

TRƯỜNG ĐẠI HỌC YERSIN ĐÀ LẠT
KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO BÀI TẬP
MÔN: LẬP TRÌNH WED 1

ĐỀ TÀI: XÂY DỰNG WEBSITE THƯƠNG MẠI ĐIỆN TỬ KINH DOANH SẢN PHẨM
TÀI LIỆU HỌC TẬP + BÁO CÁO + ĐỒ ÁN

Giáo viên hướng dẫn: **GV. Tấn Nguyễn**
Sinh viên thực hiện: **Nguyễn Ngọc Đỗ**
MSSV: **2301010012**
Lớp: **Công nghệ thông tin K20**

LỜI NÓI ĐẦU

Trong thời đại công nghệ số phát triển mạnh mẽ như hiện nay, việc xây dựng và phát triển các ứng dụng web đóng vai trò quan trọng trong hầu hết các lĩnh vực của đời sống và kinh doanh. Môn học Lập trình Web không chỉ cung cấp những kiến thức nền tảng về HTML, CSS, JavaScript, mà còn trang bị kỹ năng phát triển các trang web động với sự hỗ trợ của các công nghệ phía máy chủ như PHP, ASP.NET, cũng như các cơ sở dữ liệu như MySQL.

Cùng với sự phát triển không ngừng về kỹ thuật máy tính và mạng điện tử, công nghệ thông tin cũng được là những công nghệ đẳng cấp cao và lấn lướt chinh phục hết đỉnh cao này đến những đỉnh cao khác. Mạng Internet là một trong những sản phẩm có giá trị hết sức lớn lao và ngày càng trở nên thành một công cụ không thể thiếu, là nền tảng chính cho sự truyền tải.

Bằng internet, chúng ta đã thực hiện được nhiều công việc với tốc độ nhanh hơn và chi phí thấp hơn nhiều so với cách thức truyền thống. Chính điều này, đã thúc đẩy sự khai sinh và phát triển của thương mại điện tử và chính phủ điện tử trên khắp thế giới, làm biến đổi đáng kể về mặt văn hoá, nâng cao chất lượng cuộc sống con người.

Trong hoạt động sản xuất, kinh doanh, giờ đây, thương mại điện tử đã khẳng định được vai trò xúc tiến và thúc đẩy sự phát triển của doanh nghiệp. Đối với một cửa hàng, việc quảng bá và giới thiệu đến khách hàng các sản phẩm mới đáp ứng được nhu cầu của khách hàng sẽ là cần thiết. Vậy phải quảng bá thế nào một cách hiệu quả? Đó chính là xây dựng được một website cho cửa hàng của mình bán.

Vì vậy, em đã thực đề tài “ XÂY DỰNG WEBSITE THƯƠNG MẠI ĐIỆN TỬ KINH DOANH SẢN PHẨM TÀI LIỆU HỌC TẬP ”

Thông qua bài báo cáo này, em xin trình bày quá trình tìm hiểu, thiết kế và xây dựng một ứng dụng web đơn giản, đồng thời vận dụng các kiến thức đã học để giải quyết các yêu cầu cụ thể. Bài báo cáo cũng là cơ hội để em rèn luyện kỹ năng lập trình, tư duy logic và khả năng tự học, tự nghiên cứu.

Em xin chân thành cảm ơn thầy/cô đã tận tình giảng dạy và hỗ trợ trong suốt quá trình học tập. Mặc dù đã cố gắng hoàn thiện bài báo cáo một cách tốt nhất, nhưng chắc chắn không thể tránh khỏi những thiếu sót. Em rất mong nhận được sự góp ý quý báu của thầy/cô để em có thể rút kinh nghiệm và hoàn thiện hơn trong tương lai.

Mục Lục

Danh mục hình ảnh.....	4
Danh mục bảng biểu.....	5
1. Trang bìa.....	1
2. Lời nói đầu.....	2
3. Mục lục.....	3
4. Chương 1: Cơ sở lý thuyết.....	6
1.1. Tổng quan về lập trình Web.....	6
1.1.1 Giới thiệu về Website.....	6
1.1.2 Thành phần cấu tạo Website.....	6
1.1.3 Hoạt động ứng dụng Web.....	8
1.1.4 Quy trình phát triển Web.....	10
1.1.5 FRONT-END.....	12
1.1.6 BACK-END.....	15
1.1.7 Mô hình Client - Sever Architecture.....	17
1.2. Mô hình MVC.....	18
1.2.1 Giới thiệu về Mô hình MVC.....	18
1.2.2 Nguyên tắc hoạt động của MVC / MÔ HÌNH MVC / VIEW.....	18
1.2.3 Ưu điểm và nhược điểm của mô hình MVC.....	20
1.3. Công nghệ ASP.NET Core.....	20
1.3.1 Giới thiệu về ASP.NET CORE.....	20
1.3.2 ASP.NET CORE.....	22
1.3.3 Lịch sử hình thành và phát triển.....	23
1.3.4 ASP.NET Core và .NET Framework.....	24
1.4. ASP.NET Core MVC.....	25
1.4.1 Khái niệm cơ bản.....	26
1.4.2 Nguyên tắc hoạt động chung của MVC.....	26
1.4.3 Triển khai Controller với ASP.NET CORE.....	26
1.4.4 Nhiệm vụ của Controller.....	27
5. Chương 2: Xây dựng ứng dụng thực tế.....	30
2.1. Phát biểu bài toán.....	30
2.2. Phân tích yêu cầu.....	31
2.3. Thiết kế cơ sở dữ liệu.....	32
2.4. Thiết kế giao diện người dùng.....	35
2.5. Thiết kế thành phần MVC.....	37
2.6. Triển khai và cài đặt.....	39
6. Chương 3: Kết quả chương trình.....	41
3.1. Demo chương trình.....	41
3.2. Ảnh chụp màn hình chương trình hoạt động.....	45
3.3. Đánh giá kết quả.....	47
7. Kết luận.....	49
8. Tài liệu tham khảo.....	50

Danh mục hình ảnh

Hình 1. Sơ đồ hoạt động website.....	5
Hình 2. Mô hình hoạt động ứng dụng website.....	7
Hình 3. Quy trình phát triển website.....	9
Hình 4. vai trò của HTML, CSS và JavaScript trong phát triển web.....	12
Hình 5. Các Framework CSS hàng đầu.....	13
Hình 6. Trách nhiệm của nhà phát triển của nhà phát triển Back-end.....	14
Hình 7. Luồng hoạt động của một yêu cầu HTTP trong kiến trúc web cơ bản.....	16
Hình 8. Mô hình hoạt động của MVC.....	18
Hình 9. So sánh giữa Framework và Library.....	20
Hình 10. Software Framework.....	21
Hình 11. Lịch sử hình thành và phát triển.....	22
Hình 12. Lịch sử phát hành các phiên bản .NET Core từ 1.0 đến 3.1.....	23
Hình 13. Biểu đồ so sánh giữa ASP.NET Core và ASP.NET.....	24
Hình 14. Nguyên tắc hoạt động chung của MVC.....	25
Hình 15. Ví dụ về Controller.....	25
Hình 16. Luồng xử lý yêu cầu HTTP.....	26
Hình 17. Cơ sở dữ liệu.....	31
Hình 18. Tổng quan UseCase hệ thống.....	33
Hình 19. Trang chủ.....	35
Hình 20. Thông tin sách.....	35
Hình 21. Giỏ hàng.....	36
Hình 22. Thông tin sách.....	36
Hình 23. Controllers bán sách.....	37
Hình 24. Models bán sách.....	37
Hình 25. Views bán sách.....	38
Hình 26. Giao diện đăng nhập.....	41
Hình 27. Giao diện đăng ký.....	41
Hình 28. Trang chủ.....	42
Hình 29. Thanh toán.....	42
Hình 30. Giỏ hàng.....	43
Hình 31. Thanh toán.....	43
Hình 32. Đơn hàng.....	44
Hình 33. Trang chủ.....	44
Hình 34. Thông tin sách.....	45
Hình 35. Giỏ hàng.....	45
Hình 36. Thanh toán.....	46
Hình 37. Đơn hàng.....	46

Danh mục bảng biểu

Bảng 1. Quy trình.....	12
Bảng 2. CSS Frameworks.....	14
Bảng 3. So sánh Front-End và Back-End.....	16
Bảng 4. Ý nghĩa tổng thể.....	29
Bảng 5. Bảng NgườiDùng.....	32
Bảng 6. Bảng TàiLieu.....	33
Bảng 7. Bảng DanhMuc.....	33
Bảng 8. Bảng ĐơnHang.....	33
Bảng 9. Bảng ChiTietĐơnHang.....	33
Bảng 10. Bảng TàiKhoan.....	33
Bảng 11. Bảng QTV.....	34

CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

1.1 Tổng quan về lập trình Web

1.1.1 Giới thiệu về Website

Khái niệm:

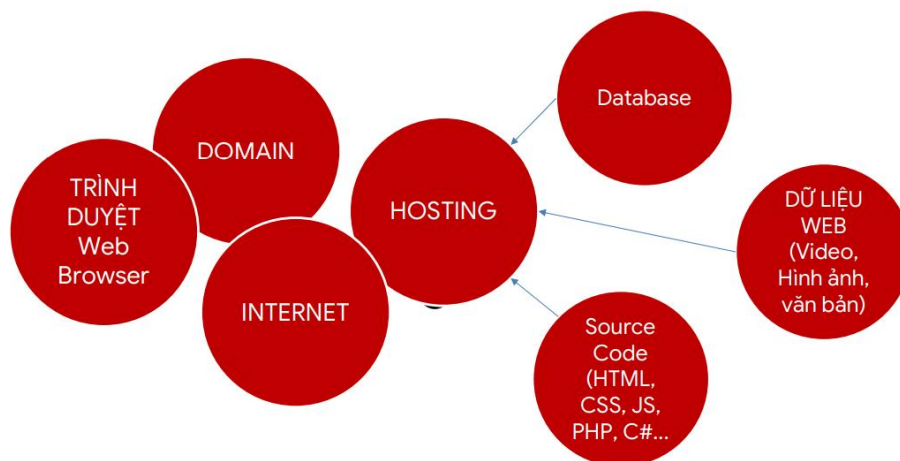
- Website là tập hợp các trang mạng chứa nội dung như văn bản, hình ảnh, âm thanh, video, được lưu trữ trên máy chủ và truy cập qua Internet.
- Được lưu trữ trực tuyến trên các máy chủ và có thể được truy cập bởi bất cứ ai, từ bất cứ đâu thông qua mạng Internet.

Phân biệt Website và Web Application:

- **Website:** Tĩnh, chủ yếu hiển thị thông tin (ví dụ: trang tin tức).
- **Web Application:** Động, có tính tương tác cao (ví dụ: ứng dụng quản lý công việc).

1.1.2 Thành phần cấu tạo Website

Thành phần cấu tạo:



Hình 1. Sơ đồ hoạt động website

Các thành phần trong ảnh:

TRÌNH DUYỆT (Web Browser)

- Là nơi người dùng nhập địa chỉ website (ví dụ: Chrome, Firefox).
- Gửi yêu cầu qua Internet để truy cập website.

DOMAIN (Tên miền)

- Ví dụ: www.example.com
- Là địa chỉ website, ánh xạ đến địa chỉ IP của máy chủ hosting.

INTERNET

- Kết nối trung gian giữa người dùng và hệ thống máy chủ.
- Chuyển yêu cầu từ trình duyệt đến Hosting và trả kết quả về.

HOSTING

- Là nơi lưu trữ toàn bộ dữ liệu, mã nguồn của website.
- Đây là trung tâm xử lý, nhận yêu cầu và gửi lại dữ liệu cho người dùng.

Database

- Cơ sở dữ liệu lưu trữ thông tin như tài khoản người dùng, sản phẩm, bài viết...
- Hosting sẽ truy cập Database khi cần lấy/ghi dữ liệu.

Source Code (HTML, CSS, JS, PHP, C#...)

- Là phần mã nguồn của website.
- Được lưu trữ trong Hosting, bao gồm cả frontend và backend.

Dữ liệu Web (Video, Hình ảnh, Văn bản)

- Tài nguyên tĩnh của trang web như ảnh, video, file văn bản...
- Được lưu trong hosting và cung cấp cho trình duyệt khi cần.

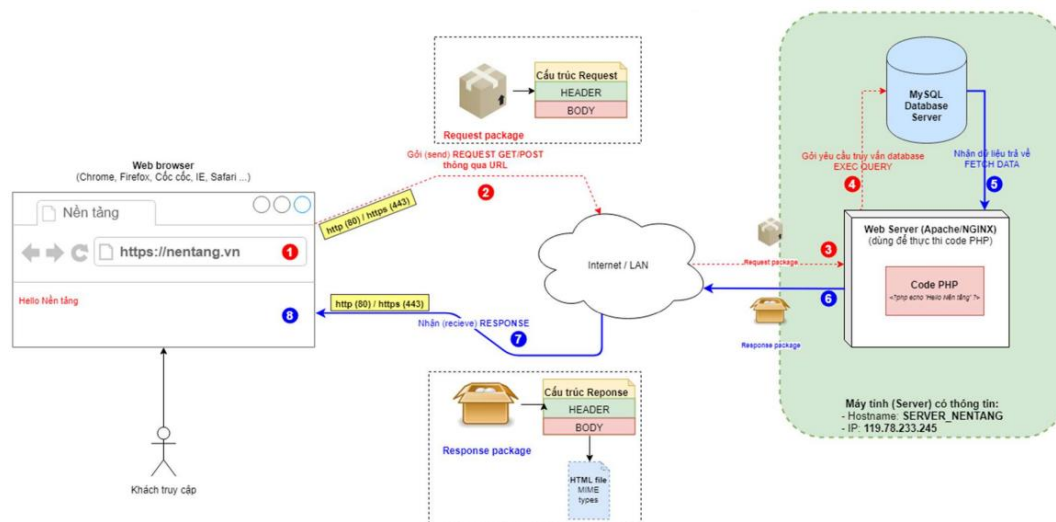
Phân loại Website

Website được phân làm 4 loại:

- **Theo cấu trúc:**
 - Web tĩnh
 - Web động
- **Theo sở hữu:**

- Doanh nghiệp
- Cá nhân
- **Theo chức năng:**
 - Tin tức
 - Bán hàng
 - Mạng xã hội
 - Lưu trữ...
- **Theo lĩnh vực:**
 - Du lịch
 - Xây dựng, giáo dục
 - Thời trang
 - Mỹ phẩm
 - Việc làm...

1.1.3 Hoạt động ứng dụng Web



Hình 2. Mô hình hoạt động ứng dụng website

1. Giới thiệu tổng quan

Hệ thống web được thiết kế dựa trên nền tảng <https://mentang.vn>, hoạt động theo mô hình Client-Server, bao gồm các thành phần chính:

- Front-end: Trình duyệt web (Chrome, Firefox, Safari, IE...)
- Back-end: Web Server (Apache/NGINX), cơ sở dữ liệu MySQL
- Hạ tầng mạng: Internet/LAN, Gateway

2. Quy trình hoạt động

Phía người dùng (Client)

- Người dùng truy cập hệ thống thông qua trình duyệt với giao diện "Hello Nền tảng".
- Hệ thống phân quyền người dùng dưới dạng "Khách truy cập".
- Các thao tác nhập liệu (bàn phím) được xử lý thông qua Keyboard Gateway.

Gửi yêu cầu (Request)

- Trình duyệt đóng gói yêu cầu thành Request Package, bao gồm:
- Request Header: Thông tin phiên, phương thức (GET/POST).
- Request Body: Dữ liệu gửi kèm (nếu có).
- Yêu cầu được gửi đến máy chủ thông qua URL (ví dụ: <https://mentang.vn>).

Xử lý phía máy chủ (Server)

- Web Server (Apache/NGINX) nhận request với thông tin:
 - IP Server: 119.78.23.245
 - Hostname: SERVICE_NENFANG
- Code PHP được thực thi để xử lý nghiệp vụ.
- MySQL Database truy vấn dữ liệu bằng các lệnh SQL (EXEC, QUERY).

Trả về kết quả (Response)

- Sau khi xử lý, server tạo Response Package gồm:
 - Response Header: Trạng thái (200, 404...), loại dữ liệu (HTML, JSON...).
 - Response Body: Nội dung phản hồi (trang web, dữ liệu, thông báo lỗi).
- Dữ liệu trả về có thể là HTML, link, hoặc các định dạng khác.

Hiển thị kết quả trên trình duyệt

Client nhận Response và hiển thị thông tin cho người dùng.

Nếu có lỗi, hệ thống thông báo (vd: "your error video with any").

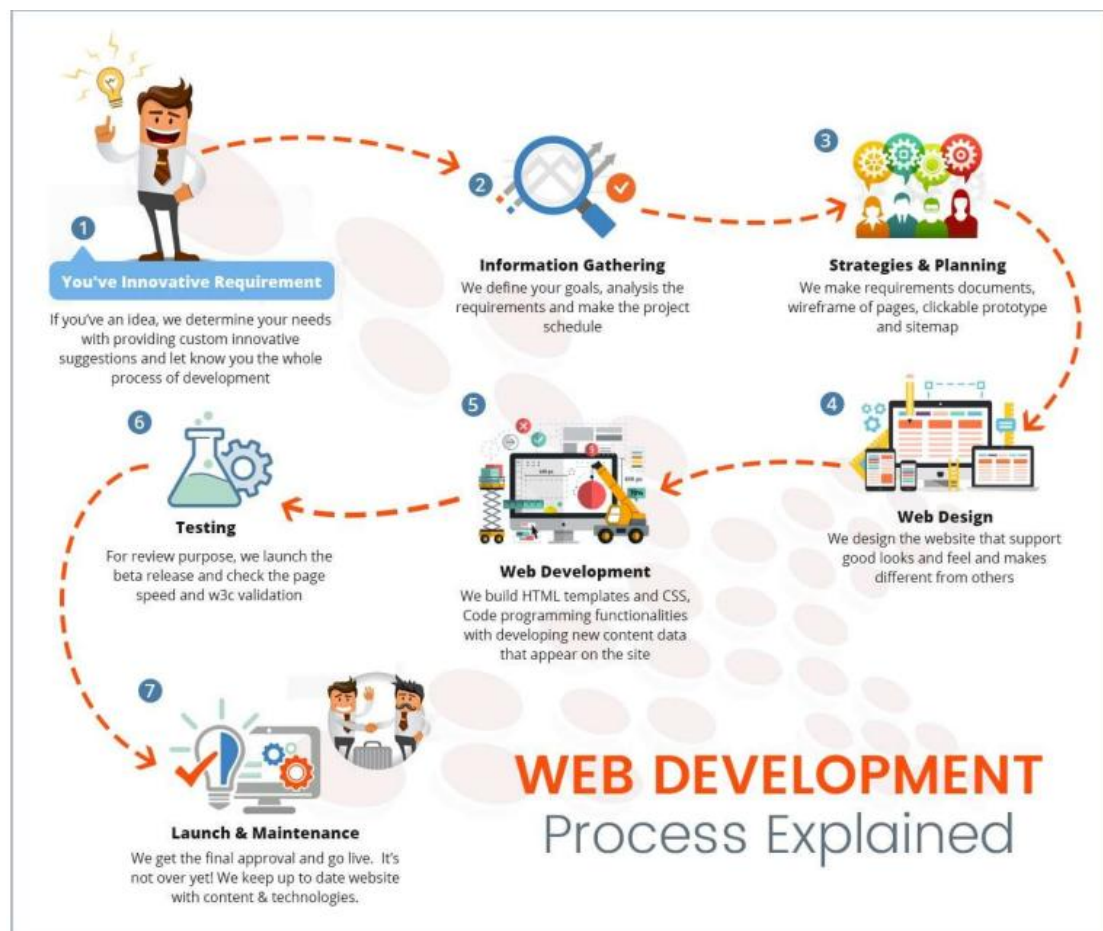
Các thành phần hỗ trợ

- Internet/LAN: Đảm bảo kết nối giữa Client và Server.
- Google Play: Có thể liên quan đến nền tảng phân phối ứng dụng di động.

3. Kết luận

- Hệ thống hoạt động theo mô hình Request-Response tiêu chuẩn:
 - Người dùng gửi yêu cầu từ trình duyệt.
 - Máy chủ xử lý và truy vấn database.
 - Kết quả được trả về và hiển thị trên giao diện người dùng.

1.1.4 Quy trình phát triển Web



Hình 3. Quy trình phát triển website

1. Các Giai Đoạn Phát Triển:

Thu Thập Thông Tin (Information Gathering):

- Mục tiêu: Xác định yêu cầu, mục đích của website.
- Hoạt động chính:
 - Phân tích nhu cầu khách hàng.
 - Lập kế hoạch dự án (thời gian, nhân lực, ngân sách).

Lập Chiến Lược & Kế Hoạch (Strategies & Planning)

- Mục tiêu: Xây dựng tài liệu định hướng phát triển.
- Hoạt động chính:
 - Tạo tài liệu yêu cầu (Requirement Document).
 - Thiết kế wireframe, prototype clickable.
 - Xây dựng sơ đồ trang (Sitemap).

Thiết Kế Web (Web Design)

- Mục tiêu: Tạo giao diện trực quan, khác biệt.
- Hoạt động chính:
 - Thiết kế UI/UX, đảm bảo tính thẩm mỹ và trải nghiệm người dùng.
 - Phê duyệt bản thiết kế trước khi chuyển sang giai đoạn lập trình.

Phát Triển Web (Web Development)

- Mục tiêu: Xây dựng website hoàn chỉnh.
- Hoạt động chính:
 - Chuyển đổi thiết kế thành HTML/CSS.
 - Lập trình chức năng (backend/frontend).
 - Phát triển nội dung hiển thị trên website.

Kiểm Thử (Testing)

- Mục tiêu: Đảm bảo chất lượng trước khi ra mắt.
- Hoạt động chính:
 - Phát hành bản beta để đánh giá.
 - Kiểm tra tốc độ tải trang, tương thích trình duyệt.
 - Xác thực chuẩn 4G Validation (tối ưu hiệu suất trên mạng di động).

Ra Mắt & Bảo Trì (Launch & Maintenance)

- Mục tiêu: Vận hành ổn định và cập nhật lâu dài.
- Hoạt động chính:
 - Triển khai chính thức (Go Live) sau khi được phê duyệt.
 - Bảo trì định kỳ: Cập nhật nội dung, công nghệ, vá lỗi.

Sơ Đồ Quy Trình

1. Thu Thập Thông Tin → 2. Lập Kế Hoạch → 3. Thiết Kế → 4. Phát Triển → 5. Kiểm Thử → 6. Ra Mắt & Bảo Trì
--

Bảng 1. Quy trình

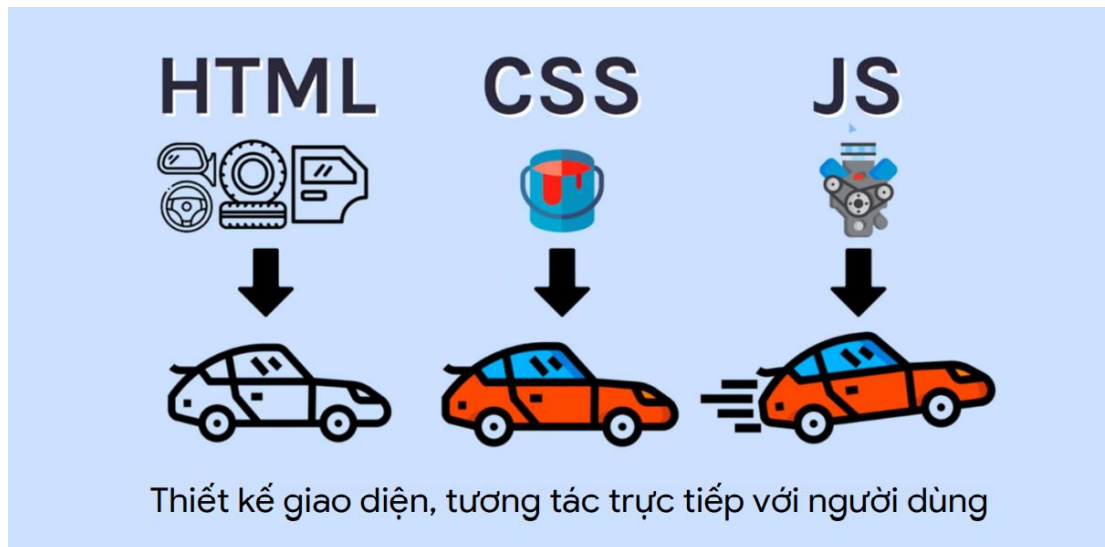
2. Kết Luận

Quy trình phát triển website là một chuỗi các bước được tổ chức chặt chẽ, đảm bảo:

- Tính sáng tạo: Đề xuất giải pháp tùy chỉnh theo nhu cầu.
- Tính khả thi: Kiểm thử kỹ lưỡng trước khi triển khai.
- Tính bền vững: Duy trì hoạt động ổn định sau khi ra mắt.

1.1.5 FRONT-END

1. Khái niệm Front-End Development:



Hình 4. vai trò của HTML, CSS và JavaScript trong phát triển web

2. Định nghĩa:

Front-end (phía người dùng) là phần giao diện và trải nghiệm tương tác trực tiếp với người dùng trên trình duyệt web hoặc ứng dụng di động. Nó bao gồm:

- HTML (HyperText Markup Language): Xây dựng cấu trúc nội dung.
- CSS (Cascading Style Sheets): Định dạng giao diện (màu sắc, bố cục, hiệu ứng).
- JavaScript (JS): Xử lý tương tác động (click, cuộn, tải dữ liệu không đồng bộ).

3. Vai trò của Front-End Developer

- Chuyển thiết kế (UI/UX) thành code hoạt động.
- Tối ưu hiệu suất và khả năng tương thích trên nhiều trình duyệt.
- Kết nối với back-end thông qua API để hiển thị dữ liệu.

4. Các Framework CSS & Front-End Phổ Biến



Hình 5. Các Framework CSS hàng đầu

5. CSS Frameworks

Giúp phát triển giao diện nhanh chóng với các thành phần được thiết kế sẵn:

Framework	Đặc điểm nổi bật	Mức độ phổ biến
Bootstrap	Hệ thống lưới mạnh, component đa dạng	★ ★ ★ ★ ★
Tailwind CSS	Utility-first, tùy chỉnh cao	★ ★ ★ ★
Bulma	Dựa trên Flexbox, không cần JS	★ ★ ★
Foundation	Linh hoạt, phù hợp cho responsive design	★ ★ ★
Materialize	Phong cách Material Design của Google	★ ★
Semantic UI	Ngữ nghĩa rõ ràng, dễ đọc code	★ ★
Pure CSS	Nhẹ ($\approx 3.5\text{KB}$), tập trung vào minimalism	★

Bảng 2. CSS Frameworks

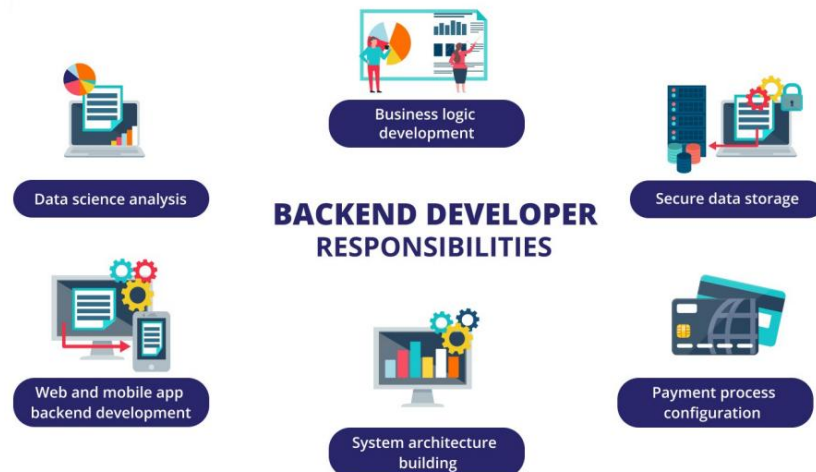
6. JavaScript Frameworks

Dùng để xây dựng ứng dụng web động:

- React.js (Facebook): Component-based, virtual DOM.
- Vue.js:
 - Dễ học, tích hợp linh hoạt.
 - Hỗ trợ cả Options API và Composition API.

- Angular (Google):
 - Framework hoàn chỉnh (MVC).
 - Sử dụng TypeScript mặc định.

1.1.6 BACK-END



Hình 6. Trách nhiệm của Nhà phát triển Back-end

1. Định Nghĩa Back-End

Back-end (phía máy chủ) là phần xử lý logic nghiệp vụ, quản lý dữ liệu và giao tiếp với front-end thông qua API. Nó bao gồm:

- Máy chủ (Server): Xử lý yêu cầu từ người dùng.
- Cơ sở dữ liệu (Database): Lưu trữ thông tin.
- Ứng dụng back-end: Code để vận hành server và kết nối database.

2. Thành Phần Chính

Server-Side Programming

- Ngôn ngữ lập trình:
 - PHP, Python (Django, Flask), Java (Spring), C# (.NET), Ruby (Ruby on Rails), Node.js (JavaScript).
- Nhiệm vụ:

- Xử lý logic (đăng nhập, thanh toán, bảo mật).
- Giao tiếp với database.
- Trả dữ liệu dạng JSON/XML cho front-end.

Database

- Hệ quản trị cơ sở dữ liệu (DBMS):
 - SQL: MySQL, PostgreSQL, SQL Server (dữ liệu có cấu trúc).
 - NoSQL: MongoDB, Firebase (dữ liệu linh hoạt).

API (Application Programming Interface)

- RESTful API hoặc GraphQL: Kết nối front-end và back-end.

Ví dụ: Khi bạn đăng nhập, front-end gửi request đến back-end để kiểm tra thông tin.

3. So Sánh Front-End và Back-End

Tiêu chí	Front-End	Back-End
Ngôn ngữ	HTML, CSS, JavaScript	PHP, Python, Java, C#, Ruby
Công việc	Giao diện người dùng	Logic nghiệp vụ, database
Framework	React, Vue, Angular	Django, Spring, Laravel
Tương tác	Trực tiếp với người dùng	Ẩn sau máy chủ

Bảng 3. So Sánh Front-End và Back-End

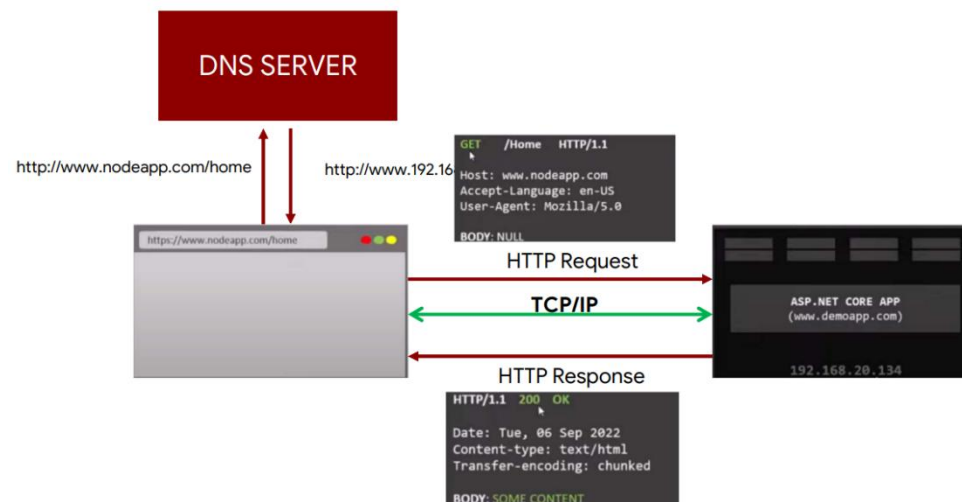
4. Công Cụ và Công Nghệ Back-End Phổ Biến

- Web Servers: Apache, Nginx.
- Authentication: JWT, OAuth.
- DevOps: Docker, Kubernetes (triển khai ứng dụng).

Tại Sao Back-End Quan Trọng?

- Xử lý dữ liệu phức tạp: Ví dụ: thuật toán gợi ý sản phẩm.
- Bảo mật: Bảo vệ thông tin người dùng (mật khẩu, giao dịch).
- Hiệu suất: Tối ưu tốc độ phản hồi từ server.

1.1.7 Mô hình Client - Sever Architecture



Hình 7. Luồng hoạt động của một yêu cầu HTTP trong kiến trúc web cơ bản

1. Khái Niệm Cơ Bản

- Mô hình Client-Server là kiến trúc phân tách hệ thống thành 2 thành phần chính:
- Client (Máy khách): Gửi yêu cầu (request) và hiển thị kết quả.
- Server (Máy chủ): Tiếp nhận, xử lý yêu cầu và trả về phản hồi (response).

2. Ưu Điểm Của Mô Hình

- Khả năng mở rộng: Nhiều client có thể kết nối đến cùng server.
- Bảo mật: Server quản lý tập trung dữ liệu nhạy cảm.
- Hiệu suất: Tài nguyên server được tối ưu cho xử lý phức tạp.

3. Giao thức TCP/IP Protocol

- Là giao thức truyền thông định nghĩa cách thức truyền dữ liệu trên mạng internet
- Nhiệm vụ giao thức TCP là phân mảnh các gói Request và Response thành các gói nhỏ hơn gọi là Packets, sau đó gửi đi.
- Khi một trong những gói này đến đích đến, TCP sẽ ráp lại những gói Packet này thành các gói

- Request và Response như ban đầu. Quá trình kết thúc khi tất cả các gói packet đến đúng đích.
- Nhiệm vụ của IP là gửi và định tuyến những gói Packet trên môi trường Internet dựa trên địa chỉ IP của mỗi gói Packet

4. Giao thức HTTP (HyperText Transfer Protocol)

- Giao thức HTTP (HyperText Transfer Protocol)
- Là giao thức được dùng để truyền tải dữ liệu dạng HTML, XML trên môi trường mạng (WorldWide Web)
- Là giao thức không trạng thái: 1 thao tác chỉ gồm 1 yêu cầu (Request) và 1 đáp ứng yêu cầu đó(Response).
- Khi một trình duyệt (client) kết nối tới một web server nó sẽ gửi một HTTP Request tới webserver
- Web Server sau khi nhận và xử lý yêu cầu sẽ gửi một HTTP Response chứa nội dung truy vấn –đáp ứng lại cho Client

1.2 Mô hình MVC

1.2.1 Giới thiệu về Mô hình MVC

- Là kiểu kiến trúc phần mềm phân tách source code của ứng dụng phần mềm thành ba thành phần M – V - C.
- Viết tắt của cụm từ **Model – View – Controller** trong đó:

Model: mô tả dữ liệu, logic nghiệp vụ và thao tác với dữ liệu

View: Khuôn mẫu giao diện người dùng để hiển thị kết quả truy vấn dữ liệu

Controller: Xử lý logic và tương tác model & lựa chọn view

- Cải thiện tính tổ chức, dễ bảo trì và mở rộng ứng dụng.

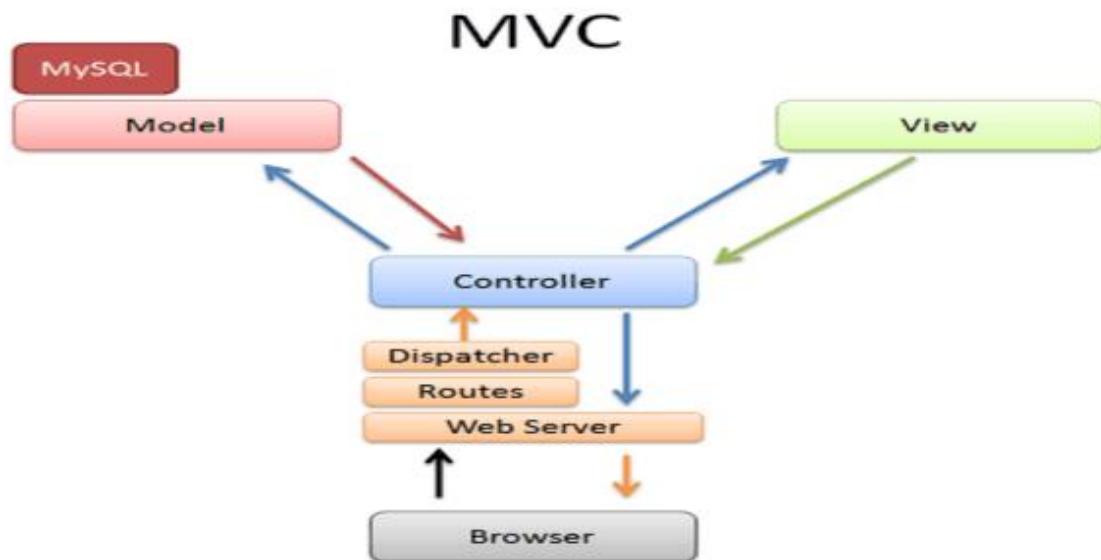
- Các phần được kết nối với nhau và mỗi thành phần đều có một nhiệm vụ riêng và độc lập với các thành phần khác.

1.2.2 Nguyên tắc hoạt động của MVC / MÔ HÌNH MVC / VIEW

- **Model:** Đây là bộ phận có nhiệm vụ lưu trữ toàn bộ dữ liệu của ứng dụng. Model cũng là bộ phận kết nối giữa hai bộ phận còn lại là View và Controller. Nó được thể hiện dưới hình thức là một CSDL hay chỉ là một

file XML bình thường. Các thao tác với CSDL như cho phép xem, xử lý hay truy xuất dữ liệu sẽ được Model được thể hiện rõ.

- **View:** Thành phần này được dùng cho tất cả các logic UI của ứng dụng. View có nhiệm vụ hiển thị thông tin, tương tác với người dùng. Nó là nơi chứa tất cả những đối tượng GUI như textbox, images,... View cũng chính là tập hợp các form hoặc file HTML. Các ứng dụng web sử dụng nó như một thành phần của hệ thống hoặc các thành phần HTML được tạo ra. View còn ghi nhận hoạt động của người dùng để tương tác với Controller.
- **Controller:** Đây chính là bộ phận có chức năng xử lý những từ yêu cầu người dùng đưa đến qua View. Controller sẽ từ đó đưa ra dữ liệu phù hợp với người dùng. Ngoài ra, Controller còn có thêm chức năng là kết nối với bộ phận Model.



Hình 8. Mô hình hoạt động của MVC

- Bước 1: Người dùng gửi yêu cầu (HTTP Request) qua một trình duyệt web bất kỳ. Yêu cầu này có thể kèm theo những dữ liệu nhập tới Controller xử lý. Bộ Routing điều hướng sẽ xác định Controller xử lý.
- Bước 2: Khi Controller nhận được yêu cầu, nó sẽ kiểm tra yêu cầu đó có cần dữ liệu từ Model không. Nếu có, nó sẽ dùng các class/function trong Model sau đó trả ra kết quả. Khi đó, Controller cũng sẽ xử lý các giá trị đó và trả ra View để hiển thị. Controller sẽ xác định các View tương ứng và hiển thị đúng theo yêu cầu.
- Bước 3: Khi View nhận được dữ liệu từ Controller, chúng sẽ xây dựng các thành phần hiển thị như hình ảnh, thông tin dữ liệu,... rồi trả về GUI content để Controller đưa ra kết quả và hiển thị lên màn hình Browser.

- Bước 4: Browser sẽ nhận được giá trị trả về và hiển thị với người dùng và kết thúc quá trình hoạt động.

1.2.3 Ưu điểm và nhược điểm của mô hình MVC:

1. Ưu điểm của MVC

Mô hình MVC có nhiều ưu điểm quan trọng. Dưới đây là một số lợi ích của sử dụng mô hình MVC:

- **Tách biệt logic:** MVC giúp tách biệt rõ ràng giữa logic xử lý dữ liệu (Model), hiển thị dữ liệu (View) và quản lý tương tác (Controller). Điều này làm cho mã nguồn dễ đọc, bảo trì và phát triển.
- **Tính linh hoạt:** Với sự phân chia rõ ràng, bạn có thể thay đổi hoặc mở rộng một thành phần trong Model mà không ảnh hưởng đến các thành phần khác. Bạn có thể thay đổi View hoặc thay đổi cách xử lý dữ liệu trong Model mà không cần sửa đổi toàn bộ hệ thống.
- **Tái sử dụng mã:** Do có sự tách biệt rõ ràng giữa các thành phần, bạn có thể tái sử dụng mã một cách dễ dàng. Ví dụ, bạn có thể sử dụng lại Model hoặc View trong các phần mềm khác nhau mà không cần viết lại từ đầu.
- **Phân công công việc:** MVC cho phép phân chia công việc giữa các thành viên trong nhóm phát triển. Nhà thiết kế giao diện có thể làm việc độc lập với nhà phát triển Model, giúp tăng hiệu suất làm việc và chất lượng sản phẩm.
- **Tiết kiệm băng thông:** Vì không sử dụng viewstate nên mô hình MVC khá tiết kiệm băng thông. Việc tiết kiệm băng thông giúp website hoạt động ổn định hơn.

2. Nhược điểm của MVC

- Đối với mô hình có tính phân tích cao như MVC thì phù hợp trong các dự án lớn. Việc ứng dụng mô hình MVC trong các dự án nhỏ sẽ dễ bị chồng chéo, tốn kém nguồn nhân lực khi phát triển.

1.3. Công nghệ ASP.NET Core

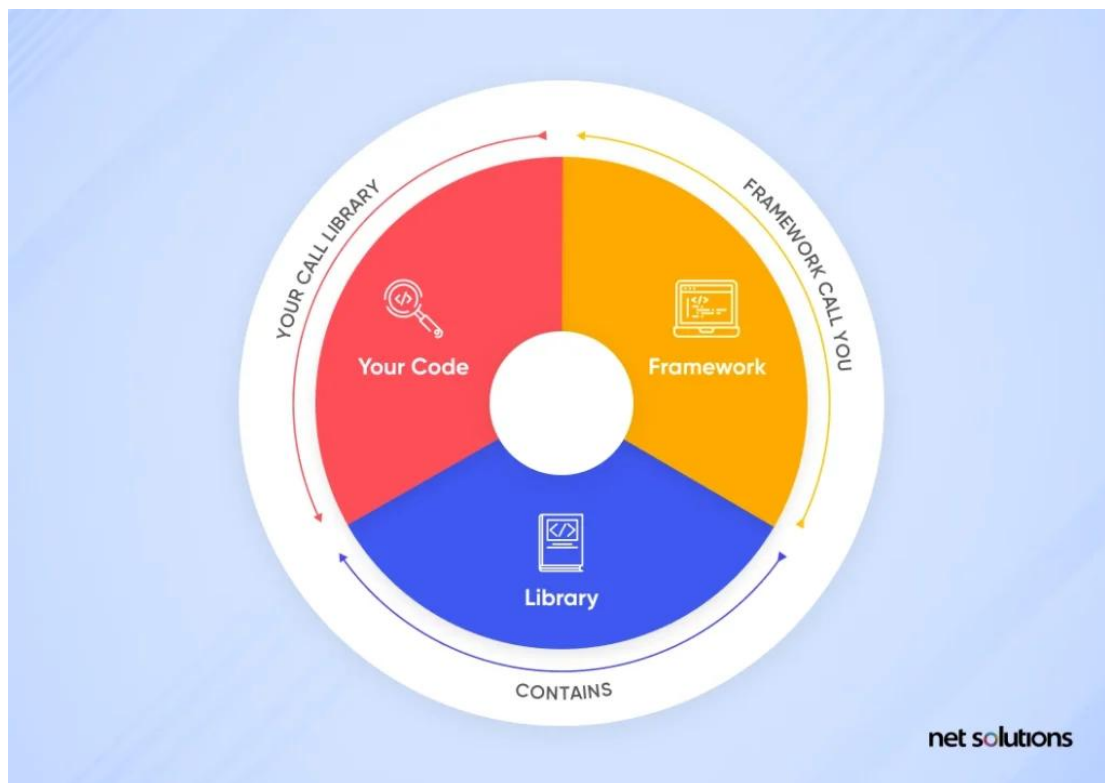
1.3.1 Giới thiệu về ASP.NET CORE

1. Framework là gì?

- Định nghĩa từ điển của framework là “cấu trúc cơ bản nằm bên dưới một hệ thống” không khác nhiều so với framework trong lập trình. Cho dù hệ

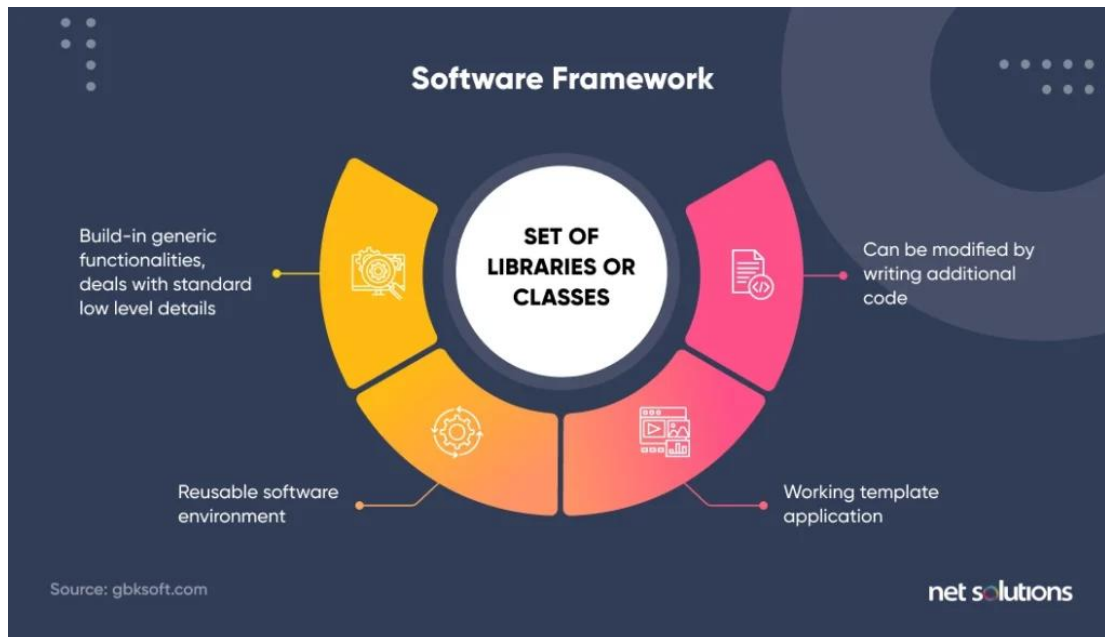
thống là một ngôi nhà, một chiếc ô tô, một lý thuyết hay một ứng dụng di động, thì khái niệm về framework vẫn như vậy: nó cung cấp sự hỗ trợ và 'hướng dẫn' thiết yếu cho cấu trúc đang được xây dựng.

- Một khuôn khổ trong lập trình là một công cụ cung cấp các thành phần hoặc giải pháp có sẵn được tùy chỉnh để tăng tốc quá trình phát triển. Một khuôn khổ có thể bao gồm một thư viện nhưng được xác định theo nguyên tắc đảo ngược điều khiển (IoC). Với lập trình truyền thống, mã tùy chỉnh gọi vào thư viện để truy cập mã có thể sử dụng lại. Với IoC, khuôn khổ gọi các đoạn mã tùy chỉnh khi cần thiết.



Hình 9. So sánh giữa Framework và Library

- Một khuôn khổ có thể bao gồm các chương trình hỗ trợ, trình biên dịch, thư viện mã, bộ công cụ và API để phát triển phần mềm và tạo hệ thống. Các khuôn khổ nguồn mở liên tục được cập nhật và cải tiến.



Hình 10. Software Framework

1.3.2 ASP.NET CORE

ASP.NET CORE là một nền tảng mã nguồn mở và đa nền tảng (cross-platform) do Microsoft phát triển, dùng để xây dựng các ứng dụng web hiện đại, dịch vụ web, và các ứng dụng kết nối đám mây.

1. Một số đặc điểm chính của ASP.NET Core:

Mã nguồn mở và đa nền tảng:

- ASP.NET Core có thể chạy trên Windows, macOS và Linux, giúp lập trình viên phát triển ứng dụng trên nhiều hệ điều hành khác nhau.

Hiệu suất cao:

- ASP.NET Core được thiết kế để tối ưu hóa hiệu suất, giúp các ứng dụng chạy nhanh và hiệu quả hơn.

Kiến trúc module:

- ASP.NET Core sử dụng kiến trúc module, cho phép bạn chỉ cần thêm các thành phần cần thiết vào ứng dụng của mình, giúp giảm thiểu tài nguyên và chi phí phát triển.

Tích hợp tốt với các công nghệ hiện đại:

- Hỗ trợ tích hợp với các framework phía client như Angular, React và Blazor, giúp xây dựng các ứng dụng web hiện đại và tương tác cao.

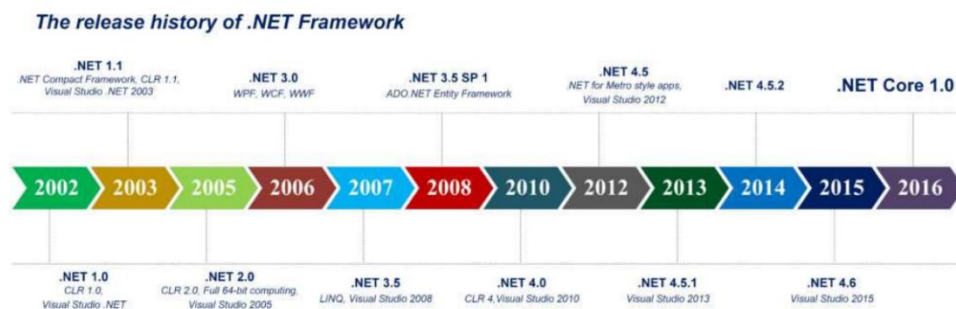
Dependency Injection:

- ASP.NET Core có sẵn hệ thống Dependency Injection, giúp quản lý các phụ thuộc trong ứng dụng một cách dễ dàng và hiệu quả.

Hỗ trợ phát triển và triển khai trên đám mây:

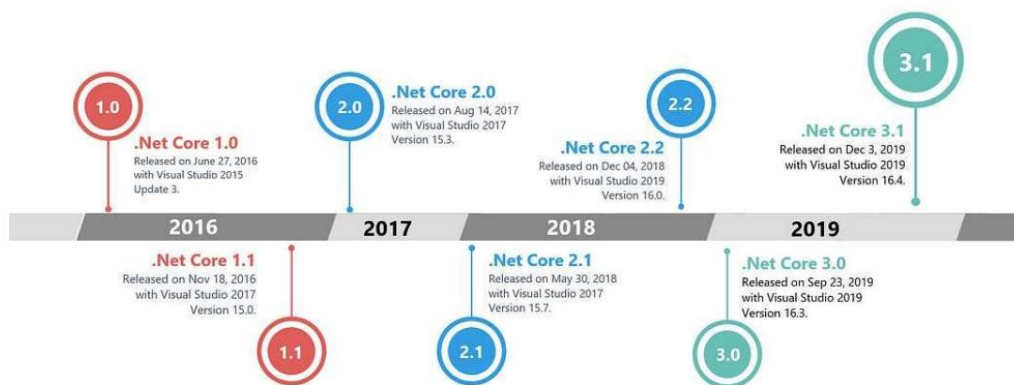
- ASP.NET Core được tối ưu hóa để triển khai trên các dịch vụ đám mây như Microsoft Azure, giúp dễ dàng mở rộng và quản lý ứng dụng.

1.3.3 Lịch sử hình thành và phát triển



Hình 11. Lịch sử hình thành và phát triển

- .NET Platform là một nền tảng thống nhất phát triển nhiều loại ứng dụng từ Mobile, Desktop cho đến Web được phát triển bởi Microsoft.
- .NET Framework được Microsoft đưa ra chính thức từ năm 2002. .NET Framework chỉ hoạt động trên Windows. Những nền tảng ứng dụng như WPF, Winforms, ASP.NET hoạt động dựa trên .NET Framework.
- Mono là phiên bản cộng đồng nhằm mang .NET đến những nền tảng ngoài Windows. Mono được phát triển chủ yếu nhằm xây dựng những ứng dụng với giao diện người dùng và được sử dụng rất rộng rãi: UnityGame, Xamarin...
- Cho đến năm 2013, Microsoft định hướng đi đa nền tảng và phát triển .NET core. .NET core hiện được sử dụng trong các ứng dụng Universal Windows platform và ASP.NET Core. Từ đây, C# có thể được sử dụng để phát triển các loại ứng dụng đa nền tảng trên các hệ điều hành khác nhau (Windows, Linux, MacOS,)



Hình 12. lịch sử phát hành của các phiên bản .NET Core từ 1.0 đến 3.1

ASP.NET CORE được thiết kế nhằm đáp ứng:

- Phát triển và hoạt động đa nền tảng
- Có kiến trúc dựa trên module
- Phát triển hoàn toàn ở dạng mã nguồn mở
- Phù hợp với xu hướng hiện đại của ứng dụng web
- Hoạt động trên Windows, macOS, Linux

1.3.4 ASP.NET Core và .NET Framework

- Để hiểu vấn đề này cần hình dung .NET Framework (và cả .NET Core) theo hai khía cạnh: (1) hệ thống thư viện API hỗ trợ phát triển ứng dụng; (2) runtime dành cho thực thi ứng dụng.

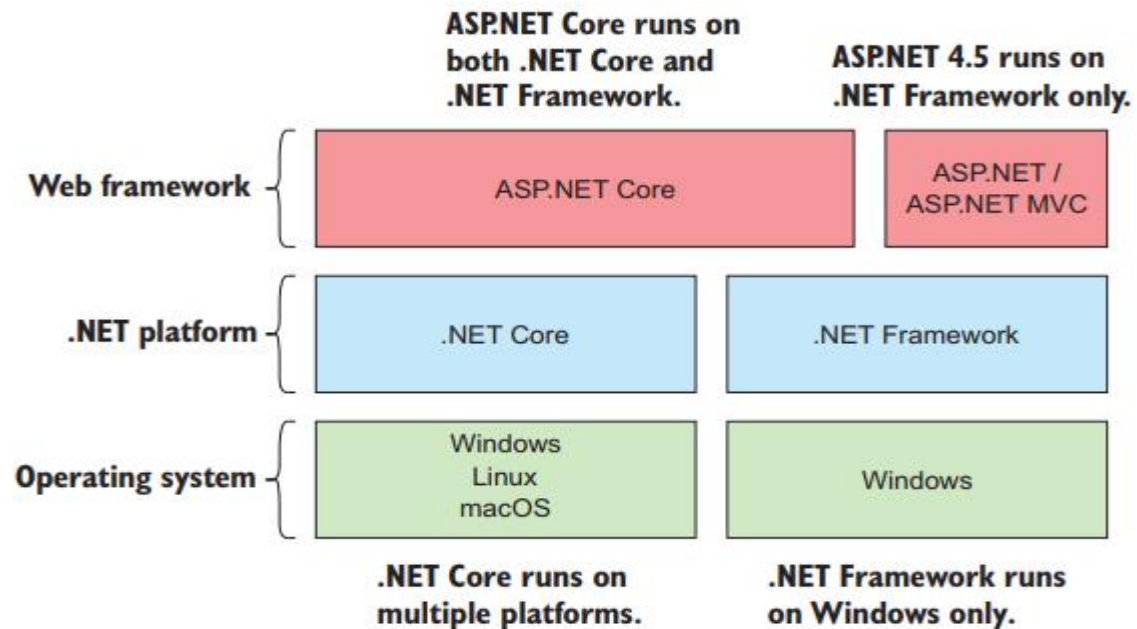
- ASP.NET Core chứa hệ thống API của riêng nó. Hệ thống API này sử dụng các API cơ bản của .NET. Thêm vào đó, .NET Core và .NET Framework có chung hệ thống API cơ bản.

- Runtime có thể hình dung như chương trình máy ảo sẽ nạp ứng dụng vào để thực thi. Ứng dụng và tất cả các thư viện của cả .NET Core và .NET Framework đều nằm ở dạng mã trung gian IL (Intermediate Language).

- Hai yếu tố trên cho phép chương trình viết bằng ASP.NET Core có thể hoạt động trên runtime (máy ảo) của .NET Framework. Ở chiều ngược lại, ASP.NET truyền thống không thể hoạt động trên .NET Core do nó phụ thuộc vào System.Web.dll của .NET Framework, vốn không có trong .NET Core.

- ASP.NET Core 2.0 tới 2.2 có thể chạy trên .NET Framework 4.6.1 (và các phiên bản cao hơn), đồng thời có thể chạy trên .NET Core 2.0 (và các bản cao hơn). Tuy nhiên ASP.NET Core 3.0 chỉ chạy trên .NET Core 3.0.

- Khi chương trình ASP.NET Core thực thi trên runtime của .NET Framework, nó sẽ gắn chặt với Windows và IIS. Do đó nó mất đi những ưu thế của .NET Core. Bù lại, nó được hưởng lợi thế từ thư viện .NET Framework.
- Mối quan hệ giữa ASP.NET Core với .NET Core và .NET Framework được minh họa



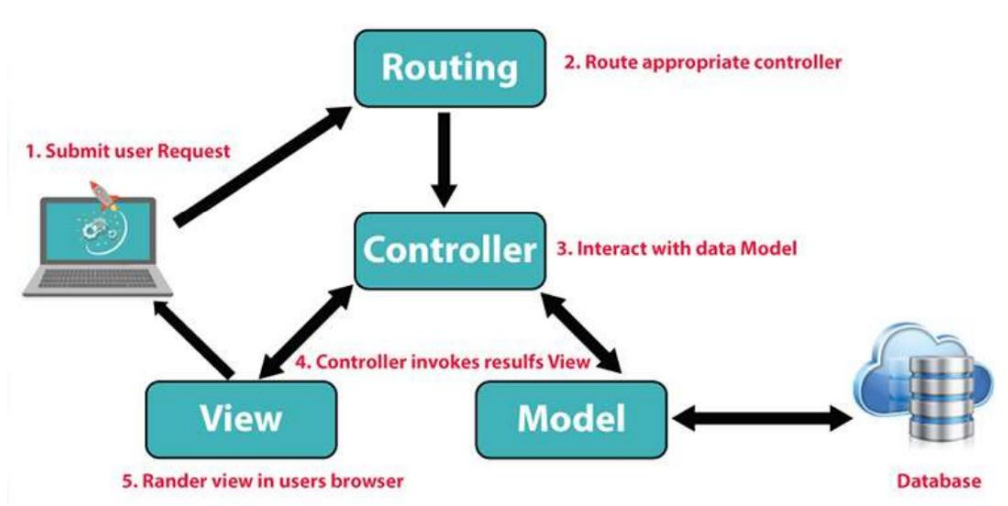
Hình 13. biểu đồ so sánh giữa ASP.NET Core và ASP.NET

1.4 ASP.NET Core MVC

1.4.1 Khái niệm cơ bản:

ASP.NET Core MVC là một framework phát triển web được Microsoft xây dựng dựa trên kiến trúc Model-View-Controller (MVC), giúp tạo ứng dụng web động, bảo trì dễ dàng và có khả năng mở rộng cao. Nó là phiên bản hiện đại và tối ưu hơn so với ASP.NET MVC truyền thống, chạy đa nền tảng (Windows, Linux, macOS) và tích hợp với .NET Core/.NET 5+.

1.4.2 Nguyên tắc hoạt động chung của MVC / ROUTING PROCESS IN ASP.NET CORE

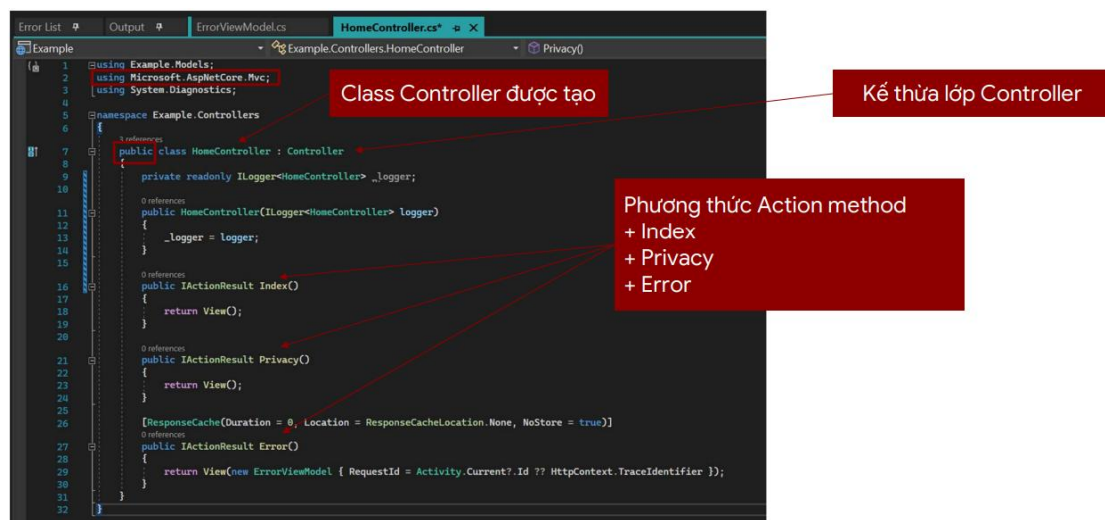


Hình 14. Nguyên tắc hoạt động chung của MVC

1.4.3 Triển khai Controller với ASP.NET CORE

- Để triển khai Controller trong ASP.NET cần tạo một CLASS chứa một hay nhiều phương thức gọi là ACTION Method.
- Được lưu với định dạng .cs
- Mỗi phương thức Action method của mỗi Controller sẽ thực thi một request xác định.

Ví dụ:



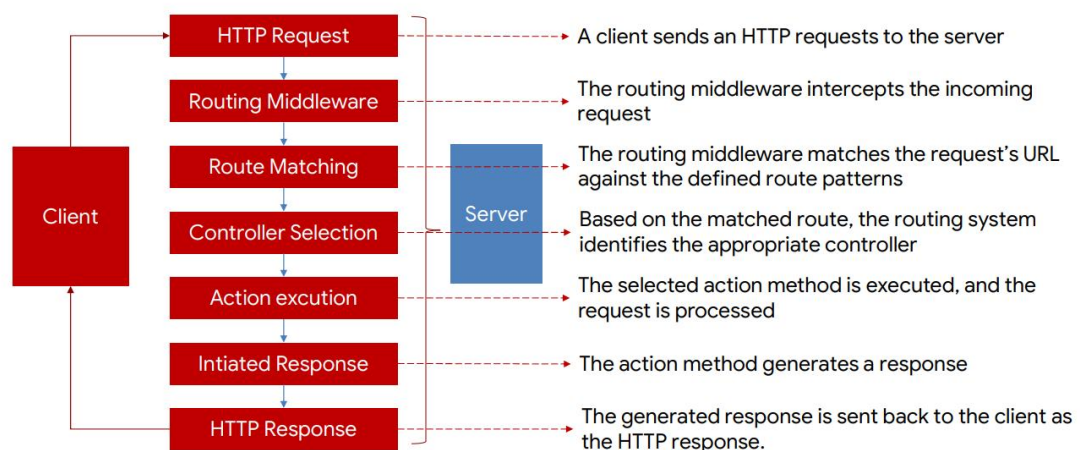
Hình 15. Ví dụ về Controller

Đối với Đối tượng Student có các thao tác:

- + Thêm thông tin một Student
 - + Chỉnh sửa thông tin một Student cụ thể
 - + Liệt kê thông tin tất cả Student
 - + Liệt kê chi tiết thông tin của một Student cụ thể
 - + Xóa thông tin của một Student cụ thể
 - + Xóa tất cả thông tin của Student
- Tất cả các hành động thao tác với Student gọi là Action của Controller

1.4.4 Nhiệm vụ của Controller:

- Xác định Action method sẽ được gọi để thực thi.
- Nhận giá trị từ URL sau đó truyền các giá trị này vào các đối số của Action method.
- Xử lý lỗi có thể xảy ra trong suốt thực thi các Action method.
- Cung cấp lớp cơ chế ViewEngine mặc định cho việc render View



Hình 16. Luồng xử lý yêu cầu HTTP

1 Client gửi HTTP Request

- Một trình duyệt hoặc ứng dụng gửi yêu cầu (GET, POST, PUT, DELETE...) tới server thông qua HTTP.
- Ví dụ: người dùng mở đường dẫn <https://example.com/products>.

2. Routing Middleware

- Đây là lớp trung gian nằm giữa client và ứng dụng server.
- Nhiệm vụ: chặn và xử lý sơ bộ yêu cầu, trước khi chuyển nó vào hệ thống định tuyến (routing).
- Có thể thực hiện: xác thực, ghi log, bảo vệ route...

3 Route Matching

- Hệ thống đối chiếu URL yêu cầu với các mẫu route đã khai báo trong server.
- Ví dụ: nếu URL là /products/42, thì server sẽ tìm route phù hợp như /products/:id.

4. Controller Selection

- Khi đã xác định được route, hệ thống sẽ gọi controller thích hợp để xử lý logic.
- Mỗi controller là lớp hoặc hàm chứa các action tương ứng.

5. Action Execution

- Đây là nơi mà logic chính được thực thi, ví dụ:
 - Truy vấn cơ sở dữ liệu
 - Tính toán
 - Xử lý file
- Kết quả có thể là JSON, HTML, dữ liệu thô...

6. Initiated Response

- Sau khi controller xử lý xong, nó sẽ tạo một đối tượng response để gửi trả lại client.
- Có thể bao gồm: dữ liệu, mã trạng thái (200, 404...), header, cookie...

7. HTTP Response

- Cuối cùng, phản hồi được trả về cho client.

- Ví dụ: trình duyệt nhận HTML và hiển thị giao diện, hoặc app nhận JSON và hiển thị dữ liệu.

Ý nghĩa tổng thể:

Thành phần	Vai trò chính
Client	Gửi yêu cầu và nhận kết quả
Routing	Middleware Lọc yêu cầu, bảo vệ, ghi log
Routing System	Điều hướng yêu cầu đến đúng nơi
Controller	Thực thi logic xử lý chính
Response	Tạo kết quả trả về

Bảng 4. Ý nghĩa tổng thể

Chương 2: Xây dựng ứng dụng thực tế

2.1. Phát biểu bài toán

1. Định nghĩa bài toán

Hệ thống mua bán và quản lý sách trực tuyến là một website thương mại điện tử cung cấp giải pháp toàn diện cho việc kinh doanh sách online. Hệ thống này cần đáp ứng các yêu cầu cơ bản sau:

- Cho phép khách hàng xem thông tin sản phẩm, tìm kiếm sách theo nhiều tiêu chí
- Hỗ trợ quy trình mua hàng trực tuyến từ khâu chọn sách đến thanh toán
- Cung cấp công cụ quản lý đơn hàng, kho sách cho nhân viên
- Theo dõi và phân tích hoạt động kinh doanh cho quản trị viên

2. Ý nghĩa thực tiễn

Giải quyết các vấn đề hiện tại

- Khắc phục hạn chế về không gian và thời gian của cửa hàng truyền thống
- Giảm thiểu chi phí vận hành (mặt bằng, nhân sự)
- Mở rộng phạm vi khách hàng ra toàn quốc
- Nâng cao trải nghiệm mua sắm với nhiều tiện ích hiện đại

Lợi ích mang lại

- Đối với khách hàng:

- Tiết kiệm thời gian mua sắm
- Dễ dàng so sánh giá và chất lượng sách
- Nhận nhiều ưu đãi hấp dẫn
- Theo dõi đơn hàng trực tuyến

- Đối với nhà sách:

- Tăng doanh thu nhờ mở rộng thị trường
- Giảm chi phí vận hành
- Quản lý kho hàng hiệu quả
- Xây dựng thương hiệu trực tuyến

Xu hướng phát triển

- Thị trường sách điện tử toàn cầu đạt 18 tỷ USD (2023)
- 63% người Việt ưa chuộng mua sách online
- Xu hướng số hóa trong ngành xuất bản

2.2. Phân tích yêu cầu

Chức năng chính

- Quản lý danh mục sách
- Tìm kiếm sản phẩm
- Quản lý giỏ hàng
- Hệ thống thanh toán đa dạng
- Quản lý đơn hàng

Đối tượng sử dụng

- Khách
- Thành viên
- Quản trị viên

Công nghệ sử dụng

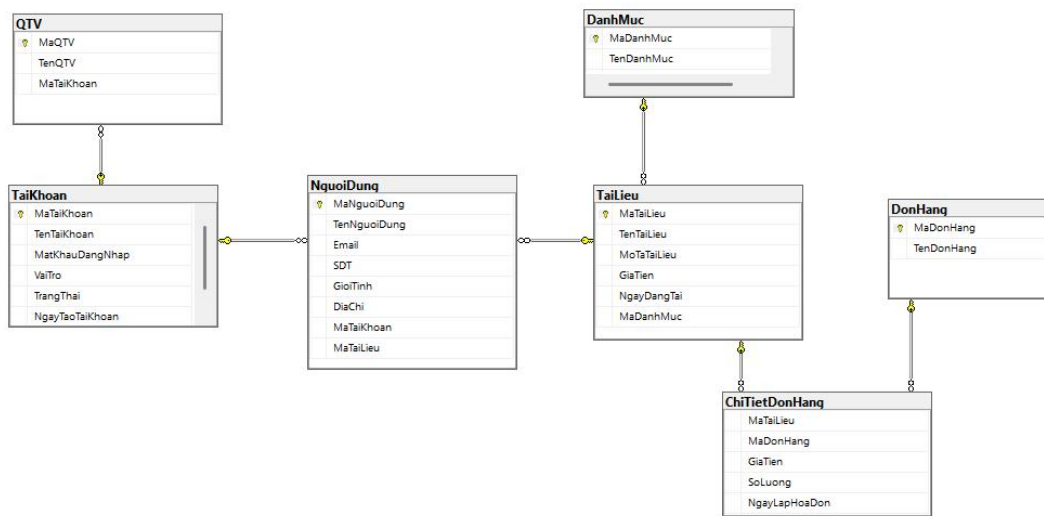
- Frontend: HTML5, CSS3, JavaScript
- Backend: ASP.NET Core, C#
- Database: SQL Server

- Search Engine
- Payment: VNPAY API + Stripe
- Hosting: Cloud server

3. Kết luận

- Website bán sách trực tuyến là giải pháp tối ưu trong bối cảnh phát triển thương mại điện tử hiện nay. Hệ thống không chỉ mang lại lợi ích kinh tế cho nhà sách mà còn đáp ứng nhu cầu ngày càng cao của độc giả về tính tiện lợi và đa dạng trong mua sắm sách.

2.3 Thiết kế cơ sở dữ liệu



Hình 17. Cơ sở dữ liệu

1. Mô tả các bảng trong cơ sở dữ liệu

Bảng NguoiDung (Người dùng)

Tên cột	Kiểu dữ liệu	Mô tả
MaNguoiDung	INT (PK)	Khóa chính
TenNguoiDung	VARCHAR	Họ tên người dùng
Email	VARCHAR	Email (duy nhất)
SDT	VARCHAR	SDT người dùng
GioiTinh	VARCHAR	Giới tính người dùng
DiaChi	VARCHAR	Địa chỉ người dùng
MaTaiKhoan	INT (FK)	Mã tài khoản
MaTaiLieu	NCHAR (FK)	Mã tài liệu

Bảng 5. Bảng NguoiDung

Bảng TaiLieu(Tài liệu)

Tên cột	Kiểu dữ liệu	Mô tả
MaTaiLieu	INT (PK)	Khóa chính
TenTaiLieu	VARCHAR	Tên tài liệu
MoTaTaiLieu	TEXT	Mô tả tài liệu
GiaTien	DECIMAL	Gía tiền tài liệu
NgayDangTai	DATETIME	Ngày đăng tài liệu
MaDanhMuc	INT(FK)	Mã danh mục

*Bảng 6. Bảng TaiLieu***Bảng DanhMuc(Danh mục)**

Tên cột	Kiểu dữ liệu	Mô tả
MaDanhMuc	INT (PK)	Khóa chính
TenDanhMuc	VARCHAR	Tên danh mục

*Bảng 7. Bảng DanhMuc***Bảng DonHang(Đơn hàng)**

Tên cột	Kiểu dữ liệu	Mô tả
MaDonHang	INT (PK)	Khóa chính
TenDonHang	INT (FK)	Người mua

*Bảng 8. Bảng DonHang***Bảng ChiTietDonHang(Chi tiết đơn hàng)**

Tên cột	Kiểu dữ liệu	Mô tả
MaTaiLieu	INT (FK)	Mã đơn hàng
MaDonHang	INT (FK)	Mã tài liệu
GiaTien	DECIMAL	Gía tại thời điểm mua
SoLuong	INT	Số lượng mua
NgayLapHoaDon	DATETIME	Ngày lập hóa đơn

*Bảng 9. Bảng ChiTietDonHang***Bảng TaiKhoan (Tài khoản đăng nhập)**

Tên cột	Kiểu dữ liệu	Mô tả
MaTaiKhoan	INT (PK)	Khóa chính
TenTaiKhoan	VARCHAR	Tên tài khoản
MatKhauDangNhap	CHAR	Mật khẩu đăng nhập
VaiTro	VARCHAR	Vai trò khi đăng nhập
NgayTaoTaiKhoan	DATETIME	Ngày tạo tài khoản

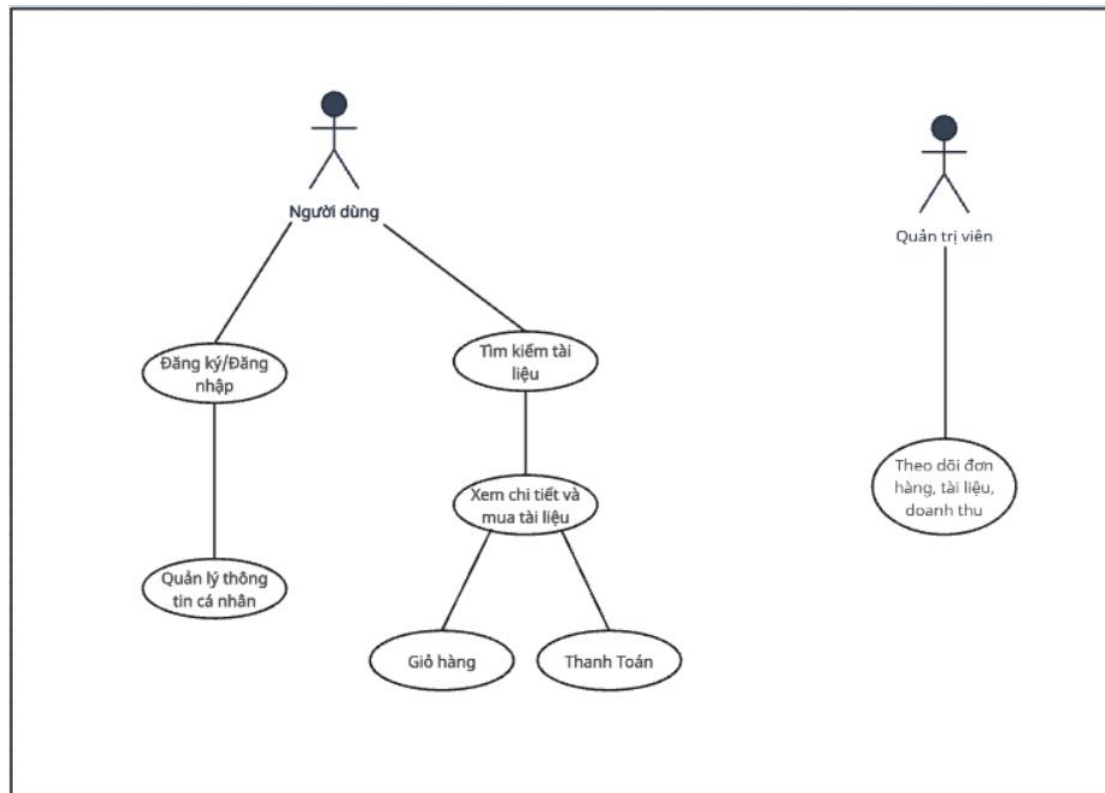
Bảng 10. Bảng TaiKhoan

Bảng QTV(Quản trị viên)

Tên cột	Kiểu dữ liệu	Mô tả
MaQTV	INT (PK)	Khóa chính
TenQTV	VARCHAR	Tên quản trị viên
MaTaiKhoan	INT(FK)	Mã tài khoản

Bảng 11. Bảng QTV

2. Sơ đồ UseCase



Hình 18. Tổng quan UseCase hệ thống

Tác nhân (Actors):

- **Người dùng (User):** Là đối tượng tương tác với hệ thống để thực hiện các chức năng liên quan đến việc tìm kiếm, xem, mua tài liệu.
- **Quản trị viên (Administrator):** Là đối tượng có vai trò quản lý và theo dõi các hoạt động tổng thể của hệ thống như đơn hàng, tài liệu và doanh thu.

Các ca sử dụng (Use Cases) và mối quan hệ:

Đối với Người dùng (User):

- **Đăng ký/Đăng nhập (Đăng ký/Đăng nhập):**

- Mô tả hành động người dùng đăng ký tài khoản mới hoặc đăng nhập vào hệ thống với tài khoản hiện có.

- **Tìm kiếm tài liệu (Tìm kiếm tài liệu):**

- Mô tả hành động người dùng tìm kiếm các tài liệu có sẵn trong hệ thống.

- Liên quan đến ca sử dụng Xem chi tiết và mua tài liệu (Xem chi tiết và mua tài liệu): Sau khi tìm thấy tài liệu, người dùng có thể xem chi tiết về tài liệu đó.

- **Xem chi tiết và mua tài liệu (Xem chi tiết và mua tài liệu):**

- Mô tả hành động người dùng xem thông tin chi tiết về một tài liệu cụ thể và có tùy chọn để mua tài liệu đó.

- Ca sử dụng này bao gồm hai ca sử dụng con (có thể là "include" hoặc "extend" tùy theo ngữ cảnh chi tiết hơn):

- **Giỏ hàng (Giỏ hàng):** Người dùng có thể thêm tài liệu vào giỏ hàng trước khi thanh toán.

- **Thanh toán (Thanh Toán):** Người dùng tiến hành thanh toán cho các tài liệu trong giỏ hàng.

Đối với Quản trị viên (Administrator):

- **Theo dõi đơn hàng, tài liệu, doanh thu (Theo dõi đơn hàng, tài liệu, doanh thu):**

- Mô tả hành động quản trị viên giám sát và quản lý các hoạt động quan trọng của hệ thống, bao gồm theo dõi tình trạng đơn hàng, quản lý danh mục tài liệu và thống kê doanh thu.

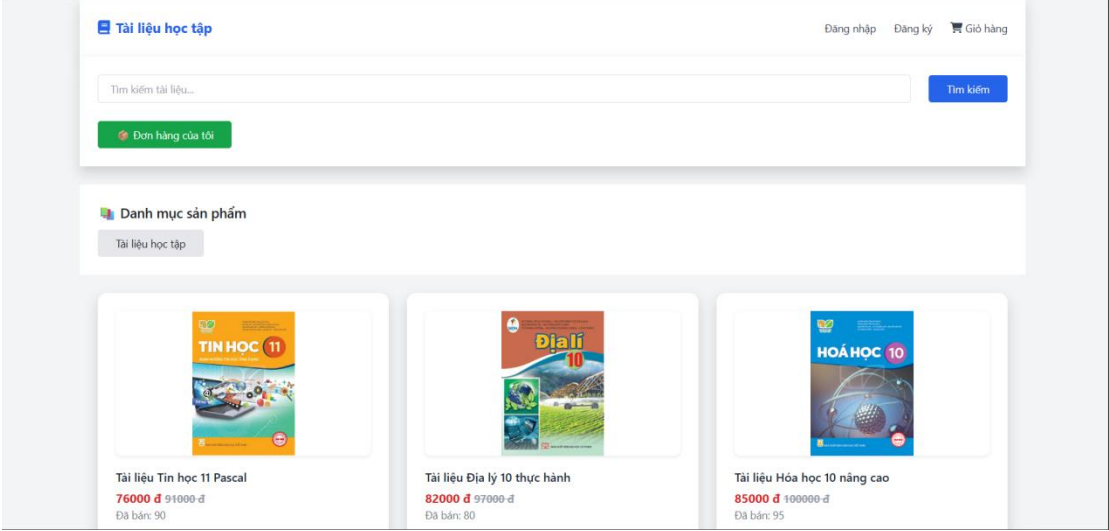
Tóm tắt luồng hoạt động:

- **Người dùng:** Có thể bắt đầu bằng việc đăng ký/đăng nhập. Sau đó, họ có thể tìm kiếm tài liệu, xem chi tiết và mua tài liệu thông qua giỏ hàng và quy trình thanh toán.

- **Quản trị viên:** Có vai trò giám sát tổng thể hoạt động kinh doanh của hệ thống, bao gồm các đơn hàng, kho tài liệu và hiệu suất doanh thu.

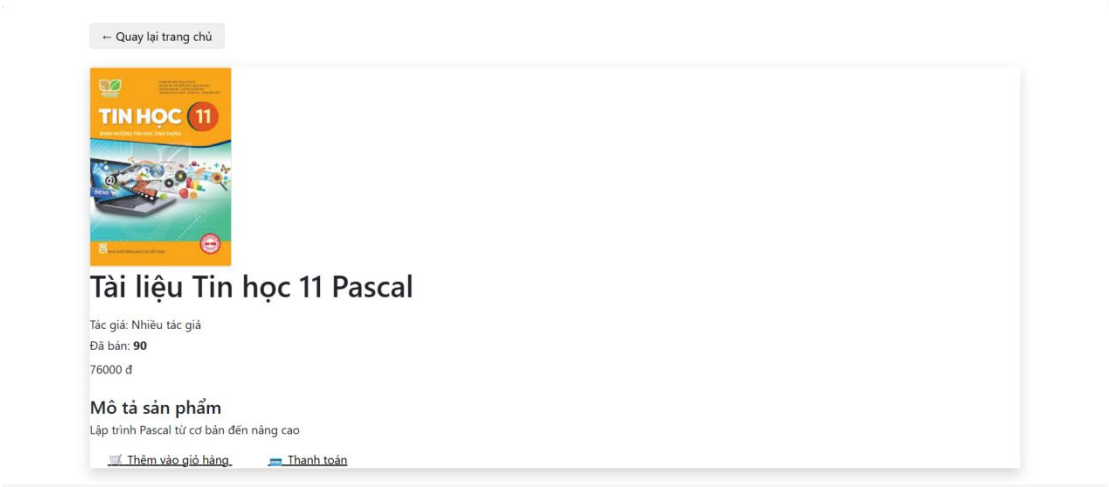
2.4 Thiết kế giao diện người dùng:

Website bán sách được thiết kế với giao diện hiện đại, thân thiện với người dùng. Giao diện chính bao gồm:



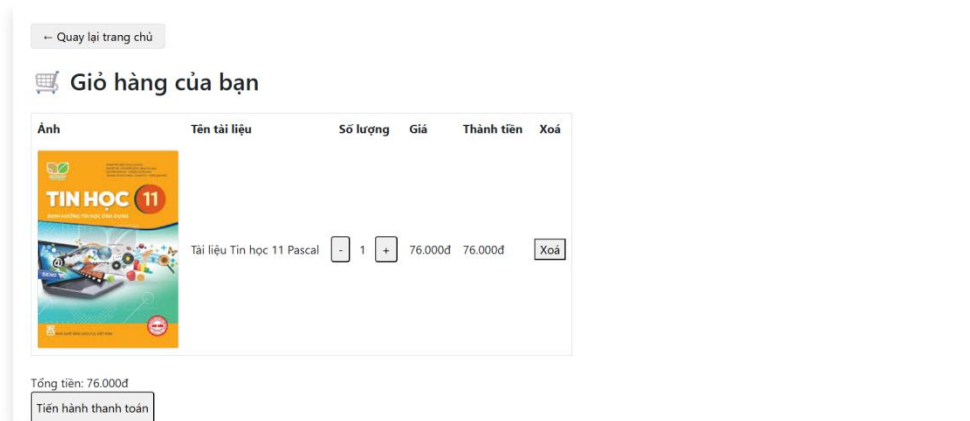
Hình 19. Trang chủ

- **Trang chủ:** Hiện thị danh sách các cuốn sách nổi bật, tìm kiếm nhanh và phân loại theo thể loại.



Hình 20. Thông tin sách

- **Trang chi tiết sách:** Cung cấp thông tin chi tiết về sách, bao gồm tên sách, tác giả, mô tả, và nút thêm vào giỏ hàng.



Hình 21. Giỏ hàng

- **Giỏ hàng:** Danh sách các sản phẩm đã chọn, hỗ trợ chỉnh sửa số lượng và xóa sản phẩm.

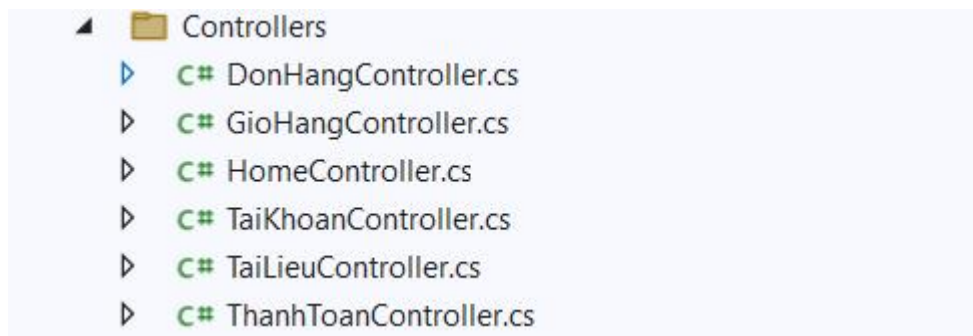
Hình 22. Thông tin thanh toán

- **Trang thanh toán:** Cho phép người dùng nhập thông tin giao hàng và xác nhận đơn hàng.

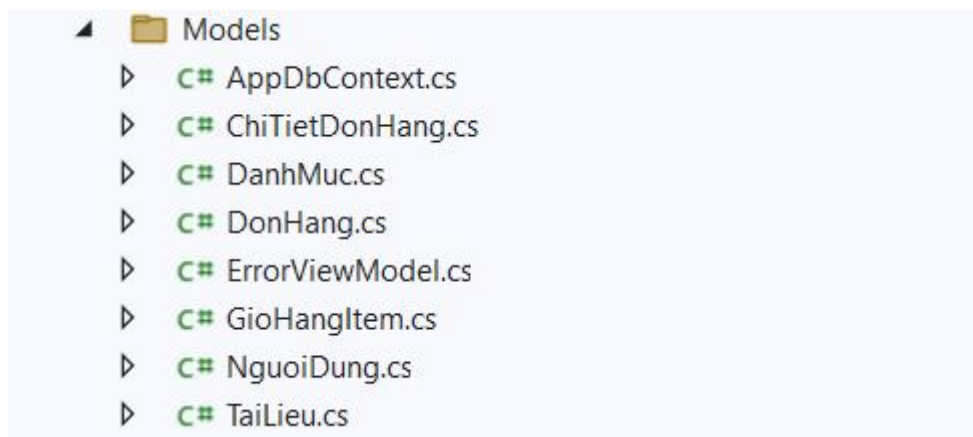
2.5. Thiết kế thành phần MVC

Website bán sách được thiết kế theo mô hình kiến trúc MVC (Model - View - Controller) nhằm tách biệt giữa giao diện người dùng, xử lý logic và quản lý dữ liệu.

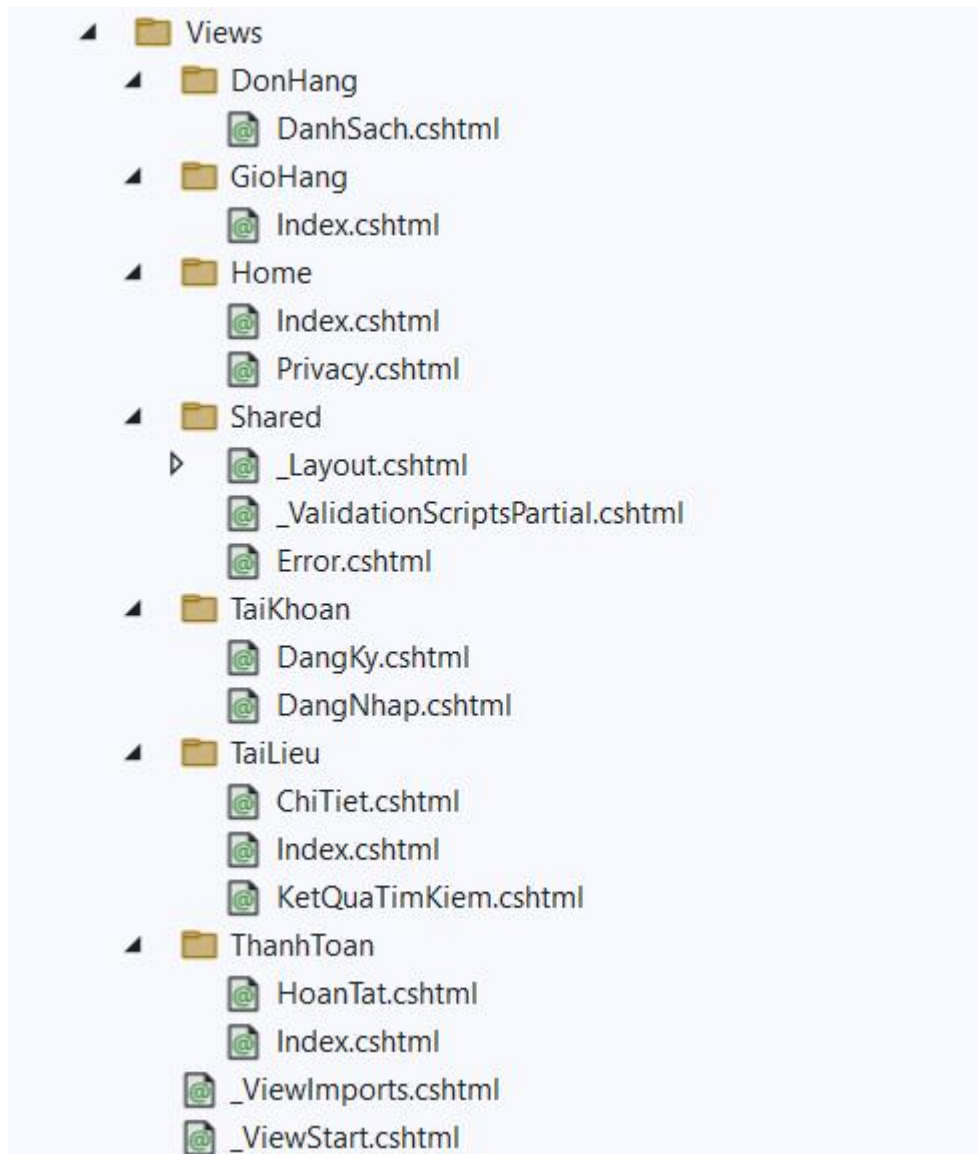
Chi tiết như sau:



Hình 23. Controllers bán sách



Hình 24. Models bán sách



Hình 25. Views bán sách

2.6 Triển khai và cài đặt

Ứng dụng web bán sách được triển khai và cài đặt thông qua các bước sau:

1. Môi trường phát triển

- Ngôn ngữ lập trình: C#
- Framework: ASP.NET MVC (.NET Framework hoặc .NET Core)
- IDE: Visual Studio 2022
- Cơ sở dữ liệu: SQL Server

- Công cụ hỗ trợ: Entity Framework, Bootstrap, jQuery

2. Các bước triển khai ứng dụng

Bước 1: Tạo project ASP.NET MVC

- Khởi tạo dự án mới bằng Visual Studio với mẫu “ASP.NET Web Application (MVC)”.

Bước 2: Thiết kế cơ sở dữ liệu

- Tạo các bảng: Sach, NguoiDung, DonHang, ChiTietDonHang, GioHang,...
- Dùng SQL Server và thiết lập kết nối với ứng dụng qua ConnectionString.

Bước 3: Xây dựng các thành phần

- Model: Tạo các lớp tương ứng với bảng dữ liệu.
- View: Dùng Razor để thiết kế giao diện động, tương tác với người dùng.
- Controller: Xử lý logic và điều hướng luồng dữ liệu, như trong hình ở mục 2.5.

Bước 4: Kết nối CSDL với Entity Framework

- Cấu hình DbContext, dùng Migration để tạo database từ mô hình dữ liệu.

Bước 5: Thiết lập giao diện người dùng

- Sử dụng Bootstrap để tạo giao diện responsive.
- Giao diện chính: Trang chủ, chi tiết sách, giỏ hàng, thanh toán, đăng nhập/đăng ký.

Bước 6: Kiểm thử và xử lý lỗi

- Kiểm tra các chức năng: thêm vào giỏ hàng, đặt hàng, đăng nhập,...
- Fix các lỗi logic và UI/UX trong quá trình kiểm thử.

Bước 7: Triển khai

- Triển khai nội bộ trên IIS hoặc công khai trên nền tảng hosting như:
- IIS Server nội bộ

- Azure App Service (nếu có)
- Localhost cho kiểm thử trước khi publish

Chương 3. Kết quả chương trình

1.1. Demo chương trình

Giao diện đăng nhập:

- Người dùng có thể tạo tài khoản và đăng nhập để quản lý đơn hàng.
- Sau khi đăng nhập, hệ thống sẽ hiển thị tên người dùng ở giao diện chính.

← Quay lại trang chủ

Đăng nhập tài khoản

Email

Mật khẩu Ẩn/Hiện

Chưa có tài khoản? [Đăng ký ngay](#)

© 2025 - BanSach - [Privacy](#)

Hình 26. Giao diện đăng nhập

Giao diện đăng ký:

← Quay lại trang chủ

Đăng ký tài khoản

Họ tên

Email

Mật khẩu Ẩn/Hiện

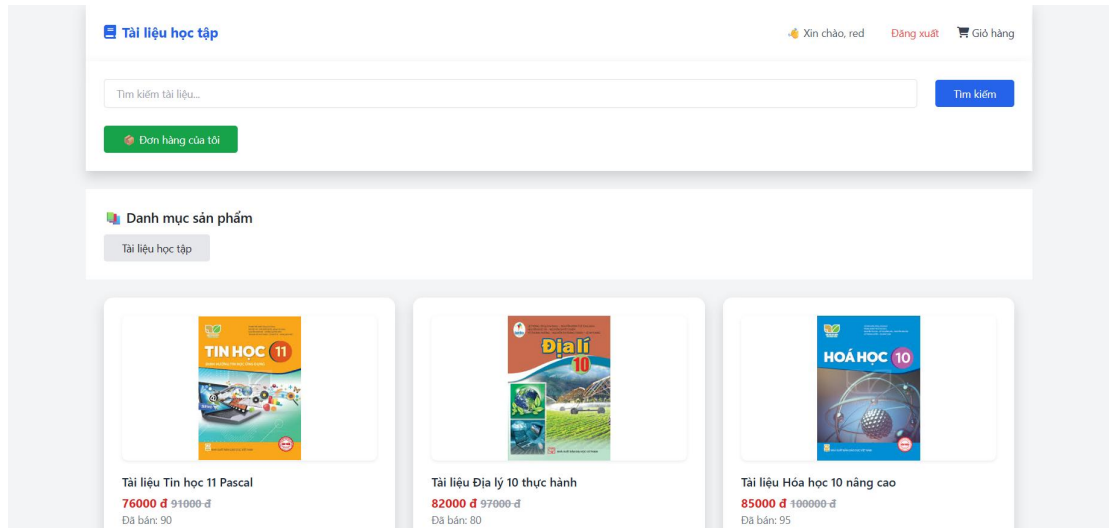
Đã có tài khoản? [Đăng nhập ngay](#)

© 2025 - BanSach - [Privacy](#)

Hình 27. Giao diện đăng ký

Trang chủ:

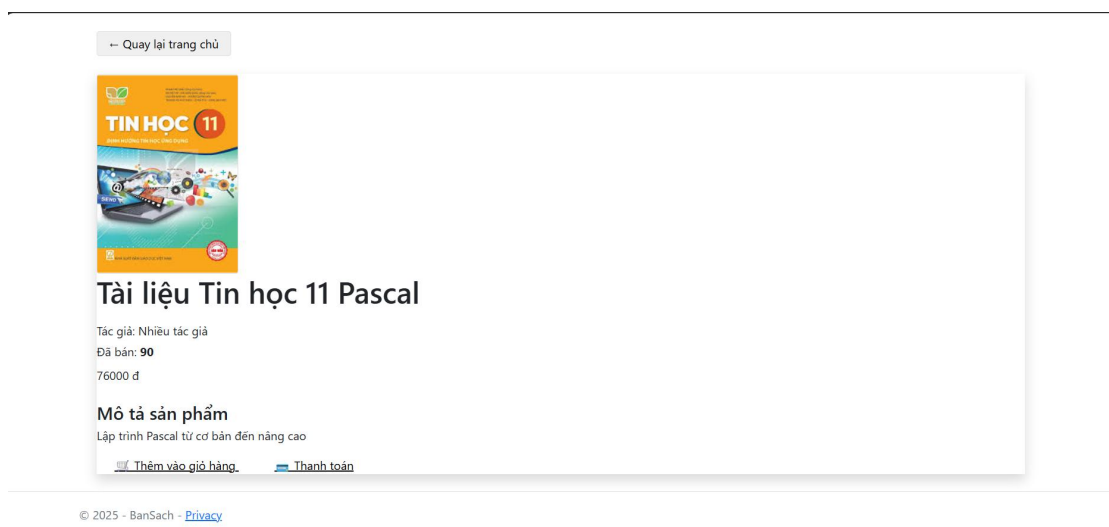
- Hiện thị danh sách sách đang bán.
- Có thanh tìm kiếm theo tên sách hoặc thể loại.
- Người dùng có thể nhấn vào một cuốn sách để xem chi tiết.



Hình 28. Trang chủ

Thông tin sách:

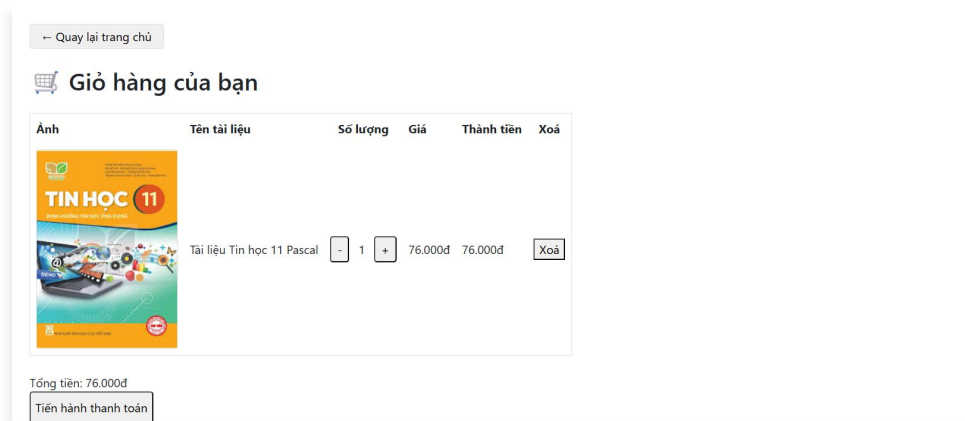
- Hiện thị ảnh bìa, tiêu đề, tác giả, giá bán, mô tả.
- Có nút "Thêm vào giỏ".
- Nút “Thanh toán”.



Hình 29. Thanh toán

Giỏ hàng:

- Khi người dùng nhấn nút "Thêm vào giỏ", sách sẽ được lưu vào session hoặc database.
- Có thể cập nhật số lượng hoặc xóa sản phẩm.



© 2025 - BanSach - [Privacy](#)

Hình 30. Giỏ hàng

Thanh toán:

- Người dùng nhập thông tin người nhận: tên, địa chỉ, số điện thoại.
- Hệ thống lưu đơn hàng và hiển thị thông báo "Đặt hàng thành công".

[← Quay lại giỏ hàng](#)

Thông tin thanh toán

Sản phẩm trong giỏ hàng

- Tài liệu Tin học 11 Pascal
- Số lượng: 1
- 76.000đ

Họ tên

Email

Số điện thoại

Địa chỉ giao hàng

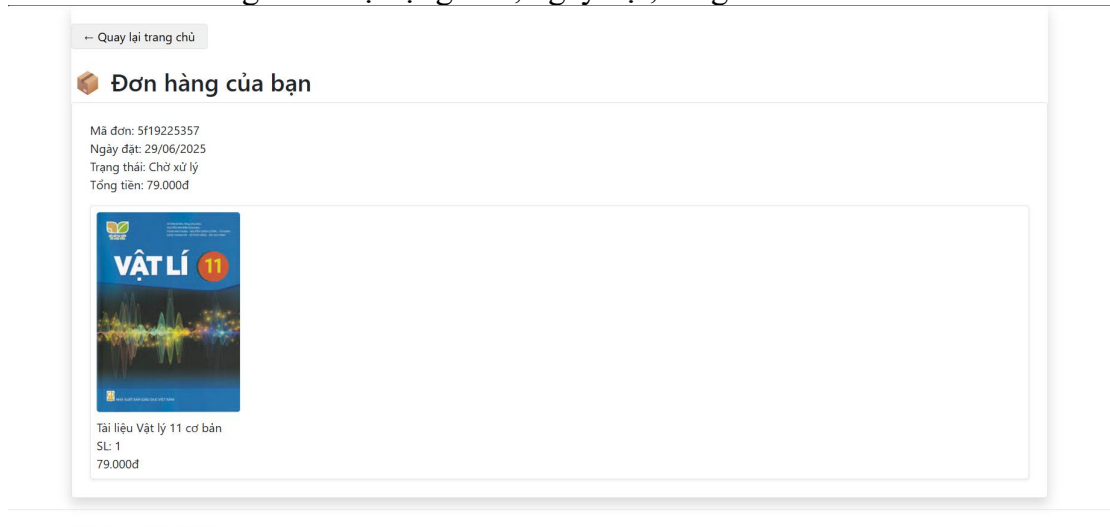
Số tài khoản ngân hàng

Hình 31. Thanh toán

Đơn hàng:

- Xem danh sách đơn hàng đã đặt.

- Mỗi đơn hàng hiển thị trạng thái, ngày đặt, tổng tiền.

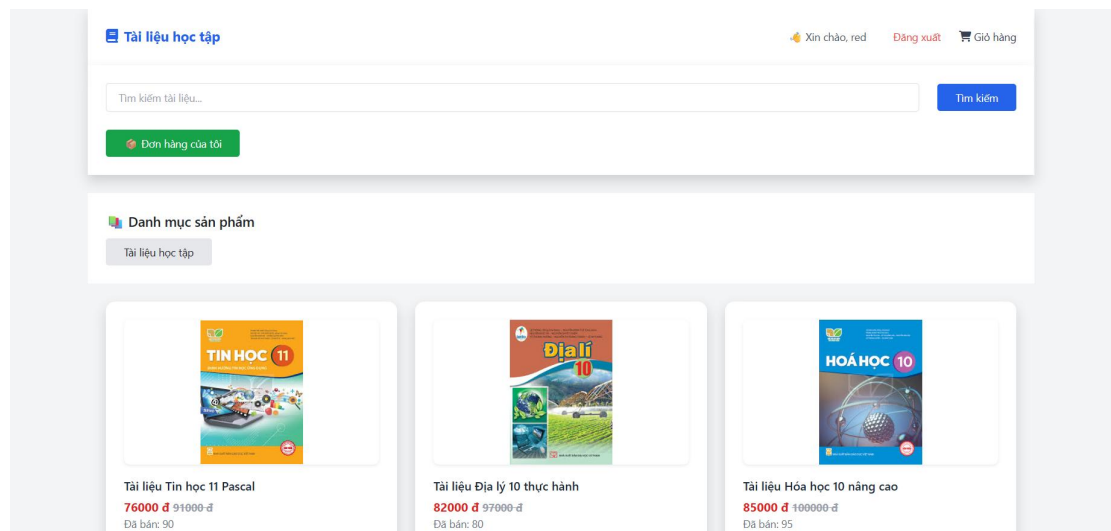


Hình 32. Đơn hàng

3.2 Ảnh chụp màn hình chương trình hoạt động

Ảnh 1: Giao diện trang chủ

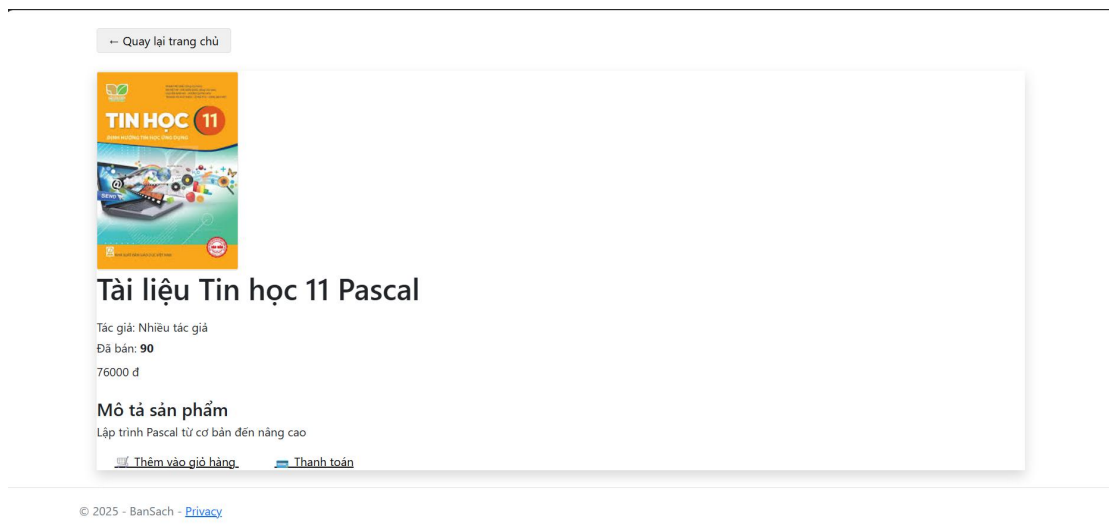
Giao diện hiển thị danh sách các sách đang bán, có thanh tìm kiếm và phân loại sách theo thể loại hoặc danh mục.



Hình 33. Trang chủ

Ảnh 2: Trang chi tiết sách

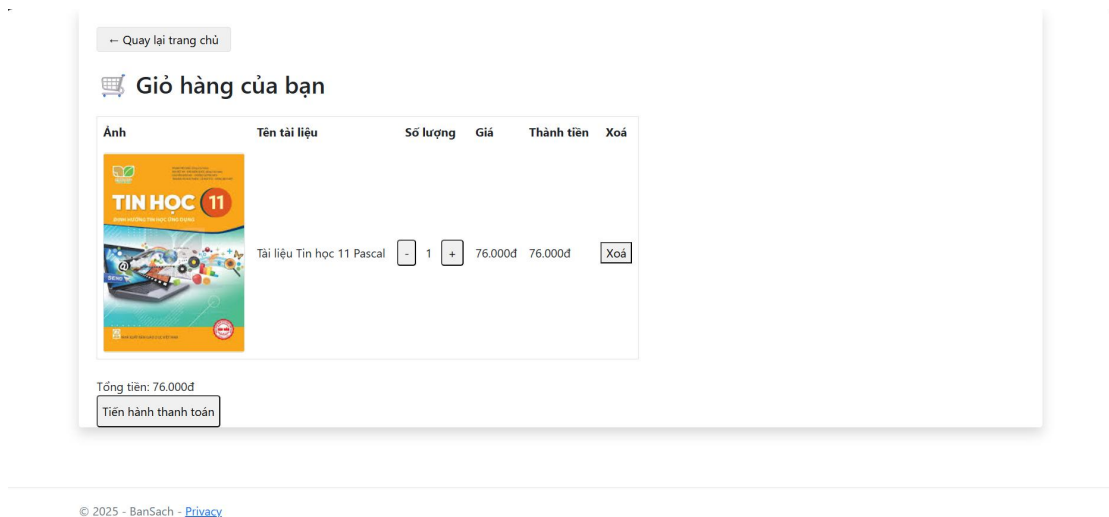
Hiển thị đầy đủ thông tin về cuốn sách: tên, tác giả, giá bán, mô tả và nút "Thêm vào giỏ" và "Thanh toán".



Hình 34. Thông tin sách

Ảnh 3: Giao diện giỏ hàng

Người dùng xem các sách đã thêm, cập nhật số lượng hoặc xóa sách khỏi giỏ.



Hình 35. Giỏ hàng

Ảnh 4: Trang thanh toán

Giao diện nhập thông tin giao hàng và xác nhận đặt hàng.

[← Quay lại giỏ hàng](#)

Thông tin thanh toán

Sản phẩm trong giỏ hàng

- Tài liệu Tin học 11 PascalSố lượng: 176.000đ

Họ tên

Email

Số điện thoại

Địa chỉ giao hàng

Số tài khoản ngân hàng

Hình 36. Thanh toán

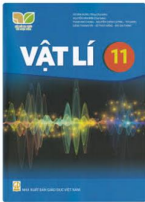
Ảnh 5: Trang quản lý đơn hàng

Hiện thị danh sách các đơn hàng đã đặt, bao gồm thông tin đơn, trạng thái và tổng giá trị.

[← Quay lại trang chủ](#)

Đơn hàng của bạn

Mã đơn: 5f19225357
Ngày đặt: 29/06/2025
Trạng thái: Chờ xử lý
Tổng tiền: 79.000đ



Tài liệu Vật lý 11 cơ bản
SL: 1
79.000đ

Hình 37. Đơn hàng

3.3 Đánh giá kết quả

Kết quả đạt được:

✔ Hoàn thành giao diện người dùng thân thiện, dễ sử dụng

- Giao diện trực quan, tương thích với nhiều độ phân giải màn hình.

- Giao diện chính gồm: trang chủ, chi tiết sách, giỏ hàng, thanh toán, quản lý đơn hàng.

✔ **Tính năng đầy đủ theo yêu cầu**

- Xem sách, tìm kiếm theo tên hoặc thể loại.
- Thêm sách vào giỏ, cập nhật số lượng.
- Đặt hàng và thanh toán.
- Đăng ký/đăng nhập.

✔ **Hệ thống phân quyền cơ bản**

- Phân biệt người dùng thông thường và quản trị viên (nếu có).
- Người dùng có thể xem đơn hàng của mình.
- Quản trị viên có thể quản lý sách và đơn hàng (nếu đã cài đặt).

Hạn chế còn tồn tại:

- Giao diện chưa thực sự tối ưu trên thiết bị di động (nếu chưa responsive).
- Chưa có tính năng thanh toán trực tuyến (chỉ mô phỏng).
- Bảo mật mới dừng ở mức cơ bản (nếu chưa mã hóa mật khẩu, chưa chống SQL Injection,...).

Kết Luận

Trong quá trình thực hiện đồ án “Xây dựng website bán sách”, áp dụng kiến thức về lập trình web, cơ sở dữ liệu và kiến trúc phần mềm để xây dựng một hệ thống hoàn chỉnh theo mô hình MVC.

Website hỗ trợ các chức năng cơ bản như: xem thông tin sách, tìm kiếm, thêm vào giỏ hàng, đặt hàng, đăng ký – đăng nhập và quản lý đơn hàng. Giao diện người dùng thân thiện, hoạt động ổn định và đáp ứng được các yêu cầu đề ra ban đầu.

Dù còn một số hạn chế về tính năng nâng cao và khả năng bảo mật, đồ án đã giúp em hiểu rõ hơn về quy trình phát triển phần mềm, kỹ năng làm việc nhóm cũng như tư duy thiết kế hệ thống.

Tài liệu tham khảo

1. Tạ Huy Cường, Phân tích và thiết kế phần mềm theo mô hình MVC, Nhà xuất bản Khoa học và Kỹ thuật, Hà Nội, 2018.
2. Nguyễn Thị Hương, Ứng dụng MVC trong phát triển web với ASP.NET, Tạp chí Công nghệ Thông tin và Truyền thông, số 12, trang 45-52, 2020.
3. Steven Sanderson, Pro ASP.NET MVC 5, Apress, 2014.
4. Martin Fowler, Patterns of Enterprise Application Architecture, Addison-Wesley, 2002.
5. Microsoft Docs, Model-View-Controller (MVC) overview, <https://learn.microsoft.com/en-us/aspnet/mvc/overview> (truy cập ngày 28/6/2025).
6. Tutorialspoint, MVC Architecture, https://www.tutorialspoint.com/mvc_framework/mvc_architecture.htm (truy cập ngày 28/6/2025).
7. Nguyễn Văn A, Thiết kế website bán sách với mô hình MVC, Tạp chí Khoa học Công nghệ, số 15, 2019.