**TRƯỜNG ĐẠI HỌC ĐẠI NAM**

**KHOA CÔNG NGHỆ THÔNG TIN**

Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**TÊN MÔN HỌC: THIẾT KẾ WEB VÀ TRIỂN KHAI HỆ THỐNG PHẦN MỀM**

**TÊN ĐỀ TÀI: THIẾT KẾ TRANG WED BÁN GIÀY DÉP**

**Giảng viên hướng dẫn: ThS. Lê Văn Phong**

**Sinh viên thực hiện:**  **Hà Phi Hùng**

**Hà Nội, 2024**

**TRƯỜNG ĐẠI HỌC ĐẠI NAM**

**KHOA CÔNG NGHỆ THÔNG TIN**

Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**TÊN MÔN HỌC: THIẾT KẾ WEB VÀ TRIỂN KHAI HỆ THỐNG PHẦN MỀM**

**TÊN ĐỀ TÀI: THIẾT KẾ TRANG WED BÁN GIÀY DÉP**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Mã Sinh Viên | Họ và Tên | Ngày Sinh | Điểm | |
| Bằng Số | Bằng Chữ |
| 01 | 1771020315 | Hà Phi Hùng | 10/07/2005 |  |  |

### 

### CÁN BỘ CHẤM THI 1 CÁN BỘ CHẤM THI 2

**Hà Nội, 2024**

**LỜI NÓI ĐẦU**

(Nếu có)

Trước khi bắt đầu thiết kế web, quan trọng để thu thập thông tin khách hàng, hiểu rõ đối tượng người dùng, và xác định mục tiêu kinh doanh. Điều này giúp xác định yêu cầu chức năng và thiết kế giao diện phù hợp, bao gồm cấu trúc trang, bố cục, màu sắc, hình ảnh và các yếu tố tương tác.

Quá trình thiết kế web thường bao gồm việc tạo ra các bản vẽ, mẫu thiết kế hoặc giao diện người dùng tương tác (UI/UX) để trình bày các ý tưởng và thiết kế ban đầu. Sau đó, các công cụ và ngôn ngữ như HTML, CSS và JavaScript được sử dụng để triển khai giao diện web thực tế.

Triển khai hệ thống phần mềm bao gồm các bước xây dựng, kiểm thử và triển khai các thành phần phần mềm. Quá trình này bao gồm việc lựa chọn công nghệ, thiết kế cơ sở dữ liệu, xây dựng các chức năng và tính năng, kiểm thử và tối ưu hóa hiệu suất. Sau đó, hệ thống phần mềm được triển khai và triển khai trên môi trường sản xuất để sử dụng bởi người dùng cuối.

Trong cả thiết kế web và triển khai hệ thống phần mềm, quan trọng để đảm bảo tính bảo mật, khả năng mở rộng, tương thích với nhiều nền tảng và trình duyệt, cũng như đáp ứng được yêu cầu và mong đợi của người dùng.

Tổng quan, việc thiết kế web và triển khai hệ thống phần mềm đòi hỏi sự phối hợp giữa thiết kế giao diện hấp dẫn và chức năng mạnh mẽ, để tạo ra trải nghiệm người dùng tốt và đáp ứng nhu cầu kinh doanh. Lời nói đầu về thiết kế web và triển khai hệ thống phần mềm thường bắt đầu bằng việc hiểu rõ yêu cầu và mục tiêu của dự án. Thiết kế web tập trung vào việc tạo ra giao diện người dùng hấp dẫn, trực quan và dễ sử dụng, trong khi triển khai hệ thống phần mềm tập trung vào việc xây dựng cơ sở hạ tầng và triển khai các phần mềm để đáp ứng yêu cầu chức năng và hiệu suất.

Trước khi bắt đầu thiết kế web, quan trọng để thu thập thông tin khách hàng, hiểu rõ đối tượng người dùng, và xác định mục tiêu kinh doanh. Điều này giúp xác định yêu cầu chức năng và thiết kế giao diện phù hợp, bao gồm cấu trúc trang, bố cục, màu sắc, hình ảnh và các yếu tố tương tác.

Quá trình thiết kế web thường bao gồm việc tạo ra các bản vẽ, mẫu thiết kế hoặc giao diện người dùng tương tác (UI/UX) để trình bày các ý tưởng và thiết kế ban đầu. Sau đó, các công cụ và ngôn ngữ như HTML, CSS và JavaScript được sử dụng để triển khai giao diện web thực tế.

Triển khai hệ thống phần mềm bao gồm các bước xây dựng, kiểm thử và triển khai các thành phần phần mềm. Quá trình này bao gồm việc lựa chọn công nghệ, thiết kế cơ sở dữ liệu, xây dựng các chức năng và tính năng, kiểm thử và tối ưu hóa hiệu suất. Sau đó, hệ thống phần mềm được triển khai và triển khai trên môi trường sản xuất để sử dụng bởi người dùng cuối.

Trong cả thiết kế web và triển khai hệ thống phần mềm, quan trọng để đảm bảo tính bảo mật, khả năng mở rộng, tương thích với nhiều nền tảng và trình duyệt, cũng như đáp ứng được yêu cầu và mong đợi của người dùng.

Tổng quan, việc thiết kế web và triển khai hệ thống phần mềm đòi hỏi sự phối hợp giữa thiết kế giao diện hấp dẫn và chức năng mạnh mẽ, để tạo ra trải nghiệm người dùng tốt và đáp ứng nhu cầu kinh doanh.

MỤC LỤC

**(Đánh tự động)**

Contents

[**CHƯƠNG I: GIỚI THIỆU VỀ THIẾT KẾ WEB VÀ TRIỂN KHAI HỆ THỐNG PHẦN MỀM** 11](#_Toc161168812)

[**1.1** **Các khái niệm cơ bản** 11](#_Toc161168813)

[1.1.2. Triển khai phần mềm (Software deployment): 11](#_Toc161168816)

[**1.2. Phân biệt Web tĩnh, Web động** 11](#_Toc161168819)

[1.1.1 Web tĩnh (Static web): 11](#_Toc161168820)

[1.1.2. Web động (Dynamic web): 12](#_Toc161168824)

[**1.3. Một số thuật ngữ (Hosting, Tên miền-Domain Name-Web Server Name, HomePage,** 13](#_Toc161168828)

[**Web Site, WebServer, URL-Uniform Resource Location, Browser, …)** 13](#_Toc161168829)

[**1.4. Một số công cụ dùng thiết kế Web (Visual Studio code, Adobe Dreamweaver, …)** 14](#_Toc161168839)

[**1.5 Kết Luận** 15](#_Toc161168840)

[1.5.1 Thuật ngữ: 15](#_Toc161168841)

[1.5.2. Công cụ thiết kế web: 15](#_Toc161168842)

[**CHƯƠNG II: HTML & HTML5** 16](#_Toc161168843)

[**(**Hyper Text Markup Language**)** 16](#_Toc161168844)

[**2.1. Tổng quan về HTML** 16](#_Toc161168845)

[**2.2. Cấu trúc tổng quát trang HTML** 16](#_Toc161168851)

[**2.3 Một số thẻ HTML** 17](#_Toc161168852)

[**2.4. Các thẻ tạo biểu mẫu (form)** 19](#_Toc161168853)

[**2.5. Một số thẻ HTML đặc biệt** 20](#_Toc161168854)

[**Kết Luận Chương 2** 22](#_Toc161168855)

[**CHƯƠNG III: CSS và CSS3** 23](#_Toc161168856)

[**3.1. CSS là gì?** 23](#_Toc161168857)

[**3.2. Cú pháp CSS** 23](#_Toc161168858)

[**3.3. Áp dụng CSS vào trang HTML** 24](#_Toc161168859)

[**3.4. Selectors** 24](#_Toc161168860)

[*3.4.1. Universal selector*: Sử dụng dấu `\*` để chọn tất cả các phần tử trên trang. 25](#_Toc161168861)

[*3.4.2. Type selector*: Sử dụng tên thẻ HTML để chọn tất cả các phần tử có cùng loại thẻ. 26](#_Toc161168862)

[*3.4.3. Identity selector* 27](#_Toc161168863)

[*3.4.4. Class selector:* 28](#_Toc161168864)

[*3.4.5. Descendant selector:* 29](#_Toc161168865)

[*3.4.6. Child selector:* 30](#_Toc161168866)

[3.4.7. Adjacent sibling selector: 31](#_Toc161168867)

[3.4.8. Attribute selector: 32](#_Toc161168868)

[*3.4.9. Pseudo class selector* 33](#_Toc161168869)

[3.4.10. Group selector: 35](#_Toc161168894)

[**3.5. Đơn vị đo lường CSS** 35](#_Toc161168895)

[**3.6. Kế thừa thuộc tính** 36](#_Toc161168896)

[**3.7. Các nhóm thuộc tính trong CSS** 37](#_Toc161168897)

[**3.8. Float & Clear** 38](#_Toc161168898)

[**3.9. Flex** 39](#_Toc161168899)

[**3.10. Grid** 40](#_Toc161168900)

[**3.11 CSS3:** 41](#_Toc161168901)

[**3.12. SCSS:** 43](#_Toc161168902)

[**3.13. SASS:** 43](#_Toc161168903)

[**Kết luận chương 3** 44](#_Toc161168904)

[CHƯƠNG 4: THIẾT KẾ WEBISTE THEO ĐỀ TÀI BẠN CHỌN 46](#_Toc161168905)

[**4.1. Ý tưởng của Website** 46](#_Toc161168906)

[**4.2. Xây dựng bố cục của trang Web** 46](#_Toc161168907)

[**4.3. Thiết kế trang Web bằng HTML và CSS** 48](#_Toc161168908)

[**4.4. Kết quả đạt được** 49](#_Toc161168909)

[**Kết luận chương 4** 52](#_Toc161168910)

[**KẾT LUẬN** 52](#_Toc161168911)

[Hướng dẫn xếp tài liệu tham khảo 55](#_Toc161168912)

# **CHƯƠNG I: GIỚI THIỆU VỀ THIẾT KẾ WEB VÀ TRIỂN KHAI HỆ THỐNG PHẦN MỀM**

## **Các khái niệm cơ bản**

### 1.1.1 Thiết kế web (Web design):

Thiết kế web là quá trình tạo ra giao diện và trải nghiệm người dùng cho một trang web. Nhiệm vụ của người thiết kế web là tạo ra một giao diện hấp dẫn, trực quan và dễ sử dụng cho người dùng. Trong quá trình thiết kế web, người thiết kế cần xác định cấu trúc trang, lựa chọn màu sắc, hình ảnh, phông chữ và các yếu tố khác để tạo nên giao diện hợp lý và thu hút người dùng. Ngoài ra, thiết kế web cũng đảm bảo tính tương thích trên các thiết bị và trình duyệt khác nhau.

### 1.1.2. Triển khai phần mềm (Software deployment):

Triển khai phần mềm là quá trình đưa phần mềm từ môi trường phát triển vào môi trường vận hành thực tế. Quá trình triển khai này bao gồm các hoạt động như chuẩn bị môi trường, cài đặt phần mềm, cấu hình hệ thống, kiểm tra và triển khai phần mềm trong môi trường sản phẩm. Mục tiêu của triển khai phần mềm là đảm bảo rằng phần mềm hoạt động đúng như mong đợi, tương thích với môi trường và đáp ứng yêu cầu của người dùng.

các nền tảng đám mây. Ngoài ra, việc triển khai cũng liên quan đến việc quản lý phiên bản, cập nhật và duy trì phần mềm sau khi triển khai.

## **1.2. Phân biệt Web tĩnh, Web động**

### 1.1.1 Web tĩnh (Static web):

- Web tĩnh là loại trang web mà nội dung và giao diện của nó không thay đổi dựa trên hành vi của người dùng. Tất cả các trang và thông tin trên web tĩnh được tạo sẵn và lưu trữ trên máy chủ web.

- Các trang web tĩnh thường được xây dựng bằng HTML (HyperText Markup Language) và CSS (Cascading Style Sheets). HTML được sử dụng để tạo cấu trúc và định dạng nội dung trang, trong khi CSS được sử dụng để định dạng giao diện và kiểu dáng của trang.

- Với web tĩnh, việc cập nhật và thay đổi nội dung trang web thường đòi hỏi sự can thiệp từ người phát triển, và sau đó phải tải lên lại máy chủ web để cập nhật nội dung mới.

### 1.1.2. Web động (Dynamic web):

- Web động là loại trang web có khả năng tạo ra nội dung và giao diện đa dạng dựa trên hành vi của người dùng. Nội dung trên trang web động được tạo ra và phục vụ theo yêu cầu từ máy chủ web.

- Các trang web động thường sử dụng ngôn ngữ lập trình phía máy chủ như PHP (Hypertext Preprocessor), ASP.NET (Active Server Pages), hoặc Node.js để tạo ra nội dung động. Các ngôn ngữ này cho phép tương tác với cơ sở dữ liệu, xử lý dữ liệu và tạo ra nội dung theo yêu cầu cụ thể của người dùng.

- Với web động, việc cập nhật nội dung trang web có thể được thực hiện thông qua giao diện quản trị hoặc hệ thống quản lý nội dung (Content Management System - CMS), cho phép người dùng không cần có kiến thức về lập trình cũng có thể thay đổi nội dung và giao diện trang web.

## **1.3. Một số thuật ngữ (Hosting, Tên miền-Domain Name-Web Server Name, HomePage,**

## **Web Site, WebServer, URL-Uniform Resource Location, Browser, …)**

1. Hosting: Hosting là dịch vụ cung cấp không gian lưu trữ trên máy chủ để lưu trữ các tập tin, dữ liệu và ứng dụng của một trang web. Các dịch vụ hosting cho phép trang web của bạn truy cập được trên Internet.

2. Tên miền (Domain Name): Tên miền là địa chỉ duy nhất trên Internet dùng để xác định một trang web hoặc địa chỉ email. Ví dụ: "google.com" là một tên miền.

3. Web Server Name: Web Server Name là tên của máy chủ web, nơi lưu trữ và phục vụ các tập tin trang web. Ví dụ: Apache, Nginx là các web server phổ biến.

4. Homepage: Homepage là trang chủ của một trang web. Đây là trang đầu tiên xuất hiện khi người dùng truy cập vào tên miền của trang web.

5. Website: Website là một tập hợp các trang web liên kết với nhau, tạo thành một thực thể hoàn chỉnh. Mỗi trang web trong website có thể chứa các nội dung và thông tin riêng biệt.

6. Web Server: Web Server là một phần mềm chạy trên máy chủ, có nhiệm vụ phục vụ các trang web cho người dùng. Web server nhận yêu cầu từ trình duyệt của người dùng thông qua giao thức HTTP và trả về các tập tin và dữ liệu trang web tương ứng.

7. URL (Uniform Resource Locator): URL là địa chỉ duy nhất cho một tài nguyên trên Internet. Nó bao gồm các thành phần như giao thức truyền tải (ví dụ: http, https), tên miền và đường dẫn đến tài nguyên cụ thể trên máy chủ.

8. Trình duyệt (Browser): Trình duyệt là phần mềm cho phép người dùng truy cập và xem các trang web trên Internet. Các trình duyệt phổ biến bao gồm Google Chrome, Mozilla Firefox, Microsoft Edge, và Safari.

Đây chỉ là một số thuật ngữ cơ bản trong lĩnh vực thiết kế web và phát triển ứng dụng web. Còn nhiều thuật ngữ khác liên quan tới các khía cạnh khác nhau của quá trình phát triển web.

## **1.4. Một số công cụ dùng thiết kế Web (Visual Studio code, Adobe Dreamweaver, …)**

1. Visual Studio Code: Visual Studio Code là một trình biên tập mã nguồn mở và miễn phí được phát triển bởi Microsoft. Nó cung cấp nhiều tính năng hữu ích cho việc lập trình web, bao gồm gợi ý mã, kiểm tra cú pháp, gỡ lỗi và tích hợp với các công cụ phát triển web khác.

2. Adobe Dreamweaver: Adobe Dreamweaver là một công cụ phát triển web chuyên nghiệp được phát triển bởi Adobe. Nó cung cấp một giao diện trực quan và các tính năng kéo và thả (WYSIWYG - What You See Is What You Get) để thiết kế và quản lý trang web. Ngoài ra, Dreamweaver cũng hỗ trợ việc viết mã và tích hợp với các công nghệ web phổ biến khác.

3. Sublime Text: Sublime Text là một trình biên tập mã nguồn được ưa chuộng trong cộng đồng phát triển web. Nó có giao diện đơn giản, tốc độ nhanh và nhiều tính năng mạnh mẽ như gợi ý mã, tìm kiếm đa dòng, và khả năng mở rộng thông qua các gói mở rộng.

4. Atom: Atom là một trình biên tập mã nguồn mở và miễn phí được phát triển bởi GitHub. Nó có giao diện đẹp và linh hoạt, hỗ trợ gợi ý mã, tìm kiếm và thay thế nâng cao, và tích hợp với các công cụ quản lý phiên bản.

5. Brackets: Brackets là một trình biên tập mã nguồn mở và miễn phí, được tạo ra bởi Adobe. Nó được thiết kế đặc biệt cho phát triển web, với các tính năng như xem trực tiếp (Live Preview) để thấy ngay kết quả của mã thay đổi trên trình duyệt.

## **1.5 Kết Luận**

### 1.5.1 Thuật ngữ:

- Hosting: Dịch vụ cung cấp không gian lưu trữ trên máy chủ để lưu trữ trang web.

- Tên miền (Domain Name): Địa chỉ duy nhất trên Internet để xác định một trang web hoặc địa chỉ email.

- Web Server Name: Tên của máy chủ web, nơi lưu trữ và phục vụ các tập tin trang web.

- Homepage: Trang chủ của một trang web.

- Website: Tập hợp các trang web liên kết với nhau, tạo thành một thực thể hoàn chỉnh.

- Web Server: Phần mềm chạy trên máy chủ, phục vụ các trang web cho người dùng.

- URL (Uniform Resource Locator): Địa chỉ duy nhất cho một tài nguyên trên Internet.

- Trình duyệt (Browser): Phần mềm cho phép người dùng truy cập và xem trang web trên Internet.

### 1.5.2. Công cụ thiết kế web:

- Visual Studio Code: Trình biên tập mã nguồn mở và miễn phí với nhiều tính năng hữu ích.

- Adobe Dreamweaver: Công cụ phát triển web chuyên nghiệp với giao diện trực quan và tính năng kéo và thả.

- Sublime Text: Trình biên tập mã nguồn phổ biến với giao diện đơn giản và tính năng mạnh mẽ.

- Atom: Trình biên tập mã nguồn mở và linh hoạt với các tính năng như gợi ý mã và tìm kiếm đa dòng.

- Brackets: Trình biên tập mã nguồn mở và miễn phí, thiết kế đặc biệt cho phát triển web với tính năng xem trực tiếp.

Những thuật ngữ và công cụ này đóng vai trò quan trọng trong quá trình thiết kế và phát triển trang web, cho phép người dùng tạo ra và quản lý giao diện web một cách dễ dàng và hiệu quả.

# **CHƯƠNG II: HTML & HTML5**

**(**Hyper Text Markup Language**)**

## **2.1. Tổng quan về HTML**

1. Thẻ (Tags): HTML sử dụng các thẻ để đánh dấu các phần tử trong trang web. Mỗi thẻ bao gồm một thẻ mở và một thẻ đóng, và nội dung được đặt giữa hai thẻ này. Ví dụ: `<tag> Nội dung </tag>`

2. Phần tử (Elements): Phần tử là một đơn vị trong HTML được định nghĩa bởi một cặp thẻ. Ví dụ, `<h1>` và `</h1>` tạo thành một phần tử tiêu đề.

3. Thuộc tính (Attributes): Thuộc tính cung cấp thông tin bổ sung về một phần tử. Chúng được thêm vào trong thẻ mở và có cú pháp `tên\_thuộc\_tính="giá\_trị"`. Ví dụ: `<img src="hinh\_anh.jpg" alt="Hình ảnh">`

4. Cấu trúc cơ bản của trang HTML: Một trang HTML thông thường bao gồm các phần chính như thẻ `<html>`, `<head>`, và `<body>`. Thẻ `<html>` là phần tử gốc và đóng vai trò bao bọc toàn bộ nội dung của trang. Thẻ `<head>` chứa các thông tin không hiển thị trực tiếp trên trình duyệt như tiêu đề trang, tệp CSS và JavaScript. Thẻ `<body>` chứa nội dung hiển thị trực tiếp trên trình duyệt.

5. Các phần tử phổ biến trong HTML: HTML cung cấp một loạt các phần tử để đánh dấu các phần tử phổ biến trên trang web như tiêu đề, đoạn văn bản, hình ảnh, liên kết, bảng, biểu đồ, biểu mẫu, video và âm thanh, v.v.

## **2.2. Cấu trúc tổng quát trang HTML**

Cấu trúc tổng quát của một trang HTML bao gồm các phần sau:

1. `<!DOCTYPE html>`: Khai báo loại tài liệu HTML.

2. `<html>`: Phần gốc của trang HTML, bao gồm toàn bộ nội dung của trang.

3. `<head>`: Phần đầu trang, chứa các thông tin không hiển thị trực tiếp trên trình duyệt như tiêu đề trang (`<title>`), các thẻ `<meta>` để định nghĩa các thông tin khác nhau về trang web, và các liên kết đến các tệp CSS và JavaScript khác.

4. `<body>`: Phần chính của trang web, chứa nội dung hiển thị trực tiếp trên trình duyệt.

5. `<header>`: Phần đầu trang, thường chứa tiêu đề, logo, menu và các yếu tố giao diện khác.

6. `<nav>`: Phần chứa các liên kết dẫn đến các trang khác trong trang web, ví dụ như thanh điều hướng.

7. `<main>`: Phần chính của nội dung trang web, thường chứa các phần tử như `<section>`, `<article>`, `<div>`, v.v.

8. `<section>`: Một phần của trang web, nằm trong phần `<main>`, có thể chứa các phần tử khác như tiêu đề (`<h1>`, `<h2>`, v.v.) và đoạn văn bản (`<p>`).

9. `<footer>`: Phần cuối trang, thường chứa thông tin bản quyền, liên hệ, liên kết đến trang chủ, v.v.

Đây chỉ là một cấu trúc tổng quát và có thể được tùy chỉnh theo yêu cầu cụ thể của từng trang web.

## **2.3 Một số thẻ HTML**

1. Các thẻ xử lý văn bản:

- Tiêu đề: Sử dụng các thẻ `<h1>`, `<h2>`, `<h3>`, ... `<h6>` để xác định tiêu đề và phân cấp tiêu đề trên trang web của bạn. Chọn tiêu đề phù hợp với mức độ ưu tiên và cấu trúc trang của bạn.

- Đoạn văn bản: Để định dạng các đoạn văn bản, sử dụng thẻ `<p>`. Đây là thẻ cơ bản để đánh dấu một đoạn văn bản.

- Văn bản nổi bật: Để làm nổi bật một phần của văn bản, bạn có thể sử dụng thẻ `<strong>` để làm đậm hoặc thẻ `<em>` để làm nghiêng văn bản. Điều này giúp tăng tính thẩm mỹ và tạo sự nhấn mạnh.

- Danh sách: Sử dụng thẻ `<ul>` để tạo danh sách không có thứ tự (danh sách dạng bullet) và thẻ `<ol>` để tạo danh sách có thứ tự (danh sách dạng số). Mỗi mục danh sách được định nghĩa bằng thẻ `<li>`.

- Đánh dấu phần văn bản: Sử dụng thẻ `<mark>` để đánh dấu một phần văn bản quan trọng hoặc để làm nổi bật trong văn bản.

2. Các thẻ tạo bảng:

- Bảng: Sử dụng thẻ `<table>` để tạo bảng trên trang web của bạn. Bảng được xây dựng dựa trên các hàng (`<tr>`) và các ô (`<td>` hoặc `<th>`).

- Tiêu đề cột: Để xác định tiêu đề cột trong bảng, sử dụng thẻ `<th>`. Thẻ `<th>` thường được đặt trong hàng đầu tiên của bảng.

- Ô dữ liệu: Để điền dữ liệu vào bảng, sử dụng thẻ `<td>`. Thẻ `<td>` thường được đặt trong các hàng (`<tr>`) của bảng.

- Tiêu đề bảng: Sử dụng thẻ `<caption>` để thêm tiêu đề cho bảng. Tiêu đề này sẽ xuất hiện phía trên bảng.

3. Thẻ liên kết:

- Liên kết: Sử dụng thẻ `<a>` để tạo liên kết đến trang web khác hoặc các tài nguyên khác trên cùng một trang. Đặt địa chỉ URL muốn liên kết vào thuộc tính `href` của thẻ `<a>`.

- Liên kết mở trong tab mới: Để mở liên kết trong một tab mới, thêm thuộc tính `target="\_blank"` vào thẻ `<a>`.

4. Các thẻ đa phương tiện:

- Hình ảnh: Để hiển thị hình ảnh trên trang web, sử dụng thẻ `<img>`. Đặt URL của hình ảnh vào thuộc tính `src` của thẻ `<img>`. Bạn cũng có thể thêm thuộc tính `alt` để cung cấp mô tả khi hình ảnh không thể hiển thị.

- Âm thanh: Sử dụng thẻ `<audio>` để nhúng âm thanh trên trang web của bạn. Đặt URL của tệp âm thanh vào thuộc tính `src` của thẻ `<audio>`. Bạn có thể sử dụng các thuộc tính như `controls` để hiển thị thanh điều khiển và `autoplay` để tự động phát âm thanh.

- Video: Sử dụng thẻ `<video>` để nhúng video trên trang web của bạn. Đặt URL của video vào thuộc tính `src` của thẻ `<video>`. Bạn có thể sử dụng các thuộc tính như `controls` để hiển thị thanh điều khiển và `autoplay` để tự động phát video.

5. Các phần tử khác:

- Định nghĩa thuật ngữ: Sử dụng thẻ `<dfn>` để định nghĩa một thuật ngữ hoặc cụm từ.

- Trích dẫn: Sử dụng thẻ `<blockquote>` để trích dẫn một đoạn văn bản dài. Thẻ `<blockquote>` thường được hiển thị với định dạng lề trái và thường có đường viền.

- Thẻ chú thích: Sử dụng thẻ `<abbr>` để xác định một từ viết tắt hoặc một từ viết tắt. Sử dụng thuộc tính `title` để cung cấp thông tin chú thích khi người dùng di chuột qua từ viết tắt.

- Ngắn gọn: Sử dụng thẻ `<hr>` để tạo một đường ngang ngắt quyền trên trang web của bạn. Đây thường được sử dụng để phân chia các phần khác nhau trên trang.

## **2.4. Các thẻ tạo biểu mẫu (form)**

Dưới đây là một số thẻ HTML thường được sử dụng để tạo biểu mẫu (form) trong trang web:

1. `<form>`: Thẻ `<form>` được sử dụng để tạo một biểu mẫu trên trang web. Nó là thẻ gốc cho tất cả các phần tử trong biểu mẫu và xác định cách dữ liệu được gửi đi sau khi người dùng hoàn thành biểu mẫu.

2. `<input>`: Thẻ `<input>` được sử dụng để tạo các ô nhập liệu trong biểu mẫu. Thẻ này có nhiều thuộc tính như `type` để xác định loại dữ liệu nhập, ví dụ như `text`, `password`, `email`, `checkbox`, `radio` và nhiều loại khác.

3. `<textarea>`: Thẻ `<textarea>` được sử dụng để tạo một ô nhập liệu văn bản đa dòng trong biểu mẫu. Thẻ này cho phép người dùng nhập nhiều dòng văn bản.

4. `<select>` và `<option>`: Thẻ `<select>` được sử dụng để tạo một danh sách thả xuống (dropdown list) trong biểu mẫu. Thẻ `<option>` được sử dụng để xác định các mục trong danh sách.

5. `<button>`: Thẻ `<button>` được sử dụng để tạo một nút trong biểu mẫu. Nút này có thể được sử dụng để gửi biểu mẫu hoặc thực hiện một hành động khác khi người dùng nhấp vào.

6. `<label>`: Thẻ `<label>` được sử dụng để tạo một nhãn cho các phần tử trong biểu mẫu, như ô nhập liệu (`<input>`) hoặc ô chọn (`<select>`). Nhãn giúp người dùng hiểu rõ hơn về ý nghĩa của các phần tử trong biểu mẫu.

7. `<fieldset>` và `<legend>`: Thẻ `<fieldset>` được sử dụng để nhóm các phần tử trong biểu mẫu thành một nhóm logic. Thẻ `<legend>` được sử dụng để xác định tiêu đề cho nhóm phần tử trong `<fieldset>`.

## **2.5. Một số thẻ HTML đặc biệt**

1. `<header>`: Thẻ `<header>` được sử dụng để xác định phần tiêu đề hoặc phần đầu trang của một trang web hoặc một phần tử. Thường được đặt ở đầu trang web và thường chứa tiêu đề, logo, menu điều hướng và các phần tử khác liên quan đến đầu trang.

2. `<footer>`: Thẻ `<footer>` được sử dụng để xác định phần chân trang hoặc phần cuối trang web hoặc một phần tử. Thường được đặt ở cuối trang web và thường chứa thông tin liên hệ, các liên kết hữu ích, bản quyền và các phần tử khác liên quan đến chân trang.

3. `<nav>`: Thẻ `<nav>` được sử dụng để xác định một khu vực chứa các liên kết điều hướng. Thường được đặt trong phần tiêu đề hoặc chân trang của một trang web và chứa các liên kết đến các trang khác hoặc các phần của trang hiện tại.

4. `<section>`: Thẻ `<section>` được sử dụng để nhóm các phần tử liên quan về một chủ đề cụ thể trong một trang web. Thẻ này giúp tạo cấu trúc và tổ chức nội dung trên trang.

5. `<article>`: Thẻ `<article>` được sử dụng để xác định một nội dung độc lập, tức là một phần tử có thể tồn tại và hoạt động một cách độc lập với các phần tử khác trong trang. Thích hợp sử dụng cho các bài viết, tin tức, blog và các nội dung tương tự.

6. `<aside>`: Thẻ `<aside>` được sử dụng để xác định một nội dung phụ, thường được hiển thị bên cạnh nội dung chính. Thích hợp để chứa các thông tin bổ sung, quảng cáo, liên kết liên quan hoặc các phần tử khác không quan trọng nhưng có ích cho nội dung chính.

7. `<figure>` và `<figcaption>`: Thẻ `<figure>` được sử dụng để nhóm các phần tử liên quan, chẳng hạn như một hình ảnh, đồ thị, hoặc video. Thẻ `<figcaption>` được sử dụng để xác định chú thích hoặc mô tả cho các phần tử được nhóm trong thẻ `<figure>`.

8. `<time>`: Thẻ `<time>` được sử dụng để định dạng và hiển thị thông tin về thời gian hoặc ngày tháng. Thẻ này hữu ích khi bạn muốn hiển thị ngày đăng, thời gian cập nhật, thời gian bắt đầu sự kiện, và các thông tin thời gian khác.

**2.6. HTML5**

HTML5 là phiên bản mới nhất của ngôn ngữ đánh dấu siêu văn bản (HTML), được phát triển và chuẩn hóa bởi World Wide Web Consortium (W3C). HTML5 cung cấp nhiều tính năng mới và cải tiến so với các phiên bản HTML trước đó, giúp phát triển ứng dụng web mạnh mẽ, linh hoạt và tương thích trên nhiều thiết bị.

1. DOCTYPE: Trong HTML5, không cần khai báo DOCTYPE dài dòng như trong các phiên bản HTML trước đó. Thay vào đó, chỉ cần sử dụng DOCTYPE ngắn gọn sau đây ở đầu tài liệu HTML5:

```html

<!DOCTYPE html>

```

2. Thẻ ngữ nghĩa: HTML5 giới thiệu nhiều thẻ ngữ nghĩa mới giúp xác định ý nghĩa và cấu trúc của các phần tử trên trang web. Một số thẻ ngữ nghĩa phổ biến bao gồm: `<header>`, `<nav>`, `<section>`, `<article>`, `<aside>`, `<footer>`, `<main>`, `<figure>`, `<figcaption>`, `<time>` và nhiều thẻ khác.

3. Webform: HTML5 cung cấp các tính năng nâng cao cho việc xây dựng biểu mẫu (form) trên trang web. Các thành phần như `<input>`, `<textarea>`, `<select>` và các loại input mới như `<email>`, `<url>`, `<date>` được hỗ trợ để kiểm soát dữ liệu đầu vào. HTML5 cũng hỗ trợ các thuộc tính mới như `placeholder`, `required`, `pattern` và hỗ trợ kiểm tra và xử lý dữ liệu nhập từ phía máy khách (client-side) bằng JavaScript.

## **Kết Luận Chương 2**

HTML là ngôn ngữ đánh dấu siêu văn bản được sử dụng để xây dựng và cấu trúc trang web. Nó sử dụng các thẻ để đánh dấu và xác định ý nghĩa của các phần tử trên trang web. HTML5 là phiên bản mới nhất của HTML, được phát triển và chuẩn hóa bởi W3C. HTML5 mang đến nhiều tính năng và cải tiến, bao gồm cú pháp khác biệt, khai báo DOCTYPE ngắn gọn, thẻ ngữ nghĩa để xác định ý nghĩa của các phần tử, hỗ trợ webform mạnh mẽ, tích hợp đa phương tiện, hỗ trợ đồ họa và đồ thị, tích hợp tốt với JavaScript, hỗ trợ Responsive Web Design và SEO. HTML5 mang đến sự linh hoạt và khả năng tương thích trên nhiều thiết bị, giúp phát triển trang web hiện đại và tương tác.

# **CHƯƠNG III: CSS và CSS3**

 (Cascading Style Sheets)

## **3.1. CSS là gì?**

CSS là viết tắt của "Cascading Style Sheets" (Bảng kiểu theo phong cách rơi). Nó là một ngôn ngữ đánh dấu được sử dụng để mô tả cách các phần tử trên trang web được hiển thị trên trình duyệt. CSS cho phép bạn kiểm soát các khía cạnh về màu sắc, font chữ, kích thước, khoảng cách và bố cục của các phần tử trên trang web.

CSS hoạt động bằng cách chọn các phần tử HTML và áp dụng các quy tắc kiểu cho chúng. Các quy tắc kiểu CSS bao gồm các thuộc tính và giá trị, ví dụ như màu sắc, độ rộng, độ cao, đường viền, định dạng chữ, vị trí và hiệu ứng. Bằng cách sử dụng CSS, bạn có thể thay đổi diện mạo của các phần tử trên trang web một cách linh hoạt và dễ dàng.

CSS được sử dụng phổ biến trong phát triển web để tạo ra các trang web có giao diện đẹp, nhất quán và dễ đọc. Nó cho phép phân tách nội dung HTML và kiểu dáng, giúp cải thiện khả năng bảo trì và mở rộng của trang web.

## **3.2. Cú pháp CSS**

Cú pháp CSS (Cascading Style Sheets) là ngôn ngữ được sử dụng để định dạng và trình bày các phần tử trên trang web. Dưới đây là cú pháp cơ bản của CSS:

1. Selector:

Selector là phần xác định phần tử HTML mà bạn muốn áp dụng các quy tắc CSS. Có nhiều loại selector như:

- Selector phần tử: áp dụng cho tất cả các phần tử có cùng tên.

- Selector ID: áp dụng cho phần tử có ID duy nhất.

- Selector lớp: áp dụng cho các phần tử có cùng tên lớp.

- Selector thuộc tính: áp dụng cho các phần tử có thuộc tính cụ thể.

2. Khối khai báo thuộc tính:

Mỗi khối khai báo thuộc tính trong CSS bao gồm một hoặc nhiều cặp thuộc tính và giá trị. Cú pháp của một khối khai báo thuộc tính như sau:

```css

selector {

property: value;

property: value;

}

```

- Property: Thuộc tính CSS mà bạn muốn đặt giá trị cho.

- Value: Giá trị của thuộc tính.

3. Bình luận:

Bạn có thể sử dụng bình luận trong CSS để thêm ghi chú hoặc giải thích cho đoạn mã. Bình luận bắt đầu bằng `/\*` và kết thúc bằng `\*/`.

```css

/\* Đây là bình luận \*/

```

Đây chỉ là một cú pháp cơ bản của CSS. CSS còn có rất nhiều thuộc tính, pseudo-classes, pseudo-elements và khái niệm kế thừa khác để tạo kiểu cho trang web.

## **3.3. Áp dụng CSS vào trang HTML**

Có nhiều cách để áp dụng CSS vào trang HTML. Bạn có thể sử dụng inline styles bằng cách thêm thuộc tính `style` trực tiếp vào các phần tử HTML. Bạn cũng có thể sử dụng CSS internal bằng cách thêm một phần tử `<style>` vào phần head của trang HTML. Ngoài ra, việc tạo một file CSS riêng và liên kết nó với trang HTML thông qua một phần tử `<link>` cũng là một cách phổ biến để áp dụng CSS.

## **3.4. Selectors**

Trình tự (selectors) trong CSS được sử dụng để xác định các phần tử HTML mà bạn muốn áp dụng CSS cho chúng. Dưới đây là một số loại trình tự phổ biến:

1. Selector phần tử (Element Selector): Chọn các phần tử dựa trên tên thẻ HTML của chúng. Ví dụ, để chọn tất cả các phần tử `<p>`, bạn sử dụng selector `p`.

2. Selector lớp (Class Selector): Chọn các phần tử dựa trên tên lớp của chúng. Bạn định nghĩa tên lớp bằng cách sử dụng thuộc tính `class` trong phần tử HTML và sau đó sử dụng `.` trước tên lớp để tạo selector. Ví dụ, để chọn tất cả các phần tử có lớp là "highlight", bạn sử dụng selector `.highlight`.

3. Selector ID: Chọn một phần tử duy nhất dựa trên thuộc tính `id` của nó. Bạn định nghĩa một ID bằng cách sử dụng thuộc tính `id` trong phần tử HTML và sau đó sử dụng `#` trước tên ID để tạo selector. Ví dụ, để chọn phần tử có ID là "header", bạn sử dụng selector `#header`.

4. Selector con (Descendant Selector): Chọn các phần tử con bên trong một phần tử khác. Bạn sử dụng khoảng trắng để chỉ định mối quan hệ chồng chéo giữa các phần tử. Ví dụ, để chọn tất cả các phần tử `<li>` bên trong một phần tử `<ul>`, bạn sử dụng selector `ul li`.

5. Selector con trực tiếp (Child Selector): Chọn các phần tử con trực tiếp của một phần tử khác. Bạn sử dụng dấu `>` để chỉ định mối quan hệ trực tiếp giữa các phần tử. Ví dụ, để chọn tất cả các phần tử `<li>` là con trực tiếp của một phần tử `<ul>`, bạn sử dụng selector `ul > li`.

Đây chỉ là một số trình tự phổ biến, và có nhiều loại trình tự khác nhau trong CSS. Bạn có thể kết hợp các trình tự này và sử dụng các phương thức khác nhau để chọn các phần tử một cách linh hoạt và chi tiết.

### *3.4.1. Universal selector*: Sử dụng dấu `\*` để chọn tất cả các phần tử trên trang.

Universal selector (trình tự toàn cục) trong CSS là một trình tự đặc biệt được sử dụng để chọn tất cả các phần tử trên trang web. Nó được biểu thị bằng ký hiệu dấu sao (\*) và được đặt làm selector đầu tiên trong quy tắc CSS.

Khi bạn sử dụng trình tự toàn cục, nó sẽ áp dụng các quy tắc CSS cho tất cả các phần tử trên trang web, mà không cần xác định tên thẻ, lớp hay ID cụ thể.

Ví dụ, nếu bạn muốn đặt màu chữ mặc định cho tất cả các phần tử trên trang là màu đỏ, bạn có thể sử dụng trình tự toàn cục như sau:

```css

\* {

color: red;

}

```

Trong ví dụ trên, mọi phần tử trên trang web sẽ có màu chữ đỏ do quy tắc CSS được áp dụng cho trình tự toàn cục.

Tuy nhiên, cần lưu ý rằng việc sử dụng trình tự toàn cục có thể ảnh hưởng đến hiệu suất của trang web, vì nó áp dụng quy tắc cho tất cả các phần tử, bao gồm cả các phần tử con và phần tử chưa được sử dụng. Vì vậy, nếu không cần thiết, nên sử dụng trình tự toàn cục một cách cẩn thận và hợp lý.

### *3.4.2. Type selector*: Sử dụng tên thẻ HTML để chọn tất cả các phần tử có cùng loại thẻ.

Type selector (trình tự loại) trong CSS là một loại trình tự sử dụng để chọn các phần tử dựa trên tên thẻ HTML của chúng. Nó được biểu thị bằng tên thẻ HTML mà bạn muốn chọn.

Ví dụ, nếu bạn muốn áp dụng một quy tắc CSS cho tất cả các phần tử `<p>` trên trang web, bạn có thể sử dụng trình tự loại như sau:

```css

p {

color: blue;

font-size: 16px;

}

```

Trong ví dụ trên, tất cả các phần tử `<p>` trên trang web sẽ có màu chữ là màu xanh và kích thước chữ là 16px.

Bạn cũng có thể kết hợp trình tự loại với các trình tự khác để chọn các phần tử cụ thể hơn. Ví dụ, để chọn các phần tử `<p>` trong một phần tử `<div>` có lớp là "container", bạn có thể sử dụng trình tự loại và trình tự lớp như sau:

```css

div.container p {

color: blue;

font-size: 16px;

}

```

Trong ví dụ trên, chỉ các phần tử `<p>` nằm trong phần tử `<div>` có lớp là "container" mới được áp dụng quy tắc CSS.

Trình tự loại là một phương pháp chọn phổ biến và mạnh mẽ trong CSS, cho phép bạn kiểm soát kiểu dáng của các phần tử dựa trên tên thẻ HTML của chúng.

### *3.4.3. Identity selector*

Identity selector (trình tự nhận dạng) trong CSS là một loại trình tự sử dụng để chọn một phần tử duy nhất dựa trên thuộc tính `id` của nó. Nó được biểu thị bằng cách sử dụng ký hiệu `#` trước tên `id` của phần tử.

Ví dụ, nếu bạn muốn áp dụng một quy tắc CSS cho một phần tử có `id` là "header", bạn có thể sử dụng trình tự nhận dạng như sau:

```css

#header {

background-color: gray;

color: white;

font-size: 24px;

}

```

Trong ví dụ trên, phần tử có `id` là "header" sẽ có màu nền là màu xám, màu chữ là màu trắng và kích thước chữ là 24px.

Một điểm quan trọng khi sử dụng trình tự nhận dạng là mỗi `id` phải là duy nhất trong tài liệu HTML. Điều này đảm bảo rằng chỉ có một phần tử duy nhất sẽ được chọn khi sử dụng trình tự nhận dạng.

Trình tự nhận dạng cho phép bạn áp dụng kiểu dáng hoặc các quy tắc CSS cụ thể cho một phần tử duy nhất trên trang web, giúp tùy chỉnh và định hình kiểu dáng của phần tử đó một cách riêng biệt.

### *3.4.4. Class selector:*

Class selector (trình tự lớp) trong CSS là một loại trình tự sử dụng để chọn các phần tử dựa trên tên lớp của chúng. Nó được biểu thị bằng ký hiệu dấu chấm (`.`) trước tên lớp.

Để sử dụng trình tự lớp, bạn cần đặt tên lớp cho các phần tử HTML bằng cách sử dụng thuộc tính `class`. Sau đó, bạn sử dụng trình tự lớp để chọn các phần tử có cùng tên lớp đó.

Ví dụ, nếu bạn muốn áp dụng một quy tắc CSS cho tất cả các phần tử có lớp là "highlight", bạn có thể sử dụng trình tự lớp như sau:

```css

.highlight {

background-color: yellow;

color: black;

}

```

Trong ví dụ trên, tất cả các phần tử có lớp là "highlight" sẽ có màu nền là màu vàng và màu chữ là màu đen.

Bạn cũng có thể sử dụng trình tự lớp cùng với các trình tự khác để chọn các phần tử cụ thể hơn. Ví dụ, để chọn các phần tử `<p>` có lớp là "highlight" trong một phần tử `<div>` có lớp là "container", bạn có thể sử dụng trình tự lớp và trình tự con như sau:

```css

div.container p.highlight {

background-color: yellow;

color: black;

}

```

Trong ví dụ trên, chỉ các phần tử `<p>` có lớp là "highlight" nằm trong phần tử `<div>` có lớp là "container" mới được áp dụng quy tắc CSS.

Trình tự lớp cho phép bạn tùy chỉnh kiểu dáng của các phần tử dựa trên lớp của chúng, cho phép bạn áp dụng các kiểu dáng tương tự cho nhiều phần tử khác nhau trên trang web của bạn.

### *3.4.5. Descendant selector:*

Descendant selector (trình tự con) trong CSS là một loại trình tự sử dụng để chọn các phần tử con nằm trong một phần tử cha cụ thể. Nó được biểu thị bằng cách sử dụng khoảng trắng để phân tách giữa các phần tử cha và con.

Ví dụ, nếu bạn muốn áp dụng một quy tắc CSS cho tất cả các phần tử `<a>` nằm trong một phần tử `<div>`, bạn có thể sử dụng trình tự con như sau:

```css

div a {

color: blue;

text-decoration: underline;

}

```

Trong ví dụ trên, tất cả các phần tử `<a>` nằm trong một phần tử `<div>` sẽ có màu chữ là màu xanh và có gạch chân.

Bạn cũng có thể sử dụng trình tự con liên tiếp để chọn các phần tử con sâu hơn. Ví dụ, để chọn các phần tử `<span>` nằm trong một phần tử `<div>` nằm trong một phần tử `<section>`, bạn có thể sử dụng trình tự con liên tiếp như sau:

```css

section div span {

font-weight: bold;

color: red;

}

```

Trong ví dụ trên, tất cả các phần tử `<span>` nằm trong một phần tử `<div>` nằm trong một phần tử `<section>` sẽ có độ dày văn bản là đậm và màu chữ là màu đỏ.

Trình tự con cho phép bạn chọn các phần tử con cụ thể trong cấu trúc cây DOM của trang web và áp dụng kiểu dáng hoặc các quy tắc CSS cho chúng.

### *3.4.6. Child selector:*

Child selector (trình tự con trực tiếp) trong CSS là một loại trình tự sử dụng để chọn các phần tử con trực tiếp nằm trong một phần tử cha cụ thể. Nó được biểu thị bằng ký hiệu dấu nhân (`>`) giữa phần tử cha và phần tử con.

Ví dụ, nếu bạn muốn áp dụng một quy tắc CSS cho tất cả các phần tử `<li>` là con trực tiếp của một phần tử `<ul>`, bạn có thể sử dụng trình tự con trực tiếp như sau:

```css

ul > li {

list-style-type: square;

}

```

Trong ví dụ trên, chỉ các phần tử `<li>` là con trực tiếp của phần tử `<ul>` mới sẽ có kiểu danh sách là hình vuông.

Trình tự con trực tiếp chỉ áp dụng cho các phần tử con trực tiếp của phần tử cha và không áp dụng cho các phần tử con sâu hơn trong cây DOM. Nó chỉ chọn các phần tử có mức độ lồng ghép trực tiếp với phần tử cha.

Lưu ý rằng trình tự con trực tiếp khác với trình tự con (descendant selector). Trình tự con trực tiếp chỉ chọn phần tử con trực tiếp, trong khi trình tự con chọn tất cả các phần tử con bất kể mức độ lồng ghép trong cây DOM.

Trình tự con trực tiếp là một công cụ hữu ích để tùy chỉnh kiểu dáng hoặc áp dụng quy tắc CSS cho các phần tử con trực tiếp của một phần tử cha cụ thể trên trang web.

### 3.4.7. Adjacent sibling selector:

Adjacent sibling selector (trình tự anh em kế tiếp) trong CSS là một loại trình tự sử dụng để chọn phần tử anh em trực tiếp kế tiếp của một phần tử cụ thể. Nó được biểu thị bằng ký hiệu dấu cộng (`+`) giữa hai phần tử.

Ví dụ, nếu bạn muốn áp dụng một quy tắc CSS cho phần tử `<h2>` ngay sau phần tử `<h1>`, bạn có thể sử dụng trình tự anh em kế tiếp như sau:

```css

h1 + h2 {

color: red;

}

```

Trong ví dụ trên, phần tử `<h2>` ngay sau phần tử `<h1>` sẽ có màu chữ là màu đỏ.

Trình tự anh em kế tiếp chỉ chọn phần tử anh em trực tiếp, có cùng mức độ lồng ghép và ngay sau phần tử cụ thể. Nó không áp dụng cho các phần tử anh em kế tiếp không trực tiếp hoặc không cùng mức độ lồng ghép.

Lưu ý rằng trình tự anh em kế tiếp khác với trình tự con trực tiếp (child selector). Trình tự anh em kế tiếp chọn phần tử đứng ngay sau phần tử cụ thể, trong khi trình tự con trực tiếp chọn các phần tử con trực tiếp của phần tử cụ thể.

Trình tự anh em kế tiếp là một công cụ hữu ích để tùy chỉnh kiểu dáng hoặc áp dụng quy tắc CSS cho các phần tử anh em trực tiếp kế tiếp của một phần tử cụ thể trên trang web.

### 3.4.8. Attribute selector:

Attribute selector (trình tự thuộc tính) trong CSS là một loại trình tự sử dụng để chọn các phần tử dựa trên các thuộc tính và giá trị của chúng. Nó cho phép bạn chọn các phần tử dựa trên các thuộc tính như `id`, `class`, hay các thuộc tính tùy chỉnh khác.

Có ba kiểu trình tự thuộc tính trong CSS: trình tự thuộc tính tồn tại (`[attribute]`), trình tự thuộc tính với giá trị cụ thể (`[attribute=value]`), và trình tự thuộc tính với giá trị bắt đầu, kết thúc, hoặc chứa một chuỗi giá trị (`[attribute^=value]`, `[attribute$=value]`, `[attribute\*=value]`).

Dưới đây là một số ví dụ về cách sử dụng trình tự thuộc tính:

1. Trình tự thuộc tính tồn tại (`[attribute]`): Chọn các phần tử có thuộc tính được chỉ định, bất kể giá trị của thuộc tính đó là gì. Ví dụ:

```css

a[href] {

color: blue;

}

```

Trong ví dụ trên, tất cả các phần tử `<a>` có thuộc tính `href` sẽ có màu chữ là màu xanh.

2. Trình tự thuộc tính với giá trị cụ thể (`[attribute=value]`): Chọn các phần tử có thuộc tính và giá trị cụ thể. Ví dụ:

```css

input[type="text"] {

border: 1px solid black;

}

```

Trong ví dụ trên, tất cả các phần tử `<input>` có thuộc tính `type` và giá trị là `"text"` sẽ có viền là đường viền đen.

3. Trình tự thuộc tính với giá trị bắt đầu, kết thúc, hoặc chứa một chuỗi giá trị (`[attribute^=value]`, `[attribute$=value]`, `[attribute\*=value]`): Chọn các phần tử có thuộc tính và giá trị bắt đầu, kết thúc, hoặc chứa một chuỗi giá trị cụ thể. Ví dụ:

```css

img[src^="https://"] {

border: 2px solid green;

}

```

Trong ví dụ trên, tất cả các phần tử `<img>` có thuộc tính `src` và giá trị bắt đầu bằng `"https://"` sẽ có viền là đường viền màu xanh lá cây.

Trình tự thuộc tính cho phép bạn chọn các phần tử dựa trên các thuộc tính và giá trị của chúng, giúp bạn tùy chỉnh kiểu dáng hoặc áp dụng các quy tắc CSS cho các phần tử cụ thể trên trang web.

### *3.4.9. Pseudo class selector*

### Pseudo class selector (trình tự giả lập lớp) trong CSS là một loại trình tự sử dụng để chọn các phần tử dựa trên trạng thái hoặc tình huống cụ thể của chúng. Pseudo class selectors được kết hợp với các yếu tố hoặc trình tự khác để xác định các phần tử cần chọn.

### Có nhiều loại pseudo class selector trong CSS, bao gồm:

### 1. Pseudo class selector dựa trên trạng thái: Chọn các phần tử dựa trên trạng thái hiện tại của chúng, ví dụ như khi rê chuột vào, khi focus, khi checked, và nhiều trạng thái khác. Ví dụ:

### ```css

### a:hover {

### color: red;

### }

### ```

### Trong ví dụ trên, khi rê chuột vào phần tử `<a>`, màu chữ của nó sẽ chuyển sang màu đỏ.

### 2. Pseudo class selector dựa trên vị trí: Chọn các phần tử dựa trên vị trí của chúng trong một danh sách hoặc một phần tử cha. Ví dụ:

### ```css

### li:first-child {

### font-weight: bold;

### }

### ```

### Trong ví dụ trên, phần tử `<li>` đầu tiên trong danh sách sẽ có độ dày văn bản là đậm.

### 3. Pseudo class selector dựa trên nội dung: Chọn các phần tử dựa trên nội dung hoặc thuộc tính của chúng. Ví dụ:

### ```css

### input[type="text"]:required {

### border: 1px solid red;

### }

### ```

### Trong ví dụ trên, các phần tử `<input>` có thuộc tính `type` là `"text"` và được yêu cầu (required) sẽ có viền là đường viền màu đỏ.

### Các pseudo class selector cho phép bạn chọn và áp dụng các quy tắc CSS cho các phần tử dựa trên trạng thái, vị trí hoặc nội dung của chúng. Điều này giúp bạn tạo ra các hiệu ứng tương tác và tùy chỉnh kiểu dáng cho các phần tử trong trang web của bạn.

### 3.4.10. Group selector:

Group selector (trình tự nhóm) trong CSS là một cách để áp dụng các quy tắc CSS cho nhiều phần tử cùng một lúc. Bằng cách sử dụng trình tự nhóm, bạn có thể chọn và áp dụng các quy tắc CSS cho một nhóm phần tử cụ thể mà không cần viết lại các quy tắc cho từng phần tử riêng lẻ.

Để sử dụng trình tự nhóm, bạn chỉ cần liệt kê các phần tử mà bạn muốn chọn và áp dụng quy tắc CSS cho chúng, và phân tách chúng bằng dấu phẩy (,). Ví dụ:

```css

h1, h2, h3 {

color: blue;

}

```

Trong ví dụ trên, các phần tử `<h1>`, `<h2>`, và `<h3>` sẽ có màu chữ là màu xanh.

Bạn cũng có thể kết hợp trình tự nhóm với các trình tự khác để chọn các phần tử cụ thể hoặc áp dụng quy tắc CSS theo yêu cầu của bạn. Ví dụ:

```css

ul.nav li, .sidebar p {

font-size: 14px;

}

```

Trong ví dụ trên, các phần tử `<li>` trong danh sách có lớp `nav` và các phần tử `<p>` trong phần sidebar sẽ có kích thước chữ là 14px.

Trình tự nhóm là một cách tiện lợi để áp dụng các quy tắc CSS cho nhiều phần tử cùng một lúc, giúp tiết kiệm thời gian và viết code hiệu quả hơn.

## **3.5. Đơn vị đo lường CSS**

Trong CSS, có nhiều đơn vị đo lường khác nhau để xác định kích thước và khoảng cách của các phần tử trên trang web. Dưới đây là một số đơn vị đo lường phổ biến:

1. Đơn vị đo tuyệt đối:

- Pixel (px): Đơn vị đo tuyệt đối thường được sử dụng. Một pixel tương đương với một điểm ảnh trên màn hình.

- Inch (in): Đơn vị đo dựa trên inch (1 inch = 2.54 cm).

- Centimeter (cm): Đơn vị đo dựa trên centimeter.

2. Đơn vị đo tương đối:

- Percentage (%): Đơn vị đo tương đối so với phần tử chứa nó.

- Em (em): Đơn vị đo tương đối so với kích thước phông chữ của phần tử cha trực tiếp.

- Rem (rem): Đơn vị đo tương đối so với kích thước phông chữ của phần tử gốc (root).

3. Đơn vị đo khác:

- Viewport Width (vw): Đơn vị đo tương đối dựa trên chiều rộng của viewport (không tính thanh cuộn).

- Viewport Height (vh): Đơn vị đo tương đối dựa trên chiều cao của viewport (không tính thanh cuộn).

- Viewport Min/Max (vmin/vmax): Đơn vị đo tương đối dựa trên giá trị nhỏ nhất hoặc lớn nhất giữa chiều rộng và chiều cao của viewport.

## **3.6. Kế thừa thuộc tính**

Trong CSS, thuộc tính có thể được kế thừa từ phần tử cha sang các phần tử con bên trong nó. Điều này có nghĩa là khi một phần tử con không có giá trị cho một thuộc tính cụ thể, nó sẽ thừa hưởng giá trị của thuộc tính từ phần tử cha.

Tuy nhiên, không phải tất cả các thuộc tính đều được kế thừa. Một số thuộc tính phổ biến có tính kế thừa và không kế thừa như sau:

1. Thuộc tính kế thừa:

- color

- font-family

- font-size

- font-weight

- text-align

- line-height

- ...

2. Thuộc tính không kế thừa:

- width

- height

- margin

- padding

- background-color

- position

- ...

Để xác định rõ liệu một thuộc tính có được kế thừa hay không, bạn có thể kiểm tra tài liệu hướng dẫn của mỗi thuộc tính cụ thể. Ngoài ra, bạn cũng có thể sử dụng giá trị "inherit" để tường minh chỉ định rằng một thuộc tính nên kế thừa giá trị từ phần tử cha.

## **3.7. Các nhóm thuộc tính trong CSS**

- Thuộc tính định dạng font chữ, văn bản (font, text): Bao gồm các thuộc tính như `font-family` (kiểu font chữ), `font-size` (kích thước font chữ), `font-weight` (độ đậm), `text-align` (canh lề văn bản), và `text-decoration` (gạch chân, gạch ngang văn bản).

- Thuộc tính định dạng nền (background): Bao gồm các thuộc tính như `background-color` (màu nền), `background-image` (hình ảnh nền), `background-repeat` (lặp lại hình ảnh), và `background-position` (vị trí hình ảnh nền).

- Mô hình hộp (box model): Bao gồm các thuộc tính như `width` (chiều rộng), `height` (chiều cao), `margin` (phần lề ngoài), `padding` (phần lề bên trong), và `border` (viền).

- Các thuộc tính định margin, padding, border: Bao gồm các thuộc tính như `margin-top`, `margin-right`, `margin-bottom`, `margin-left` (phần lề ngoài), `padding-top`, `padding-right`, `padding-bottom`, `padding-left` (phần lề bên trong), và `border-width`, `border-style`, `border-color` (viền).

- Thuộc tính định dạng cách hiển thị (display): Bao gồm các thuộc tính như `display` (cách hiển thị), `visibility` (tính hiển thị), `float` (trôi), và `clear` (xóa trôi).

- Thuộc tính xác định vị trí (position): Bao gồm các thuộc tính như `position` (vị trí), `top`, `right`, `bottom`, `left` (vị trí đối tượng).

## **3.8. Float & Clear**

Float và Clear là hai thuộc tính trong CSS được sử dụng để điều chỉnh vị trí và xếp chồng các phần tử trên trang web. Dưới đây là mô tả về Float và Clear:

1. Float:

Thuộc tính `float` trong CSS cho phép bạn đặt một phần tử để nổi lề trái hoặc lề phải của phần tử cha của nó. Khi một phần tử được float, nó sẽ di chuyển sang cạnh trái hoặc cạnh phải của phần tử cha và các phần tử khác trong cùng phần tử cha sẽ chồng lên nó.

Cú pháp:

```css

.example-element {

float: left | right | none;

}

```

- `left`: Đặt phần tử nổi lề trái.

- `right`: Đặt phần tử nổi lề phải.

- `none` (giá trị mặc định): Không áp dụng float cho phần tử.

Khi sử dụng float, phần tử cha sẽ không giữ ô bao quanh phần tử float, dẫn đến sự thay đổi về bố cục và vị trí của các phần tử khác trong cùng phần tử cha.

2. Clear:

Thuộc tính `clear` trong CSS được sử dụng để xác định liệu một phần tử sẽ phải chạy dọc theo phần tử float (không được chồng lên float) hay không. Nó được sử dụng để khắc phục tình trạng các phần tử sau phần tử float bị chồng lên float.

Cú pháp:

```css

.example-element {

clear: left | right | both | none;

}

```

- `left`: Chạy dọc theo phần tử float bên trái.

- `right`: Chạy dọc theo phần tử float bên phải.

- `both`: Chạy dọc theo cả hai phần tử float bên trái và bên phải.

- `none` (giá trị mặc định): Không áp dụng clear.

Bằng cách sử dụng clear, bạn có thể đảm bảo rằng các phần tử sau phần tử float sẽ không bị chồng lên float và sẽ hiển thị đúng vị trí của chúng.

Lưu ý rằng việc sử dụng float có thể ảnh hưởng đến bố cục tổng thể của trang web và có thể gây ra các vấn đề về trình tự đọc trang và khả năng phản hồi. Do đó, các phương pháp bố cục hiện đại như flexbox và grid layout thường được ưu tiên hơn float trong thiết kế web hiện đại.

## **3.9. Flex**

Để sử dụng Flexbox, bạn cần định nghĩa một container (thường là một phần tử cha) và áp dụng thuộc tính `display: flex;` cho container đó. Sau đó, các phần tử con bên trong container sẽ được xếp theo cách mà bạn đã chỉ định.

Các thuộc tính quan trọng trong Flexbox bao gồm:

- `flex-direction`: Xác định hướng xếp của các phần tử con trong container. Có các giá trị như `row` (xếp theo hàng mặc định), `column` (xếp theo cột), `row-reverse`, `column-reverse`.

- `flex-wrap`: Xác định cách xếp các phần tử con khi không đủ không gian trong container. Có các giá trị như `nowrap` (không xếp hàng mới), `wrap` (xếp hàng mới khi cần), `wrap-reverse`.

- `justify-content`: Điều chỉnh căn chỉnh ngang của các phần tử con trong container. Có các giá trị như `flex-start` (trái), `flex-end` (phải), `center` (giữa), `space-between` (căn đều khoảng cách giữa các phần tử), `space-around` (căn đều khoảng cách xung quanh các phần tử).

- `align-items`: Điều chỉnh căn chỉnh dọc của các phần tử con trong container. Có các giá trị như `flex-start` (đỉnh), `flex-end` (đáy), `center` (giữa), `baseline` ( căn theo dòng cơ bản), `stretch` (kéo căng các phần tử để phù hợp với chiều cao container).

- `align-content`: Điều chỉnh căn chỉnh dọc của các hàng trong container khi có nhiều hàng. Có các giá trị như `flex-start` (đỉnh), `flex-end` (đáy), `center` (giữa), `space-between` (căn đều khoảng cách giữa các hàng), `space-around` (căn đều khoảng cách xung quanh các hàng), `stretch` (kéo căng các hàng để phù hợp với chiều cao container).

Ngoài ra, các phần tử con trong container có thể sử dụng các thuộc tính flex để điều chỉnh kích thước và sự phân bổ không gian, bao gồm `flex-grow`, `flex-shrink`, và `flex-basis`.

Flexbox là một công cụ mạnh mẽ trong CSS3 cho phép xây dựng bố cục linh hoạt và phản hồi cho các trang web và ứng dụng web.

## **3.10. Grid**

Để sử dụng Grid, bạn cần định nghĩa một container (thường là một phần tử cha) và áp dụng thuộc tính `display: grid;` cho container đó. Sau đó, bạn có thể xác định cấu trúc lưới bằng cách sử dụng các thuộc tính như `grid-template-columns`, `grid-template-rows`, và `grid-template-areas`.

Các thuộc tính quan trọng trong Grid bao gồm:

- `grid-template-columns` và `grid-template-rows`: Xác định kích thước và tỷ lệ của các cột và hàng trong lưới. Bạn có thể sử dụng các giá trị như pixel (`px`), phần trăm (`%`), hoặc các đơn vị linh hoạt như `fr` (chia tỷ lệ).

- `grid-template-areas`: Xác định vị trí của các ô trong lưới bằng cách đặt tên cho các ô và sắp xếp chúng thành các mẫu (patterns). Bạn có thể sử dụng các tên ô và dấu chấm (`.`) để biểu thị ô trống.

- `grid-gap` hoặc `gap`: Xác định khoảng cách giữa các ô trong lưới. Bạn có thể chỉ định khoảng cách ngang (`grid-column-gap`) và khoảng cách dọc (`grid-row-gap`), hoặc chỉ sử dụng `gap` để đồng thời đặt cả hai khoảng cách.

- `grid-column` và `grid-row`: Xác định vị trí và kích thước của một phần tử trong lưới. Bạn có thể sử dụng các giá trị như số cột/hàng (`1`, `2`,...) hoặc tên ô đã xác định (`header`, `sidebar`,...). Bạn cũng có thể sử dụng các giá trị như `span` để mở rộng phần tử qua nhiều cột/hàng.

Ngoài ra, Grid còn cung cấp các thuộc tính khác như `justify-items`, `align-items`, `justify-content`, và `align-content` để căn chỉnh và căn lề các phần tử trong ô.

Grid là một công cụ mạnh mẽ trong CSS, cho phép bạn tạo ra các bố cục phức tạp và tùy chỉnh theo ý muốn. Nó cung cấp khả năng linh hoạt và kiểm soát cao hơn so với các hệ thống xếp lưới truyền thống như Floating hoặc Flexbox.

## **3.11 CSS3:**

- CSS3 là phiên bản thứ ba của CSS (Cascading Style Sheets), là một ngôn ngữ được sử dụng để định dạng và trình bày các phần tử trên trang web. CSS3 là phiên bản tiếp theo sau CSS2 và đưa vào sử dụng nhiều tính năng mới và cải tiến so với CSS2.

- CSS3 Selectors: CSS3 cung cấp những bộ chọn (selectors) mới giúp lựa chọn các phần tử dựa trên các tiêu chí khác nhau như lớp (class), id, các quan hệ giữa các phần tử, thuộc tính, v.v. Ví dụ: :nth-child(), :not(), :first-child, v.v.

- CSS3 Rounded Corners: CSS3 cho phép làm tròn góc của các phần tử bằng thuộc tính `border-radius`. Điều này cho phép tạo ra các hình dạng bo góc tròn hoặc bo tròn các phần tử trên trang web.

- CSS3 Border Images: CSS3 cho phép định dạng hình ảnh nền cho đường viền của một phần tử bằng thuộc tính `border-image`. Điều này cho phép tạo ra các hiệu ứng đường viền độc đáo bằng cách sử dụng các hình ảnh tùy chỉnh.

CSS3 Backgrounds: CSS3 cung cấp nhiều tính năng mới liên quan đến nền của một phần tử. Điều này bao gồm việc định dạng hình ảnh nền, tạo gradient nền, chèn văn bản hoặc hình ảnh lặp lại trên nền, và nhiều tính năng khác liên quan đến nền.

- CSS Gradients: CSS3 cho phép tạo gradient (sự chuyển đổi màu từ một màu đến một màu khác) bằng thuộc tính `linear-gradient` và `radial-gradient`. Điều này cho phép tạo các hiệu ứng màu sắc phong phú trên các phần tử web.

- CSS3 Shadows: CSS3 cung cấp các thuộc tính `box-shadow` và `text-shadow` để tạo ra hiệu ứng đổ bóng cho các phần tử và văn bản.

- CSS3 Fonts: CSS3 cung cấp các tính năng cải tiến liên quan đến định dạng và trình bày phông chữ trên trang web. Điều này bao gồm sử dụng phông chữ tùy chỉnh, tạo hiệu ứng văn bản đặc biệt và điều chỉnh khoảng cách kí tự.

- CSS3 2D Transforms: CSS3 cung cấp các thuộc tính `transform`, `translate`, `rotate`, `scale`, `skew` để thay đổi vị trí, kích thước và hình dạng của các phần tử trong không gian hai chiều.

- CSS3 3D Transforms: CSS3 cung cấp các thuộc tính `rotateX`, `rotateY`, `rotateZ`, `translate3d`, `scale3d`, `perspective` để thực hiện các phép biến đổi trong không gian ba chiều.

- CSS3 Transitions: CSS3 cho phép tạo hiệu ứng chuyển đổi mượt mà giữa các trạng thái của một phần tử thông qua thuộc tính `transition`, cho phép kiểm soát thời gian và kiểu chuyển đổi.

- CSS3 Multiple Columns: CSS3 cho phép chia một phần tử thành nhiều cột bằng thuộctính `columns`. Điều này cho phép tạo ra bố cục đa cột linh hoạt trên trang web.

- CSS3 Animations: CSS3 cung cấp các thuộc tính `@keyframes`, `animation-name`, `animation-duration`, `animation-timing-function`, `animation-delay`, `animation-iteration-count`, `animation-direction` để tạo ra các hiệu ứng hoạt hình trên trang web.

## **3.12. SCSS:**

- SCSS (Sassy CSS) là một phiên bản mở rộng của CSS và được hiểu là "CSS với cú pháp cải tiến". Nó là một ngôn ngữ giúp tăng khả năng tái sử dụng, dễ bảo trì và dễ đọc hơn CSS thông thường. SCSS sử dụng cú pháp giống CSS nhưng bổ sung các tính năng như biểu thức, biến, nesting (lồng ghép), import, mixins, và thậm chí cả quản lý màu sắc.

- SCSS được biên dịch thành CSS truyền thống trước khi được sử dụng trên trình duyệt.

## **3.13. SASS:**

- SASS (Syntactically Awesome Style Sheets) cũng là một ngôn ngữ mở rộng của CSS. Nó cung cấp một cú pháp mạnh mẽ và linh hoạt hơn CSS thông thường, giúp viết CSS dễ dàng hơn và có tính sáng tạo cao hơn.

- SASS cho phép sử dụng các tính năng như biến, nesting, mixins, import, inheritance (kế thừa), và nhiều tính năng khác để tạo ra mã CSS dễ đọc, tái sử dụng và dễ bảo trì hơn.

- SASS có hai phiên bản: SASS cổ điển và SCSS. SASS cổ điển sử dụng cú pháp không giống CSS, trong khi SCSS sử dụng cú pháp giống CSS. Cả hai phiên bản đều có thể được biên dịch thành CSS truyền thống để sử dụng trên trình duyệt.

## **Kết luận chương 3**

CSS là ngôn ngữ định dạng để xác định cách trang web được hiển thị.

Cú pháp CSS sử dụng các quy tắc và khối mã để áp dụng các thuộc tính định dạng cho các phần tử trên trang.

CSS được áp dụng vào trang HTML bằng cách sử dụng các quy tắc CSS trong thẻ `<style>` hoặc từ một tệp CSS riêng biệt.

Selectors là cách chọn phần tử HTML mà bạn muốn áp dụng các quy tắc CSS vào.

Các loại selector bao gồm universal selector, type selector, ID selector, class selector, descendant selector, child selector, adjacent sibling selector, attribute selector, pseudo-class selector và group selector.

Đơn vị đo lường CSS được sử dụng để xác định kích thước và khoảng cách trong CSS, bao gồm px, %, em, rem và vw/vh.

Kế thừa thuộc tính cho phép một phần tử kế thừa các thuộc tính từ phần tử cha.

Các nhóm thuộc tính trong CSS bao gồm thuộc tính định dạng font chữ, văn bản, nền, mô hình hộp, margin, padding, border, cách hiển thị, vị trí.

Float & Clear được sử dụng để điều chỉnh vị trí và ảnh hưởng của các phần tử dựa trên dòng chữ xung quanh.

Flex là một phương pháp xây dựng bố cục linh hoạt trong CSS, cho phép căn chỉnh, điều chỉnh kích thước và sắp xếp các phần tử.

Grid là một hệ thống xếp lưới trong CSS cho phép tạo bố cục phức tạp và linh hoạt.

CSS3 là phiên bản tiếp theo của CSS, bao gồm các cải tiến và tính năng mới như CSS3 Selectors, Rounded Corners, Border Images, Backgrounds, Gradients, Shadows, Fonts, 2D Transforms, 3D Transforms, Transitions, Multiple Columns, Animations.

SCSS là một biến thể của CSS với cú pháp tương tự như CSS nhưng hỗ trợ các tính năng mở rộng và cú pháp gọn hơn.

SASS là một ngôn ngữ định dạng CSS mở rộng, tương tự SCSS nhưng có cú pháp khác biệt và hỗ trợ các tính năng bổ sung như biểu thức, mixin và nested rules.

# CHƯƠNG 4: THIẾT KẾ WEBISTE THEO ĐỀ TÀI BẠN CHỌN

## **4.1. Ý tưởng của Website**

- Phân loại sản phẩm: Trang web phân loại sản phẩm theo các danh mục cụ thể như hãng, loại giày, giỏ hàng và liên hệ, giúp người dùng dễ dàng tìm kiếm và chọn lựa sản phẩm theo nhu cầu của họ.

- Hình ảnh và thông tin sản phẩm: Trang web cung cấp thông tin chi tiết và hình ảnh rõ ràng về từng sản phẩm, giúp người dùng hiểu rõ về sản phẩm trước khi quyết định mua hàng.

- Giá cả hấp dẫn: Trang web giới thiệu các sản phẩm với giá cả hấp dẫn, bao gồm cả giá cũ và giá mới, cùng với các chương trình khuyến mãi như "FLASH SALE" để thu hút người mua hàng.

- Tính năng đặt hàng: Trang web cung cấp tính năng đặt hàng trực tuyến, giúp người dùng dễ dàng thêm sản phẩm vào giỏ hàng và tiến hành thanh toán một cách thuận tiện.

- Thông tin liên hệ: Trang web cung cấp thông tin liên hệ rõ ràng như hotline và địa chỉ cửa hàng, giúp người dùng có thể liên hệ và tìm hiểu thêm về sản phẩm hoặc dịch vụ.

## **4.2. Xây dựng bố cục của trang Web**

**- Header (<div class="header">):**

+ Mô tả: Phần header của trang web, bao gồm một hình ảnh nền và tiêu đề của trang web.

+ Đặc điểm:

+ Sử dụng một hình ảnh làm nền với đường dẫn được chỉ định trong CSS.

+ Chứa một tiêu đề <h1> cho trang web.

**- Menu điều hướng (<div class="menu-web">):**

+ Mô tả: Phần menu điều hướng chứa các liên kết đến các trang khác nhau trong trang web.

+ Đặc điểm:

+Sử dụng danh sách không thứ tự <ul> và các mục menu <li> để sắp xếp các liên kết.

+ Các mục menu có thể có các mục con, được hiển thị khi di chuột qua.

+ Một số mục menu có các mục con đi kèm.

**- Content (<div class="content">):**

+ Mô tả: Phần này hiện chưa có nội dung cụ thể, chỉ là một phần để điền nội dung sau này.

+ Đặc điểm:

+ Không có nội dung cụ thể được hiển thị.

+ Nội dung sản phẩm (<div class="noidung">):

+ Mô tả: Phần này chứa danh sách sản phẩm và thông tin chi tiết về mỗi sản phẩm.

+ Đặc điểm:

+ Sử dụng một bảng để sắp xếp các sản phẩm.

+ Mỗi sản phẩm được đại diện bằng một hàng trong bảng, bao gồm hình ảnh, mô tả, giá cũ, giá mới và nút đặt hàng.

**- Footer (<div class="footer">):**

+ Mô tả: Phần footer chứa thông tin về việc đổi hàng và bảo hành của cửa hàng, cùng với thông tin liên hệ.

+ Đặc điểm:

+ Sử dụng một bảng để sắp xếp các thông tin liên hệ và chính sách đổi trả.

**- CSS:**

+ Mô tả: Định dạng các phần tử HTML trên trang web.

+ Đặc điểm: Sử dụng CSS để tạo kiểu cho header, menu điều hướng, nội dung sản phẩm và footer.

+Định dạng các phần tử như màu nền, màu chữ, kích thước v.v.

**- Tổng quan:**

Trang web này có một bố cục cơ bản với header, menu điều hướng, phần nội dung sản phẩm và footer. Mỗi phần được định dạng bằng CSS để hiển thị theo cách mong muốn. Đây chỉ là một bố cục cơ bản và bạn có thể tùy chỉnh và mở rộng nó theo nhu cầu cụ thể của mình.

## **4.3. Thiết kế trang Web bằng HTML và CSS**

- HTML:

+ Định cấu trúc trang web: Sử dụng các thẻ HTML để phân chia trang thành các phần như header, menu, content, và footer.

+ Sử dụng thẻ <style>: Trong phần <head>, sử dụng thẻ <style> để nhúng mã CSS trực tiếp vào tài liệu HTML.

+ Sử dụng các thẻ semantic: Sử dụng các thẻ như <header>, <nav>, <main>, <section>, <article>, <footer> để cung cấp ý nghĩa ngữ cảnh cho nội dung.

+ Cấu trúc menu đa cấp: Sử dụng các thẻ <ul> và <li> để tạo menu chính và menu con.

+ Sử dụng thẻ <table> để hiển thị dữ liệu: Sử dụng các thẻ <table>, <tr>, <td> để hiển thị danh sách sản phẩm và thông tin liên hệ.

- CSS:

+ Định dạng và trang trí giao diện: Sử dụng CSS để định dạng và trang trí giao diện của trang web, bao gồm màu sắc, kích thước, vị trí và kiểu chữ.

+ Responsive design: Cố gắng tạo ra một giao diện đáp ứng với các thiết bị khác nhau bằng cách sử dụng các thuộc tính như width, margin, và padding.

+ Hiệu ứng hover: Sử dụng pseudo-class :hover để thêm hiệu ứng hover cho các phần tử như menu và sản phẩm.

+ Định dạng và trang trí sản phẩm: Sử dụng CSS để định dạng và trang trí các sản phẩm, bao gồm việc thêm hiệu ứng hover và bo tròn ảnh.

+ Chia cột và căn chỉnh nội dung: Sử dụng CSS để chia cột và căn chỉnh nội dung của trang web một cách hợp lý và đồng nhất.

## **4.4. Kết quả đạt được**

- Giao diện hấp dẫn: Trang web có một giao diện hấp dẫn với màu sắc phù hợp, sự kết hợp giữa hình ảnh và văn bản tạo nên sự cân đối và thu hút người xem.

- Menu đa cấp: Việc sử dụng menu đa cấp giúp người dùng dễ dàng điều hướng trên trang web và tìm kiếm các loại sản phẩm khác nhau.

- Hiệu ứng hover: Các hiệu ứng hover được thêm vào menu và sản phẩm giúp tăng tính tương tác và hấp dẫn cho người dùng khi di chuột qua các phần tử.

- Hiển thị sản phẩm: Sản phẩm được hiển thị rõ ràng trong bảng và được phân loại theo từng danh mục, giúp người dùng dễ dàng tìm thấy và so sánh các sản phẩm.

- Thông tin liên hệ: Thông tin liên hệ được đặt ở phía trên cùng của trang web, giúp người dùng dễ dàng tìm thấy thông tin liên lạc khi cần.

- Responsive design: Mặc dù không có mã CSS cụ thể cho responsive design, trang web vẫn có vẻ phản hồi tương đối tốt trên các thiết bị di động và trình duyệt khác nhau.

#### 

#### 

#### 

#### 

## **Kết luận chương 4**

- Trang web sử dụng một tổ chức giao diện rõ ràng và hấp dẫn để hiển thị sản phẩm giày và dép. Giao diện được thiết kế một cách chuyên nghiệp, với việc sử dụng màu sắc và hình ảnh hài hòa, tạo ra một trải nghiệm trực quan cho người dùng. Menu đa cấp được sử dụng một cách hiệu quả để phân loại sản phẩm, giúp người dùng dễ dàng tìm kiếm và duyệt qua các danh mục.

- Thông tin sản phẩm được hiển thị một cách rõ ràng và trực quan, bao gồm hình ảnh, tên sản phẩm, giá bán cũ và giá giảm, cùng với nút "Đặt hàng" tiện lợi. Thông tin liên hệ được đặt ở vị trí dễ thấy, giúp người dùng dễ dàng liên hệ khi cần hỗ trợ.

- Mặc dù trang web có một số điểm mạnh như giao diện hấp dẫn và tổ chức thông tin tốt, nhưng có thể cần cải thiện thêm về tính năng và trải nghiệm người dùng, bao gồm việc thêm tính năng giỏ hàng và thanh toán trực tuyến để tăng tính tương tác và thuận tiện cho người dùng.

# **KẾT LUẬN**

**- Ưu điểm:**

Giao diện hấp dẫn: Trang web có giao diện đẹp mắt và hấp dẫn với việc sử dụng hình ảnh sản phẩm lớn, màu sắc tương phản và tổ chức thông tin rõ ràng, thu hút người dùng từ cái nhìn đầu tiên. Tổ chức menu: Menu được tổ chức cấp độ, giúp người dùng dễ dàng điều hướng và tìm kiếm sản phẩm theo các danh mục cụ thể như hãng, loại giày, giỏ hàng, và liên hệ. Thông tin sản phẩm: Thông tin về sản phẩm được trình bày một cách rõ ràng, bao gồm hình ảnh, giá bán, giá cũ, và nút "Đặt hàng", giúp người dùng dễ dàng quyết định mua hàng.

**- Nhược điểm:**

Thiếu tính năng trang web không cung cấp tính năng giỏ hàng và thanh toán trực tuyến, điều này có thể làm giảm tính tương tác và thuận tiện cho người dùng khi mua sắm trực tuyến. Responsive design trang web không hoàn toàn responsive, điều này có thể gây khó khăn cho người dùng khi truy cập từ các thiết bị di động hoặc kích thước màn hình khác nhau.Tối ưu hóa SEO thiếu một số yếu tố tối ưu hóa SEO cơ bản như thẻ meta, tiêu đề trang và mô tả, có thể làm giảm khả năng trang web được tìm thấy trên các công cụ tìm kiếm.

**- Hướng phát triển:**

Thêm tính năng giỏ hàng và thanh toán: Bổ sung tính năng giỏ hàng và thanh toán trực tuyến để cải thiện trải nghiệm mua sắm của người dùng và tăng doanh số bán hàng.Tối ưu hóa cho thiết bị di động: Cải thiện khả năng responsive của trang web để đảm bảo hiển thị mượt mà trên mọi loại thiết bị di động.Tối ưu hóa SEO tối ưu hóa trang web cho các yếu tố SEO cơ bản như thẻ meta, tiêu đề trang và mô tả để tăng khả năng xuất hiện trên các công cụ tìm kiếm và thu hút lượng truy cập hơn.

**DANH MỤC TÀI LIỆU THAM KHẢO**

1. Nguyễn Hồng Sơn (2007), *Giáo trình hệ thống Mạng máy tính CCNA* (Semester 1), NXB Lao động xã hội.
2. Phạm Quốc Hùng (2017), *Đề cương bài giảng Mạng máy tính*, Đại học SPKT Hưng Yên.
3. James F. Kurose and Keith W. Ross (2013), *Computer Networking: A top-down approach sixth Edition*, Pearson Education.

Hướng dẫn xếp tài liệu tham khảo

1. Tài liệu tham khảo xếp theo thứ tự trích dẫn trong bài tập lớn.
2. Tài liệu tham khảo là sách, luận án, báo cáo phải ghi đầy đủ các thông tin sau:

* Tên các tác giả hoặc cơ quan ban hành (không có dấu ngăn cách)
* (năm xuất bản), (đặt trong ngoặc đơn, dấu phẩy sau ngoặc đơn)
* Tên sách, luận án hoặc báo cáo, (in nghiêng, dấu phẩy cuối tên)
* Nhà xuất bản, (dấu phẩy cuối tên nhà xuất bản)
* Nơi xuất bản, (dấu chấm kết thúc tài liệu tham khảo).

Tài liệu tham khảo là bài báo trong tạp chí, bài trong một cuốn sách… ghi đầy đủ các thông tin sau:

* Tên các tác giả (không có dấu ngăn cách);
* (Năm công bố), (đặt trong ngoặc đơn, dấu phẩy sau ngoặc đơn)
* “Tên bài báo”, (đặt trong ngoặc kép, không in nghiên, dấu phẩy cuối tên)
* Tên tạp chí hoặc tên sách, (in nghiên, dấu phẩy cuối tên)
* Tập (không có dấu ngăn cách)
* (Sổ), (đặt trong ngoặc đơn, dấu phẩy sau ngoặc đơn)
* Các số trang, (gạch ngang giữa hai chữ số, dấu chấm kết thúc)

Cần chú ý những chi tiết về trình bày nêu trên. Nếu tài liệu dài hơn một dòng thì nên trình bày sau cho từ dòng thứ hai lùi vào so với dòng thứ nhất 1 cm để phần tài liệu tham khảo được rõ ràng và dễ theo dõi.

Ví dụ:

1. Nguyễn Hồng Sơn (2007), *Giáo trình hệ thống Mạng máy tính CCNA* (Semester 1), NXB Lao động xã hội.
2. Phạm Quốc Hùng (2017), *Đề cương bài giảng Mạng máy tính*, Đại học SPKT Hưng Yên.
3. James F. Kurose and Keith W. Ross (2013), *Computer Networking: A top-down approach sixth Edition*, Pearson Education.