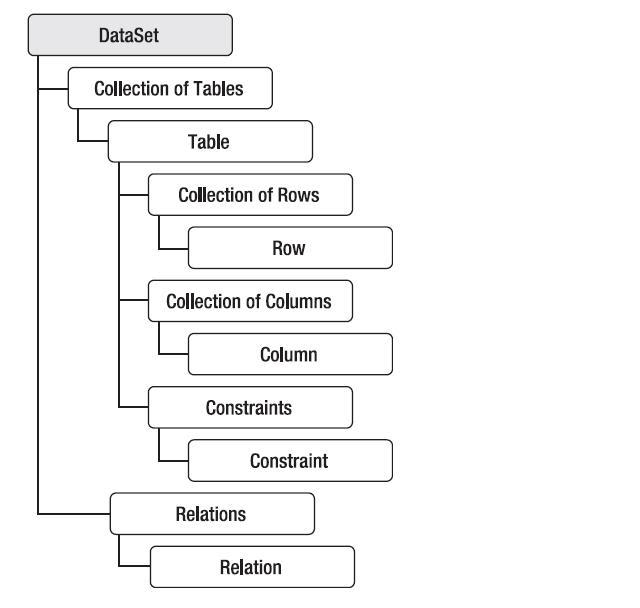
**LINQ to XML**

LINQ to XML cung cấp phương pháp đơn giản và hiệu quả để truy vấn, thao tác dữ liệu XML mà không cần sử dụng đến các kỹ thuật phức tạp như XmlDocument, XPath, hay XQuery. Nó giúp làm việc với tài liệu XML trở nên trực quan và dễ đọc hơn trong mã nguồn C#.

**LINQ to Datasets**

Trong ADO.NET, DataSet là một giải pháp linh hoạt để làm việc với cơ sở dữ liệu trong môi trường ngắt kết nối. Khi thao tác với dữ liệu, việc duy trì kết nối liên tục đến nguồn dữ liệu có thể gây tốn tài nguyên. DataSet giải quyết vấn đề này bằng cách lưu dữ liệu cục bộ (offline) để chỉnh sửa.

Sau khi người dùng thực hiện các thay đổi trên dữ liệu tại local cache, hệ thống có thể kết nối trở lại để đồng bộ dữ liệu vào Data Source. Dữ liệu trong DataSet được tổ chức dưới dạng một tập các bảng (DataTables), và bạn sẽ làm việc với các lớp như DataTable, DataRow, và DataColumn.

****

Hình 2.3. Hình ảnh cấu trúc dataset

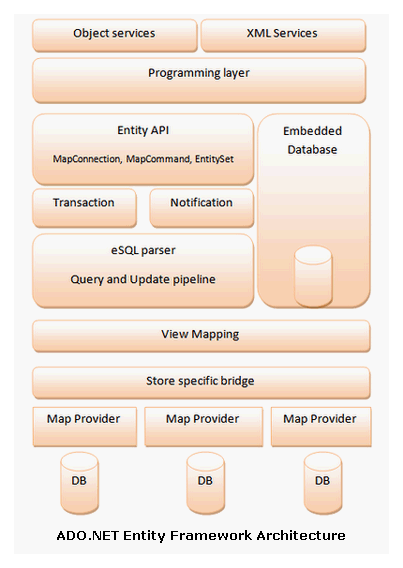
**LINQ to Entities**

**Khái Niệm về Entity Framework**

Entity Framework là một công cụ giúp ánh xạ các đối tượng trong ứng dụng thành các dữ liệu được lưu trữ trong cơ sở dữ liệu, gọi là mô hình giản đồ (schema). Mô hình này được tạo ra dựa trên cấu trúc của cơ sở dữ liệu và giúp lập trình viên thao tác dữ liệu theo hướng đối tượng mà không cần quan tâm trực tiếp đến câu lệnh SQL hay cách lưu trữ vật lý.

Entity Framework hoạt động độc lập với loại cơ sở dữ liệu cụ thể, nghĩa là nó có thể làm việc với nhiều hệ quản trị khác nhau. Nó xây dựng trên nền tảng chuẩn của ADO.NET và cho phép truy cập dữ liệu từ các nguồn bên thứ ba thông qua việc sử dụng ADO.NET làm lớp trung gian.

Ứng dụng của Entity Framework giúp đơn giản hóa quá trình phát triển phần mềm, tăng tính linh hoạt và giảm thiểu sự phụ thuộc vào cơ sở dữ liệu cụ thể, từ đó giúp việc bảo trì và mở rộng hệ thống trở nên dễ dàng hơn.

****

Hình 2.4. Hình ảnh kiến trúc entity

**Kiến trúc của Entity Framework**

**LINQ to Entities** cho phép thực hiện các truy vấn trực tiếp trên các thực thể được quản lý bởi Entity Framework. Điều này giúp truy xuất dữ liệu từ cơ sở dữ liệu dưới dạng các đối tượng trong ứng dụng một cách dễ dàng và linh hoạt.

Dưới đây là danh sách các toán tử (operator) phổ biến và chuẩn thường được sử dụng trong LINQ:

1. Select / SelectMany – Lấy và chuyển đổi dữ liệu.
2. Where – Lọc dữ liệu theo điều kiện.
3. Sum / Min / Max / Average / Aggregate – Thực hiện các phép toán tổng hợp.
4. Join / GroupJoin – Kết hợp các tập dữ liệu dựa trên khóa chung.
5. Take / TakeWhile – Lấy một số phần tử theo điều kiện hoặc số lượng.
6. Skip / SkipWhile – Bỏ qua một số phần tử theo điều kiện hoặc số lượng.
7. OfType – Lọc phần tử theo kiểu dữ liệu cụ thể.
8. Concat – Nối hai tập hợp lại với nhau.
9. OrderBy / ThenBy – Sắp xếp dữ liệu theo thứ tự tăng dần.
10. Reverse – Đảo ngược thứ tự phần tử.
11. GroupBy – Gom nhóm các phần tử theo một khóa.
12. Distinct – Lọc các phần tử duy nhất.
13. Union / Intersect / Except – Các phép toán tập hợp: hợp, giao và hiệu.
14. EqualAll – So sánh sự bằng nhau của tất cả phần tử.
15. First / FirstOrDefault / Last / LastOrDefault – Lấy phần tử đầu hoặc cuối, có thể trả về giá trị mặc định nếu không tìm thấy.
16. Single – Lấy duy nhất một phần tử, gây lỗi nếu không đúng điều kiện.
17. ElementAt – Lấy phần tử tại vị trí cụ thể.
18. Any / All / Contains – Kiểm tra điều kiện tồn tại, tất cả hoặc chứa phần tử.
19. Count – Đếm số lượng phần tử.

**LINQ vs ADO.NET**

Nhiều người thường băn khoăn về mối quan hệ giữa LINQ và ADO.NET, liệu chúng có phải là hai công nghệ độc lập hay hỗ trợ nhau? Thực tế, LINQ là một tập hợp các mở rộng giúp bạn có thể viết câu truy vấn ngay trong ngôn ngữ lập trình. Nó cho phép làm việc với nhiều kiểu dữ liệu khác nhau như XML, collection, mảng, cũng như cơ sở dữ liệu.

Trong khi đó, ADO.NET là một công nghệ hỗ trợ kết nối và thao tác với nhiều loại cơ sở dữ liệu khác nhau, bao gồm các thao tác truy vấn, cập nhật, thêm, xóa, hoặc gọi các thủ tục lưu trữ.

LINQ không được tạo ra để thay thế ADO.NET mà có thể sử dụng độc lập mà không cần cơ sở dữ liệu. Tuy nhiên, LINQ to SQL, một phần mở rộng của LINQ, cho phép làm việc trực tiếp với cơ sở dữ liệu SQL Server. Khi sử dụng LINQ to SQL, bạn không cần phải viết các câu lệnh ADO.NET truyền thống mà chỉ cần tập trung vào cú pháp truy vấn mà LINQ cung cấp.

# **CHƯƠNG III: HIỆN THỰC HÓA NGHIÊN CỨU**

* 1. **Xác định thành phần và chức năng chính của hệ thống**

Hệ thống website bán vé xem phim trực tuyến được thiết kế nhằm hỗ trợ hiệu quả cả người dùng cuối (khách hàng) và nhân viên quản trị hệ thống. Các chức năng chính bao gồm:

* **Quản lý thông tin người dùng**  
  Hệ thống cho phép quản trị viên hoặc nhân viên cập nhật và quản lý thông tin cá nhân của người dùng như họ tên, địa chỉ email, số điện thoại, tên đăng nhập, mật khẩu và vai trò (khách hàng, nhân viên, quản trị viên...). Điều này hỗ trợ phân quyền truy cập và đảm bảo an toàn dữ liệu người dùng.
* **Quản lý thông tin đạo diễn**  
  Cho phép thêm mới, chỉnh sửa và xóa thông tin đạo diễn như họ tên, tiểu sử tóm tắt, quốc tịch, và các bộ phim đã tham gia. Chức năng này giúp hệ thống dễ dàng tra cứu các phim theo đạo diễn hoặc thống kê theo danh mục.
* **Quản lý nhân viên hệ thống**  
  Chức năng này dành cho người quản trị để tạo mới, sửa đổi hoặc xóa thông tin nhân sự nội bộ như mã nhân viên, tên, địa chỉ, số điện thoại, giới tính, chức vụ và quyền hạn truy cập hệ thống.
* **Quản lý danh sách phim**  
  Nhân viên được cấp quyền có thể cập nhật cơ sở dữ liệu phim, bao gồm mã phim, tên phim, thể loại, đạo diễn, thời lượng chiếu, mô tả nội dung, trailer và hình ảnh đại diện (poster). Việc quản lý tập trung giúp đảm bảo thông tin phim luôn được cập nhật kịp thời và chính xác.
* **Tìm kiếm và lọc phim**  
  Hệ thống hỗ trợ người dùng tìm kiếm phim theo nhiều tiêu chí: tên phim, thể loại, đạo diễn hoặc ngày khởi chiếu. Tính năng này giúp nâng cao trải nghiệm người dùng và tiết kiệm thời gian khi tìm kiếm nội dung yêu thích.
* **Đăng ký tài khoản người dùng**  
  Cho phép người dùng lần đầu truy cập hệ thống có thể tạo tài khoản bằng cách điền thông tin cá nhân cơ bản như họ tên, email, số điện thoại, tên đăng nhập và mật khẩu. Tài khoản này sẽ được sử dụng để đặt vé, lưu lịch sử giao dịch, và nhận các thông báo ưu đãi.
* **Đăng nhập và xác thực người dùng**  
  Người dùng sau khi có tài khoản có thể đăng nhập vào hệ thống để truy cập các chức năng cá nhân như đặt vé, theo dõi lịch chiếu đã đặt, hoặc cập nhật thông tin cá nhân.
* **Quản lý lịch chiếu phim**  
  Chức năng này cho phép nhân viên cấu hình các suất chiếu bao gồm ngày, giờ, rạp chiếu, phòng chiếu và mức giá vé tương ứng. Tính năng này được xây dựng linh hoạt nhằm dễ dàng điều chỉnh lịch trình theo từng rạp hoặc sự kiện cụ thể.
  1. **Phân tích vai trò người dùng trong hệ thống**

Để đảm bảo phân quyền rõ ràng và hoạt động hiệu quả, hệ thống chia người dùng thành các nhóm vai trò với quyền hạn khác nhau như sau:

* **Người quản trị hệ thống (Admin)**  
  Là người chịu trách nhiệm điều hành và vận hành toàn bộ hệ thống. Admin có toàn quyền truy cập và điều khiển các chức năng như quản lý người dùng, nhân viên, phim, lịch chiếu, rạp chiếu, giá vé và theo dõi báo cáo tổng hợp. Ngoài ra, admin còn có thể phân quyền và cấp quyền truy cập cho các tài khoản khác.
* **Nhân viên**  
  Nhân viên có quyền truy cập một phần hệ thống tùy theo chức năng được phân công. Họ có thể thực hiện các tác vụ như thêm/sửa phim, quản lý lịch chiếu, cập nhật thông tin đạo diễn, và hỗ trợ khách hàng trong quá trình đặt vé hoặc thanh toán.
* **Khách hàng**   
  Là người sử dụng dịch vụ đặt vé trực tuyến. Họ có thể đăng ký tài khoản, đăng nhập hệ thống, tìm kiếm phim, xem lịch chiếu, chọn suất chiếu phù hợp, đặt vé, thanh toán và lưu trữ vé điện tử. Ngoài ra, khách hàng còn có thể nhận thông báo phim mới, khuyến mãi và phản hồi đánh giá về trải nghiệm sử dụng.
  1. **Sơ đồ cơ sở dữ liệu**

**A screenshot of a computer flow chart

AI-generated content may be incorrect.**

Hình 3.1. Hình ảnh sơ đồ cơ sở dữ liệu

* 1. **Cơ sở dữ liệu**

**Bảng User**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Ràng buộc** |
| 1 | Id | string (GUID) | **PK** |
| 2 | UserName | string |  |
| 3 | NormalizedUserName | string |  |
| 4 | Email | string |  |
| 5 | NormalizedEmail | string |  |
| 6 | EmailConfirmed | bool |  |
| 7 | PasswordHash | string |  |
| 8 | SecurityStamp | string |  |
| 9 | ConcurrencyStamp | string |  |
| 10 | PhoneNumber | string |  |
| 11 | PhoneNumberConfirmed | bool |  |
| 12 | TwoFactorEnabled | bool |  |
| 14 | LockoutEnabled | bool |  |
| 15 | AccessFailedCount | int |  |
| 16 | Name | string |  |
| 17 | DayOfBirth | DateTime |  |
| 18 | Gender | string |  |

Bảng 3.1. User – Dùng để lưu thông tin người dùng

**Bảng Movie**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Ràng buộc** |
| 1 | ID | int | **PK** |
| 2 | Poster | string |  |
| 3 | Title | string |  |
| 4 | Director | string |  |
| 5 | Cast | string |  |
| 6 | Genre | string |  |
| 7 | ReleaseDate | DateTime |  |
| 8 | Duration | int |  |
| 9 | Language | string |  |
| 10 | Rated | string |  |
| 11 | Summary | string |  |

Bảng 3.2. Movie – Dùng để quản lý thông tin về phim

**Bảng Theater**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Ràng buộc** |
| 1 | ID | int | **PK** |
| 2 | Name | string |  |
| 3 | NoSeat | int |  |
| 4 | Format | string |  |

Bảng 3.3. Theater – Dùng để lưu trữ dữ liệu của rạp phim

**Bảng Showtime**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Ràng buộc** |
| 1 | ID | int | **PK** |
| 2 | MovieID | int | FK |
| 3 | TheaterID | int | FK |
| 4 | StartTime | DateTime |  |
| 5 | NoSeatBooked | int |  |
| 6 | Format | string |  |

Bảng 3.4. Showtime – Dùng để lưu thông tin về suất chiếu

**Bảng Ticket**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Ràng buộc** |
| 1 | ID | int | **PK** |
| 2 | ShowtimeID | int | **FK** |
| 3 | Seat | string |  |
| 4 | UserID | string | **FK** |
| 5 | StaffID | string | **FK** |
| 6 | CodeToGet | string |  |
| 7 | Got | bool |  |
| 8 | Money | decimal |  |

Bảng 3.5. Ticket – Dùng để lưu dữ liệu của vé

* 1. **Sơ đồ use case**
* **Sơ đồ use case tổng quát**

**A diagram of a company

AI-generated content may be incorrect.**

Hình 3.2. Hình ảnh sơ đồ use case tổng quát

* **Sơ đồ use case đăng nhập**

**A diagram of a person with a diagram

AI-generated content may be incorrect.**

Hình 3.3. Hình ảnh sơ đồ use case đăng nhập

* **Sơ đồ use case xem thông tin cá nhân**

**Ảnh có chứa vòng tròn, biểu đồ, hàng, văn bản

Nội dung do AI tạo ra có thể không chính xác.**

Hình 3.4. Hình ảnh sơ đồ use case xem thông tin cá nhân

* **Sơ đồ use case đặt vé**

**A black and white image of a white background

AI-generated content may be incorrect.**

Hình 3.5. Hình ảnh sơ đồ use case đặt vé

* **Sơ đồ use case quản lý phim**

**A close-up of a diagram

AI-generated content may be incorrect.**

Hình 3.6. Hình ảnh sơ đồ use case quản lý phim

* **Sơ đồ use case quản lý người dùng**

A black and blue text in a circle

AI-generated content may be incorrect.

Hình 3.7. Hình ảnh sơ đồ use case quản lý người dùng

**CHƯƠNG IV: KẾT QUẢ NGHIÊN CỨU**

* 1. **Màn hình giao diện chính**

**A screenshot of a computer

AI-generated content may be incorrect.**

Hình 4.1. Hình ảnh trang chủ

* 1. **Màn hình đăng nhập**

**A screenshot of a computer

AI-generated content may be incorrect.**

Hình 4.2. Hình ảnh màn hình đăng nhập

* 1. **Màn hình đăng ký**

**A screenshot of a computer

AI-generated content may be incorrect.**

Hình 4.3. Hình ảnh màn hình đăng ký

* 1. **Màn hình chi tiết phim**

**A computer screen shot of a computer

AI-generated content may be incorrect.**

Hình 4.4. Hình ảnh chi tiết phim

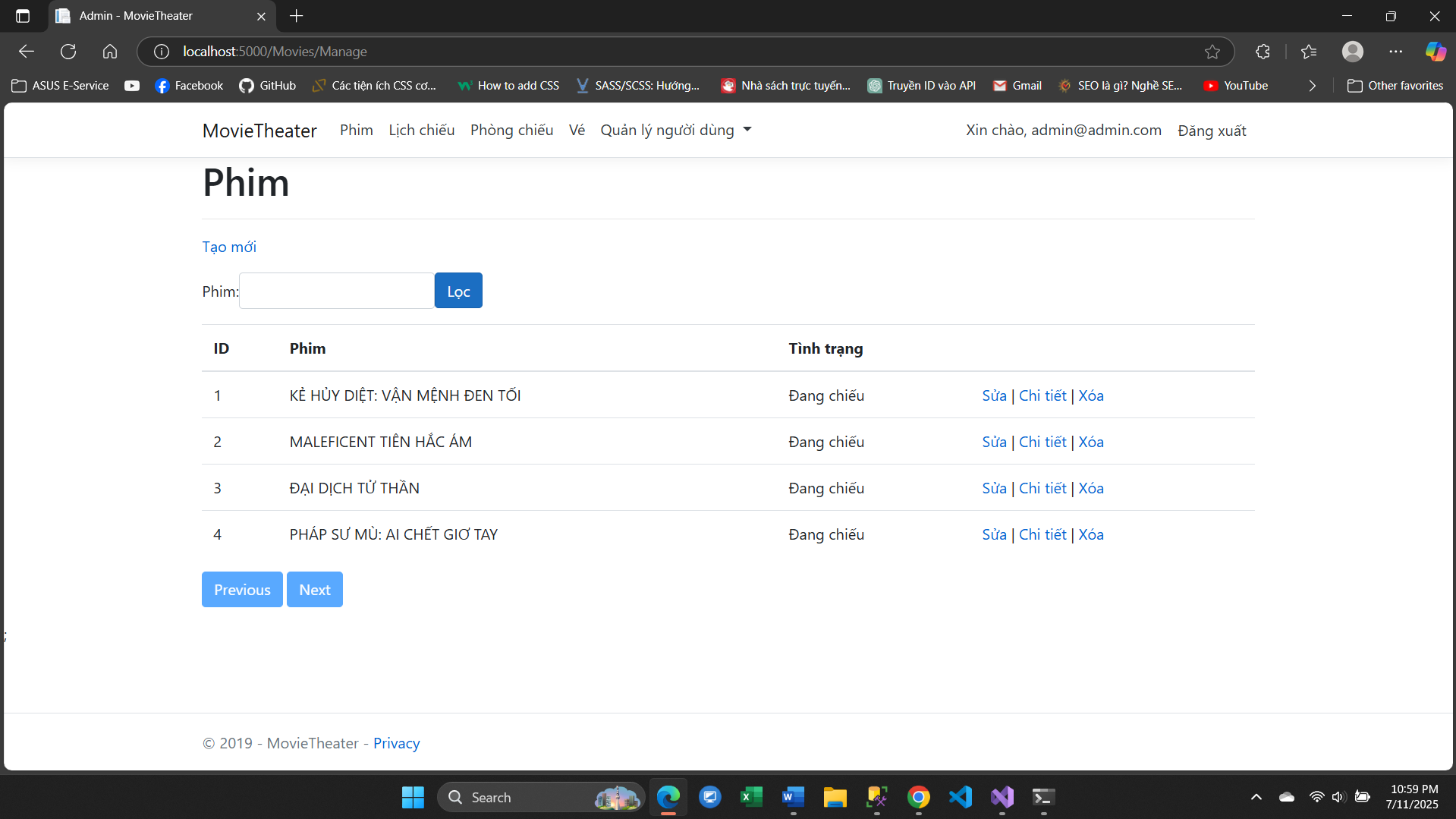
* 1. **Màn hình danh sách phim**

**A screenshot of a computer

AI-generated content may be incorrect.**

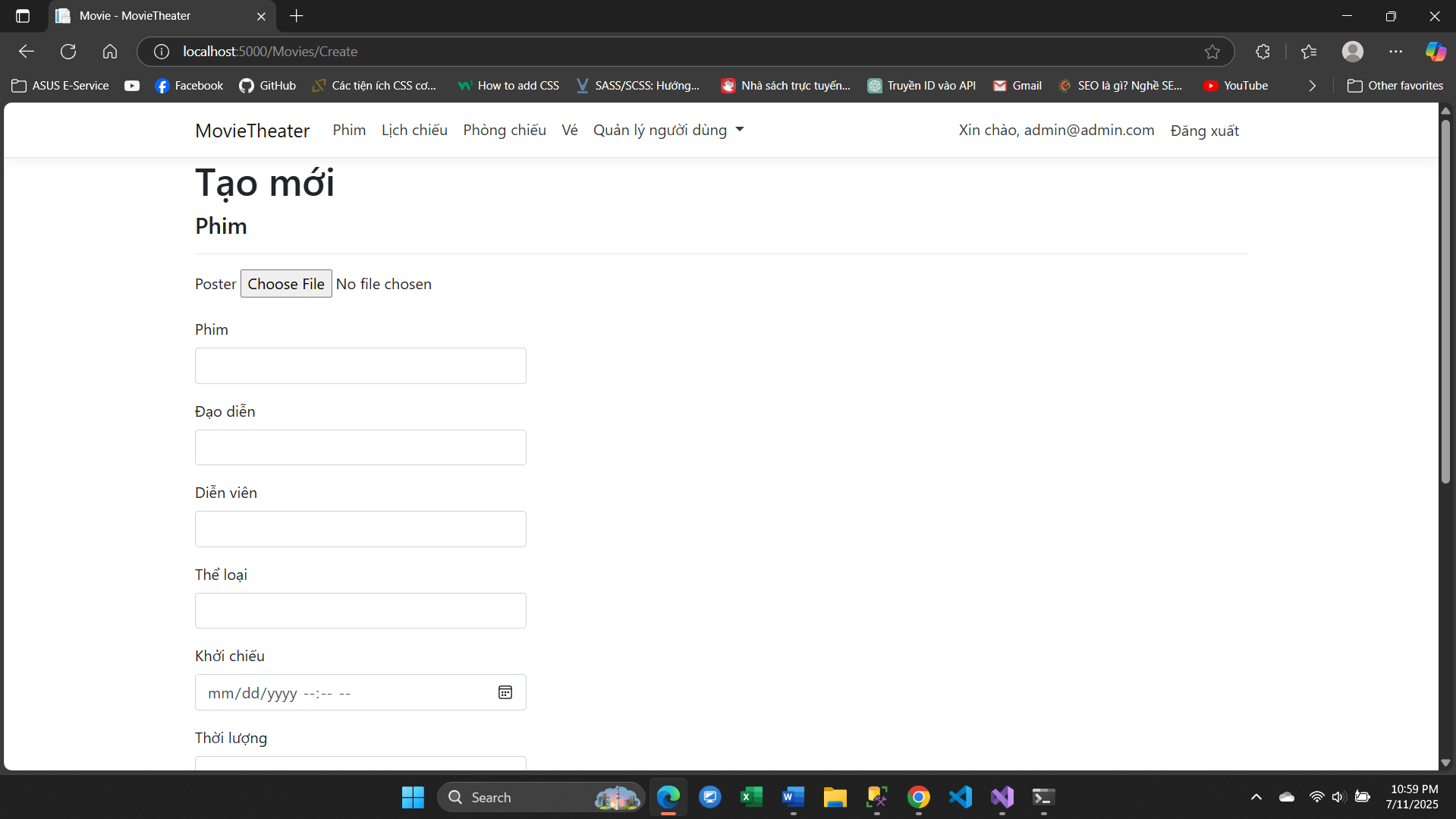
Hình 4.5. Hình ảnh màn hình danh sách phim

* 1. **Màn hình trang quản lý phim**

****

Hình 4.6. Hình ảnh quản lý phim

* 1. **Màn hình trang thêm phim mới**

****

Hình 4.7. Hình ảnh màn hình thêm phim mới

* 1. **Màn hình quản lý lịch chiếu**

***A screenshot of a computer

AI-generated content may be incorrect.***

Hình 4.8. Hình ảnh quản lý lịch chiếu

# **CHƯƠNG V:** **KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

## **Kết luận**

Dự án “Phát triển Website bán vé xem phim trực tuyến” đã được triển khai thành công, đáp ứng đầy đủ các chức năng cơ bản của một hệ thống đặt vé phim hiện đại, bao gồm: quản lý thông tin phim, thể loại phim, suất chiếu, rạp chiếu, đặt vé và quản lý người dùng.

Hệ thống được phát triển dựa trên nền tảng **ASP.NET Core MVC** kết hợp với **SQL Server**, giúp đảm bảo hiệu năng tốt, độ bảo mật cao và khả năng mở rộng linh hoạt trong tương lai.

Website có giao diện thân thiện, dễ sử dụng, cho phép người dùng tra cứu lịch chiếu, tìm kiếm phim và đặt vé một cách nhanh chóng, tiện lợi. Hệ thống quản trị hỗ trợ nhân viên và quản trị viên trong việc quản lý phim, suất chiếu, vé, rạp chiếu, người dùng và các hoạt động nghiệp vụ liên quan, từ đó góp phần nâng cao hiệu quả vận hành hệ thống rạp.

## **Hướng phát triển**

**Cải tiến hệ thống đặt vé:** Hỗ trợ chọn ghế tương tác trực tiếp trên sơ đồ rạp, thêm tính năng thanh toán online qua ví điện tử và ngân hàng.

**Phát triển ứng dụng di động:** Xây dựng phiên bản ứng dụng di động trên iOS và Android, giúp người dùng đặt vé mọi lúc mọi nơi và nhận thông báo lịch chiếu hoặc khuyến mãi.

**Tối ưu hóa SEO và Marketing:** Nâng cao khả năng xuất hiện trên công cụ tìm kiếm như Google, đồng thời tích hợp chiến dịch quảng bá phim và chương trình ưu đãi đến người dùng.

**Tích hợp đa ngôn ngữ:** Phát triển giao diện website đa ngôn ngữ để phục vụ người dùng ở nhiều khu vực và khách du lịch, mở rộng khả năng tiếp cận thị trường quốc tế.

# **DANH MỤC TÀI LIỆU THAM KHẢO**

1. Microsoft. (không rõ năm). *Tổng quan về ASP.NET Core MVC*. Truy cập từ: <https://learn.microsoft.com/en-us/aspnet/core/mvc>
2. Microsoft. (không rõ năm). *Tài liệu về Entity Framework Core*. Truy cập từ: <https://learn.microsoft.com/en-us/ef/core>
3. W3Schools. (không rõ năm). *Hướng dẫn sử dụng Bootstrap 4*. Truy cập từ: <https://www.w3schools.com/bootstrap4>
4. Microsoft. (không rõ năm). *Tài liệu hướng dẫn SQL Server*. Truy cập từ: <https://learn.microsoft.com/en-us/sql/sql-server>
5. Stack Overflow. (không rõ năm). *Các phương pháp tối ưu truy vấn SQL trong ASP.NET Core*. Truy cập từ: <https://stackoverflow.com>