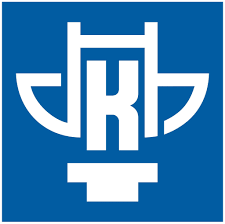
**TRƯỜNG ĐẠI HỌC KIẾN TRÚC HÀ NỘI**

**KHOA CÔNG NGHỆ THÔNG TIN**

-----🙡🕮🙣-----



**ĐỒ ÁN MÔN HỌC CÔNG NGHỆ WEB**

***ĐỀ TÀI:***

**XÂY DỰNG WEBSITE CỬA HÀNG**

**BÁN ĐỒ CÔNG NGHỆ**

**Giảng viên hướng dẫn: Ths Nguyễn Hồng Thanh**

**Nhóm thực hiện: Nhóm 13**

**Nguyễn Văn Sơn**

**Nguyễn Đức Huy**

**Vũ Quốc Tuấn**

**Hà Nội, Ngày tháng năm 2023**

# **LỜI CẢM ƠN**

Sau thời gian học tập và rèn luyện bằng sự biết ơn và kính trọng, nhóm em xin gửi lời cảm ơn sâu sắc tới thầy giáo, giảng viên ThS. Nguyễn Hồng Thanh đã giảng dạy và hỗ trợ kiến thức cho chúng em thời gian qua. Qua sự nổ lực và tâm huyết của cô, chúng em đã tiếp thu cho bản thân được rất nhiều kiến thức dể nhóm có thể hoàn thành được bài báo cáo này.

Bên cạnh đó cũng cảm ơn tất cả các thành viên trong nhóm đã nỗ lực, đoàn kết để có thể hoàn thành dự án một cách suôn sẻ. Hy vọng các bạn sẽ tiếp tục học tập và trau dồi kiến thức để hoàn thiện bản thân hơn.

Tuy điều kiện về năng lực của nhóm còn hạn chế, đề tài môn học chắc chắn không tránh khỏi những thiếu sót. Kính mong nhận được sự đóng góp ý kiến của cô để bài báo cáo nhóm em được tốt hơn.

Em xin chân thành cảm ơn!

**MỤC LỤC**

[LỜI CẢM ƠN 2](#_Toc149583054)

[CHƯƠNG I: TỔNG QUAN ĐỀ TÀI 5](#_Toc149583055)

[**1.1 Giới thiệu đề tài** 5](#_Toc149583056)

[**1.2 Lý do chọn đề tài:** 5](#_Toc149583057)

[**1.3 Mục tiêu và phạm vi của dự án** 5](#_Toc149583058)

[1.3.1 Mục tiêu 5](#_Toc149583059)

[1.3.2 Phạm vi 5](#_Toc149583060)

[CHƯƠNG II: CƠ SỞ LÝ THUYẾT 6](#_Toc149583061)

[**2.1 Kiến thức cơ bản về ngôn ngữ C#** 6](#_Toc149583062)

[2.1.1. C# là gì? 6](#_Toc149583063)

[**2.2 Các framework sử dụng** 7](#_Toc149583064)

[2.2.1 Framework ASP.NET CORE: 7](#_Toc149583065)

[2.2.2 Framework ASP.NET MVC: 8](#_Toc149583066)

[2.2.3 Entity framework 8](#_Toc149583067)

[2.2.5 Hệ quản trị cơ sở dữ liệu MySQL 10](#_Toc149583068)

[2.2.6 Nguyên lý SOLID 11](#_Toc149583069)

[CHƯƠNG III: PHÂN TÍCH THIẾT KẾ 12](#_Toc149583070)

[**3.1. Phân tích yêu cầu của trang web** 12](#_Toc149583071)

[3.1.1. Mô tả nghiệp vụ 12](#_Toc149583072)

[3.2. Phân tích hệ thống 13](#_Toc149583073)

[**3.3. Thiết kế cơ sở dữ liệu cho hệ thống** 30](#_Toc149583074)

[**3.4. Thiết kế giao diện người dùng** 33](#_Toc149583075)

[3.4.1 Người dùng 33](#_Toc149583076)

[3.4.2 Addmin: 35](#_Toc149583077)

[CHƯƠNG IV: XÂY DỰNG ỨNG DỤNG 38](#_Toc149583078)

[**4.1. Cấu trúc dự án** 38](#_Toc149583079)

[**4.2. Một số mã code xây dựng backend** 39](#_Toc149583080)

[4.2.1 Cấu hình 39](#_Toc149583081)

[4.2.2. Repository 41](#_Toc149583082)

[4.2.3. Service 42](#_Toc149583083)

[4.2.4 Controllers 43](#_Toc149583084)

[**4.3 Một số mã code xây dựng Fontend** 49](#_Toc149583085)

[**4.4. Kiểm thử** 54](#_Toc149583086)

[4.4.1. Chức năng đăng nhập 54](#_Toc149583087)

[4.4.2. Chức năng quản lý thông tin cá nhân 54](#_Toc149583088)

[CHƯƠNG V: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 56](#_Toc149583089)

[**5.1 Kết luận:** 56](#_Toc149583090)

[**5.2 Ưu và nhược điểm** 56](#_Toc149583091)

[5.2.1 Ưu điểm 56](#_Toc149583092)

[5.2.2 Nhược điểm 56](#_Toc149583093)

[**5.3 Các chức năng đạt được** 56](#_Toc149583094)

[**5.4 Hướng phát triển** 56](#_Toc149583095)

# **CHƯƠNG I: TỔNG QUAN ĐỀ TÀI**

## **1.1 Giới thiệu đề tài**

Trong thời đại số hóa mạnh mẽ, trang web của một cửa hàng bán điện thoại không chỉ là một công cụ quảng cáo sản phẩm mà còn là một giao diện quan trọng giữa khách hàng và doanh nghiệp. Việc phát triển và tối ưu hóa trang web của cửa hàng bán điện thoại là một nhiệm vụ quan trọng để cải thiện trải nghiệm của khách hàng và tối ưu hóa hiệu suất kinh doanh.

## **1.2 Lý do chọn đề tài:**

Lý do chúng em quyết định tập trung vào đề tài này còn liên quan đến tầm quan trọng của trang web trong mô hình kinh doanh hiện đại. Trang web không chỉ là nơi giới thiệu sản phẩm, mà còn đóng vai trò quan trọng trong việc tạo ấn tượng ban đầu với khách hàng, thu hút họ và tạo trải nghiệm mua sắm thú vị. Điều này đặc biệt quan trọng trong một ngành đòi hỏi sự đa dạng và sáng tạo như ngành công nghiệp điện thoại di động. Nếu cửa hàng có thể tận dụng tối đa tiềm năng của trang web, họ có thể tạo sự khác biệt và thành công đáng kể trong thị trường cạnh tranh này.

## **1.3 Mục tiêu và phạm vi của dự án**

### ***1.3.1 Mục tiêu***

-Mục tiêu chính của dự án là phát triển và tối ưu hóa trang web của cửa hàng bán điện thoại để cải thiện trải nghiệm khách hàng và tối ưu hóa hiệu suất kinh doanh trực tuyến. Mục tiêu cụ thể bao gồm:

* Tối ưu hóa trải nghiệm khách hàng*:* Cải thiện giao diện và tính năng trang web để làm cho việc duyệt và mua sắm trở nên dễ dàng, hấp dẫn và thoải mái hơn đối với khách hàng.
* Tối ưu hóa tốc độ tải trang*:* Đảm bảo rằng trang web hoạt động nhanh chóng và hiệu quả trên cả máy tính và thiết bị di động.
* Cải thiện tính năng mua sắm trực tuyến: Tăng khả năng tùy chỉnh sản phẩm, thêm vào giỏ hàng, và quá trình thanh toán để nâng cao tỷ lệ chuyển đổi.
* Tích hợp công cụ quảng cáo và phân tích: Kết hợp công cụ quảng cáo hiệu quả và phân tích dữ liệu để theo dõi và cải thiện hiệu suất trực tuyến.

### ***1.3.2 Phạm vi***

-Thiết kế giao diện người dùng:

Thiết kế một giao diện hấp dẫn, tương tác và dễ sử dụng để tạo trải nghiệm mua sắm trực tuyến tốt cho khách hàng.

-Quản lý sản phẩm và đặt hàng:

Xây dựng hệ thống quản lý thông tin sản phẩm, lưu trữ và xử lý đơn hàng một cách hiệu quả, giúp đảm bảo việc quản lý sản phẩm và giao hàng đúng thời hạn.

# **CHƯƠNG II: CƠ SỞ LÝ THUYẾT**

## **2.1 Kiến thức cơ bản về ngôn ngữ C#**

### **2.1.1. C# là gì?**

-C# (hay C Sharp) là một ngôn ngữ lập trình đơn giản, được phát triển bởi đội ngũ kỹ sư của Microsoft vào năm 2000. C# là ngôn ngữ lập trình hiện đại, hướng đối tượng và được xây dựng trên nền tảng của hai ngôn ngữ mạnh nhất là C++ và Java.

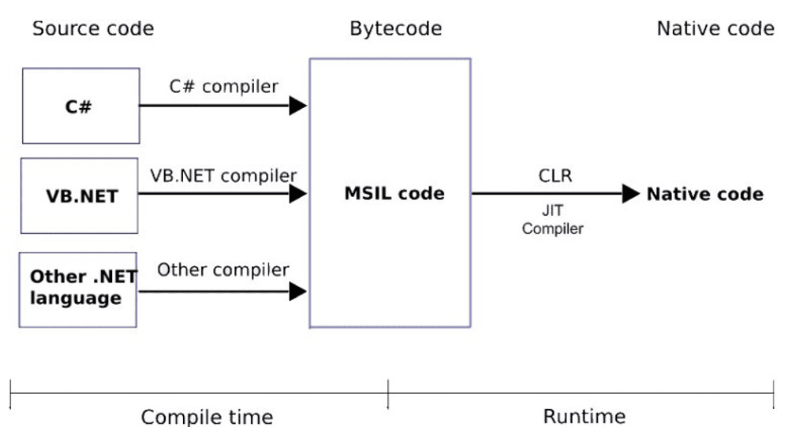
-Ngôn ngữ C# khá đơn giản, chỉ khoảng 80 từ khóa và hơn mười mấy kiểu dữ liệu được xây dựng sẵn. Tuy nhiên, ngôn ngữ C# có ý nghĩa cao khi nó thực thi những khái niệm lập trình hiện đại. C# là một ngôn ngữ lập trình hướng đối tượng (OOP), có khả năng kế thừa, đa hình và đóng gói. Nó cũng hỗ trợ các tính năng lập trình chức năng, các cấu trúc điều khiển, các biểu thức chính quy và các tính năng của các ngôn ngữ lập trình khác.

-C# được sử dụng rộng rãi trong phát triển ứng dụng máy tính, ứng dụng di động, ứng dụng web, các dịch vụ đám mây và các ứng dụng trò chơi điện tử. Nó được tích hợp trong các sản phẩm của Microsoft như Windows, Office và các dịch vụ đám mây của họ.

-C# cũng được xem là một trong những ngôn ngữ lập trình phổ biến nhất trên thế giới, đứng thứ 5 trong bảng xếp hạng Tiobe Index năm 2021.

-Trong các ứng dụng Windows truyền thống, mã nguồn chương trình được biên dịch trực tiếp thành mã thực thi của hệ điều hành. Trong các ứng dụng sử dụng .NET Framework, mã nguồn chương trình (C#, VB.NET) được biên dịch thành mã ngôn ngữ trung gian MSIL (Microsoft intermediate language).

-Sau đó mã này được biên dịch bởi Common Language Runtime (CLR) để trở thành mã thực thi của hệ điều hành. Hình bên dưới thể hiện quá trình chuyển đổi MSIL code thành native code.



Hình Quá trình thực thi chuyển đổi mã nguồn trong HĐH

-C# với sự hỗ trợ mạnh mẽ của .NET Framework giúp cho việc tạo một ứng dụng windows Forms hay WPF (Windows Presentation Foundation), phát triển game, ứng dụng web, ứng dụng mobile trở nên dễ dàng.

**2.1.2. Đặc trưng của ngôn ngữ lập trình C#**

2.1.2.1 C# là ngôn ngữ đơn giản, mạnh mẽ

-C# được dựng trên nền tảng C++ và java, ảnh hưởng bởi Delphi, VisuaBasic nên ngôn ngữ C# được thừa hưởng các ưu điểm và loại bỏ các yếu điểm cảu các ngôn ngữ trên, vì vậy nó khá đơn giản, đồng thười loại bỏ các củ pháp dư thừa và thêm vào đó các củ pháp cải tiến hơn.

-C# là ngôn ngữ lập trình bậc cao, đa nền tảng vì vậy nó dễ dàng tiếp cận và phù hợp cho người mới bắt đầu học, ví dụ câu lệnh kinh điển dành cho người mới bắt đầu học.

2.1.2.2. C# là ngôn ngữ đa năng và hiện đại

-C# phù hợp cho việc phát triển trong thời đại 4.0, bao gồm việc phát triển Web, mobile app, game, học máy và trí tuệ nhân tạp, phát triển đám mây ...

2.1.2.3. C# là một ngôn ngữ lập trình hướng đối tượng đồng thời hỗ trợ lập trình chức năng.

-C# hỗ trợ mạnh mẽ cho phương pháp lập trình hướng đối tượng, ngoài ra C# còn hỗ trợ các phương pháp lập trình chức năng thông qua các biểu thức lamba, khớp mẫu. function, các thuộc tính bất biến.

2.1.2.4. C# là một ngôn ngữ lập trình phổ biến và phát triển nhất

-Theo TIOBE Index, tính đến tháng 10/2020, C# là ngôn ngữ phổ biến thứ 5 thế giới

-Theo PYPL, tính đến tháng 10/2020, C# là ngôn ngữ được cộng đồng quan tâm và chia sẻ nhiều thứ 4 thế giới.

-Cộng đồng phát triển và số người theo học ngôn ngữ C# tăng không ngừng theo mối năm. Theo ước tính 10/2020, cộng đồng phát triển C# là hơn 6 triệu người.

2.1.2.5. C# kết hợp mạnh mẽ với nền tảng .NET một khung nền tảng đươc đầu tư rất mạnh của Microsoft.

-Ngoài ra, C# còn có những ưu điểm:

* C# là ngôn ngữ lập trình mã nguồn mở, vì vậy C# là miễn phí với tất cả mọi người, đồng thời mọi người đều có thể tham gia phát triển, đề xuất thiết kế ngôn ngữ C#.
* C# là ngôn ngữ đa nền tảng vì vật có thể biên dịch trên nhiều nền tảng mạng máy tính khác nhau (Windows, Linux, MacOS)
* C# có hiệu suất cao và tốc độ thực thi nhanh do sử dụng trình biên dịch trung gian, điểm cộng nữa là tốc độ phát triển phần mềm nhanh chóng so với đa số các ngôn ngữ hiện tại.
* C# có IDE Visual Studio Code, Netbeans, ...
* C# có cấu trúc khá gần gũi với các ngôn ngữ lập trình truyền thống, song cũng được bổ sung các yếu tố mang tính hiện đại nên dễ dàng tiếp cận cho người mới học và học nhanh với c#.
* C# và NET được đánh giá có design tốt, vì vậy cú pháp và logic rất nhất quán, mã nguồn C# dễ đọc và mở rộng.
* C# được thiết kế và phát triển bới Microsoft nên được rất nhiều Micosoft quan tâm và hỗ trợ.
* C# cũng được hỗ trợ bởi một cộng đồng rộng lớn, có nhiều tài liệu và công cụ hỗ trợ để giúp lập trình viên tạo ra các ứng dụng chất lượng cao.

## **2.2 Các framework sử dụng**

### ***2.2.1 Framework ASP.NET CORE:***

-Khái niệm: ASP.NET là một framework phát triển ứng dụng web được phát triển bởi Microsoft. Nó cung cấp một nền tảng mạnh mẽ để xây dựng các ứng dụng web đa dạng, bao gồm cả trang web bán đồ công nghệ.

-Ưu điểm: ASP.NET cung cấp tính năng mạnh mẽ như xử lý sự kiện trên phía máy chủ, quản lý trạng thái của ứng dụng, bảo mật, xử lý dữ liệu, và hỗ trợ các ngôn ngữ lập trình như C# và VB.NET. Nó cũng tích hợp tốt với các công nghệ khác của Microsoft như SQL Server và Azure.

### ***2.2.2 Framework ASP.NET MVC:***

-Khái niệm: ASP.NET MVC là một phần mở rộng của Framework ASP.NET, tập trung vào mô hình phát triển Model-View-Controller (MVC). Nó giúp phân tách logic ứng dụng thành các phần riêng biệt như Model (mô hình dữ liệu), View (giao diện người dùng) và Controller (xử lý logic).

-Ưu điểm: ASP.NET MVC cung cấp sự linh hoạt trong việc phát triển ứng dụng web, giúp tách biệt giao diện người dùng và logic ứng dụng. Nó cung cấp các công cụ và tính năng hỗ trợ cho việc xây dựng các chức năng như định tuyến URL, quản lý trạng thái, xử lý form, và kiểm thử đơn vị.

### ***2.2.3 Entity framework***

-Khái niệm: Entity Framework (EF) là một thư viện ORM được phát triển từ năm 2008 bởi Microsoft. Framework này tra đời nhằm hỗ trợ sự tương tác giữa các ứng dụng trên nền tảng .NET với các CSDL quan hệ. ORM là một tập hợp công nghệ có thể làm việc với các hệ quản trị CSDL quan hệ từ các ngôn ngữ lập trình hướng đối tượng mà không phải trực tiếp để xử lý truy vấn SQL.

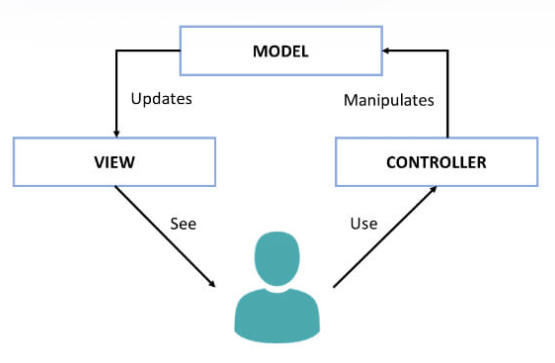
🡪 Entity Framework sẽ giúp các nhà phát triển web tương tác với CSDL theo phương pháp hướng đối tượng với ít code hơn so với các ứng dụng truyền thống. Ưu điểm nổi bật nhất của ứng dụng này là giúp lập trình viên giảm thiểu việc lập trình mã nguồn để thực hiện truy cập và tương tác với CSDL.

-Đặc điểm:

* Đa nền tảng: Entity Framework Core là một Framework đa nền tảng có thể chạy trên Window, Linux và MacOS.
* Mô hình hóa: Entity Framework tạo ra EDM (Entity Data Model) dựa trên các entity (thực thể) POCO (Plain Old CLR Object) với các thuộc tính get/set của các loại dữ liệu khác nhau. Framework sẽ sử dụng mô hình này khi truy vấn hoặc lưu dữ liệu vào CSDL.
* Truy vấn: Entity Framework cho phép lập trình viên sử dụng các truy vấn LINQ (C#/VB.NET) để truy xuất dữ liệu từ CSDL. Ngoài ra, EF cũng thực hiện các truy vấn SQL thô trực tiếp tới CSDL.
* Theo dõi thay đổi: Entity Framework theo dõi các thay đổi xảy ra đối với các phiên bản entity cần được gửi tới CSDL.
* Giao dịch: Entity Framework thực hiện quản lý giao dịch tự động trong khi truy vấn hoặc lưu dữ liệu. Framework cũng cung cấp các tùy chọn để tùy chỉnh quản lý giao dịch.
* Bộ nhớ đệm: Entity Framework bao gồm cấp bộ nhớ đệm cấp độ đầu tiên. Vì vậy, truy vấn lặp đi lặp lại sẽ trả về dữ liệu từ bộ đệm thay vì truy cập vào CSDL.
* Các quy ước tích hợp sẵn: Entity Framework tuân theo các quy ước về mẫu lập trình cấu hình, bao gồm một bộ quy tắc mặc định tự động định cấu hình mô hình Entity Framework.
* Cấu hình: Entity Framework cho phép cấu hình mô hình bằng cách sử dụng các thuộc tính chú thích dữ liệu hoặc API để ghi đè các quy ước mặc định.
* Di chuyển: Entity Framework cung cấp một tập hợp các lệnh di chuyển có thể được thực thi trên NuGet Package Manager Console hoặc Command Line Interface (Giao diện dòng lệnh để tạo).

**2.2.4 Mô hình MVC**

-Khái niệm: MVC là viết tắt của cụm từ “Model-View-Controller“. Đây là mô hình thiết kế được sử dụng trong kỹ thuật phần mềm. MVC là một mẫu kiến trúc phần mềm để tạo lập giao diện người dùng trên máy tính. MVC chia thành ba phần được kết nối với nhau và mỗi thành phần đều có một nhiệm vụ riêng của nó và độc lập với các thành phần khác.



Hình Mô hình MVC

-Các thành phần: Mô hình MVC gồm 3 loại chính là thành phần bên trong không thể thiếu khi áp dụng mô hình này:

* Model: Là bộ phận có chức năng lưu trữ toàn bộ dữ liệu của ứng dụng và là cầu nối giữa 2 thành phần bên dưới là View và Controller. Một model là dữ liệu được sử dụng bởi chương trình. Đây có thể là cơ sở dữ liệu, hoặc file XML bình thường hay một đối tượng đơn giản. Chẳng hạn như biểu tượng hay là một nhân vật trong game.
* View: Đây là phần giao diện (theme) dành cho người sử dụng. View là phương tiện hiển thị các đối tượng trong một ứng dụng. Chẳng hạn như hiển thị một cửa sổ, nút hay văn bản trong một cửa sổ khác. Nó bao gồm bất cứ thứ gì mà người dùng có thể nhìn thấy được.
* Controller: Là bộ phận có nhiệm vụ xử lý các yêu cầu người dùng đưa đến thông qua View. Một controller bao gồm cả Model lẫn View. Nó nhận input và thực hiện các update tương ứng.

-Luồng xử lý: Luồng xử lý trong của mô hình MVC, bạn có thể hình dung cụ thể và chi tiết qua từng bước dưới đây:

Khi một yêu cầu của từ máy khách (Client) gửi đến Server. Thì bị Controller trong MVC chặn lại để xem đó là URL request hay sự kiện.

Sau đó, Controller xử lý input của user rồi giao tiếp với Model trong MVC.

Model chuẩn bị data và gửi lại cho Controller.

Cuối cùng, khi xử lý xong yêu cầu thì Controller gửi dữ liệu trở lại View và hiển thị cho người dùng trên trình duyệt. Ở đây, View không giao tiếp trực tiếp với Model. Sự tương tác giữa View và Model sẽ chỉ được xử lý bởi Controller.

### ***2.2.5 Hệ quản trị cơ sở dữ liệu MySQL***

-Khái niệm: Hệ quản trị cơ sở dữ liệu MySQL được hiểu như là chương trình dùng để quản lý hệ thống cơ sở dữ liệu, trong đó, cơ sở dữ liệu là một hệ thống lưu trữ thông tin được sắp xếp rõ ràng, phân lớp ngăn nắp. Nó giúp bạn có thể truy cập dữ liệu một cách thuận lợi và nhanh chóng nhất. Vì hỗ trợ đa số các ngôn ngữ lập trình nên MySQL chính là hệ quản trị cơ sở dữ liệu tự do nguồn mở phổ biến nhất trên thế giới. Hiện MySQL đang được các nhà phát triển rất ưa chuộng trong quá trình phát triển ứng dụng. MySQL là cơ sở dữ liệu tốc độ cao, ổn định hoạt động trên nhiều hệ điều hành, cung cấp một hệ thống lớn các hàm tiện ích rất mạnh. Đặc biệt, hệ quản trị cơ sở dữ liệu MySQL hoàn toàn miễn phí nên người dùng có thể thoải mái tải về từ trang chủ. Nó có rất nhiều những phiên bản cho các hệ điều hành khác nhau.

-Đặc điểm: Hệ quản trị cơ sở dữ liệu MySQL chính là một trong những phần mềm quản trị CSDL dạng server based, hệ gần giống với SQL SERVER. MySQL là phần mềm quản lý dữ liệu thông qua CSDL. Và mỗi một CSDL đều có bảng quan hệ chứa dữ liệu riêng biệt. MySQL có cơ chế quản lý sử dụng riêng giúp cho mỗi người sử dụng đều có thể quản lý cùng lúc một hay nhiều CSDL khác nhau. Và mỗi người dùng đều có 1 username và password để truy nhập và truy xuất đến CSDL. Khi truy vấn đến CSDL của MySQL, bạn phải cung cấp tài khoản và mật khẩu có quyền sử dụng cơ sở dữ liệu đó.

***2.2.6. Identity Framework***  
-Identity Framework là một framework xác thực và cấp quyền cho .NET Core và .NET Framework. Nó cung cấp một số tính năng, bao gồm:

* Xác thực người dùng: Identity Framework cho phép bạn xác thực người dùng bằng tên người dùng và mật khẩu, mã thông báo hoặc phương thức xác thực hai yếu tố.
* Cấp quyền người dùng: Identity Framework cho phép bạn cấp quyền cho người dùng truy cập vào các tài nguyên và chức năng trong ứng dụng của bạn.
* Cấu hình linh hoạt: Identity Framework có thể được cấu hình để đáp ứng nhu cầu của ứng dụng của bạn.
* Để sử dụng Identity Framework, bạn cần tạo một lớp IdentityUser để đại diện cho người dùng trong ứng dụng của bạn. Bạn cũng cần tạo một lớp IdentityRole để đại diện cho vai trò trong ứng dụng của bạn.
* Để xác thực người dùng, bạn có thể sử dụng phương thức SignInAsync() của lớp IdentityManager. Để cấp quyền cho người dùng, bạn có thể sử dụng phương thức AddClaim() của lớp IdentityUser.
* Identity Framework là một framework xác thực và cấp quyền mạnh mẽ và linh hoạt có thể giúp bạn xây dựng các ứng dụng web an toàn và bảo mật.
* Dưới đây là một số lợi ích của việc sử dụng Identity Framework:
* Tăng tính bảo mật: Identity Framework cung cấp một số tính năng bảo mật nâng cao, chẳng hạn như xác thực hai yếu tố và bảo vệ chống lại tấn công brute force.
* Tăng năng suất: Identity Framework giúp bạn tiết kiệm thời gian và công sức bằng cách cung cấp một framework tiêu chuẩn để xác thực và cấp quyền người dùng.
* Linh hoạt: Identity Framework có thể được cấu hình để đáp ứng nhu cầu của ứng dụng của bạn.

### ***2.2.6 Nguyên lý SOLID***

-Dependency Injection (DI) là một nguyên tắc thiết kế phổ biến trong lập trình phần mềm, đặc biệt trong lập trình hướng đối tượng. Nó đề cập đến cách các đối tượng hoặc thành phần của một ứng dụng tương tác và phụ thuộc vào nhau.

-Trong DI, các phụ thuộc (dependencies) của một đối tượng không được đối tượng đó tự tạo ra, mà thay vào đó, chúng được cung cấp từ bên ngoài. Điều này giúp giảm sự phụ thuộc trực tiếp giữa các đối tượng và làm cho ứng dụng linh hoạt hơn. DI thường được sử dụng để đảm bảo nguyên tắc "Inversion of Control" (IoC), trong đó quá trình quản lý các đối tượng và luồng điều khiển của ứng dụng được đảo ngược.

-Các ưu điểm của DI bao gồm:

* Linh hoạt: Bạn có thể thay đổi hoặc nâng cấp một phụ thuộc mà không ảnh hưởng đến các đối tượng khác trong ứng dụng.
* Dễ kiểm tra và đảm bảo chất lượng: DI giúp bạn dễ dàng kiểm tra độc lập từng phần của ứng dụng, giúp dễ dàng phát hiện và sửa lỗi.
* Tái sử dụng: Bạn có thể tái sử dụng các phụ thuộc cho nhiều đối tượng, giảm lặp lại mã.
* Dễ quản lý và bảo trì: Các phụ thuộc được quản lý và cấu hình từ một nơi duy nhất, dễ dàng theo dõi và bảo trì.

# **CHƯƠNG III: PHÂN TÍCH THIẾT KẾ**

## **3.1. Phân tích yêu cầu của trang web**

### 3.1.1. Mô tả nghiệp vụ

Xây dựng website bán đồ công nghệ với yêu cầu phục vụ cho việc bán hàng online và quản lý sản phẩm, thống kê doanh thu theo tháng, năm. Hệ thống bao gồm các đối tượng như sau:

Admin: là người quản trị hệ thống, có quyền hạn cao nhất

Người mua hàng: là người mua đăng ký tài khoản để mua hàng

Mỗi đối tượng sẽ được phân một loại tài khoản riêng với những chức năng phù hợp với nhiệm vụ của mình

3.1.2. Yêu cầu về chức năng

Đăng nhập

Đăng nhập hệ thống: có thể thực hiện đăng nhập để xác minh danh tính trong hệ thống.

Đăng Xuất hệ thống: trong quá trình sử dụng, người dùng muốn sử dụng tài khoản khác hay không muốn sử dụng hệ thống nữa.

Trang thông tin người dùng

Xem thông tin cá nhân

Sửa thông tin cá nhân

Trang sản phẩm

Xem danh sách các sản phẩm

Tìm sản phẩm theo danh mục

Trang chi tiết sản phẩm

Xem thông tin chi tiết của từng sản phẩm

Thêm sản phẩm vào giỏ hàng

Trang giỏ hàng

Xem danh sách các sản phẩm được chọn

Tính tổng tiền của đơn hàng

Điền thông tin chi tiết của người mua

Thanh toán đơn hàng

Quản lý sản phẩm

Xem danh sách các sản phẩm

Thêm mới sản phẩm

Sửa thông tin sản phẩm

Xoá khỏi hệ thống

Quản lý đơn hàng

Xem danh sách các đơn hàng

Sửa trạng thái đơn hàng

Quản lý người dùng

Xem danh sách người dùng

Thêm mới tài khoản

Sửa thông tin người dùng

Xoá khỏi hệ thống

3.1.3. Yêu cầu phi chức năng

Giao diện bắt mắt, dễ dàng sử dụng

Các chức năng rõ ràng, tối ưu cho người sử dụng, tiết kiệm thao tác.

### ***3.2. Phân tích hệ thống***

3.2.1. Xác định tác nhân

Admin: là người quản trị hệ thống, có quyền hạn cao nhất

Người mua hàng: là người mua đăng ký tài khoản để mua hàng

3.2.2. Các chức năng của tác nhân

|  |  |
| --- | --- |
| Tác nhân | Chức năng |
| Admin | Có toàn bộ quyền, chức năng của hệ thống |
| Người mua hàng | 1. Đăng nhập, đăng xuất;  2. Xem, tìm kiếm thông tin sản phẩm  3. Quản lý thông tin cá nhân  4. Xem trạng thái đơn hàng  5. Thêm sản phẩm vào giỏ hàng, thanh toán |

3.2.3. Biểu đồ Use case tổng quát

A picture containing diagram, sketch, line, drawing

Description automatically generated

Hình 3 Biểu đồ Use case tổng quát

3.2.4. Biểu đồ chức năng tổng quát

A picture containing text, diagram, screenshot, line

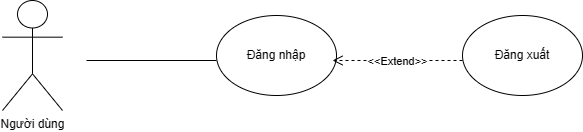
Description automatically generated

Hình 4 Biểu đồ chức năng tổng quát

3.2.5. Chức năng đăng nhập, đăng xuất

3.2.5.1. Biểu đồ Use Case

3.2.5.1.1 Người dùng



Hình 5 Biểu đồ Use case chức năng đăng nhập, xuất người dùng.

3.2.5.1.2 Admin

A picture containing diagram, circle, line

Description automatically generated

Hình 6 Biểu đồ Use Case chức năng đăng nhập, xuất Admin

3.2.5.2. Kịch bản

3.2.5.2.1 Người dùng

|  |  |
| --- | --- |
| Tên Use Case | Đăng nhập, đăng xuất |
| Tác nhân | Người dùng |
| Tiền điều kiện(đăng nhập) | Người dùng đã có tài khoản trong hệ thống và đăng nhập thành công. |
| Mục đích | Xác thực Người dùng đăng nhập vào hệ thống thông qua “Tài khoản” và “Mật khẩu” |
| Luồng tương tác chính | 1. Hệ thống hiển thị giao diện chính website.  2. Người dùng N chọn đăng nhập vào hệ thống.  3. Hệ thống hiển thị giao diện đăng nhập.  4. Người dùng N nhập “Tên đăng nhập” và “Mật khẩu”.  5. Hệ thống kiểm tra và xác nhận thông tin đăng nhập. Hiển thị giao diện chính của hệ thống. |
| Luồng tương tác ngoại lệ | 5. Người dùng N nhập thiếu hoặc sai định dạng thông tin đăng nhập.  5.1. Hệ thống hiển thị thông báo nhập sai và yêu cầu nhập lại  5.2. Thực hiện lại từ bước 3 đến bước 5 |
| Hậu điều kiện | - Trường hợp thành công: Hệ thống sẽ hiển thị giao diện quản lý chính. Người dùng có thể thực hiện các chức năng theo đúng  quyền hạn của mình.  - Trường hợp thất bại: Hệ thống sẽ đưa ra thông báo” Tài khoản nhập sai hoặc không tồn tại” và yêu cầu đăng nhập lại. |
| Tiền điều kiện(đăng xuất) | - Người dùng đã đăng nhập |
| Mục đích | - Đăng xuất khỏi hệ thống |
| Luồng tương tác chính | 1. Người dùng N ấn vào đăng xuất  2. Hệ thống hiển thị giao diện chưa đăng nhập |

3.2.5.2.2 Admin

|  |  |
| --- | --- |
| Tên Use Case | Đăng nhập, đăng xuất |
| Tác nhân | Admin |
| Tiền điều kiện(đăng nhập) | Admin đã có tài khoản trong hệ thống và đăng nhập thành công. |
| Mục đích | Xác thực Admin đăng nhập vào hệ thống thông qua “Tài khoản” và “Mật khẩu” |
| Luồng tương tác chính | 1. Hệ thống hiển thị giao diện chính website.  2. Admin A chọn đăng nhập vào hệ thống.  3. Hệ thống hiển thị giao diện đăng nhập.  4. Admin A nhập “Tên đăng nhập” và “Mật khẩu”.  5. Hệ thống kiểm tra và xác nhận thông tin đăng nhập. Hiển thị giao diện chính của hệ thống. |
| Luồng tương tác ngoại lệ | 5. Admin A nhập thiếu hoặc sai định dạng thông tin đăng nhập.  5.1. Hệ thống hiển thị thông báo nhập sai và yêu cầu nhập lại  5.2. Thực hiện lại từ bước 3 đến bước 5 |
| Hậu điều kiện | - Trường hợp thành công: Hệ thống sẽ hiển thị giao diện quản lý chính. Người dùng có thể thực hiện các chức năng theo đúng  quyền hạn của mình.  - Trường hợp thất bại: Hệ thống sẽ đưa ra thông báo” Tài khoản nhập sai hoặc không tồn tại” và yêu cầu đăng nhập lại. |
| Tiền điều kiện(đăng xuất) | - Admin đã đăng nhập |
| Mục đích | - Đăng xuất khỏi hệ thống |
| Luồng tương tác chính | - Admin A ấn vào đăng xuất  - Hiển thị giao diện chưa đăng nhập |

3.2.5.3. Biểu đồ hoạt động

A picture containing diagram, line, screenshot, text

Description automatically generated

Hình 7 Biểu đồ hoạt động chức năng đăng nhập, xuất

3.2.5.4. Biểu đồ trình tự

A picture containing text, diagram, parallel, line

Description automatically generated

Hình 8 Biểu đồ trình tự chức năng đăng nhập, xuất

3.2.6. Chức năng quản lý thông tin cá nhân

3.2.6.1. Biểu đồ Use Case

3.2.6.1.1 Người dùng

A picture containing diagram, text, circle, line

Description automatically generated

Hình 9 Biểu đồ Use Case chức năng quản lý thông tin cá nhân người dùng

3.2.6.1.2 Admin

A diagram of an extinguisher

Description automatically generated with low confidence

Hình 10 Biểu đồ Use Case chức năng quản lý thông tin cá nhân admin.

3.2.6.2. Kịch bản

3.2.6.2.1 Người dùng

|  |  |
| --- | --- |
| Tên Use Case | Quản lý thông tin cá nhân |
| Tác nhân | Người dùng |
| Tiền điều kiện | Người dùng đã có tài khoản trong hệ thống và đăng nhập thành công. |
| Mục đích | Xem, sửa thông tin cá nhân của mình |
| Luồng tương tác chính | 1. Người dùng N click vào sửa thông tin cá nhân.  2. Hệ thống hiển thị form thông tin hiện tại của người dùng.  3. Người dùng N nhập vào thông tin còn thiếu hoặc cần sửa rồi lưu thông tin.  4. Hệ thống kiểm tra và lưu thông tin vào csdl. |
| Luồng tương tác ngoại lệ | 4. Người dùng N nhập thiếu hoặc sai định dạng thông tin cá nhân.  4.1. Hệ thống hiển thị thông báo nhập sai và yêu cầu nhập lại  4.2. Thực hiện lại từ bước 2 đến bước 4 |

3.2.6.2.2 Admin

|  |  |
| --- | --- |
| Tên Use Case | Quản lý thông tin cá nhân |
| Tác nhân | Admin |
| Tiền điều kiện | Admin đã có tài khoản trong hệ thống và đăng nhập thành công. |
| Mục đích | Xem, sửa thông tin cá nhân của mình |
| Luồng tương tác chính | 1. Admin A click vào sửa thông tin cá nhân.  2. Hệ thống hiển thị form thông tin hiện tại của người dùng.  3. Admin A nhập vào thông tin còn thiếu hoặc cần sửa rồi lưu thông tin.  4. Hệ thống kiểm tra và lưu thông tin vào csdl. |
| Luồng tương tác ngoại lệ | 4. Admin A nhập thiếu hoặc sai định dạng thông tin cá nhân.  4.1. Hệ thống hiển thị thông báo nhập sai và yêu cầu nhập lại  4.2. Thực hiện lại từ bước 2 đến bước 4 |

3.2.6.3. Biểu đồ hoạt động

A picture containing diagram, screenshot, text, line

Description automatically generated

Hình 11 Biểu đồ hoạt động chức năng quản lý thông tin cá nhân

3.2.6.4. Biểu đồ trình tự

A picture containing text, diagram, parallel, line

Description automatically generated

Hình 12 Biểu đồ trình tự chức năng quản lý thông tin cá nhân

3.2.7. Chức năng thêm vào giỏ hàng

3.2.7.1. Biểu đồ Use Case

3.2.7.1.1 Người dùng

A picture containing diagram, circle, text, line

Description automatically generated

Hình 13 Biểu đồ Use Case chức năng thêm vào giỏ hàng

3.2.7.2. Kịch bản

3.2.7.2.1 Người dùng

|  |  |
| --- | --- |
| Tên Use Case | Thêm vào giỏ hàng |
| Tác nhân | Người dùng |
| Tiền điều kiện | Người dùng đã có tài khoản trong hệ thống và đăng nhập thành công. |
| Mục đích | Thêm sản phẩm của người mua vào giỏ hàng |
| Luồng tương tác chính | 1. Người dùng N click vào nút thêm vào giỏ hàng  2. Hệ thống hiển thị form giỏ hàng  3. Người dùng N nhập vào thông tin số lượng sản phẩm, số điện thoại, địa chỉ giao hàng rồi bấm nút xác nhận thông tin.  4. Hệ thống kiểm tra và lưu thông tin vào csdl.  5. Người dùng N click vào thanh toán, chọn hình thức thanh toán: COD, banking  6. Hệ thống tạo ra đơn hàng của người dùng N, thay đổi số lượng của sản phẩm được mua |
| Luồng tương tác ngoại lệ | 4. Người dùng N nhập thiếu hoặc sai định dạng thông tin cá nhân.  4.1. Hệ thống hiển thị thông báo nhập sai và yêu cầu nhập lại  4.2. Thực hiện lại từ bước 2 đến bước 4 |

3.2.7.3. Biểu đồ hoạt động

A picture containing text, screenshot, diagram, line

Description automatically generated

Hình 14 Biểu đồ hoạt động chức năng thêm vào giỏ hàng

3.2.7.4. Biểu đồ trình tự

A picture containing text, diagram, parallel, line

Description automatically generated

Hình 15 Biểu đồ trình tự chức năng thêm vào giỏ hàng

3.2.8. Chức năng quản lý sản phẩm

3.2.8.1. Biểu đồ Use Case

3.2.8.1.2 Admin

A picture containing diagram, text, circle, line

Description automatically generated

Hình 16 Biểu đồ Use Case chức năng quản lý sản phẩm

3.2.8.2. Kịch bản

3.2.8.2.2 Admin

|  |  |
| --- | --- |
| Tên Use Case | Quản lý sản phẩm |
| Tác nhân | Admin |
| Tiền điều kiện | Admin đăng nhập thành công. |
| Mục đích | Xem, thêm sản phẩm |
| Luồng tương tác chính | 1. Admin A click vào quản lý sản phẩm.  2. Hệ thống hiển thị form sản phẩm  3. Admin A nhập vào sản phẩm muốn thêm rồi lưu thông tin.  4. Hệ thống kiểm tra và lưu thông tin vào csdl. |
| Luồng tương tác ngoại lệ | 4. Admin A nhập thiếu hoặc sai định dạng thông tin cá nhân.  4.1. Hệ thống hiển thị thông báo nhập sai và yêu cầu nhập lại  4.2. Thực hiện lại từ bước 2 đến bước 4 |

3.2.8.3. Biểu đồ hoạt động

A picture containing text, screenshot, diagram, line

Description automatically generated

Hình 17 Biều đồ hoạt động chức năng quản lý sản phẩm

3.2.8.4. Biểu đồ trình tự

A picture containing text, diagram, parallel, line

Description automatically generated

Hình 18 Biểu đồ trình tự chức năng quản lý sản phẩm

3.2.9. Chức năng quản lý đơn hàng

3.2.9.1. Biểu đồ Use Case

3.2.9.1.1 Người dùng

A picture containing diagram, text, circle, line

Description automatically generated

Hình 19 Biểu đồ Use Case chức năng quản lý đơn hàng người dùng

3.2.9.1.2 Admin

A picture containing diagram, text, circle, line

Description automatically generated

Hình 20 Biểu đồ Use Case chức năng quản lý đơn hàng Admin

3.2.9.2. Kịch bản

3.2.9.2.1 Người dùng

|  |  |
| --- | --- |
| Tên Use Case | Quản lý đơn hàng |
| Tác nhân | Người dùng |
| Tiền điều kiện | Người dùng đăng nhập thành công. |
| Mục đích | Xem trạng thái đơn hàng |
| Luồng tương tác chính | 1. Người dùng N click vào quản lý trạng thái đơn hàng  2. Hệ thống hiển thị form xem đơn hàng |

3.2.9.2.2 Admin

|  |  |
| --- | --- |
| Tên Use Case | Quản lý đơn hàng |
| Tác nhân | Admin |
| Tiền điều kiện | Admin đăng nhập thành công. |
| Mục đích | Xem, sửa thông tin đơn hàng |
| Luồng tương tác chính | 1. Admin A click vào quản lý trạng thái đơn hàng  2. Hệ thống hiển thị form đơn hàng  3. Admin A nhập vào thông tin còn thiếu hoặc cần sửa rồi lưu trạng thái  4. Hệ thống kiểm tra và lưu thông tin vào csdl. |
| Luồng tương tác ngoại lệ | 4. Admin A nhập thiếu hoặc sai định dạng thông tin cá nhân.  4.1. Hệ thống hiển thị thông báo nhập sai và yêu cầu nhập lại  4.2. Thực hiện lại từ bước 2 đến bước 4 |

3.2.9.3. Biểu đồ hoạt động

A picture containing text, diagram, screenshot, line

Description automatically generated

Hình 21 Biểu đồ hoạt động chức năng quản lý đơn hàng

3.2.9.4. Biểu đồ trình tự

A picture containing text, diagram, parallel, line

Description automatically generated

Hình 22 Biểu đồ trình tự chức năng quản lý đơn hàng

3.2.10. Chức năng quản lý người dùng

3.2.10.1. Biểu đồ Use Case

3.2.10.1.2 Admin

A picture containing diagram, circle, text, line

Description automatically generated

Hình 23 Biểu đồ Use Case chức năng quản lý Admin

3.2.10.2. Kịch bản

3.2.10.2.2 Admin

|  |  |
| --- | --- |
| Tên Use Case | Quản lý người dùng |
| Tác nhân | Admin |
| Tiền điều kiện | Admin đăng nhập thành công. |
| Mục đích | Xem, xoá thông tin của người dùng |
| Luồng tương tác chính | 1. Admin A click vào quản lý người dùng  2. Hệ thống hiển thị form thông tin hiện tại của người dùng.  3. Admin A xem hoặc xoá thông tin người dùng rồi bấm lưu thông tin.  4. Hệ thống kiểm tra và lưu thông tin vào csdl. |
| Luồng tương tác ngoại lệ | 4. Người dùng A nhập thiếu hoặc sai định dạng thông tin cá nhân.  4.1. Hệ thống hiển thị thông báo nhập sai và yêu cầu nhập lại  4.2. Thực hiện lại từ bước 2 đến bước 4 |

3.2.10.3. Biểu đồ hoạt động

A picture containing text, diagram, line, parallel

Description automatically generated

Hình 24 Biểu đồ hoạt động chức năng quản lý người dùng

3.2.10.4. Biểu đồ trình tự

A picture containing text, diagram, parallel, line

Description automatically generated

Hình 25 Biểu đồ trình tự chức năng quản lý người dùng

## **3.3. Thiết kế cơ sở dữ liệu cho hệ thống**

-Bảng CSDL tự tạo

A computer screen shot of a diagram

Description automatically generated

-Bảng Identity Framework tạo:

A screenshot of a computer

Description automatically generated

3.3.1. Bảng orderItems

Bảng 1 Bảng orderItems

|  |  |  |
| --- | --- | --- |
| Tên trường | Kiểu dữ liệu | Mô tả |
| OrderItemID | Int | Khoá chính |
| OrderID | Int | Khoá ngoại |
| ProductID | Int | Khoá ngoại |
| Quantity | Int | Số lượng |

3.3.2. Bảng orders

Bảng 2 Bảng orders

|  |  |  |
| --- | --- | --- |
| Tên trường | Kiểu dữ liệu | Mô tả |
| OrderId | Int | Khoá chính |
| UserId | Int | Khoá ngoại |
| OrderDate | Date | Ngày đặt |
| Status | Int | Trạng thái |

3.3.3. Bảng products

Bảng 3 Bảng products

|  |  |  |
| --- | --- | --- |
| Tên trường | Kiểu dữ liệu | Mô tả |
| ProductId | Int | Khoá chính |
| Name | Varchar(255) | Tên sản phẩm |
| Description | Text | Mô tả |
| Price | Float | Giá |
| ImageUrl | Varchar(255) | Url ảnh |
| StockQuantity | Int | Số lượng |
| CategoryId | Int | Khoá ngoại |

3.3.4. Bảng users

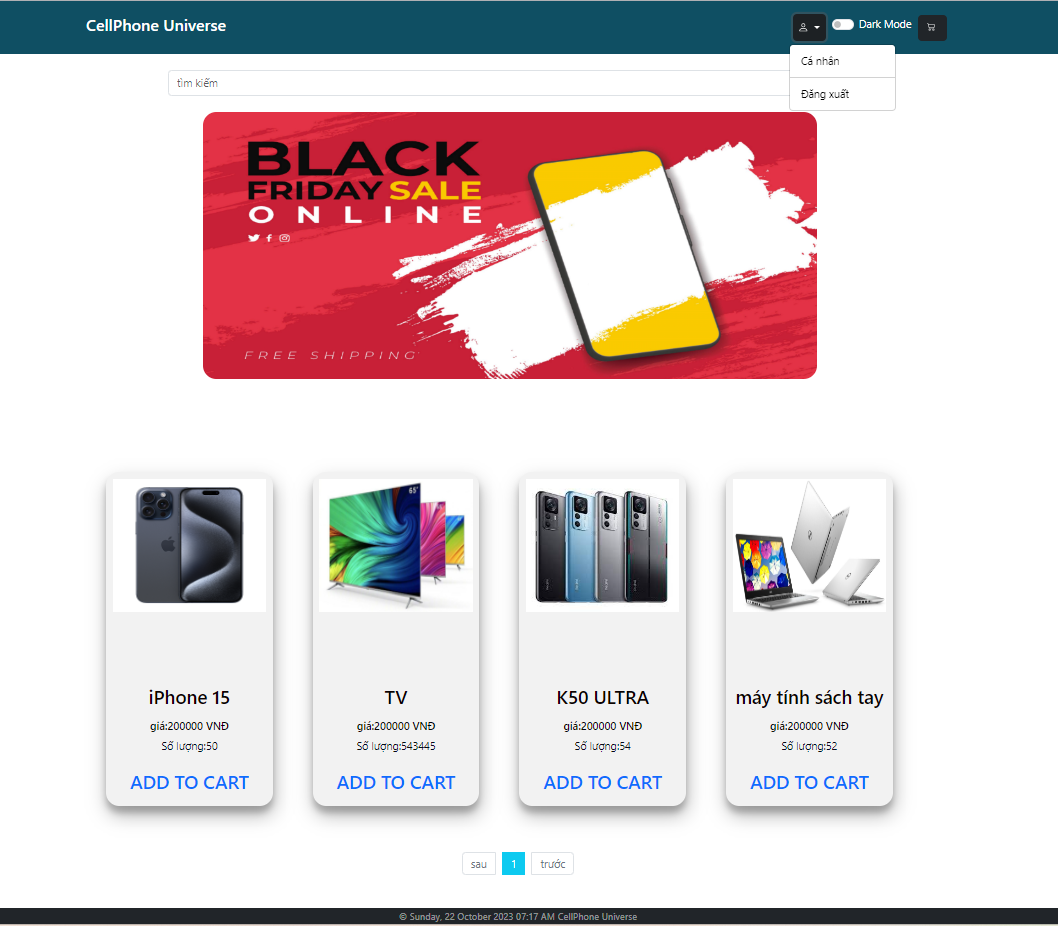
Bảng 4 Bảng users

|  |  |  |
| --- | --- | --- |
| Tên trường | Kiểu dữ liệu | Mô tả |
| UserId | Nvarchar(50) | Khoá chính |
| NameUser | Varchar(255) | Tên người dùng |
| AddressUser | Varchar(255) | Địa chỉ |
| EmailUser | Varchar(255) | Email |
| PhoneUser | Varchar(50) | Số điện thoại |
| RegistrationDate | Date | Ngày đăng ký |

## **3.4. Thiết kế giao diện người dùng**

### ***3.4.1 Người dùng***

Menu



Form đăng nhập

A screenshot of a computer

Description automatically generated

Form đăng kí

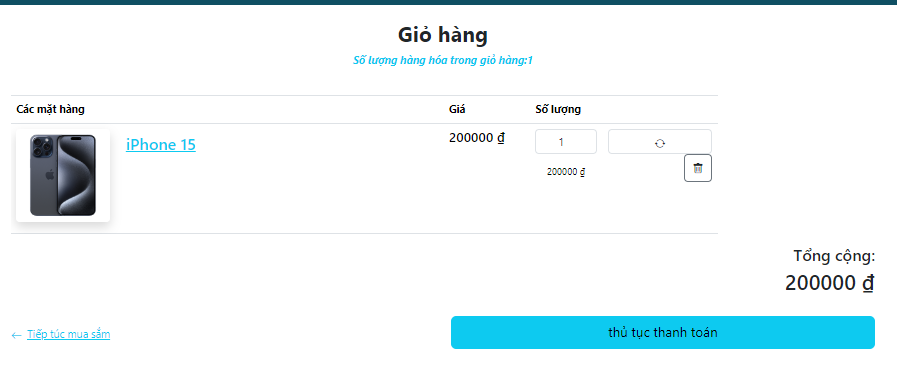


Form quản lý thông tin cá nhân

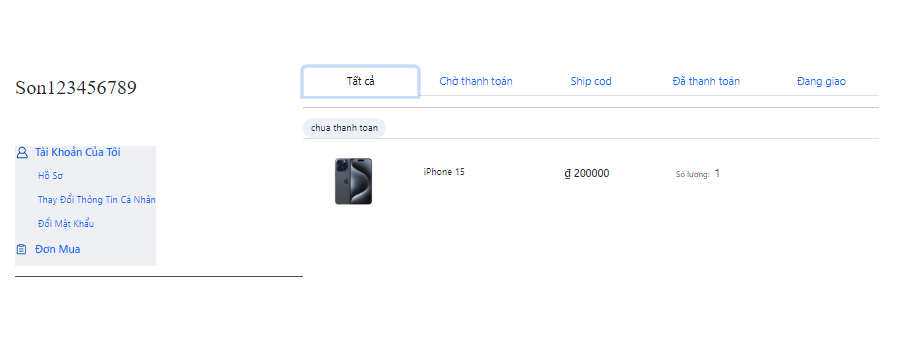
A screenshot of a computer

Description automatically generated

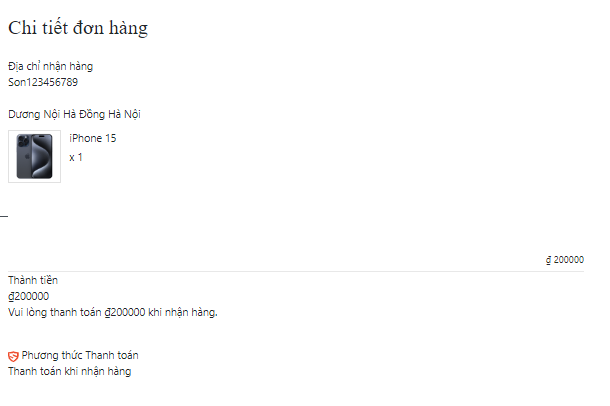
Form quản lý giỏ hàng



Form quản lý đơn hàng



Form quản lý chi tiết đơn hàng



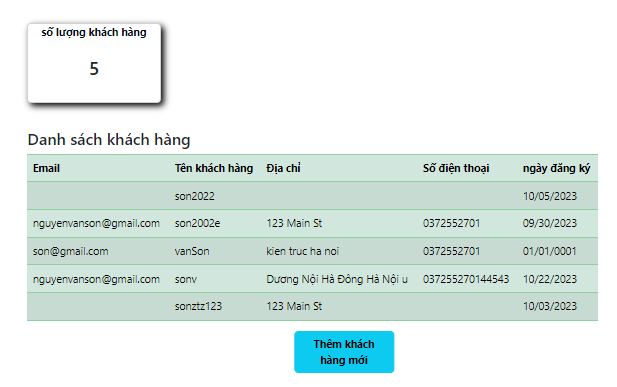
Form thanh toán

A screenshot of a computer

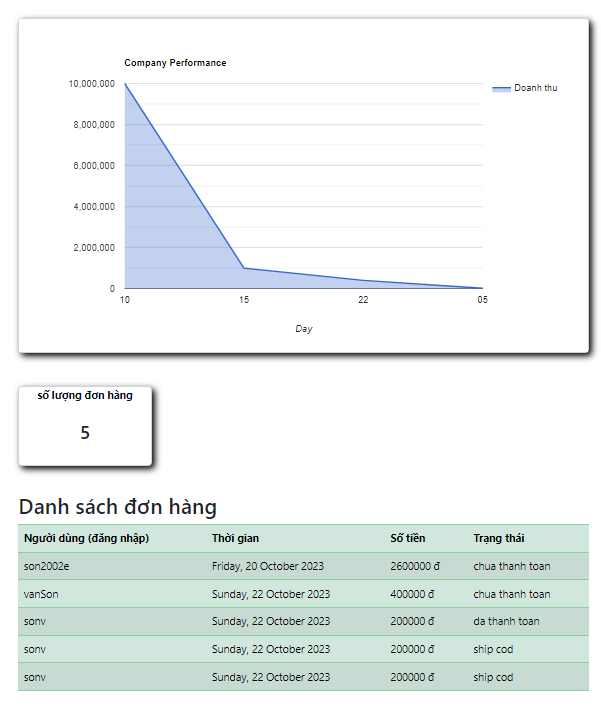
Description automatically generated

### ***3.4.2 Addmin:***

Form quản lý người dùng



Quản lý đơn hàng



Quản lý vai trò

A screenshot of a computer

Description automatically generated

Quản lý sản phẩm

A screenshot of a computer

Description automatically generated

# **CHƯƠNG IV: XÂY DỰNG ỨNG DỤNG**

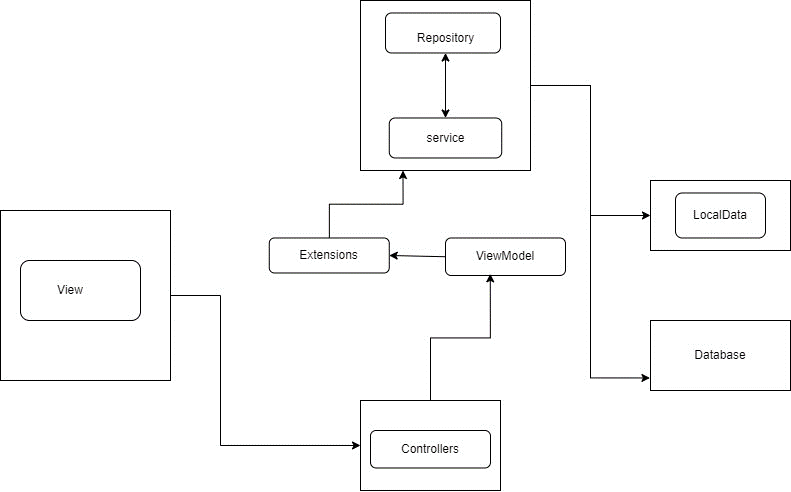
## **4.1. Cấu trúc dự án**

A screenshot of a computer

Description automatically generated

Dự án được xây dựng theo cấu trúc MVC nên các folder được mang tên của từng tầng xử lý

Mô hình dự án:



Mô tả mô hình:

Repository thực hiện những hàm lấy dữ liệu, thêm, sửa, xóa.

Service sẽ thực hiện những công việc sử lý dữ liệu theo yêu cầu cầu chức năng của hệ thống

Extensions sẽ thực hiện việc chuyển đổi dữ liệu để hiện thị ra view, quản lý những error

Controller sẽ đẩy dữ liệu, cũng như nhận dữ liệu từ view

## **4.2. Một số mã code xây dựng backend**

### 4.2.1 Cấu hình

-Dependency Inject

builder.Services.AddSingleton(configuration);

builder.Services.AddScoped<IPaymentService, PaymentService>();

builder.Services.AddScoped<ProductService>();

builder.Services.AddTransient<UserService>();

builder.Services.AddTransient<OrderItemService>();

builder.Services.AddTransient<OrderItemRepository, OderDbRepository>();

builder.Services.AddTransient<ProductRepository, ProductDbRepository>();

builder.Services.AddTransient<UserRepository, UserDbRepository>();

builder.Services.AddTransient<RoleManager<IdentityRole>>();

builder.Services.AddTransient<UserManager<ApplicationUser>>();

builder.Services.AddTransient<AppIdentityDbContextSeeder>();

AddSingleton:

Vòng đời: đối tượng này được sử dụng 1 lần duy nhất trong duy trì suốt vòng đời ứng dụng. Nó sẽ được tạo khi ứng dụng khởi tạo và sẽ tồn tại cho đến khi ứng dụng kết thúc

Sử dụng: thường được sủ dụng để đảm bảo cấu hình ứng dụng chỉ được đọc một lần và được chia sẻ trên toàn ứng dụng

AddScoped:

Vòng đời: Là một intance của class cụ thể sẽ được tạo cho mỗi yêu cầu http và sau khi yêu cầu kết thúc, instance này sẽ được kết thúc và giải phóng.

Sử dụng:

Thường được sủ dụng cho các dịch vụ hoặc lớp có thể chia se trong phạm vi của 1 yêu cầu web

AddTransient:

Vòng đời: Một instance mới của lớp tương ứng sẽ được tạo ra mỗi khi một đối tượng yêu cầu dịch vụ này (transient). Không có sự chia sẻ giữa các yêu cầu hoặc đối tượng, và instance sẽ bị giải phóng và thu hồi bộ nhớ sau khi nó không còn cần thiết.

Sử dụng: Thường được sử dụng cho các lớp hoặc dịch vụ có thể tạo ra và hủy bỏ một cách an toàn mà không cần quan tâm đến vòng đời của chúng.

-cookie:

static void ConfigureCookieSettings(IServiceCollection services)

{

services.Configure<CookiePolicyOptions>(options =>

{

options.CheckConsentNeeded = context => true;

options.MinimumSameSitePolicy = SameSiteMode.None;

});

services.ConfigureApplicationCookie(options =>

{

options.Cookie.HttpOnly = true;

options.ExpireTimeSpan = TimeSpan.FromHours(1);

options.LoginPath = "/Account/Login";

options.LogoutPath = "/Account/Logout";

options.Cookie = new CookieBuilder

{

IsEssential = true

};

});

}

-thiết lập dữ liệu ban đầu

var seedTask = AppIdentityDbContextSeeder.SeedAsync(userManager, roleManager);

seedTask.Wait();

-Cấu hình userManager và roleManager

var userManager = serviceProvider.GetRequiredService<UserManager<ApplicationUser>>();

var roleManager = serviceProvider.GetRequiredService<RoleManager<IdentityRole>>();

### 4.2.2. Repository

- UserRepository: thực hiện những thao tác với CSDL với những trường liên quan đến người dùng

public interface UserRepository

{

User TryGetByUserId(string id);

List<User> GetAll();

void AddInformation(User userInfo);

void DeleteOrder(int id,Product product,int quantity);

void CreateUser(User user);

}

- ProductRepository: thực hiện những thao tác với CSDL với những trường liên quan đến sản phẩm

public interface ProductRepository

{

List<Product> GetAll();

Product Get(int id);

void Create(Product product);

void Update (Product product);

void Delete(int id);

}

- OrderItemRepository: thực hiện những thao tác với CSDL với những trường liên quan trường đến Order và ItemOrder

public interface OrderItemRepository

{

Order GetOrder(int IdOrder);

List<Order> GetAll();

Order Create(string userId, Product product,int quantity);

void CreateItem(int Id, Product product);

void Update(OrderItem OrderItem);

void DeleteItem(int OrderItem);

void DeleteOder(int orderId);

void UpdateOrder(Order order, Product product, int quantity);

List<Order> TryGetByOrderUserId(string UserId);

List<Order> GetAllOrderPay();

List<OrderItem> getAllItem(int idItem);

Order getOrderPay(string userId);

void ChangeStatus(int id, int status);

}

### 4.2.3. Service

UserService:

public interface IUserService

{

void AddInformation(string userId, UserViewModel userViewModel);

List<UserViewModel> GetAll();

UserViewModel GetUser(string id);

User GetUserl(string id);

void createUser(string IdUser);

}

ProductService:

public interface IProductService

{

List<ProductViewModel> GetAllProducts();

void Update(ProductViewModel productViewModel);

ProductViewModel GetProduct(int id);

void UpdateProduct(int IdProduct, int quanty);

void Create(ProductViewModel productViewModel);

}

OrderItemService:

public interface IOrderItemService

{

void AddCart(ProductViewModel productViewModel, string userId);

List<OrderViewModel> GetCurrentAllOrder();

List<OrderViewModel> GetCurrentAllOrderUser(string userId);

OrderViewModel GetOrder(int IdOrder);

List<OrderViewModel> GetAllOrdernPay();

List<OrderItemViewModel> GetOrderItem(int OrderId);

void DeleteItem(string userId, int orderItem);

void DeleteOrder(int OrderId);

void ChangeStatus(int orderId, int status);

}

IpaymentService:

public interface IPaymentService

{

string PaymentCall(UserOrderViewModel userOrderView);

List< string> CheckPaymentStatus(string IdUser, string urlResponse);

}

### 4.2.4 Controllers

Đăng ký, đăng nhập

[HttpPost]

public IActionResult Register(RegisterViewModel model)

{

if (ModelState.IsValid)

{

var user = new ApplicationUser

{

UserName = model.nameLogin,

DateTime = DateTime.Now,

Name = model.nameUser,

Address = model.Address

};

var result = userManager.CreateAsync(user, model.Password).Result;

if (result.Succeeded)

{

signInManager.SignInAsync(user, false);

return RedirectToAction("Login");

}

else

{

result.AddErrorsTo(ModelState);

}

}

return View(model);

}

[HttpPost]

[ValidateAntiForgeryToken]

public IActionResult Login(LoginViewModel model)

{

if (ModelState.IsValid)

{

var result = signInManager.PasswordSignInAsync(model.NameLogin, model.Password, model.RememberMe, false).Result;

if (result.Succeeded)

{

if (!string.IsNullOrEmpty(model.ReturnUrl) && Url.IsLocalUrl(model.ReturnUrl))

{

return Redirect(model.ReturnUrl);

}

else

{

return RedirectToAction("Index", "Home");

}

}

ModelState.AddModelError("", "Tên đăng nhập hoặc mật khẩu không đúng");

}

else

{

ModelState.AddModelError("", "Các trường phải được điền đầy đủ");

}

return View(model);

}

-Dùng hàm đăng nhập đăng ký có sẵn của Identity famrwork là var result = userManager.CreateAsync(user, model.Password).Result; và var result = signInManager.PasswordSignInAsync(model.NameLogin, model.Password, model.RememberMe, false).Result;

AddmindController

public async Task<IActionResult> GetOrders()

{

var ListUserViewModelOrder = new List<UserOrderViewModel>();

var orders = OrderItemService.GetCurrentAllOrder();

for (int i = 0; i < orders.Count; i++)

{

var l = new UserOrderViewModel();

var user = await userManager.FindByIdAsync(orders[i].UserId);

if (user != null)

{

l.Order = orders[i];

l.User = new UserViewModel { Id = user.Id, EmailUser = user.Email, NameUser = user.UserName };

ListUserViewModelOrder.Add(l);

}

}

return View(ListUserViewModelOrder);

}

Hiển thị thông tin tất cả những order của người dùng

public IActionResult GetUsers() => View(userManager.Users.ToList());

lấy những người dùng từ userManager

CartController

Hiển thị itemOrder

public IActionResult Index()

{

var orderViewModel = OrderItemService.GetOrderPay(userManager.GetUserId(User));

return View(orderViewModel);

}

Lấy Id người dùng hiện tại và lấy thông tin những itemOrder và hiển thị ra view

Thêm item vào order của mình:

public IActionResult Add(int id)

{

var IdUser = userManager.GetUserId(User);

var product = productService.GetProduct(id);

var user = userService.GetUserl(IdUser);

if (user == null)

{

userService.createUser(IdUser);

}

OrderItemService.AddCart(product, IdUser);

return RedirectToAction("Index");

}

Lấy id người dùng hiện tại và tạo 1 item từ sản phẩm minh thêm

CheckOutController

Thực hiện những thao tác liên quan đến việc thanh toán

[HttpPost]

[Route("api/v1/payment")]

public async Task<IActionResult> Payment(string status)

{

var user = userService.GetUser(userManager.GetUserId(User));

var userOrderView = new UserOrderViewModel();

ProductViewModel product;

userOrderView.Order = OrderItemService.GetOrderPay(userManager.GetUserId(User));

userOrderView.User = user;

if (status == "1")

{

if (userOrderView != null)

{

var respone = paymentService.PaymentCall(userOrderView);

if (!string.IsNullOrEmpty(respone))

{

return Redirect(respone);

}

}

return NotFound();

}

if(status == "2")

{

OrderItemService.ChangeStatus(userOrderView.Order.Id,2);

foreach(var itemOrder in userOrderView.Order.ItemViewModels)

{

productService.UpdateProduct(itemOrder.Product.Id, itemOrder.quantity);

}

}

return RedirectToAction("Index", "Cart");

}

Hàm Payment() thực hiện việc gọi API của ngân hàng để phục vụ việc thanh toán cụ thể là VNPay

[HttpGet]

public ActionResult ResponseVnp()

{

string queryString = Request.QueryString.ToString();

var response = paymentService.CheckPaymentStatus(userManager.GetUserId(User), queryString);

return View(response);

}

Hàm ResponseVnp() thực hiện lấy kết quả trả về từ ngân hàng và hiển thị ra view

HomeController

Để hiển thị sản phẩm ra trang home

public IActionResult Index(int page)

{

var products = productService.GetAllProducts();

var pager = CreatePagination(products, page);

return View(pager);

}

ProductController

Hiển thị thông tin chi tiết của từng sản phẩm

public IActionResult Index(int id)

{

var product = productService.GetProduct(id);

return View(product);

}

UserProfileController

Thực hiện những chức năng của người dùng

public async Task<IActionResult> Index()

{

var user = await userManager.FindByIdAsync(userManager.GetUserId(User));

var model = new UserViewModel { Id = user.Id, EmailUser = user.Email, NameUser = user.Name, PhoneUser = user.PhoneNumber,RegistrationDate = user.DateTime,AddressUser = user.Address };

return View(model);

}

Lấy id người dùng hiện tại để tìm kiếm và hiển thị thông tin người dùng hiện tại

Thay đổi thông tin người dùng

[HttpPost]

public async Task<IActionResult> Edit(UserViewModel model)

{

if (model != null)

{

var user = await userManager.FindByIdAsync(model.Id);

if (user != null)

{

user.Email = model.EmailUser;

user.Name = model.NameUser;

user.PhoneNumber = model.PhoneUser;

user.Address = model.AddressUser;

var result = await userManager.UpdateAsync(user);

if (result.Succeeded)

{

return RedirectToAction("Index");

}

else

{

result.AddErrorsTo(ModelState);

}

}

}

return View(model);

}

Lấy thông tin người dùng hiện tại dựa vào Id và dùng var result = await userManager.UpdateAsync(user); của Identity để cập nhật thông tin người dùng

Thay đổi mật khẩu

[HttpPost]

public async Task<IActionResult> ChangePassword(UserViewModel model)

{

if (ModelState.IsValid)

{

var user = await userManager.FindByIdAsync(model.Id);

if (user != null)

{

var result = await userManager.ChangePasswordAsync(user, model.Password, model.NewPassword);

if (result.Succeeded)

{

return RedirectToAction("Index");

}

else

{

result.AddErrorsTo(ModelState);

}

}

else

{

ModelState.AddModelError(string.Empty, "vui lòng nhập thông tin");

}

}

return View(model);

}

Lấy thông tin người dùng hiện tại dựa vào Id và dùng var result = await userManager.ChangePasswordAsync(user, model.Password, model.NewPassword); của Identity để cập nhật mật khẩu người dùng

## **4.3 Một số mã code xây dựng Fontend**

Form login

<body class="text-center">

<div class="album py-5">

<form method="post" asp-controller="Account" asp-action="Login" asp-route-returnUrl="@Model.ReturnUrl">

<div asp-validation-summary="ModelOnly" class="text-danger"></div>

<h1 class="h3 mb-3 font-weight-normal">ĐĂNG NHẬP</h1>

<div class="container col-md-4 align-self-center">

<label asp-for="NameLogin" class="sr-only">Tên tài khoản</label>

<input asp-for="NameLogin" class="form-control" placeholder="Tên tài khoản" />

<span asp-validation-for="NameLogin" class="sr-only"></span>

<div class="album py-1"></div>

<label asp-for="Password" class="sr-only">Mật khẩu</label>

<input asp-for="Password" class="form-control" placeholder="Mật khẩu" />

<span asp-validation-for="Password" class="sr-only"></span>

<div class="album py-1"></div>

</div>

<div class="checkbox mb-3">

<label asp-for="RememberMe">

<input asp-for="RememberMe" type="checkbox" /> có nhớ mật khẩu?

</label>

</div>

<button class="btn btn-dark col-md-3" type="submit">Đăng Nhập</button>

<div class="album py-1"></div>

<a style="color:black" asp-action="Register">Đăng ký Người dùng mới</a>

</form>

</div>

</body>

Form Register

<body class="text-center">

<div class="album py-5">

<form method="post" asp-controller="Account" asp-action="Register">

<div asp-validation-summary="All" class="text-danger"></div>

<h1 class="h3 mb-3 font-weight-normal">Đăng ký</h1>

<div class="container col-md-4 align-self-center">

<label asp-for="nameLogin" class="sr-only"></label>

<input asp-for="nameLogin" class="form-control" placeholder="Đăng nhập" />

<span asp-validation-for="nameLogin" class="sr-only"></span>

<div class="album py-1"></div>

<label asp-for="nameUser" class="sr-only"></label>

<input asp-for="nameUser" class="form-control" placeholder="tên khách hàng" />

<span asp-validation-for="nameUser" class="sr-only"></span>

<div class="album py-1"></div>

<label asp-for="Address" class="sr-only"></label>

<input asp-for="Address" class="form-control" placeholder="địa chỉ" />

<span asp-validation-for="Address" class="sr-only"></span>

<div class="album py-1"></div>

<label asp-for="Password" class="sr-only"></label>

<input asp-for="Password" class="form-control" placeholder="Mật khẩu" />

<span asp-validation-for="Password" class="sr-only"></span>

<div class="album py-1"></div>

<label asp-for="PasswordConfirm" class="sr-only"></label>

<input asp-for="PasswordConfirm" class="form-control" placeholder="Xác nhận mật khẩu" />

<span asp-validation-for="PasswordConfirm" class="sr-only"></span>

<div class="album py-3"></div>

</div>

<button class="btn btn-dark col-md-3" type="submit">Đăng ký</button>

</form>

</div>

</body>

Form thay đổi thông tin người dùng

<div class="col-sm-6 col-md-8">

<div class="breadcrumb">

<a>Thay đổi thông tin cá nhân</a>

</div>

<div class="container text-center">

<div class="row g-2">

<form asp-action="Edit" asp-controller="UserProfile">

<div asp-validation-summary="All" class="text-danger offset-4 col-8"></div>

<div class="form-group row">

<label asp-for="NameUser" class="col-4 col-form-label">tên khách hàng</label>

<div class="col-8">

<input asp-for="NameUser" placeholder="Tên khách hàng" class="form-control here" type="text">

<span asp-validation-for="NameUser" class="sr-only"></span>

</div>

</div>

<div class="form-group row">

<label asp-for="PhoneUser" class="col-4 col-form-label">số điện thoại</label>

<div class="col-8">

<input asp-for="PhoneUser" placeholder="Số điện thoại" class="form-control here" type="number">

<span class="sr-only"></span>

</div>

</div>

<div class="form-group row">

<label asp-for="EmailUser" class="col-4 col-form-label">Email </label>

<div class="col-8">

<input asp-for="EmailUser" placeholder="Email khách hàng" class="form-control here"

type="email" required>

<span asp-validation-for="EmailUser" class="sr-only"></span>

</div>

</div>

<div class="form-group row">

<label asp-for="AddressUser" class="col-4 col-form-label">Địa chỉ</label>

<div class="col-8">

<input asp-for="AddressUser" placeholder=" địa chỉ " class="form-control here"

type="tel">

<span class="sr-only"></span>

</div>

</div>

<div class="row">

<div class="d-flex col-md-12">

<div class="ml-auto">

<button type="submit" class="btn btn-info">xác nhận</button>

</div>

</div>

</div>

</form>

</div>

</div>

Form lấy xem order của người dùng

<div id="partialContainer">

@foreach (var order in Model.Orders)

{

<hr>

<div class="shop shop-cursor" id="o-list">

<div class="shop-header border-bottom">

<div class="shop-right">

<div class="shop-right-status">@order.NameStatus</div>

<div class="shop-right-action"></div>

</div>

</div>

@foreach (var item in order.ItemViewModels)

{

<a asp-action="GetItemOrder" asp-route-id="@order.Id" style=" text-decoration: none;">

<div class="card border-0">

<img src="@item.Product.ImageUrl" style="width: 80px; height: 80px;" />

<div class="mask" style="background-color: rgba(251, 251, 251, 0.15);"></div>

<div class="card-body" style="display:flex;">

<div class="item-main item-main-mini">

<div>

<div class="text title item-title" data-spm="details\_title">

@item.Product.Name

</div>

</div>

</div>

<div class="item-status">

<div class="item-price text bold">₫ @item.Product.Price</div>

</div>

<div class="item-quantity" style="margin-left:100px">

<span>

<span class="text desc info multiply">

Số

lượng:

</span><span class="text">&nbsp;@item.quantity</span>

</span>

</div>

</div>

</div>

</a>

}

</div>

}

</div>

</div>

</div>

<script>

document.querySelectorAll(".nav-link").forEach(function (tab) {

tab.addEventListener("click", function (e) {

e.preventDefault();

document.querySelectorAll(".nav-link").forEach(function (otherTab) {

otherTab.classList.remove("active");

});

tab.classList.add("active");

var status = tab.getAttribute("data-status");

fetch("/UserProfile/fragment?status=" + status)

.then(response => response.text())

.then(data => {

const fragment = document.getElementById("partialContainer");

fragment.innerHTML = data;

})

.catch(error => console.error(error));

});

});

</script>

Sử dụng AJAX để load lại những phần thay đổi không cần phải load lại cả trang

## **4.4. Kiểm thử**

### 4.4.1. Chức năng đăng nhập

-Kịch bản kiểm thử: Người dùng nhập các thông tin đăng nhập, nếu thông tin sai thì đưa ra cảnh báo, còn thông tin đăng nhập chính xác sẽ vào bên trong.

-Kết quả kiểm thử: Thành công, hệ thống cho phép người dùng truy cập các chức năng theo vai trò của mình

### 4.4.2. Chức năng quản lý thông tin cá nhân

-Kịch bản kiểm thử: Người dùng click vào thông tin cá nhân để thay đổi, kiểm tra các dữ liệu, nếu đạt yêu cầu thì lưu vào csdl nếu sai thì hiện thông báo cho người dùng biết.

-Kết quả kiểm thử: Thành công, hệ thống cho phép người dùng thay đổi thông tin cá nhân.

4.4.3. Chức năng thêm vào giỏ hàng

-Kịch bản kiểm thử: Người dùng click vào thêm vào giỏ hàng của sản phẩm nào thì thông tin sản phẩm đó được thêm vào giỏ hàng

-Kết quả kiểm thử: Thành công.

4.4.4. Chức năng quản lý sản phẩm

-Kịch bản kiểm thử: Người dùng có tài khoản được cấp quyền admin mới có thể truy cập các chức năng xem, tìm kiếm, thêm, sửa, xóa sản phẩm. Những tài khoản không được cấp quyền sẽ hiện thông báo không có quyền truy cập chức năng.

-Kết quả kiểm thử: Thành công, hệ thống cho phép những tài khoản người dùng được cấp quyền truy cập vào các chức năng.

4.4.5. Chức năng quản lý đơn hàng

-Kịch bản kiểm thử: Người dùng có tài khoản được cấp quyền admin mới có thể truy cập các chức năng xem, tìm kiếm, thêm, sửa, xóa đơn hàng. Những tài khoản không được cấp quyền sẽ hiện thông báo không có quyền truy cập chức năng.

-Kết quả kiểm thử: Thành công, hệ thống cho phép những tài khoản người dùng được cấp quyền truy cập vào các chức năng

4.4.6. Chức năng quản lý người dùng

-Kịch bản kiểm thử: Người dùng có tài khoản được cấp quyền admin mới có thể truy cập các chức năng xem, tìm kiếm, thêm, sửa, xóa người dùng. Những tài khoản không được cấp quyền sẽ hiện thông báo không có quyền truy cập chức năng.

-Kết quả kiểm thử: Thành công, hệ thống cho phép những tài khoản người dùng được cấp quyền truy cập vào các chức năng.

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

**5.1 Kết luận:**

* Nhờ quá trình làm đồ án em đã học được một vài kiến thức và kĩ năng cần thiết đáp ứng cho quá trình làm việc sau này. Một số nội dung em tích lũy qua quá trình thực hiện đồ án như:
* Nắm được quy trình và hướng phát triển để xây dựng một dự án;
* Các bước để thiết kế website, từ thiết kế hệ thống, thiết kế đối tượng, thiết kế dữ liệu cho đến thiết kết giao diện;
* Tạo ra giao diện thân thiện với người dùng
* Xây dựng được hệ thống quản lý bán hàng
* Xây dựng được các chức năng thêm vào giỏ hàng, đặt hàng trực tuyến

**5.2 Ưu và nhược điểm**

**5.2.1 Ưu điểm**

* Trang web đơn giản, người dùng dễ sử dụng
* Tốc độ load nhanh, ít giật lag

**5.2.2 Nhược điểm**

* Web còn sơ sài, ít chức năng
* Còn tồn tại nhiều lỗi tiềm ẩn
* Vấn đề bảo mật chưa được chú trọng

**5.3 Các chức năng đạt được**

* Đăng nhập, đăng xuất
* Quản lý thông tin cá nhân
* Thêm vào giỏ hàng, thanh toán
* Quản lý sản phẩm
* Quản lý đơn hàng
* Quản lý người dùng

**5.4 Hướng phát triển**

* Phát triển thêm chức năng thanh toán bẳng thẻ ATM, thẻ ngân hàng, ví điện tử, internet banking,…
* Bổ sung các chức năng cần thiết
* Tăng cường bảo mật

**TÀI LIỆU THAM KHẢO**

[1] Andrew, T. & Philip J.(2015), *C# 6.0 and the .NET 4.6 Framework*

[2] Christian, N. (2014), *Professional C# 5.0 and .NET 4.5.12014*

[3] Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom. (2013), *Database Systems: The Complete Book*

[4] Jon Duckett. (2011), *HTML and CSS: Design and Build Websites*

[5] Andrew Lock. (2018), *ASP.NET Core in Action*

[6] RB Whitaker. The C# Player’s Guide (2nd Edition). (2012),*Starbound Software. U.S.A.*

[7] Adam FreeMan. (2013), *Professional ASP.NET MVC 5 (1st Edition). U.S.A*

[8] Nguyễn Văn Đạo, Nguyễn Thanh Sơn. (2016), *Lập trình website với ASP.NET MVC 5*

[9] Trần Duy Thanh. (2016), *Cơ sở dữ liệu SQL Server*

[10] Nguyễn Mạnh Hùng, Trương Anh Dũng. (2016), *Lập trình web ASP.NET MVC 5 cơ bản và nâng cao*

[11] Dương Quang Thiện. (2005), .*NET Toàn Tập*

[12] Nguyễn Ngọc Bình Phương, Thái Thanh Phong. (2006), *Các Giải Pháp Lập Trình C#.*