

**TRƯỜNG ĐẠI HỌC YERSIN ĐÀ LẠT**  
**KHOA CÔNG NGHỆ THÔNG TIN**



**YERSIN UNIVERSITY**

**BÁO CÁO MÔN HỌC**  
**LẬP TRÌNH WEB 1**

**Xây dựng website thương mại điện tử kinh doanh sản phẩm Tài liệu  
học tập + báo cáo + đồ án**

GVHD : Nguyễn Đức Tấn

SVTH : Trần Anh Khoa

Mã số SV : 2301010024

Khóa học : 2024 - 2025

Đà Lạt, tháng 6- 2025

## PHẦN NHẬN XÉT CỦA GIẢNG VIÊN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Điểm: .....

Ngày ... tháng ... năm...

*Ký và ghi rõ họ tên*

# Mục Lục

<b>CHƯƠNG 1 :TÌM HIỂU VỀ LẬP TRÌNH WEB MVC (CƠ SỞ LÝ THUYẾT).....</b>	<b>6</b>
<b>1. Tổng quan về ASP.NET Core:.....</b>	<b>6</b>
1.1 Giới thiệu về ASP.NET Core: .....	6
1.2 Tìm hiểu về mô hình lập trình web MVC của ASP.NET Core: .....	6
1.3 Nguyên lý hoạt động ASP.NET MVC CORE: .....	8
1.3.1 Khái niệm: .....	8
1.3.2 Ví dụ minh họa .....	9
1.4 Đặc Điểm: .....	11
1.4.1 Tại sao phải sử dụng ASP.NET Core? : .....	11
1.5 Công nghệ triển khai ASP.NET Core: .....	14
1.5.1 Môi trường triển khai:.....	14
1.5.2 Các hình thức triển khai: .....	15
1.5.3 Các nền tảng đám mây hỗ trợ: .....	15
1.5.4 Công cụ hỗ trợ triển khai:.....	16
<b>CHƯƠNG 2 :XÂY DỰNG ỨNG DỤNG THỰC TẾ .....</b>	<b>16</b>
2.1 Phát biểu bài toán ứng dụng:.....	16
2.1.1 Giới thiệu dự án: .....	16
2.1.2 mục tiêu của dự án .....	17
2.1.3 Ý nghĩa thực tiễn .....	17
2.2 Phân Tích yêu cầu của ứng dụng: .....	19
2.2.1 Mô tả chức năng ứng dụng: .....	19
2.2.2 Mô tả yêu cầu phi chức năng của ứng dụng:.....	22
2.2.3 Sơ đồ UseCase: .....	25
2.3 Thiết kế cơ sở dữ liệu: .....	28
2.3.1 Thiết kế logic ERD:.....	28
2.3.2 Mô hình vật lý.....	29
2.4 Thiết kế giao diện:.....	30
2.4.1 Trang chủ.....	30
2.4.2 Trang Shop .....	31
2.4.3 Trang liên hệ.....	32
2.4.4 Trang giỏ hàng .....	33

2.4.5 Trang đăng nhập.....	34
2.5 Thiết kế các thành phần MVC: .....	34
2.5.1 Model.....	34
2.5.2 View .....	37
2.5.3 Controller.....	39
2.6 Triển khai cài đặt.....	42
2.6.1 Cấu hình máy phát triển.....	42
2.6.2 Môi trường phát triển.....	42
2.6.3 Cài đặt.....	43
CHƯƠNG 3 KẾT QUẢ CHƯƠNG TRÌNH .....	44
3.1 Tranh chủ .....	44
3.2 Trang Shop .....	45
3.3 Trang liên hệ.....	46
3.4 Trang Giỏ hàng .....	47
3.5 Trang Login.....	47
CHƯƠNG 4 : Kết Luận.....	48
4.1 Tổng kết kiến thức đạt được .....	48
4.2 Điểm tồn tại .....	48
4.3 Hướng phát triển dự án.....	49
4.4 Tài liệu tham khảo:.....	49

## LỜI NÓI ĐẦU

Trong bối cảnh kỷ nguyên số bùng nổ, công nghệ thông tin đã trở thành mạch nguồn định hình mọi khía cạnh của đời sống và kinh doanh. Đặc biệt, sự phát triển vượt bậc của Trí tuệ nhân tạo (AI) và các nền tảng kỹ thuật số đã tạo ra một làn sóng mạnh mẽ, thúc đẩy thương mại điện tử vươn lên tầm cao mới, đồng thời mở ra cả cơ hội lẫn thách thức cho doanh nghiệp và người dùng cá nhân. Giờ đây, các ứng dụng web quản lý và bán hàng trực tuyến không chỉ là công cụ hỗ trợ mà đã trở thành yếu tố chiến lược then chốt, quyết định khả năng cạnh tranh và mở rộng thị trường của bất kỳ tổ chức nào.

Trước những biến chuyển sâu sắc đó, việc nghiên cứu, phát triển và ứng dụng các giải pháp công nghệ mới để đáp ứng kịp thời nhu cầu thị trường là vô cùng cấp thiết. Đồ án này là thành quả của quá trình tìm hiểu, nghiên cứu và triển khai một ứng dụng web cụ thể, nhằm góp phần vào xu thế chung của ngành công nghệ thông tin. Báo cáo sẽ trình bày chi tiết về toàn bộ hành trình xây dựng một ứng dụng web hoàn chỉnh, từ khâu phân tích yêu cầu, lựa chọn công nghệ, thiết kế kiến trúc, đến triển khai và đánh giá kết quả thực tế. Qua đó, em mong muốn minh chứng khả năng chuyển hóa lý thuyết thành giải pháp thực tiễn, giải quyết một bài toán cụ thể trong lĩnh vực thương mại điện tử.

Cuối cùng, em xin gửi lời tri ân sâu sắc đến thầy Nguyễn Đức Tấn, người đã luôn tận tình hướng dẫn và hỗ trợ em trong suốt quá trình thực hiện đồ án. Những kiến thức chuyên môn sâu rộng, kinh nghiệm quý báu cùng với sự định hướng kịp thời của thầy chính là kim chỉ nam giúp em hoàn thành dự án này.

# CHƯƠNG 1 :TÌM HIỂU VỀ LẬP TRÌNH WEB MVC (CƠ SỞ LÝ THUYẾT)

## 1. Tổng quan về ASP.NET Core:

### 1.1 Giới thiệu về ASP.NET Core:

Đầu năm 2002, Microsoft giới thiệu một kỹ thuật lập trình Web khá mới mẻ với tên gọi ban đầu là ASP+, sau này chính thức là ASP.NET. Với ASP.NET, bạn không cần phải biết các thẻ HTML hay thiết kế web mà vẫn được hỗ trợ mạnh mẽ lập trình hướng đối tượng trong quá trình xây dựng và phát triển ứng dụng Web. ASP.NET là kỹ thuật lập trình và phát triển ứng dụng web ở phía máy chủ (Server-side) dựa trên nền tảng của Microsoft .NET Framework.

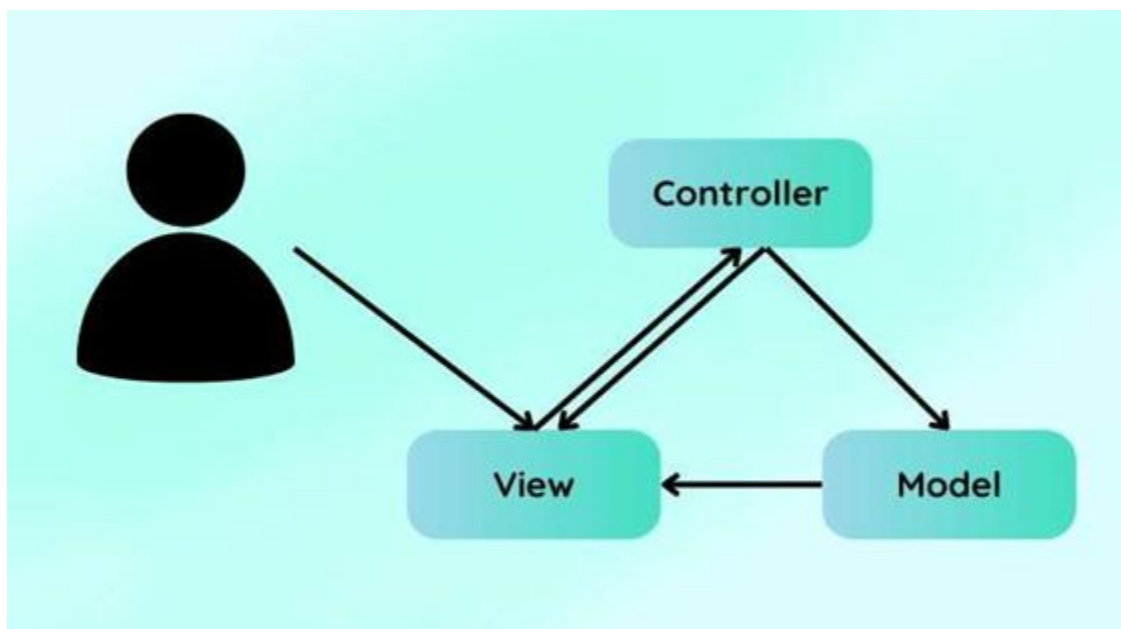
Hầu hết những người mới làm quen với lập trình web đều bắt đầu với các kỹ thuật phía máy khách (Client-side) như HTML, JavaScript, CSS (Cascading Style Sheets). Khi trình duyệt web yêu cầu một trang web (sử dụng kỹ thuật Client-side), máy chủ web sẽ tìm trang đó và gửi về cho máy khách. Máy khách nhận kết quả từ máy chủ và hiển thị lên màn hình. ASP.NET Core sử dụng kỹ thuật lập trình phía máy chủ hoàn toàn khác. Mã lệnh ở phía máy chủ (ví dụ: mã lệnh trong trang ASP.NET) sẽ được biên dịch và thực thi tại Web Server. Sau khi được Server đọc, biên dịch và thực thi, kết quả sẽ tự động được chuyển sang HTML/JavaScript/CSS và trả về cho Client. Tất cả các xử lý lệnh ASP.NET Core đều được thực hiện tại Server, do đó được gọi là kỹ thuật lập trình phía máy chủ.

### 1.2 Tìm hiểu về mô hình lập trình web MVC của ASP.NET Core:

Mô hình MVC (viết tắt của Model - View - Controller) là một kiến trúc phần mềm hay mô hình thiết kế được sử dụng trong kỹ thuật phần mềm (đặc biệt đối với phát triển ứng dụng web). Nó giúp tổ chức ứng dụng (phân chia mã nguồn) thành ba phần khác nhau: Model, View và Controller. Mỗi thành phần có một nhiệm vụ riêng biệt và độc lập với các thành phần khác.

**Model:** Chứa tất cả các nghiệp vụ logic, phương thức xử lý, truy xuất cơ sở dữ liệu (CSDL), đối tượng mô tả dữ liệu như các Class, hàm xử lý... Model có nhiệm vụ cung cấp dữ liệu và lưu dữ liệu vào các kho chứa. Tất cả các nghiệp vụ logic được thực thi ở Model. Dữ liệu từ người dùng sẽ thông qua View để kiểm tra ở Model trước khi lưu vào cơ sở dữ liệu. Việc truy xuất, xác nhận và lưu dữ liệu là một phần của Model. **View:** Hiển thị thông tin cho người dùng của ứng dụng và có nhiệm vụ nhận dữ liệu đầu vào từ người dùng, gửi yêu cầu người dùng đến bộ điều khiển (Controller), sau đó nhận phản hồi từ bộ điều khiển và hiển thị kết quả cho người dùng. Các trang HTML, các công cụ tạo giao diện (như Razor Pages trong ASP.NET Core) và các tệp nguồn liên quan là một phần của View.

**Controller:** Là tầng trung gian giữa Model và View. Controller có nhiệm vụ nhận các yêu cầu từ người dùng (phía máy khách). Một yêu cầu được nhận từ máy khách sẽ được xử lý bởi một chức năng logic thích hợp từ thành phần Model và sau đó sinh ra kết quả cho người dùng, được thành phần View hiển thị.



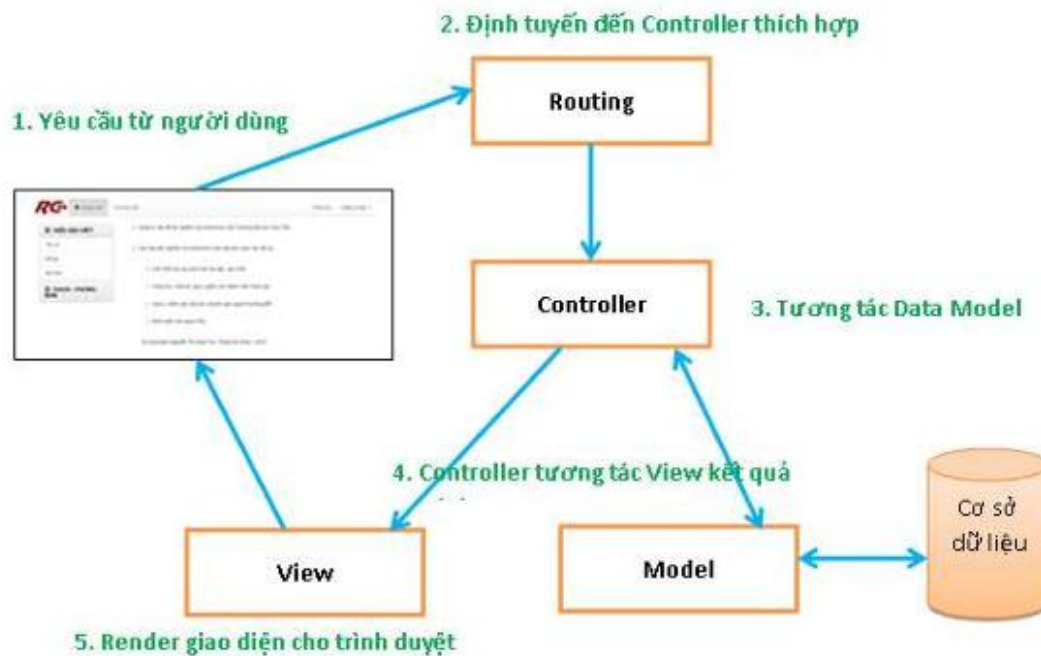
Hình 1.1: Mô hình MVC

### 1.3 Nguyên lý hoạt động ASP.NET MVC CORE:

#### 1.3.1 Khái niệm:

- Yêu cầu từ trình duyệt đến Web Server: Khi một yêu cầu (request) được gửi từ trình duyệt web đến Web Server (ví dụ: Kestrel, IIS, Nginx), yêu cầu đó sẽ được chuyển đến Middleware Pipeline của ASP.NET Core. Trong pipeline này, cơ chế Routing Middleware sẽ đóng vai trò quan trọng trong việc xác định lộ trình phù hợp.
- Routing và chọn Controller: Routing Middleware có nhiệm vụ phân tích URL của yêu cầu và quyết định yêu cầu này sẽ được xử lý bởi Controller và Action Method cụ thể nào. Dựa vào cấu hình định tuyến (routes) đã được thiết lập, hệ thống sẽ ánh xạ URL đến đúng Controller và Action. Sau khi xác định được, một instance của Controller sẽ được tạo ra.
- Thực thi Action Method: Controller sau đó sẽ xác định Action Method cụ thể để xử lý yêu cầu và tiến hành thực thi Action Method đó. Trong quá trình này, Action Method có thể tương tác với các Model Class để truy xuất dữ liệu từ cơ sở dữ liệu hoặc thực thi các logic nghiệp vụ cần thiết.
- Trả về Action Result: Sau khi hoàn tất xử lý, Action Method sẽ trả về một Action Result. ASP.NET Core MVC cung cấp nhiều loại Action Result khác nhau (ví dụ: ViewResult, JsonResult, ContentResult, RedirectResult, v.v.). Trong đó, ViewResult là một loại Action Result đặc biệt. ViewResult có nhiệm vụ làm việc với một View cụ thể để tạo ra mã HTML.
- Hiển thị View với View Engine: View Engine (ví dụ: Razor View Engine) là thành phần thực hiện việc hiển thị một View. View Engine sẽ nhận dữ liệu từ Model (nếu có) và sử dụng cú pháp của mình để kết xuất thành mã HTML. Mã HTML này sau đó sẽ được trả về cho trình duyệt web và hiển thị cho người dùng, hoàn tất quá trình xử lý yêu cầu.





Hình 1.2: Mô tả Nguyên Lý hoạt động MVC

### 1.3.2 Ví dụ minh họa

#### Model:

```

namespace Atsumaru.Models
{
    public class Product { public int Id { get; set; }
    {
        public string Name { get; set; }
        public decimal Price { get; set; }
        public string Description { get; set; }
    }
}

```

#### Controller:

```

namespace Atsumaru.Controllers
{
    private static List<Product> _products = new List<Product>
    {

```

```

new Product { Id = 1, Name = "Truyện One Piece Tập 1", Price = 30.000M,
Description = "Khởi đầu hành trình của Luffy." },
new Product { Id = 2, Name = "Mô hình Naruto Sage Mode", Price =
500.000M, Description = "Mô hình cao cấp." },
new Product { Id = 3, Name = "Light Novel Sword Art Online Tập 1", Price
= 95.000M, Description = "Thế giới ảo đầy kịch tính." }
};
public IActionResult Index()
{
    return View(_products);
}

```

**View:**

```

@model List<Atsumaru.Models.Product>
<h1>Danh sách sản phẩm Anime</h1>
<table class="table">
    <thead>
        <tr>
            <th>Tên sản phẩm</th>
            <th>Giá</th>
            <th>Mô tả</th> <th></th>
        </tr>
    </thead>
    <tbody>
        @foreach (var item in Model)
        {
            <tr>
                <td>@Html.DisplayFor(modelItem=>item.Name)</td>
                <td>@Html.DisplayFor(modelItem=>tem.Price)VNĐ</td>
                <td>@Html.DisplayFor(modelItem => item.Description)</td>
            </tr>

```

```
}  
</tbody>  
</table>
```

## 1.4 Đặc Điểm:

### 1.4.1 Tại sao phải sử dụng ASP.NET Core? :

Yêu cầu về xây dựng các ứng dụng thương mại điện tử ngày càng phát triển và nâng cao. Khi đó, các công nghệ cũ như ASP không còn đáp ứng được yêu cầu đặt ra. ASP được thiết kế riêng biệt và nằm ở tầng phía trên hệ điều hành Windows và Internet Information Service, do đó các công dụng của nó hết sức rời rạc và giới hạn. ASP.NET Core ra đời mang đến một phương pháp phát triển hoàn toàn mới, khác hẳn so với ASP trước kia và đáp ứng được các yêu cầu khắt khe của ứng dụng hiện đại. Ưu điểm và khuyết điểm của ASP.NET.

#### Ưu điểm:

- Hỗ trợ đa ngôn ngữ lập trình: Trong khi ASP chỉ sử dụng VBScript và JavaScript, ASP.NET Core cho phép bạn viết code bằng nhiều ngôn ngữ mạnh mẽ như C#, F#, và Visual Basic .NET.
- Phong cách lập trình Code-behind hiện đại: ASP.NET Core thúc đẩy phong cách lập trình Code-behind, giúp tách biệt rõ ràng giữa logic nghiệp vụ (code) và giao diện người dùng (markup). Điều này giúp mã nguồn dễ đọc, dễ quản lý và bảo trì hơn rất nhiều.
- Hỗ trợ kiểm tra dữ liệu đầu vào mạnh mẽ: Thay vì phải viết mã thủ công để kiểm tra dữ liệu nhập từ người dùng, ASP.NET Core hỗ trợ các Validation Controls (trong ASP.NET truyền thống) và Data Annotations (trong ASP.NET Core) giúp việc kiểm tra dữ liệu trở nên đơn giản và hiệu quả, giảm thiểu lượng code phải viết.

- Phát triển đa nền tảng và thiết bị: ASP.NET Core hỗ trợ phát triển các ứng dụng web có thể truy cập trên nhiều thiết bị khác nhau, từ máy tính để bàn đến các thiết bị di động như PocketPC, Smartphone, v.v., nhờ vào khả năng đa nền tảng của nó.
- Hỗ trợ Web Server Controls (trong ASP.NET truyền thống) và Tag Helpers (trong ASP.NET Core): Cung cấp nhiều control sẵn có giúp tăng tốc độ phát triển giao diện.
- Cá nhân hóa giao diện: Cho phép người dùng thiết lập giao diện trang web theo sở thích cá nhân sử dụng các tính năng như Theme, Profile, và WebPart (trong ASP.NET truyền thống) hoặc các cơ chế tùy chỉnh giao diện linh hoạt trong ASP.NET Core.
- Tăng cường tính năng bảo mật: ASP.NET Core được thiết kế với các tính năng bảo mật mạnh mẽ, giúp bảo vệ ứng dụng khỏi các lỗ hổng phổ biến.
- Hỗ trợ kỹ thuật truy cập dữ liệu LINQ: Tích hợp mạnh mẽ với LINQ (Language Integrated Query), giúp truy vấn dữ liệu một cách trực quan và hiệu quả.
- Tận dụng .NET Framework (nay là .NET): ASP.NET Core kế thừa và tận dụng mạnh mẽ bộ thư viện phong phú và đa dạng của nền tảng .NET, hỗ trợ làm việc với XML, Web Service, truy cập cơ sở dữ liệu qua ADO.NET (hoặc Entity Framework Core), v.v.
- Kiến trúc lập trình quen thuộc: Kiến trúc lập trình của ASP.NET Core có nhiều điểm tương đồng với việc phát triển ứng dụng trên Windows, giúp lập trình viên dễ dàng tiếp cận và làm quen.
- Quản lý ứng dụng ở mức toàn cục: Hỗ trợ quản lý ứng dụng ở mức toàn cục với các tính năng như Global.asax (trong ASP.NET truyền thống) hoặc middleware trong ASP.NET Core, quản lý session trên nhiều Server, và không nhất thiết cần Cookies.

Nhược điểm:

- Phức tạp hơn cho dự án nhỏ: Đối với các dự án web quy mô rất nhỏ và đơn giản, việc áp dụng mô hình MVC có thể gây cảm giác cồng kềnh và tốn thời gian hơn trong quá trình phát triển ban đầu, do cần cấu hình và thiết lập nhiều thành phần.
- Thời gian trung chuyển dữ liệu giữa các thành phần: Mặc dù là nhược điểm nhỏ, việc phân tách các thành phần Model, View, Controller có thể đôi khi làm tăng thời gian trung chuyển dữ liệu giữa chúng so với các kiến trúc monolithic đơn giản hơn. Tuy nhiên, lợi ích về khả năng mở rộng và bảo trì thường vượt trội hơn.

Bảng 1.1 Sự khác biệt mô hình lập trình Web ASP.NET MVC, ASP.NET Webform và ASP.NET Core:

Các tính năng	ASP.NET WebForm	ASP.NET MVC	ASP.NET Core MVC
Kiến trúc chương trình	Kiến trúc mô hình WebForm → Business → Database	Phân chia chương trình thành: Models, Views, Controllers	Giống ASP.NET MVC, nhưng tối ưu hơn, hỗ trợ kiến trúc Modular, Middleware, Dependency Injection
Cú pháp chương trình	Dùng cú pháp WebForm, sự kiện & controls do server quản lý	Sự kiện do controllers điều khiển, controls không do server quản lý	Cú pháp linh hoạt hơn, hỗ trợ Razor Pages, Tag Helpers, dễ tích hợp JavaScript/SPA
Truy cập dữ liệu	Dùng các công nghệ truyền thống như ADO.NET	Chủ yếu dùng LINQ, Entity Framework, SQL Class	Ưu tiên Entity Framework Core, LINQ, hỗ trợ tốt cho NoSQL, DbContext được tối ưu hóa
Debug	Debug toàn bộ (data layer, UI, controls...)	Dùng unit test để kiểm tra controllers dễ dàng hơn	Tích hợp sâu với unit test/xUnit, hỗ trợ tốt test DI, test middleware, kiểm thử REST API

Tốc độ phân tải	Chậm nếu nhiều controls do ViewState lớn	Nhanh hơn vì không dùng ViewState	Nhanh và tối ưu hơn nhờ không dùng ViewState, caching tốt, hỗ trợ async/await
Tương tác với JavaScript	Khó khăn vì controls bị server kiểm soát	Dễ hơn vì không có server-side controls	Tối ưu nhất: hỗ trợ full SPA frameworks như React, Angular, Vue, tích hợp SignalR dễ dàng
URL Address	Dạng <filename>.aspx?&<các tham số>	Cấu trúc rõ ràng: Controllers/Action/ID	Cấu trúc rõ hơn, hỗ trợ routing tùy chỉnh, RESTful API, endpoint routing mới

## 1.5 Công nghệ triển khai ASP.NET Core:

ASP.NET Core là một nền tảng phát triển ứng dụng web hiện đại, đa nền tảng và mã nguồn mở, được Microsoft phát triển để thay thế cho ASP.NET truyền thống. Với thiết kế linh hoạt và hiệu năng cao, ASP.NET Core hỗ trợ nhiều mô hình triển khai phù hợp với các môi trường khác nhau từ máy chủ nội bộ cho đến các nền tảng đám mây hiện đại.

### 1.5.1 Môi trường triển khai:

ASP.NET Core có thể triển khai linh hoạt trên nhiều hệ điều hành:

- Windows: Thường sử dụng IIS hoặc self-host thông qua Kestrel kết hợp với IIS làm reverse proxy.
- Linux: Phổ biến với mô hình sử dụng Kestrel làm web server kết hợp với Nginx hoặc Apache reverse proxy.
- macOS: Dùng chủ yếu để phát triển và kiểm thử cục bộ, ít dùng trong sản xuất.

### 1.5.2 Các hình thức triển khai:

ASP.NET Core hỗ trợ nhiều phương thức triển khai, giúp lập trình viên linh hoạt hơn khi phát hành ứng dụng:

- Framework-dependent deployment (FDD): Máy chủ cần cài sẵn .NET Core Runtime để chạy ứng dụng.
- Self-contained deployment (SCD): Ứng dụng được đóng gói cùng runtime, không phụ thuộc vào hệ thống máy chủ.
- Single-file executable: Toàn bộ ứng dụng được gói gọn trong một tệp thực thi duy nhất (ví dụ: .exe hoặc .dll), tiện lợi cho việc phân phối.
- Deployment qua CI/CD: Kết hợp cùng các công cụ tự động hóa như GitHub Actions, Azure DevOps hoặc Jenkins để tự động hóa quy trình build và triển khai.

### 1.5.3 Các nền tảng đám mây hỗ trợ:

ASP.NET Core tương thích tốt với nhiều nền tảng cloud hiện đại:

- Microsoft Azure: Hỗ trợ trực tiếp thông qua Azure App Services, Azure Container Instances và Azure Kubernetes Service.
- Amazon Web Services (AWS): Triển khai trên Elastic Beanstalk, EC2, ECS, hoặc Lambda.
- Google Cloud Platform (GCP): Có thể chạy trên App Engine, Cloud Run hoặc GKE (Google Kubernetes Engine).
- Docker & Kubernetes: ASP.NET Core có thể dễ dàng được container hóa và triển khai trên các nền tảng điều phối container như Docker Swarm hoặc Kubernetes.

#### 1.5.4 Công cụ hỗ trợ triển khai:

Một số công cụ phổ biến được sử dụng để triển khai ứng dụng ASP.NET Core:

- Visual Studio: Hỗ trợ publish trực tiếp lên IIS, Azure hoặc thư mục cục bộ.
- Dotnet CLI: Cung cấp các lệnh như dotnet publish, dotnet run để build và đóng gói ứng dụng.
- Docker CLI: Sử dụng để đóng gói ứng dụng vào container và triển khai trên các môi trường containerized.
- CI/CD Tools: Các công cụ như GitHub Actions, GitLab CI/CD hoặc Azure DevOps giúp tự động hóa toàn bộ quy trình triển khai.

## CHƯƠNG 2 :XÂY DỰNG ỨNG DỤNG THỰC TẾ

### 2.1 Phát biểu bài toán ứng dụng:

#### 2.1.1 Giới thiệu dự án:

Trong bối cảnh kỷ nguyên số bùng nổ và xu hướng học tập trực tuyến (e-learning) ngày càng phổ biến, nhu cầu về các nguồn tài liệu học tập chất lượng cao, đa dạng và dễ tiếp cận đang trở nên cấp thiết hơn bao giờ hết. Các phương pháp truyền thống như mua sách giấy tại hiệu sách hay tìm kiếm tài liệu rời rạc trên mạng Internet thường tốn thời gian, công sức và đôi khi không đảm bảo chất lượng.

Bài toán đặt ra là **nghiên cứu, phân tích, thiết kế và phát triển một hệ thống thương mại điện tử chuyên biệt**, tập trung vào việc **cung cấp các loại tài liệu học tập toàn diện** (bao gồm sách điện tử, giáo trình, bài giảng video, khóa học trực tuyến ngắn hạn, đề thi mẫu, tài liệu tham khảo chuyên ngành, v.v.) dưới nhiều định dạng khác nhau (PDF, EPUB, MP4, SCORM, v.v.). Hệ thống này sẽ được xây dựng trên nền tảng **ASP.NET Core** hiện đại, kết hợp với các công nghệ front-end tiên tiến và hệ quản trị cơ sở dữ liệu mạnh mẽ.



### 2.1.2 mục tiêu của dự án

Mục tiêu của hệ thống là tạo ra một **kênh phân phối hiệu quả và thân thiện**, nơi người học (học sinh, sinh viên, người đi làm muốn nâng cao kiến thức) có thể dễ dàng:

- **Tìm kiếm và khám phá** các tài liệu phù hợp với nhu cầu học tập của mình thông qua các công cụ lọc và gợi ý thông minh.
- **Xem thông tin chi tiết** về tài liệu, đọc các phần xem trước (preview), và tham khảo các đánh giá từ cộng đồng.
- **Thực hiện quy trình mua sắm trực tuyến** một cách an toàn và thuận tiện, bao gồm thêm vào giỏ hàng, đặt hàng và thanh toán qua nhiều hình thức (công thanh toán trực tuyến, chuyển khoản ngân hàng, ví điện tử).
- **Truy cập và quản lý** các tài liệu đã mua thông qua một thư viện cá nhân trực tuyến hoặc tải về thiết bị.
- **Tham gia cộng đồng**, đưa ra đánh giá, bình luận và tương tác với các tài liệu cũng như những người học khác.

### 2.1.3 Ý nghĩa thực tiễn

Đối với người học (Khách hàng mục tiêu):

- **Tiếp cận tri thức không giới hạn:** Phá bỏ rào cản về địa lý và thời gian, giúp người học ở mọi nơi, mọi lúc đều có thể tiếp cận kho tàng tri thức khổng lồ. Đặc biệt hữu ích cho học sinh, sinh viên ở vùng sâu vùng xa hoặc những người có lịch trình bận rộn.
- **Tối ưu chi phí và thời gian:** Tiết kiệm đáng kể chi phí đi lại, in ấn và thời gian tìm kiếm so với việc mua sắm truyền thống. Tài liệu số thường có giá thành phải chăng hơn và có thể sử dụng ngay lập tức sau khi mua.
- **Nguồn tài liệu đa dạng và phong phú:** Cung cấp nhiều lựa chọn hơn từ các nguồn khác nhau (nhà xuất bản, tác giả độc lập, trung tâm đào tạo), giúp người học dễ dàng tìm thấy nội dung phù hợp với phong cách và trình độ của mình.

- **Cá nhân hóa trải nghiệm học tập:** Hệ thống có thể ghi nhớ lịch sử mua hàng, đề xuất các tài liệu liên quan dựa trên sở thích, giúp người học khám phá những kiến thức mới mẻ.
- **Góp phần bảo vệ môi trường:** Việc sử dụng tài liệu số giảm thiểu lượng giấy in, góp phần vào mục tiêu phát triển bền vững.

Đối với người cung cấp nội dung (Tác giả, Giáo viên, Nhà xuất bản):

- **Mở rộng kênh phân phối:** Cung cấp một nền tảng hiệu quả để các tác giả, giáo viên, chuyên gia và nhà xuất bản có thể dễ dàng tiếp cận hàng ngàn người học tiềm năng mà không cần qua các kênh phân phối truyền thống phức tạp.
- **Tăng nguồn thu nhập thụ động:** Tạo cơ hội cho những người có kiến thức chuyên sâu có thể "số hóa" và bán tài liệu của mình, biến tri thức thành nguồn thu nhập bền vững.
- **Phản hồi trực tiếp từ người học:** Hệ thống đánh giá và bình luận giúp người cung cấp nội dung nhận được phản hồi giá trị để cải thiện và nâng cao chất lượng tài liệu.
- **Bảo vệ bản quyền:** Hệ thống có thể tích hợp các biện pháp bảo vệ bản quyền cho tài liệu số, giảm thiểu tình trạng sao chép trái phép.

Đối với xã hội và cộng đồng:

- **Thúc đẩy văn hóa học tập suốt đời:** Tạo ra một môi trường khuyến khích mọi người không ngừng học hỏi và trau dồi kiến thức.
- **Nâng cao chất lượng giáo dục:** Đóng góp vào việc phổ biến kiến thức chất lượng cao, giúp nâng cao trình độ dân trí và chất lượng nguồn nhân lực.
- **Phát triển kinh tế số:** Góp phần vào sự phát triển của ngành thương mại điện tử và kinh tế số tại Việt Nam.

## 2.2 Phân Tích yêu cầu của ứng dụng:

### 2.2.1 Mô tả chức năng ứng dụng:

Mục đích đề tài xây dựng hệ thống phần mềm hoạt trên môi trường web để quản lý việc bán hàng được thuận tiện, rất dễ quản lý các sản phẩm giúp cho người quản lý sẽ dễ dàng trong việc quản lý bán hàng cho khách có thể kiểm soát được khách hàng mua hàng và thống kê doanh thu của cửa hàng. Còn khách hàng mua hàng trên web sẽ được thoải mái hơn không tốn thời gian để đến cửa hàng, mà vẫn mua được sản phẩm mà mình thích.

Đối với Người dùng (Khách hàng):

#### ❖ Module Đăng ký/Đăng nhập & Quản lý tài khoản:

- Đăng ký tài khoản mới (email, số điện thoại, tích hợp Google/Facebook OAuth).
- Đăng nhập, quên mật khẩu, cập nhật thông tin cá nhân (hồ sơ, địa chỉ giao hàng).
- Lịch sử đơn hàng và quản lý tài liệu đã mua (Thư viện của tôi).

#### ❖ Module Tìm kiếm & Khám phá Tài liệu:

- **Bộ lọc mạnh mẽ:** Theo môn học, cấp độ, định dạng (Ebook, Video, Đề thi), tác giả, nhà xuất bản, khoảng giá, đánh giá.
- **Tính năng tìm kiếm nâng cao:** Gợi ý từ khóa, tìm kiếm theo tiêu đề/mô tả/nội dung (đối với tài liệu có thể crawl).
- **Trang chủ và Trang danh mục:** Hiện thị tài liệu mới nhất, bán chạy nhất, tài liệu nổi bật theo từng chuyên mục.

#### ❖ Module Chi tiết Tài liệu:

- **Thông tin đầy đủ:** Mô tả chi tiết, hình ảnh bìa, mục lục, thông tin tác giả, ngôn ngữ, số trang/thời lượng.
- **Phần xem trước (Preview):** Cung cấp một phần nội dung mẫu (ví dụ: vài trang PDF đầu tiên, đoạn video ngắn) để người dùng xem trước khi mua.

- **Đánh giá & Bình luận:** Hiện thị đánh giá sao trung bình, danh sách các bình luận từ người dùng đã mua, cho phép người dùng viết bình luận mới.

❖ Module Giỏ hàng & Thanh toán:

- **Quản lý giỏ hàng:** Thêm/bớt sản phẩm, cập nhật số lượng, lưu giỏ hàng.
- **Quy trình thanh toán an toàn:** Tích hợp đa dạng cổng thanh toán (ví dụ: VNPay, Momo, ZaloPay, OnePay), chuyển khoản ngân hàng, COD (đối với bản in).
- **Áp dụng mã giảm giá/khuyến mãi.**
- Xác nhận đơn hàng qua email/SMS.

❖ Module Thư viện Cá nhân & Tải xuống:

- Sau khi thanh toán thành công, tài liệu số sẽ tự động xuất hiện trong "Thư viện của tôi".
- Cho phép tải xuống tài liệu (với số lượt tải giới hạn để chống chia sẻ trái phép) hoặc xem trực tuyến (đối với video, khóa học).
- Quản lý quyền truy cập tài liệu.

Đối với Người quản trị (Admin):

❖ Quản lý Sản phẩm (Tài liệu):

- **CRUD (Create, Read, Update, Delete) tài liệu:** Nhập liệu thông tin chi tiết (tiêu đề, mô tả, giá, định dạng, tác giả, hình ảnh, file đính kèm), quản lý phiên bản tài liệu.
- **Phân loại và Gắn thẻ (Tagging):** Quản lý danh mục, chủ đề, từ khóa để tối ưu hóa tìm kiếm.
- **Quản lý file đính kèm:** Upload, quản lý các file tài liệu số.
- **Quản lý khuyến mãi/mã giảm giá:** Tạo, chỉnh sửa, áp dụng các chương trình khuyến mãi.

❖ Quản lý Đơn hàng:

- Xem danh sách đơn hàng, lọc theo trạng thái (mới, đang xử lý, đã hoàn thành, đã hủy, đã thanh toán, chưa thanh toán).
- Cập nhật trạng thái đơn hàng, xác nhận thanh toán.
- Xem chi tiết đơn hàng, thông tin khách hàng.

❖ Quản lý Người dùng:

- Xem danh sách người dùng, tìm kiếm, lọc.
- Kích hoạt/vô hiệu hóa tài khoản, đặt lại mật khẩu.
- Phân quyền người dùng (Admin, Biên tập viên, Khách hàng).

❖ Quản lý Nội dung Website:

- **Quản lý Bài viết/Tin tức:** Thêm, sửa, xóa các bài viết blog, tin tức khuyến mãi, thông báo.
- Quản lý các trang tĩnh (Giới thiệu, Liên hệ, Chính sách bảo mật).

❖ Thống kê & Báo cáo:

- Báo cáo doanh thu theo ngày/tháng/năm, theo sản phẩm, theo khách hàng.
- Thống kê lượt truy cập, lượt tải xuống.
- Báo cáo về hành vi mua sắm của người dùng.

## 2.2.2 Mô tả yêu cầu phi chức năng của ứng dụng:

### 1. Hiệu năng (Performance)

Hiệu năng là yếu tố quyết định tốc độ và sự phản hồi của hệ thống. Một ứng dụng chậm chạp có thể làm mất đi người dùng tiềm năng.

#### ❖ Tốc độ tải trang:

- Trang chủ và các trang danh mục tài liệu phải được **tải hoàn chỉnh trong vòng tối đa 3 giây** trên kết nối internet băng thông rộng tiêu chuẩn (ví dụ: cáp quang 50 Mbps).
- Các trang chi tiết sản phẩm và giỏ hàng phải tải trong vòng **dưới 4 giây** do có nhiều nội dung và tương tác hơn.

#### ❖ Thời gian xử lý giao dịch:

- Quá trình xử lý thanh toán, từ khi người dùng xác nhận đến khi nhận được thông báo giao dịch thành công từ hệ thống, không được vượt quá **5 giây**.

#### ❖ Khả năng chịu tải (Scalability under Load):

- Hệ thống phải có khả năng hỗ trợ **ít nhất 1.000 người dùng đồng thời** (concurrent users) thực hiện các tác vụ duyệt, tìm kiếm, thêm vào giỏ hàng mà không làm giảm đáng kể hiệu suất (thời gian phản hồi tăng không quá 20%).
- Đối với các sự kiện hoặc chương trình khuyến mãi lớn, hệ thống cần có khả năng mở rộng để đáp ứng đột biến lên tới **3.000 người dùng đồng thời** trong thời gian ngắn (tối đa 30 phút).

#### ❖ Tốc độ tìm kiếm:

- Kết quả tìm kiếm tài liệu phải hiển thị trong vòng **dưới 2 giây**, ngay cả khi cơ sở dữ liệu có hàng chục ngàn tài liệu.

## 2. Bảo mật (Security)

Bảo mật là tối quan trọng, đặc biệt khi xử lý thông tin cá nhân và tài chính của người dùng. Mất mát dữ liệu hoặc lộ thông tin có thể gây ra hậu quả nghiêm trọng.

### ❖ **Bảo vệ dữ liệu người dùng:**

- Tất cả **dữ liệu nhạy cảm** của người dùng (như mật khẩu, thông tin cá nhân, chi tiết địa chỉ) phải được **mã hóa mạnh** (ví dụ: sử dụng thuật toán mã hóa AES-256 hoặc bcrypt cho mật khẩu) cả khi lưu trữ trong cơ sở dữ liệu và khi truyền tải.

### ❖ **Giao thức truyền thông an toàn:**

- Toàn bộ website phải sử dụng giao thức **HTTPS** để mã hóa mọi giao tiếp giữa trình duyệt của người dùng và máy chủ, ngăn chặn việc đánh cắp dữ liệu khi truyền tải.

### ❖ **Chống tấn công phổ biến:**

- Hệ thống phải được thiết kế và triển khai để **ngăn chặn các lỗ hổng bảo mật phổ biến** được liệt kê trong danh sách OWASP Top 10 (ví dụ: SQL Injection, Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF), Broken Authentication).

### ❖ **Cơ chế phân quyền chặt chẽ:**

- Phải có cơ chế **phân quyền người dùng rõ ràng và chặt chẽ**. Chỉ Quản trị viên mới có quyền truy cập vào các chức năng quản lý và cấu hình hệ thống. Các quyền này cần được kiểm tra nghiêm ngặt ở cả phía máy chủ.

### ❖ **Bảo vệ bản quyền tài liệu số:**

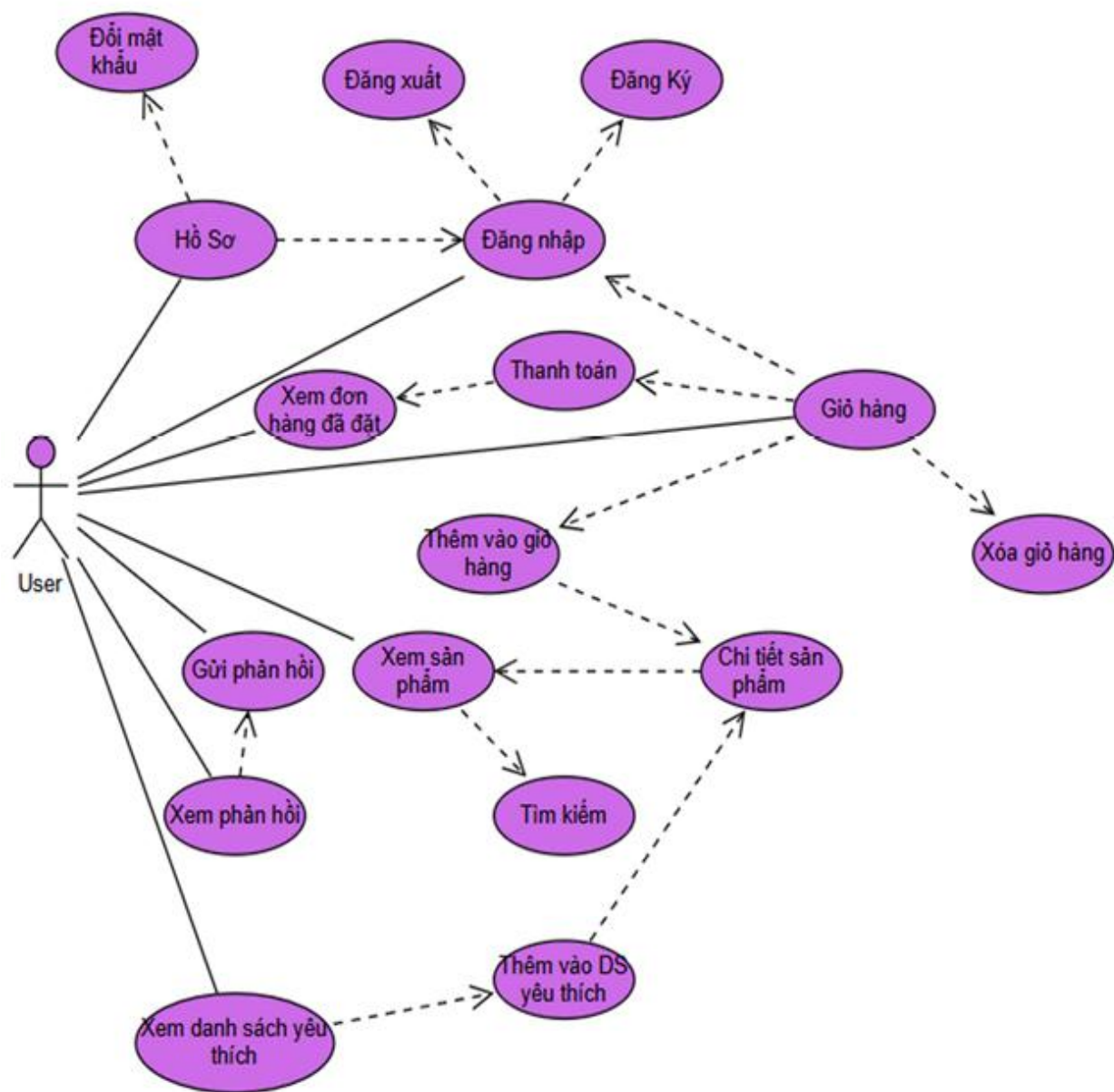
- Cần triển khai các biện pháp để **ngăn chặn việc sao chép và phân phối trái phép** tài liệu số đã mua. Các biện pháp này có thể bao gồm:
  - ✓ Giới hạn số lượt tải xuống của một tài liệu.
  - ✓ Thêm watermark cá nhân hóa vào tài liệu PDF (ví dụ: ID người mua hoặc email) để dễ dàng truy vết.

- ✓ Sử dụng các công nghệ DRM (Digital Rights Management) cơ bản nếu khả thi.



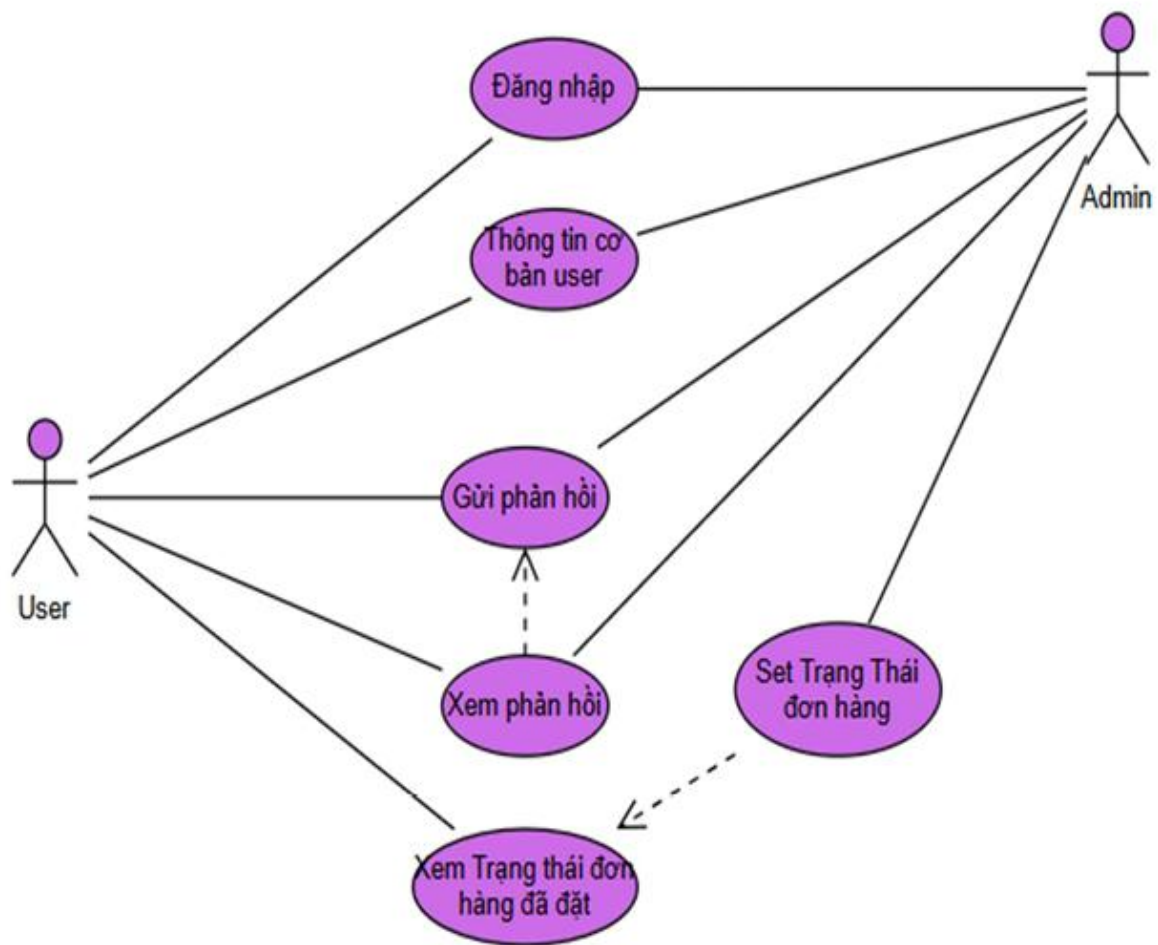
## 2.2.3 Sơ đồ UseCase:

### 2.2.3.1 UseCase người dùng (User)



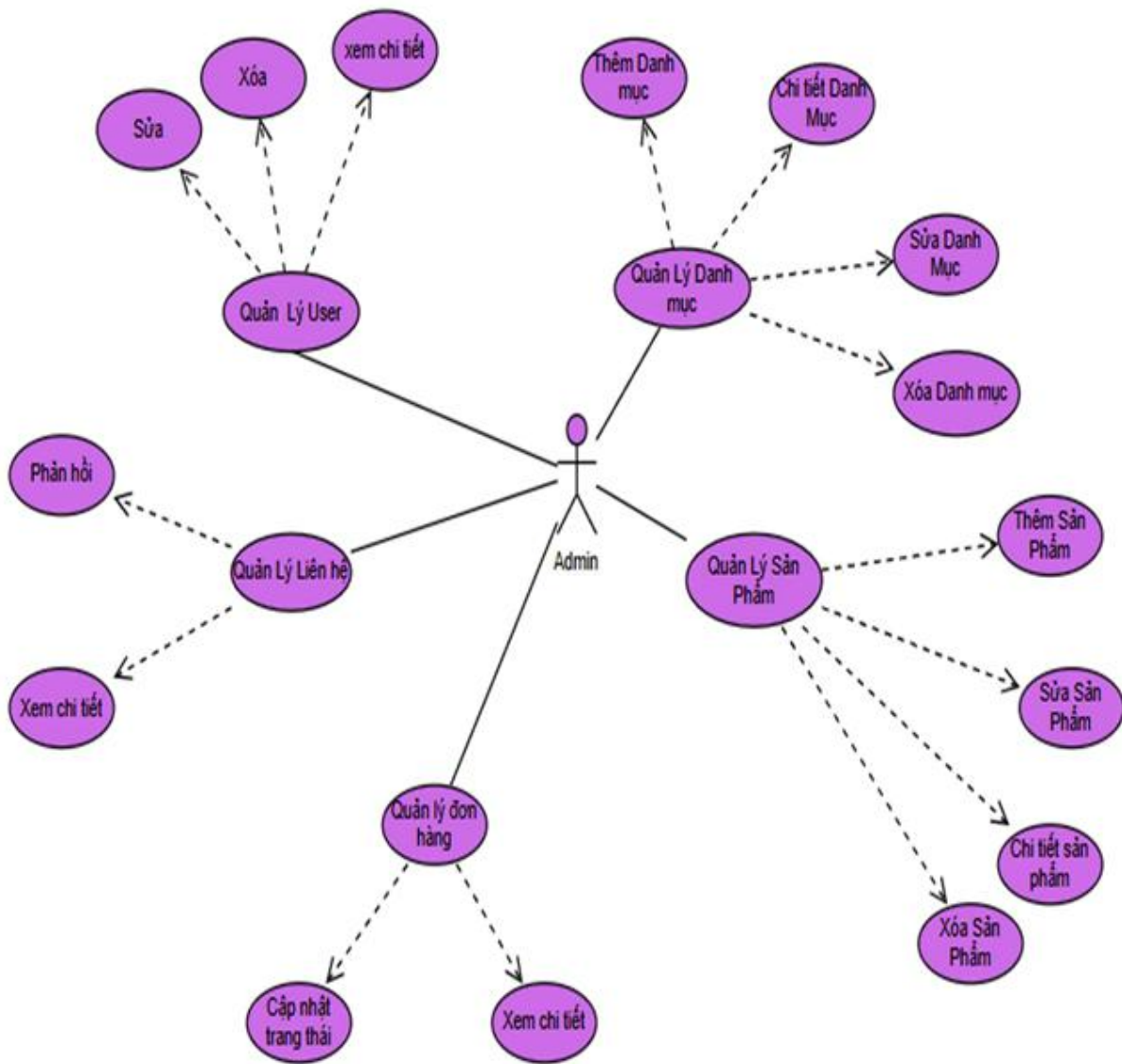
Hình 2.1: Sơ đồ UseCase mô tả chức năng của người dùng (User)

### 2.2.3.2 UseCase người dùng (User) tương tác với người quản trị (Admin)



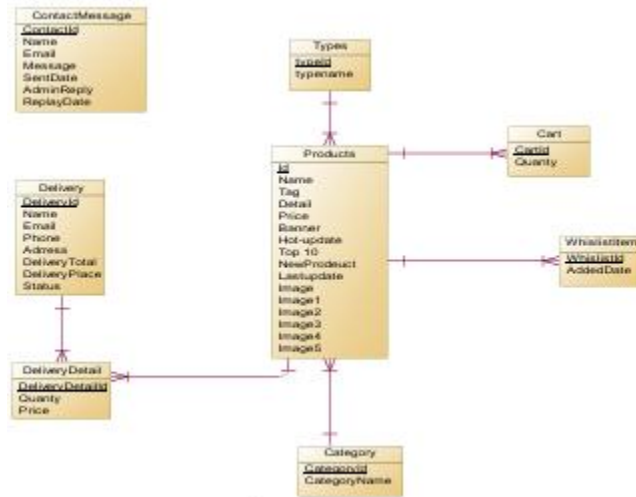
Hình 2.2: Sơ đồ UseCase mô tả sự tương tác giữa người dùng(User) với người quản trị(Admin)

### 2.2.3.3 UseCase người quản trị viên (Admin)

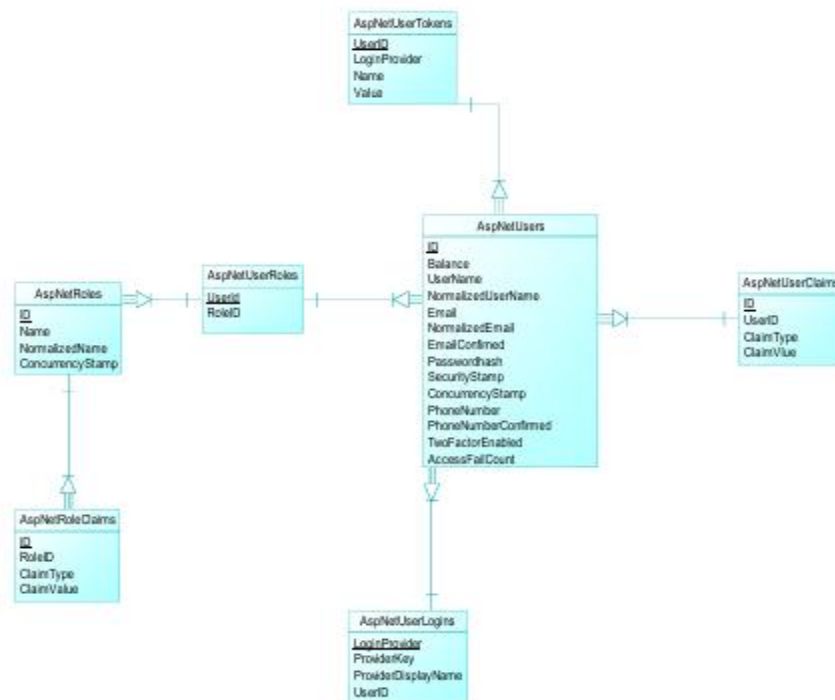


## 2.3 Thiết kế cơ sở dữ liệu:

### 2.3.1 Thiết kế logic ERD:

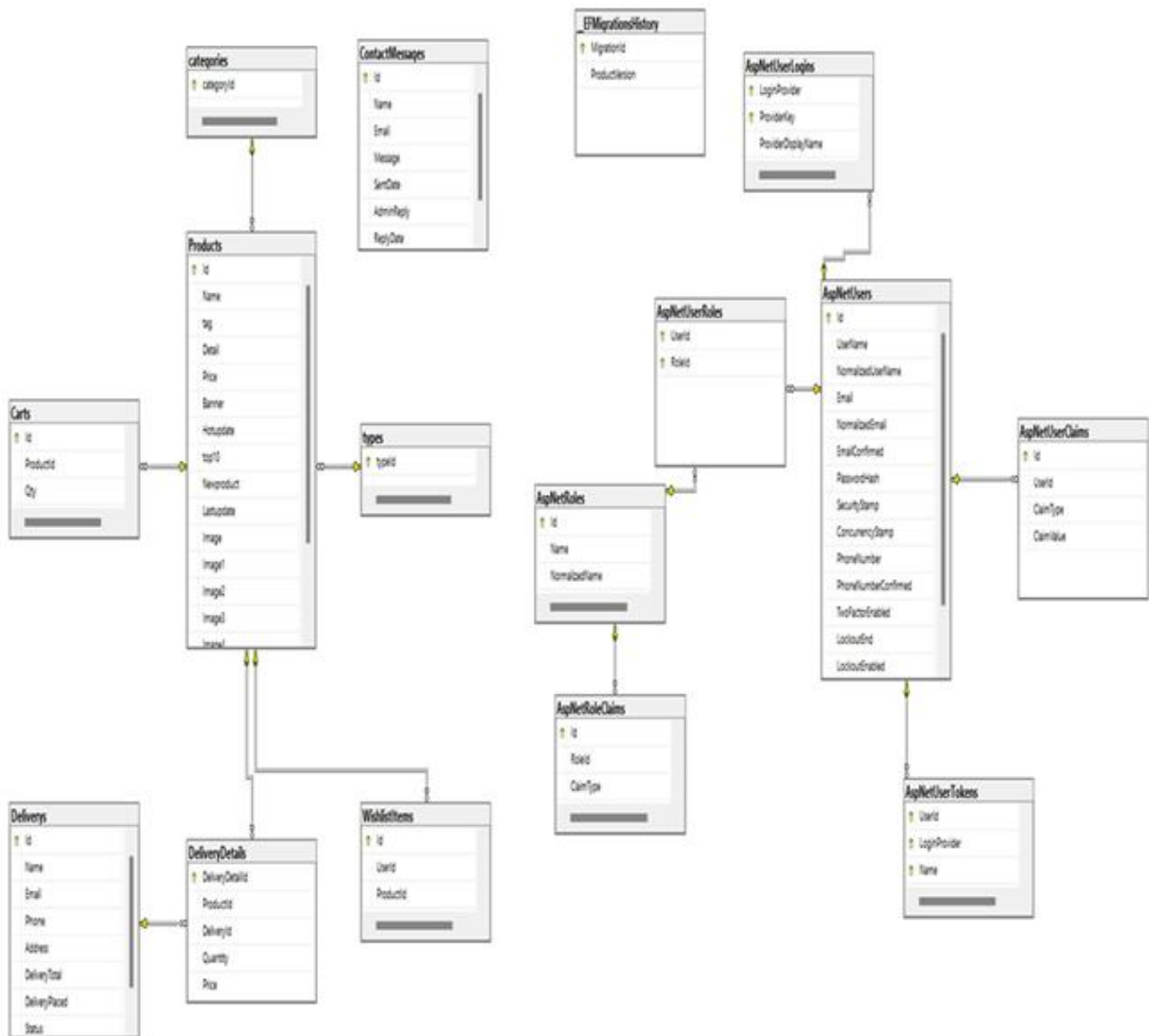


Hình 2.4: Phần dữ liệu nghiệp vụ (domain data)



Hình 2.5: Phần hệ thống người dùng (Identity)

### 2.3.2 Mô hình vật lý

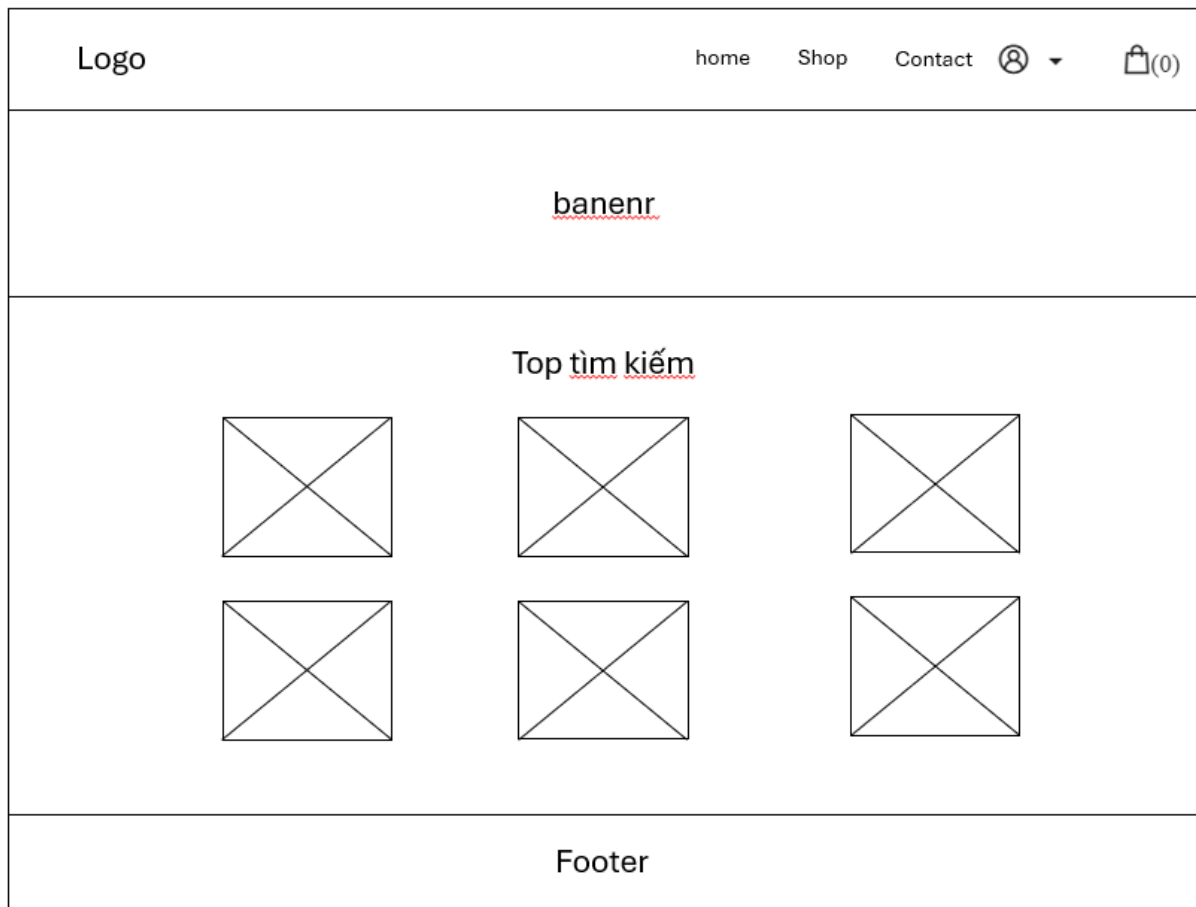


Hình 2.6: Mô hình Cơ sở dữ liệu

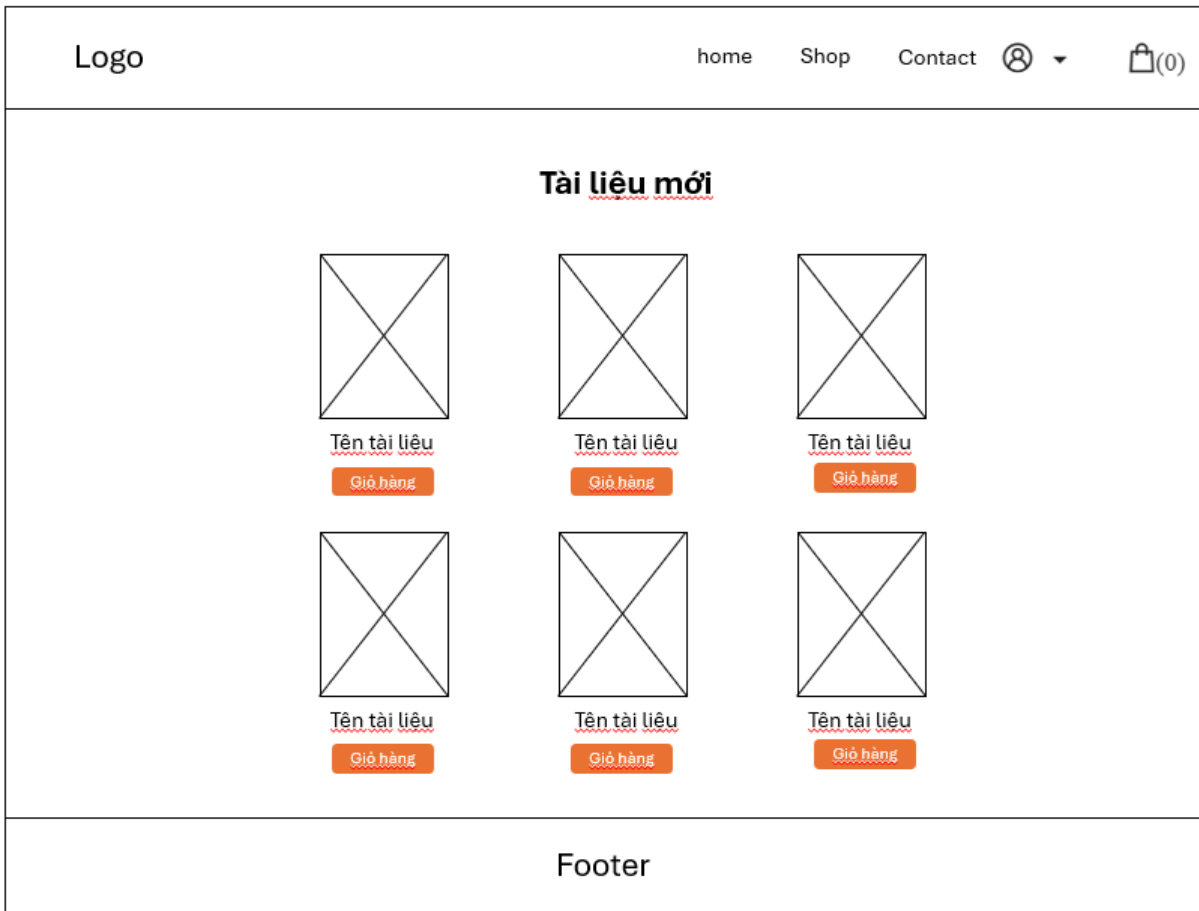
## 2.4 Thiết kế giao diện:

### 2.4.1 Trang chủ



Hiển thị banner, sản phẩm nổi bật, menu danh mục.



## 2.4.2 Trang Shop

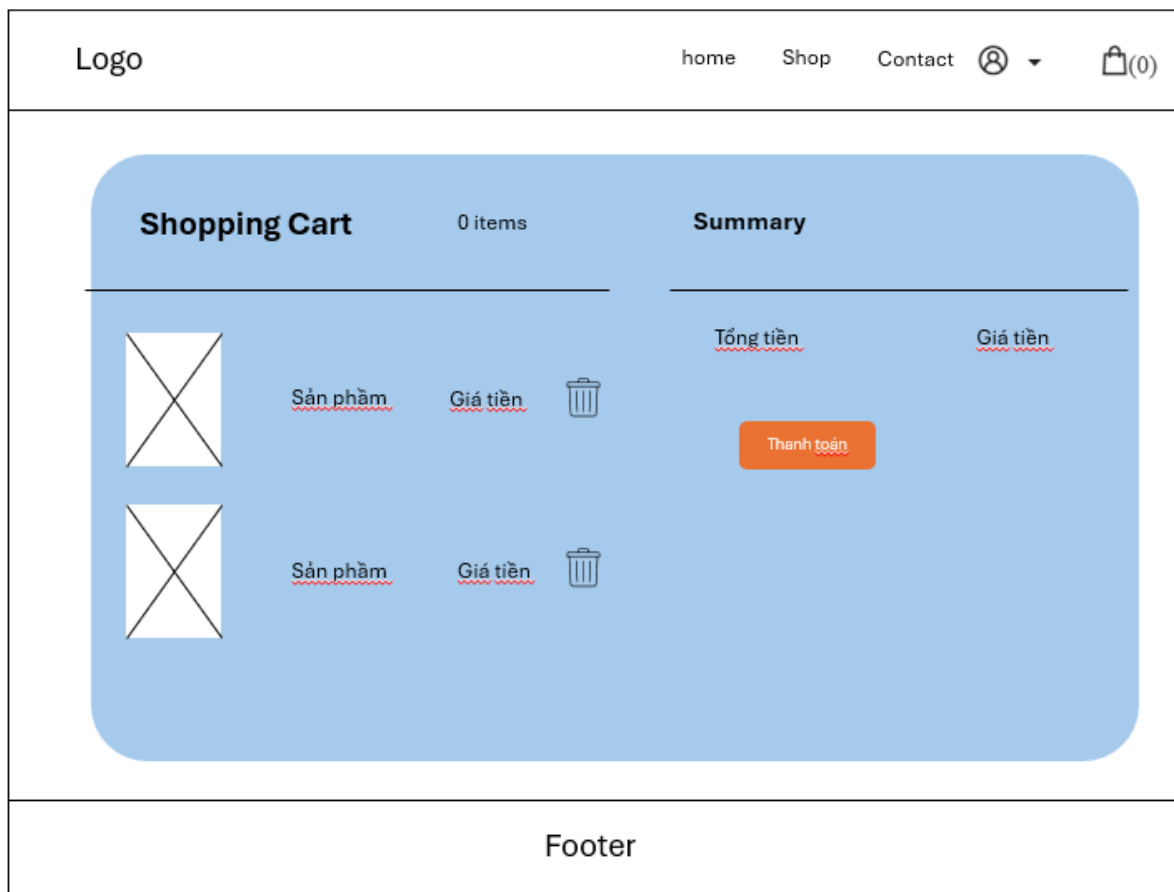


### 2.4.3 Trang liên hệ

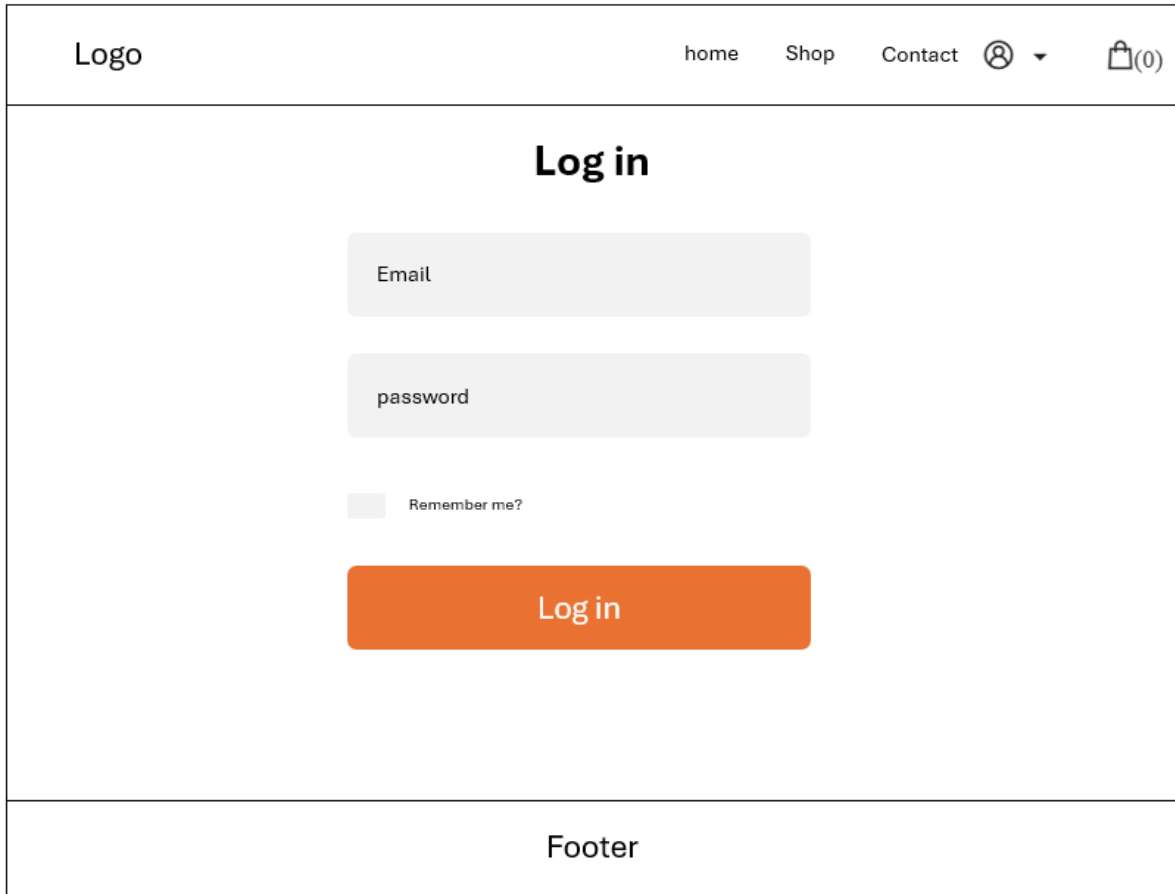
Logo	home	Shop	Contact	 ▾	 (0)
<h2>Liên hệ</h2> <div><p>Website của chúng tôi sẽ sớm liên hệ với bạn.</p><div><div>Tên</div><div>Email</div><div>Ghi chú</div></div></div>					
Footer					



## 2.4.4 Trang giỏ hàng



## 2.4.5 Trang đăng nhập



The diagram illustrates the layout of a login page. It is divided into three main horizontal sections: a top navigation bar, a central content area, and a bottom footer. The top navigation bar contains a 'Logo' on the left, and links for 'home', 'Shop', and 'Contact' in the center. On the right side of the navigation bar are a user profile icon with a dropdown arrow and a shopping cart icon labeled '(0)'. The central content area is titled 'Log in' and contains a form with three elements: an 'Email' input field, a 'password' input field, and a 'Remember me?' checkbox. Below these fields is a large orange 'Log in' button. The bottom footer section is labeled 'Footer'.

## 2.5 Thiết kế các thành phần MVC:

### 2.5.1 Model

Tầng Model trong kiến trúc MVC chịu trách nhiệm quản lý dữ liệu, định nghĩa cấu trúc của dữ liệu và các quy tắc nghiệp vụ liên quan. Nó tương tác trực tiếp với cơ sở dữ liệu để lấy, lưu trữ, cập nhật và xóa dữ liệu. Trong ứng dụng, tầng Model được triển khai dưới dạng các lớp C# (POCOs - Plain Old CLR Objects) kết hợp với Entity Framework Core để ánh xạ với các bảng trong cơ sở dữ liệu.

### 2.5.1.1 Class Product

```
public class Product
{
    16 references
    public int Id { get; set; }
    7 references
    public string? Name { get; set; }
    7 references
    public string? Detail { get; set; }
    7 references
    public string? ImageUrl { get; set; }
    11 references
    public decimal Price { get; set; }
    1 reference
    public bool IsTrendingProduct { get; set; }
}
```

### 2.5.1.2 Class ShoppingCartItem

```
public class ShoppingCartItem
{
    0 references
    public int Id { get; set; }

    9 references
    public Product? Product { get; set; }

    9 references
    public int Qty { get; set; }

    6 references
    public string? ShoppingCartId { get; set; }
}
```

### 2.5.1.3 Class Order

```
public class Order
{
    0 references
    public int Id { get; set; }
    2 references
    public string? FirstName { get; set; }
    2 references
    public string? LastName { get; set; }
    2 references
    public string? Email { get; set; }
    2 references
    public string? Phone { get; set; }
    2 references
    public string? Address { get; set; }
    1 reference
    public decimal OrderTotal { get; set; }
    1 reference
    public DateTime OrderPlaced { get; set; }
    2 references
    public List<OrderDetail>? OrderDetails { get; set; }
}
```

### 2.5.1.4 Class OrderDetail

```
public class OrderDetail
{
    0 references
    public int OrderDetailId { get; set; }
    1 reference
    public int ProductId { get; set; }
    0 references
    public Product? Product { get; set; }
    0 references
    public int OrderId { get; set; }
    0 references
    public Order? Order { get; set; }
    1 reference
    public int Quantity { get; set; }
    1 reference
    public decimal Price { get; set; }
}
```

## 2.5.2 View

View là phần giao diện hiển thị dữ liệu từ model và cho phép người dùng tương tác với hệ thống. Trong dự án này các View được mình xây dựng bằng công nghệ HTML/CSS kết hợp với Razor và các framework giao diện như Bootstrap để đảm bảo tính thẩm mỹ, phản hồi nhanh và thân thiện với người dùng.

### 2.5.2.1 Home

```
@model IEnumerable<CoffeeShop.Models.Product>

<section data-bs-version="5.1" class="header18 cid-tsEQyI2Rcx mbr-fullscreen mbr-parallax-background" id="header18-1">
  <div class="mbr-overlay" style="opacity:0.9; background-image: image();">
  </div>
  <div class="mbr-overlay" style="opacity: 0.7; background-color: #0a71de;">
  <div class="align-center container">
    <div class="row justify-content-center">
      <div class="col-12 col-lg-18">
        <h1 class="mbr-section-title mbr-fonts-style mbr-white mb-3 display-1">
          <strong>Giảm ngay 38% khi đăng ký hội viên</strong>
        </h1>
        <p class="mbr-text mbr-fonts-style mbr-white display-7">Tham quan shop</p>
        <div class="btn btn-primary display-4">
          <a class="btn btn-primary display-4" asp-controller="Product" asp-action="Shop">
            <span class="mobi-mbri mobi-mbri-shopping-bag mbr-iconfont mbr-iconfont-btn"></span>Shop now</a>
        </div>
      </div>
    </div>
  </div>
</section>

<section data-bs-version="5.1" class="content2 cid-tsEQ5Cggh" id="content2-2">
  <div class="container">
    <div class="mbr-section-head">
      <h2 class="mbr-section-title mbr-fonts-style align-center mb-0 display-2">
        <strong>top tile kiể</strong>
      </h2>
    </div>
    <div class="row mt-4">
      @foreach (var item in Model)
      {
        <div class="item features-image col- col-md-6 col-lg-4">
          <div class="item-wrapper">
            <div class="item-img">
              
            </div>
            <div class="item-content">
              <h3 class="item-subtitle mbr-fonts-style st-1 display-7">
                <strong>Sở THPT 2025</strong>
              </h3>
              <p class="mbr-text mbr-fonts-style st-3 display-7">@item.Price.ToString("c")</p>
            </div>
            <div class="mbr-section-btm item-footer st-2">
              <a asp-controller="Shoppingcart" asp-action="AddToShoppingCart" asp-route-pid="@item.Id" class="btn item-btn btn-primary display-4" target="self">
                <span class="mbr-iconfont mbr-iconfont-btn"></span>Add to cart
              </a>
            </div>
          </div>
        </div>
      }
    </div>
  </div>
</section>
```

### 2.5.2.2 Contact

```
<section data-bs-version="5.1" class="form7 cid-tsJ8wGtENs" id="form7-q">
  <div class="container">
    <div class="mbr-section-head">
      <h3 class="mbr-section-title mbr-fonts-style align-center mb-0 display-2">
        <strong>Liên Hệ</strong>
      </h3>
    </div>
    <div class="row justify-content-center mt-4">
      <div class="col-lg-8 mx-auto mbr-form">
        <form method="POST" class="mbr-form form-with-styler mx-auto">
          <p class="mbr-text mbr-fonts-style align-center mb-4 display-7">
            Website của chúng tôi sẽ sớm liên hệ với bạn.
          </p>
          <div class="dragArea row">
            <div class="col-lg-12 col-md-12 col-sm-12 form-group mb-3" data-for="name">
              <input type="text" name="name" placeholder="Tên" data-form-field="name"
                class="form-control" value="" id="name-form7-q">
            </div>
            <div class="col-lg-12 col-md-12 col-sm-12 form-group mb-3" data-for="email">
              <input type="email" name="email" placeholder="Email" data-form-field="email"
                class="form-control" value="" id="email-form7-q">
            </div>
            <div data-for="phone" class="col-lg-12 col-md-12 col-sm-12 form-group mb-3">
              <textarea type="text" class="form-control" rows="5" placeholder="Ghi chú"></textarea>
            </div>
            <div class="col-auto mbr-section-btn align-center">
              <button type="submit" class="btn btn-primary display-4">Send</button>
            </div>
          </div>
        </form>
      </div>
    </div>
  </div>
</section>
```

### 2.5.2.3 Shop

```
<section data-bs-version="5.1" class="infos cid-tsE7zVmh5a mbr-parallax-background" id="infos-0">
  <div class="mbr-overlay" style="opacity: 0.9; background-image: url('https://images.unsplash.com/photo-1588933873521-dc49ac8d4e6a?ixlib=rb-4.0.3&ixid=MnwxMjA3fDB8MHxwaG90byl1ZWQ6aW50fD88fHx8&auto=format&fit=crop&w=2000&q=80');">
  </div>
  <div class="mbr-overlay" style="opacity: 0.7; background-color: rgb(0, 0, 0);"></div>
  <div class="container">
    <div class="row justify-content-center">
      <div class="card col-12 col-lg-10">
        <div class="card-wrapper">
          <div class="card-box align-center">
            <h1 class="card-title mbr-fonts-style align-center mb-4 display-1">
              <strong>Shop Tailieu</strong>
            </h1>
          </div>
        </div>
      </div>
    </div>
  </div>
</section>

<section data-bs-version="5.1" class="content2 cid-tsEZVFgBri" id="content2-g">
  <div class="container">
    <div class="mbr-section-head">
      <h3 class="mbr-section-subtitle mbr-fonts-style align-center mb-0 mt-2 display-5">
        Tài liệu mới
      </h3>
    </div>
    <div>
      <partial name="Product"></partial>
    </div>
  </div>
</section>
```

#### 2.5.2.4 Detail

```
<div class="container">
  <div class="card-wrapper">
    <div class="row align-items-center">
      <div class="col-12 col-lg-6">
        <div class="image-wrapper">
          
        </div>
      </div>
      <div class="col-12 col-lg">
        <div class="text-box">
          <h5 class="mbr-title mbr-fonts-style display-2">
            <strong>@Model.Name</strong>
          </h5>
          <p class="mbr-text mbr-fonts-style display-7">
            @Model.Detail
          </p>
          <div class="cost">
            <span class="currentcost mbr-fonts-style pr-2 display-2">@Model.Price.ToString("c")</span>
          </div>
          <div class="mbr-section-btn pt-3">
            <a href="" class="btn btn-primary display-4">
              <span class="mbr-iconfont mbr-iconfont-btn"></span>Add to cart&nbsp;
            </a>
          </div>
        </div>
      </div>
    </div>
  </div>
</div>
<br />
```

#### 2.5.3 Controller

Controller là thành phần trung gian giữa Model và View. Nó tiếp nhận yêu cầu (request) từ người dùng, xử lý logic phù hợp bằng cách sử dụng dữ liệu từ Model, sau đó trả về kết quả thông qua View. Controller giúp quản lý luồng điều khiển của ứng dụng và định nghĩa các hành động mà người dùng có thể thực hiện. Trong hệ thống, mỗi Controller thường tương ứng với một nhóm chức năng chính của ứng dụng, ví dụ như quản lý bài viết, người dùng, sản phẩm, đơn hàng.

### 2.5.3.1 HomeController

```
public class HomeController : Controller
{
    private IProductRepository productRepository;
    0 references
    public HomeController(IProductRepository productRepository)
    {
        this.productRepository = productRepository;
    }
    0 references
    public IActionResult Index()
    {
        return View(productRepository.GetAllProducts());
    }
    0 references
    public IActionResult Contact()
    {
        return View();
    }
    0 references
    public IActionResult Shop()
    {
        return View();
    }
}
```

### 2.5.3.2 OrdersController

```
public class OrdersController : Controller
{
    private IOrderRepository OrderRepository;
    private IShoppingCartRepository shoppingCartRepository;
    0 references
    public OrdersController(IOrderRepository orderRepository,
        IShoppingCartRepository shoppingCartRepository)
    {
        this.OrderRepository = orderRepository;
        this.shoppingCartRepository = shoppingCartRepository;
    }
    0 references
    public IActionResult Checkout()
    {
        return View();
    }
    [HttpPost]
    0 references
    public IActionResult Checkout(Order order)
    {
        OrderRepository.PlaceOrder(order);
        shoppingCartRepository.ClearCart();
        HttpContext.Session.SetInt32("CartCount", 0);

        return RedirectToAction("CheckoutComplete");
    }
    0 references
    public IActionResult CheckoutComplete()
    {
        return View();
    }
}
```



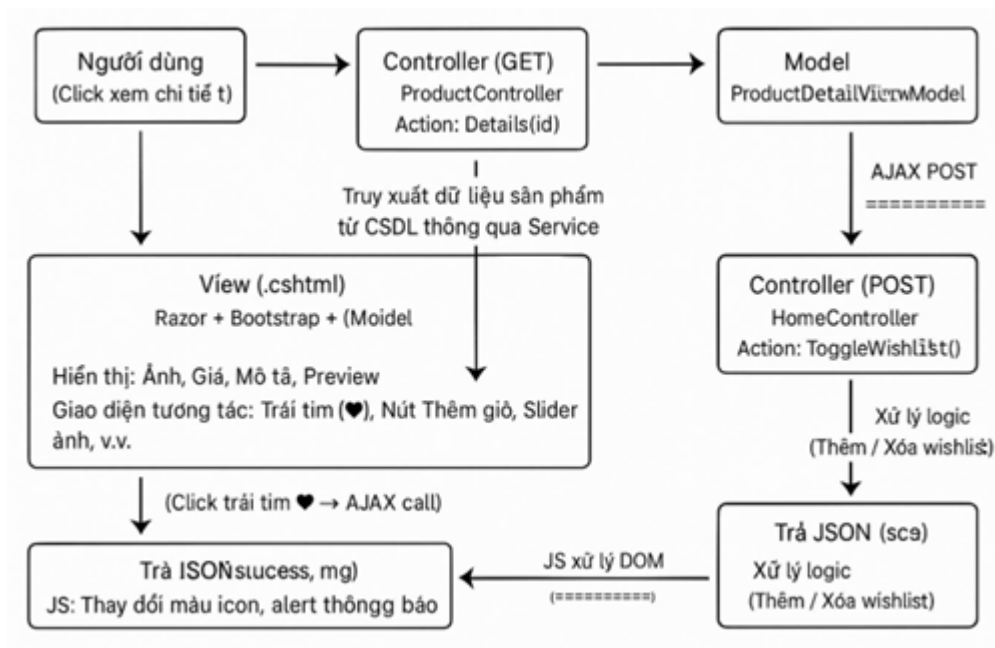
### 2.5.3.3 ProductController

```
public class ProductController : Controller
{
    private IProductRepository productRepository;
    // references
    public ProductController(IProductRepository productRepository)
    {
        this.productRepository = productRepository;
    }
    // references
    public IActionResult Shop()
    {
        return View(productRepository.GetAllProducts());
    }
    // references
    public IActionResult Detail(int id)
    {
        var product = productRepository.GetProductDetail(id);
        if (product != null)
        {
            return View(product);
        }
        return NotFound();
    }
}
```

### 2.5.3.4 ShoppingCartController

```
public class ShoppingCartController : Controller
{
    private IShoppingCartRepository shoppingCartRepository;
    private IProductRepository productRepository;
    // references
    public ShoppingCartController(IShoppingCartRepository shoppingCartRepository, IProductRepository productRepository)
    {
        this.shoppingCartRepository = shoppingCartRepository;
        this.productRepository = productRepository;
    }
    // references
    public IActionResult Index()
    {
        var items = shoppingCartRepository.GetAllShoppingCartItems();
        shoppingCartRepository.shoppingcartItems = items;
        ViewBag.TotalCart = shoppingCartRepository.GetShoppingCartTotal();
        return View(items);
    }
    // references
    public RedirectToActionResult AddToShoppingCart(int pId)
    {
        var product = productRepository.GetAllProducts().FirstOrDefault(p => p.Id ==
d);
        if (product != null)
        {
            shoppingCartRepository.AddToCart(product);
            int cartCount = shoppingCartRepository.GetAllShoppingCartItems().Count();
            HttpContext.Session.SetInt32("CartCount", cartCount);
        }
        return RedirectToAction("Index");
    }
    // references
    public RedirectToActionResult RemoveFromShoppingCart(int pId)
    {
        var product = productRepository.GetAllProducts().FirstOrDefault(p => p.Id ==
d);
        if (product != null)
        {
            shoppingCartRepository.RemoveFromCart(product);
            int cartCount = shoppingCartRepository.GetAllShoppingCartItems().Count();
            HttpContext.Session.SetInt32("CartCount", cartCount);
        }
        return RedirectToAction("Index");
    }
}
```

## 2.5.4 Sơ đồ minh họa cấu trúc hoạt động MVC



## 2.6 Triển khai cài đặt

### 2.6.1 Cấu hình máy phát triển

- Hệ điều hành: Windows 11 64-bit
- CPU: Intel Core i7-11400H
- GPU: GTX 1650 GDDR6
- IDE: Visual Studio 2022 Community
- Trình duyệt kiểm thử: Google Chrome, Microsoft Edge

### 2.6.2 Môi trường phát triển

- Ngôn ngữ: C#
- Kiến trúc: ASP.NET MVC (Model – View – Controller)
- Razor View Engine: để xây dựng giao diện người dùng
- Entity Framework Core (Code First): để làm việc với cơ sở dữ liệu
- Microsoft SQL Server (LocalDB): hệ quản trị cơ sở dữ liệu
- ASP.NET Core Identity: để xử lý đăng nhập, phân quyền, quản lý người dùng

- Bootstrap : xây dựng giao diện responsive, thân thiện
- Font Awesome: sử dụng icon
- CSS & JavaScript : để tạo hiệu ứng, cải thiện trải nghiệm giao diện
- CSS & JavaScript : để tạo hiệu ứng, cải thiện trải nghiệm giao diện.

### 2.6.3 Cài đặt

1. Mở Visual studio 2022 tạo project ASP.Net Core (MVC)
2. Cài bộ thư viện EF và Identity
3. Khởi Tạo Model và Data
4. Cấu hình appsetting.json và program
5. Chạy lệnh add-migration và update-database để tạo CSDL theo CodeFisrt
6. Thực hiện viết các controller xử lý logic và view để hiển thị
7. Nhấn Run/Build để chạy


### 3.1 Tranh chủ



**top tìm kiếm**

Add to cart

[illegible]

### 3.3 Trang liên hệ



HomeShopContact (0)

## Liên Hệ

Website của chúng tôi sẽ sớm liên hệ với bạn.

Tên

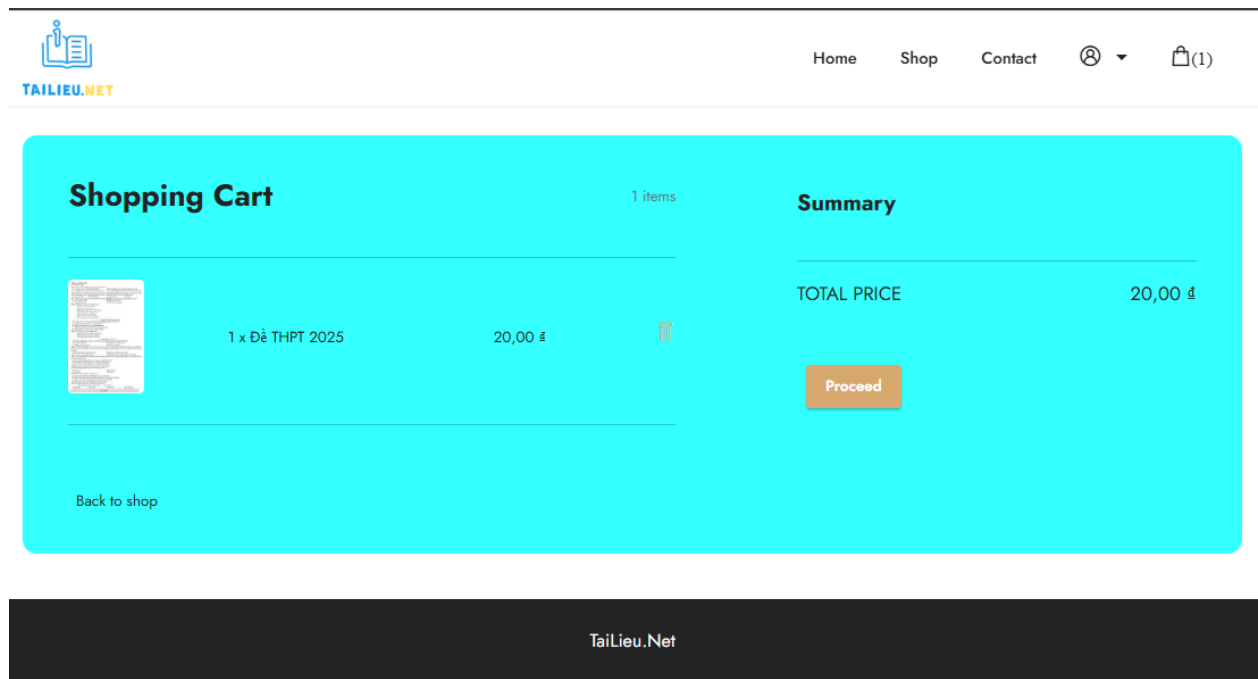
Email

Ghi chú

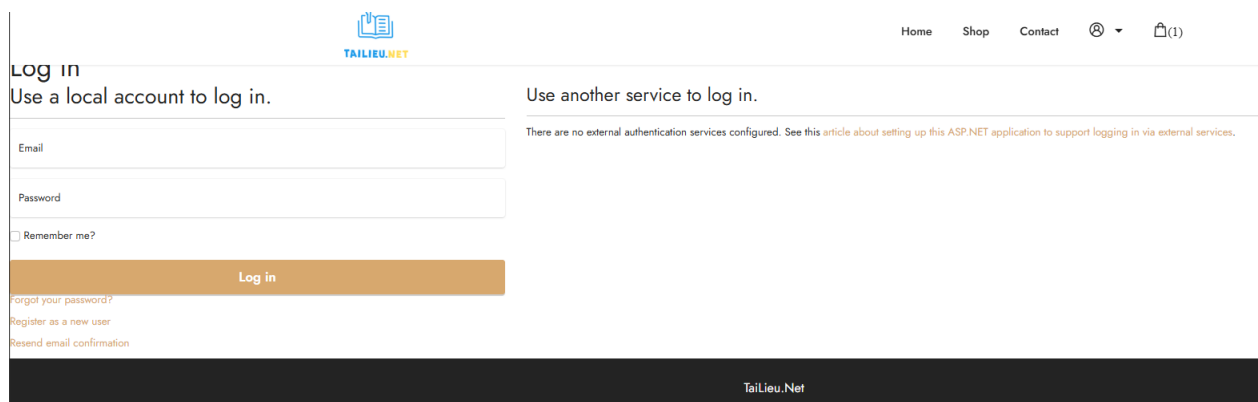
Send

TaiLieu.Net

### 3.4 Trang Giỏ hàng



### 3.5 Trang Login



## CHƯƠNG 4 : Kết Luận

### 4.1 Tổng kết kiến thức đạt được

Qua quá trình thực hiện đồ án, mình đã củng cố và vận dụng nhiều kiến thức lý thuyết và thực tiễn, bao gồm:

- Hiểu rõ và áp dụng mô hình MVC (Model – View – Controller) trong xây dựng ứng dụng web.
- Làm việc với ASP.NET Core MVC, sử dụng Razor View Engine để xây dựng giao diện động
  - Áp dụng Entity Framework Core theo hướng Code First để thiết kế cơ sở dữ liệu và thao tác với dữ liệu.
- Tích hợp ASP.NET Core Identity để xử lý đăng ký, đăng nhập, phân quyền người dùng.
- Sử dụng Bootstrap, CSS/JS, AJAX để tạo giao diện thân thiện, hiện đại và nâng cao trải nghiệm người dùng.
- Rèn luyện kỹ năng làm việc với Visual Studio, SQL Server và tư duy thiết kế hệ thống web thực tế.

### 4.2 Điểm tồn tại

Mặc dù đã hoàn thành hầu hết các chức năng đề ra, nhưng đồ án vẫn còn một số hạn chế như:

- Tính bảo mật cơ bản có nhưng chưa được kiểm thử kỹ lưỡng.
- Chưa có API xác thực thanh toán
- Chưa có hệ thống chatbox
- Chưa tích hợp chức năng gửi email thông báo.
- Giao diện chưa thực sự tối ưu cho mobile.
- Chưa có hệ thống chức năng theo dõi vị trí đơn hàng cách còn bao xa



- Chưa có tính năng thống kê doanh thu cho admin
- Chưa có hệ thống nhập kho, số lượng sản phẩm, tồn kho.
- Chưa có đầy đủ các trang

#### 4.3 Hướng phát triển dự án

Trong tương lai, để nâng cấp và hoàn thiện hệ thống, em định hướng mở rộng các tính năng sau:

- Tích hợp API cho phép thanh toán nhiều ngân hàng.
- Tự động gửi email thông báo cho đơn hàng.
- Tích hợp tính năng quên mật khẩu.
- Tối ưu giao diện cho mobile
- Cải thiện hiệu suất và bảo mật
- Tích hợp Google map API để cho khách hàng dễ theo dõi đơn hàng
- Thêm số lượng tồn kho, nhập kho.

#### 4.4 Tài liệu tham khảo:

- [1] Microsoft Docs – ASP.NET Core MVC – Microsoft – Online – 2023  
<https://learn.microsoft.com/en-us/aspnet/core/mvc>
- [2] Entity Framework Core Documentation – Microsoft – Online – 2023  
<https://learn.microsoft.com/en-us/ef/core/>
- [3] ASP.NET Core Identity Overview – Microsoft – Online – 2023  
<https://learn.microsoft.com/en-us/aspnet/core/security/authentication/identity>
- [4] Xây dựng website với ASP.NET Core MVC – Ths. Nguyễn Đức Tấn – 2025 Đại học Yersin (Bài giảng nội bộ)
- [5] ASP.NET Core MVC Full Project Tutorial – IAmTimCorey (YouTube) – 2021  
<https://www.youtube.com/watch?v=FbfkYjIg8Fw>