**TRƯỜNG ĐẠI HỌC ĐẠI NAM**

**KHOA CÔNG NGHỆ THÔNG TIN**

Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**TÊN MÔN HỌC: THIẾT KẾ LẬP TRÌNH FROND-END**

**TÊN ĐỀ TÀI: THIẾT KẾ WEB NHÀ HÀNG BÁN ĐỒ ĂN NHANH**

**Giảng viên hướng dẫn: ThS. Vũ Đình Thắng**

**Sinh viên thực hiện:**  **Nguyễn Đình Huy**

**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 LẬP TRÌNH FROND-END**

**TÊN ĐỀ TÀI: THIẾT KẾ WEB NHÀ HÀNG BÁN ĐỒ ĂN NHANH**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Mã Sinh Viên | Họ và Tên | Ngày Sinh | Điểm | |
| Bằng Số | Bằng Chữ |
| 01 | 1771020357 | Nguyễn Đình Huy | 03/09/2005 |  |  |

### 

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

**Hà Nội, 2024**

**MỤC LỤC**

[CHƯƠNG I: GIỚI THIỆU VỀ THIẾT KẾ WEB VÀ TRIỂN KHAI HỆ THỐNG PHẦN MỀM 7](#_Toc16829)

[1.1. Các khái niệm cơ bản 7](#_Toc11978)

*[1.1.1. Thiết kế web](#_Toc31329)* [7](#_Toc31329)

*[1.1.2. Triển khai hệ thống phần mềm](#_Toc14996)* [7](#_Toc14996)

[1.2. Phân biệt Web tĩnh, Web động 7](#_Toc16257)

*[1.2.1. Web tĩnh.](#_Toc5339)* [7](#_Toc5339)

*[1.2.2. Web động.](#_Toc2654)* [8](#_Toc2654)

[1.3. Một số thuật ngữ (Hosting, Tên miền-Domain Name-Web Server Name,HomePage, Web Site, WebServer, URL-Uniform Resource Location, Browser, …) 9](#_Toc9041)

[1.4. Một số công cụ dùng thiết kế Web (Visual Studio code, Adobe Dreamweaver, …) 10](#_Toc7536)

*[1.4.1. Visual Studio Code.](#_Toc3037)* [10](#_Toc3037)

[KẾT LUẬN CHƯƠNG I 13](#_Toc19162)

[CHƯƠNG II: HTML & HTML5 14](#_Toc18921)

[2.1. Tổng quan về HTML 14](#_Toc15281)

*[2.1.1. Khái niệm.](#_Toc23662)* [14](#_Toc23662)

[2.2. Cấu trúc tổng quát trang HTML 16](#_Toc11977)

*[2.2.1. Khai báo DOCTYPE:](#_Toc20311)* [16](#_Toc20311)

[2.3. Các thẻ HTML thông dụng 16](#_Toc17980)

[2.4. Các thẻ tạo biểu mẫu (form) 18](#_Toc14437)

[2.5. Một số thẻ HTML đặc biệt 19](#_Toc3093)

[2.6. HTML5 19](#_Toc20619)

[KẾT LUẬN CHƯƠNG 2 23](#_Toc6283)

[CHƯƠNG III: CSS và CSS3 24](#_Toc17554)

[3.1. CSS là gì? 24](#_Toc11672)

[3.2. Cú pháp CSS 24](#_Toc7307)

[3.3. Áp dụng CSS vào trang HTML 25](#_Toc25183)

[3.4. Các chức năng và cách hoạt động của CSS 25](#_Toc24392)

*[3.4.1. Universal selector](#_Toc10771)* [25](#_Toc10771)

*[3.4.2. Type selector](#_Toc16144)* [26](#_Toc16144)

*[3.4.3. Identity selector](#_Toc12845)* [26](#_Toc12845)

*[3.4.4. Class selector](#_Toc19467)* [26](#_Toc19467)

*[3.4.5. Descendant selector](#_Toc18562)* [26](#_Toc18562)

*[3.4.6. Child selector](#_Toc18908)* [27](#_Toc18908)

*[3.4.7. Adjacent sibling selector](#_Toc8897)* [27](#_Toc8897)

*[3.4.9. Pseudo class selector](#_Toc1773)* [28](#_Toc1773)

*[3.4.10. Group selector](#_Toc32033)* [29](#_Toc32033)

[3.5. Đơn vị đo lường CSS 29](#_Toc1175)

[3.6. Kế thừa thuộc tính 30](#_Toc30113)

[3.7. Các nhóm thuộc tính trong CSS 30](#_Toc26489)

*[3.7.1. Thuộc tính định dạng font chữ, văn bản (font, text)](#_Toc24547)* [30](#_Toc24547)

*[3.7.2. Thuộc tính định dạng nền (background)](#_Toc13694)* [31](#_Toc13694)

*[3.7.3. Mô hình hộp (box model)](#_Toc15892)* [32](#_Toc15892)

*[3.7.4. Các thuộc tính định margin, padding, border](#_Toc30289)* [32](#_Toc30289)

*[3.7.5. Thuộc tính định dạng cách hiển thị (display)](#_Toc19726)* [33](#_Toc19726)

*[3.7.6. Thuộc tính xác định vị trí (position)](#_Toc10795)* [33](#_Toc10795)

[3.8. Float & Clear 34](#_Toc4042)

[3.9. Flex 35](#_Toc17724)

[3.10. Grid 36](#_Toc6760)

[3.11. CSS3 36](#_Toc269)

*[3.11.1. CSS3 là gì?](#_Toc1681)* [36](#_Toc1681)

*[3.11.2. CSS3 Selectors](#_Toc17887)* [37](#_Toc17887)

*[3.11.3. CSS3 Rounded Corners](#_Toc25553)* [38](#_Toc25553)

*[3.11.4. CSS3 Border Images](#_Toc21529)* [38](#_Toc21529)

*[3.11.5. CSS3 Backgrounds](#_Toc19417)* [39](#_Toc19417)

*[3.11.6. CSS Gradients](#_Toc22014)* [40](#_Toc22014)

*[3.11.7. CSS3 Shadows](#_Toc16849)* [41](#_Toc16849)

*[3.11.8. CSS3 Fonts](#_Toc1092)* [41](#_Toc1092)

*[3.11.9. CSS3 2D Transforms](#_Toc32022)* [43](#_Toc32022)

*[3.11.10. CSS3 3D Transforms](#_Toc7242)* [44](#_Toc7242)

*[3.11.11. CSS3 Transitions](#_Toc801)* [45](#_Toc801)

*[3.11.12. CSS3 Multiple Columns](#_Toc7412)* [46](#_Toc7412)

*[3.11.13. CSS3 Animations](#_Toc20838)* [47](#_Toc20838)

[3.12. SCSS 48](#_Toc10107)

[3.13. SASS 48](#_Toc8045)

[KẾT LUẬN CHƯƠNG 3 50](#_Toc2997)

[CHƯƠNG 4: THIẾT KẾ WEBISTE CỬA HÀNG BÁN ĐỒ ĂN NHANH BẰNHTML5 & CSS5,JAVASCRIPT. 51](#_Toc927)

[4.1. Ý tưởng của Website 51](#_Toc18264)

*[4.1.1. Mục tiêu và đối tượng](#_Toc46)* [51](#_Toc46)

*[4.1.2. Nội dung](#_Toc15516)* [51](#_Toc15516)

*[4.1.3 Giao diện](#_Toc14770)* [51](#_Toc14770)

*[4.1.4. Chức năng](#_Toc13761)* [51](#_Toc13761)

*[4.1.5. Trải nghiệm người dùng](#_Toc2131)* [51](#_Toc2131)

[4.2. Xây dựng bố cục của trang Web 51](#_Toc6736)

[4.3. Thiết kế trang Web bằng HTML và CSS, JavaScript. 52](#_Toc28014)

[4.5. Kết quả đạt được 52](#_Toc9796)

[KẾT LUẬN CHƯƠNG IV 54](#_Toc15580)

[KẾT LUẬN 55](#_Toc28346)

[DANH MỤC TÀI LIỆU THAM KHẢO 56](#_Toc3428)

**MỤC LỤC HÌNH ẢNH**

[Hình 1.1: Web tĩnh & Web động 8](#_Toc25617)

[Hình 1.2: Visual Studio Code. 1](#_Toc31270)1

[Hình 1.3: Adobe Dreamweaver 1](#_Toc16158)1

[Hình 2.0: HTML 1](#_Toc32383)4

[Hình 2.1:HTML & HTML5 1](#_Toc27062)6

[Hình 4:Bố cục trang web 5](#_Toc28530)2

**BẢNG CÁC TỪ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| **STT** | **TỪ VIẾT TẮT** | **VIẾT ĐẦY ĐỦ** |
| 1 | HTML | Hyper Text Markup Language |
| 2 | CSS | Cascading Style Sheets |
| 3 | XML | Extensible Markup Language |
| 4 | SCSS | Sassy CSS |
| 5 | SASS | Syntactically Awesome Style Sheets |

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

**1.1. Các khái niệm cơ bản**

***1.1.1. Thiết kế web***

Thiết kế web là quá trình tạo ra giao diện và trải nghiệm người dùng cho một trang web. Nó bao gồm việc lên ý tưởng, lựa chọn các yếu tố thiết kế, xác định cấu trúc thông tin và tạo ra các mẫu thiết kế để hiển thị nội dung trên trình duyệt web.

* Trong quá trình thiết kế web, các nhà thiết kế sẽ tạo ra các phần tử giao diện như đồ họa, hình ảnh, biểu đồ, màu sắc, kiểu chữ và các yếu tố tương tác như nút bấm, biểu mẫu và menu. Họ cũng xác định cách các phần tử này sẽ được sắp xếp và tương tác với nhau trên trang web.
* Mục tiêu của thiết kế web là tạo ra một trang web hấp dẫn, dễ sử dụng, thân thiện với người dùng và phù hợp với mục đích và nội dung của trang web. Thiết kế web cũng cần quan tâm đến việc tạo ra một trải nghiệm người dùng tốt, đảm bảo rằng người dùng có thể tìm kiếm thông tin một cách dễ dàng và tương tác với trang web một cách thuận tiện.

***1.1.2. Triển khai hệ thống phần mềm***

Triển khai hệ thống phần mềm là quá trình triển khai và đưa hệ thống phần mềm vào hoạt động thực tế. Nó bao gồm các hoạt động như chuẩn bị môi trường triển khai, triển khai mã nguồn, cấu hình hệ thống, kiểm tra và đảm bảo rằng hệ thống hoạt động đúng và đáp ứng yêu cầu.

* Một số hoạt động cần chú ý trong quá trình triển khai như:
* Môi trường triển khai
* Triển khai mã nguồn
* Cấu hình hệ thống
* Kiểm tra và xác nhận
* Quản lý phiên bản
* Hỗ trợ triển khai

**1.2. Phân biệt Web tĩnh, Web động**

***1.2.1. Web tĩnh.***

Web tĩnh là loại trang web mà nội dung trên trang không thay đổi hoặc thay đổi rất ít. Thông thường, nội dung trên các trang web tĩnh được xây dựng trước và lưu trữ dưới dạng các tập tin HTML, CSS và hình ảnh.

* Khi người dùng yêu cầu truy cập vào trang web tĩnh, máy chủ chỉ cần trả về các tập tin đã được tạo sẵn đó, không cần thực hiện xử lý hay truy xuất dữ liệu từ cơ sở dữ liệu.
* Web tĩnh thích hợp cho các trang web không đòi hỏi tính tương tác phức tạp và thường không thay đổi thường xuyên, ví dụ như các trang web thông tin, trang web tĩnh cá nhân, trang web tĩnh quảng cáo.



Hình 1.1: Web tĩnh & Web động

Nguồn: Internet

***1.2.2. Web động.***

Web động là loại trang web có nội dung có thể thay đổi động dựa trên hành vi của người dùng hoặc dữ liệu từ cơ sở dữ liệu. Các trang web động thường sử dụng các ngôn ngữ lập trình như PHP, Python, Ruby hoặc Java để tạo ra nội dung động và tương tác với người dùng.

* Khi người dùng yêu cầu truy cập vào trang web động, máy chủ cần thực hiện xử lý và truy xuất dữ liệu từ cơ sở dữ liệu để tạo ra nội dung phản hồi.
* Web động phù hợp cho các trang web có tính tương tác cao, chẳng hạn như các trang web thương mại điện tử, trang web xã hội, trang web tin tức, vì chúng cho phép tạo và hiển thị nội dung động dựa trên yêu cầu và dữ liệu cụ thể của người dùng.
* Web tĩnh có nội dung không thay đổi hoặc thay đổi rất ít, trong khi web động có thể tạo nội dung dộng và thay đổi dựa trên yêu cầu và dữ liệu, Vì nội dung đã được lưu sẵn trong web tĩnh nên không yêu cầu máy chủ xử lý thông tin, còn web động cần yêu cầu máy chủ để tạo ra nội dung động. Tuy nhiên web tĩnh khả năng tương tác khá thấp, còn web động cho phép người dùng tương tác cùng. Web tĩnh trong lập trình sử dụng HTML, CSS còn Web động sử dụng Java, Python, Ruby…,

**1.3. Một số thuật ngữ (Hosting, Tên miền-Domain Name-Web Server Name, HomePage, Web Site, WebServer, URL-Uniform Resource Location, Browser, …)**

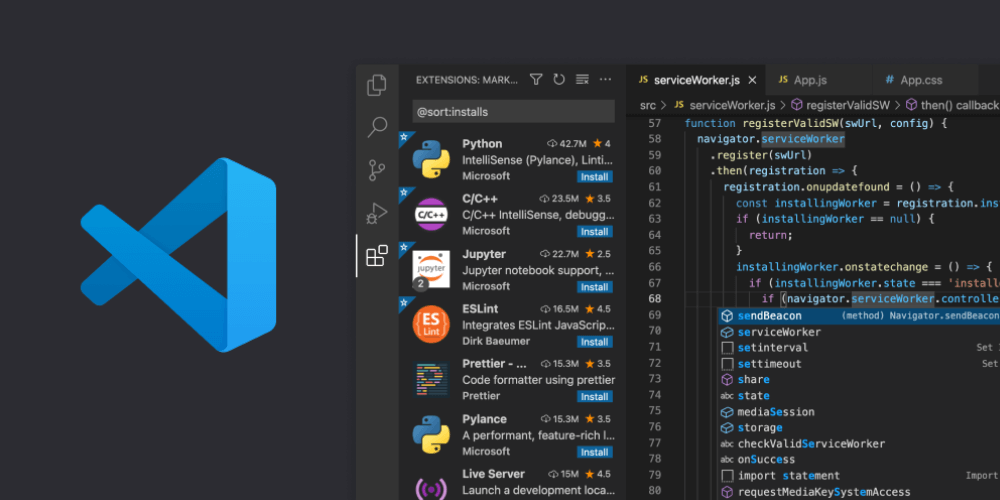
* Hosting: Hosting là dịch vụ cung cấp không gian lưu trữ trên máy chủ (server) để lưu trữ các tệp tin, dữ liệu và tài nguyên liên quan đến một trang web. Nhà cung cấp dịch vụ hosting cung cấp không gian lưu trữ và các tài nguyên kỹ thuật cần thiết để cho phép trang web truy cập trên Internet.
* Web Server Name: Web Server Name là tên được gán cho máy chủ web, nơi các tệp tin và trang web được lưu trữ. Đây là một tên duy nhất để xác định và truy cập vào máy chủ web cụ thể.
* Trang chủ (HomePage): Trang chủ là trang đầu tiên xuất hiện khi truy cập vào một trang web. Đây thường là trang đầu tiên mà người dùng thấy khi truy cập vào một tên miền cụ thể.
* Trang web (Website): Trang web là một tập hợp các trang liên kết với nhau và được lưu trữ trên máy chủ web. Nó bao gồm các thành phần như văn bản, hình ảnh, video, liên kết và các yếu tố khác để cung cấp thông tin và tương tác với người dùng.
* Máy chủ web (Web Server): Máy chủ web là một máy tính hoặc hệ thống máy tính được cấu hình để lưu trữ và phục vụ các trang web cho người dùng truy cập qua Internet. Nó xử lý các yêu cầu truy cập và gửi lại các tệp và dữ liệu tương ứng của trang web.
* URL (Uniform Resource Locator): URL là một chuỗi ký tự duy nhất được sử dụng để xác định địa chỉ của một tài nguyên trên Internet. Nó bao gồm các phần như giao thức, tên miền, đường dẫn và các tham số khác để xác định và truy cập vào một trang web cụ thể.
* Trình duyệt (Browser): Trình duyệt là phần mềm được sử dụng để truy cập và duyệt các trang web trên Internet. Nó hiển thị nội dung của trang web và cho phép người dùng tương tác với các yếu tố trên trang.

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

***1.4.1. Visual Studio Code.***

Visual Studio Code (VS 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 phát triển web, bao gồm cú pháp kiểm tra, gợi ý mã, gỡ lỗi và tích hợp Git. VS Code hỗ trợ nhiều ngôn ngữ lập trình và có thể mở rộng với các tiện ích và phần mở rộng của cộng đồng.

* Adobe Dreamweaver: Adobe Dreamweaver là một công cụ phát triển web tương đối phổ biến của Adobe. Nó cung cấp một giao diện đồ họa cho phép người dùng thiết kế và xây dựng trang web một cách trực quan. Dreamweaver hỗ trợ nhiều ngôn ngữ lập trình và cung cấp tính năng như xem trước, gỡ lỗi, quản lý tệp tin và tích hợp với các dịch vụ khác của Adobe.
* Sublime Tex: Sublime Text là một trình biên tập mã nguồn với giao diện đơn giản và dễ sử dụng. Nó cung cấp các tính năng như gợi ý mã, cú pháp kiểm tra, gỡ lỗi và tùy chỉnh cao. Sublime Text có thể mở rộng với các gói mở rộng của cộng đồng, giúp nâng cao hiệu suất và tính năng.
* Atom: Atom là một trình biên tập mã nguồn mở và miễn phí được phát triển bởi GitHub. Nó có giao diện đẹp và tùy chỉnh cao. Atom hỗ trợ gợi ý mã, cú pháp kiểm tra, gỡ lỗi và tích hợp Git. Nó cũng có thể mở rộng với các gói và chủ đề của cộng đồng.
* Brackets: Brackets là một trình biên tập mã nguồn mở và miễn phí do Adobe tài trợ. Nó được thiết kế đặc biệt cho phát triển web và tập trung vào việc xây dựng giao diện người dùng trực quan. Brackets cung cấp các tính năng như gợi ý mã, xem trước trực tiếp và tích hợp với trình duyệt để phát triển và kiểm tra trực tiếp trong quá trình xây dựng.



Hình 1.2: Visual Studio Code.

Nguồn: Internet

***1.4.2. Adobe Dreamweaver***

Adobe Dreamweaver là một ứng dụng phát triển và thiết kế web được phát triển bởi Adobe Systems. Đây là một công cụ phổ biến được sử dụng bởi các nhà thiết kế và nhà phát triển web để tạo, chỉnh sửa và xuất bản các trang web và ứng dụng web. Dreamweaver cung cấp giao diện trực quan cho phép người dùng thiết kế trang web bằng cách sử dụng một sự kết hợp giữa chỉnh sửa WYSIWYG (What You See Is What You Get) và chỉnh sửa mã.



Hình 1.3: Adobe Dreamweaver

Nguồn: internet

Dreamweaver hỗ trợ nhiều công nghệ web khác nhau như HTML, CSS, JavaScript và PHP, cho phép người dùng tạo ra các trang web động và tương tác. Nó cung cấp các tính năng như tô sáng mã, kiểm tra cú pháp, hoàn thành mã và đoạn mã mẫu để hỗ trợ việc viết mã. Người dùng cũng có thể xem trước thiết kế của mình trong thời gian thực bằng cách sử dụng chế độ xem trước tích hợp trình duyệt web.

Dreamweaver bao gồm một loạt các mẫu và tài sản thiết kế để giúp người dùng bắt đầu nhanh chóng. Nó cũng hỗ trợ thiết kế web đáp ứng, cho phép nhà thiết kế tạo ra các trang web tự thích nghi với kích thước màn hình và thiết bị khác nhau. Nó cung cấp các công cụ để quản lý các tệp và tài sản trang web, cũng như tính năng FTP (File Transfer Protocol) tích hợp để tải lên và xuất bản trang web lên máy chủ web.

Ngoài việc thiết kế và phát triển web, Dreamweaver tích hợp với các ứng dụng khác trong Adobe Creative Cloud như Photoshop và Illustrator, giúp làm việc với đồ họa và nội dung đa phương tiện trong các dự án web dễ dàng hơn.

**KẾT LUẬN CHƯƠNG I**

Qua những câu hỏi trên, ta có thể kết luận rằng việc thiết kế web là một quá trình phức tạp và đa dạng, đòi hỏi sự hiểu biết về các công cụ và kỹ năng lập trình. Có nhiều công cụ khác nhau có thể được sử dụng để thiết kế web, và lựa chọn công cụ phù hợp phụ thuộc vào yêu cầu và sở thích của người dùng.

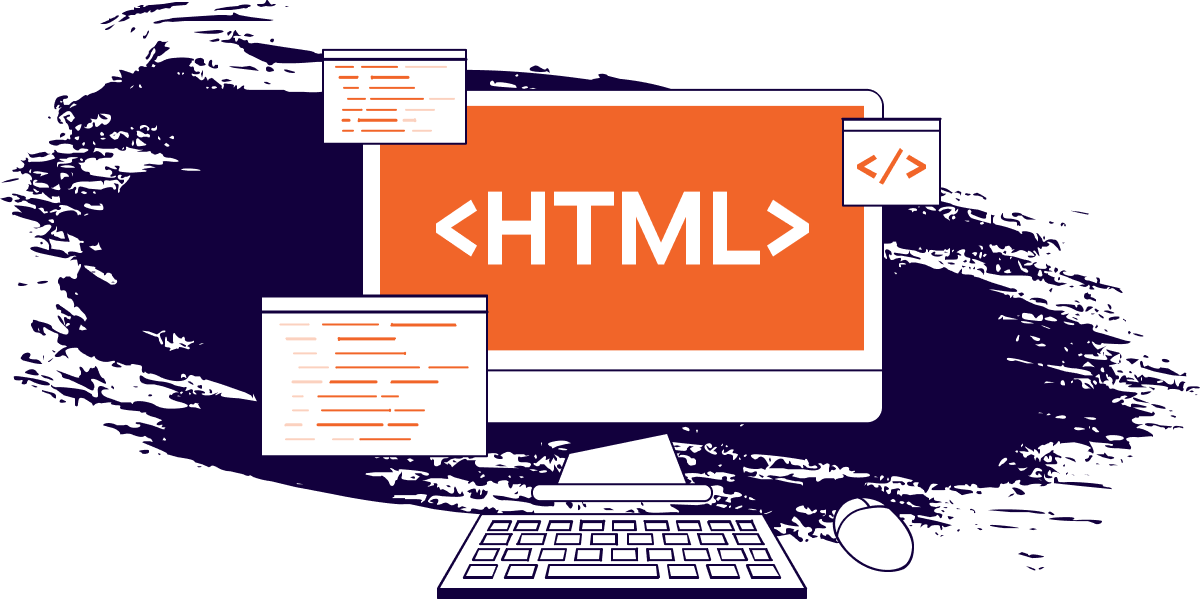
**CHƯƠNG II: HTML & HTML5**

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

**2.1. Tổng quan về HTML**

***2.1.1. Khái niệm.***

Là ngôn ngữ đánh dấu được sử dụng để xây dựng và cấu trúc nội dung trên các trang web. Nó định nghĩa cách mà các phần tử và thành phần của trang web được hiển thị và tương tác với nhau trên trình duyệt.



Hình 2.0: HTML

Nguồn: Internet

* Cú pháp: HTML sử dụng các thẻ (tag) và các thuộc tính (attribute) để đánh dấu và mô tả các phần tử trên trang web. Mỗi thẻ được bao quanh bởi dấu nhọn như "<tag></tag>". Các thuộc tính cung cấp thông tin bổ sung về phần tử.
* Cấu trúc: Một tài liệu HTML thông thường bắt đầu bằng thẻ "<html>" và kết thúc bằng thẻ "</html>". Phần nội dung của trang web được bao quanh bởi thẻ "<body>". Tiêu đề trang được đặt trong thẻ "<head>".
* Phần tử và thành phần: HTML định nghĩa nhiều phần tử và thành phần để hiển thị nội dung trên trang web. Ví dụ, "<h1>" đến "<h6>" được sử dụng để định dạng tiêu đề với các mức độ khác nhau, "<p>" được sử dụng để định dạng đoạn văn bản, "<img>" để chèn hình ảnh, "<a>" để tạo liên kết, vv.
* CSS và Javejcipt: HTML thường kết hợp với CSS (Cascading Style Sheets) và JavaScript để tạo ra các trang web đẹp và tương tác. CSS được sử dụng để định dạng và trình bày các phần tử HTML, trong khi JavaScript được sử dụng để thêm tính năng động và tương tác vào trang web.
* Phiên bản HTML: Có nhiều phiên bản HTML khác nhau, từ HTML cơ bản như HTML 4.01 đến các phiên bản tiên tiến hơn như HTML5. Mỗi phiên bản có các tính năng và cú pháp khác nhau.
* Ưu điểm và nhược điểm
* Ưu điểm
* Dễ học và sử dụng: HTML là một ngôn ngữ đánh dấu đơn giản và dễ hiểu. Người mới học web có thể nhanh chóng tiếp cận và bắt đầu xây dựng trang web cơ bản chỉ sau một thời gian ngắn.
* Độ tương thích cao: HTML là một tiêu chuẩn web được hỗ trợ rộng rãi trên các trình duyệt và nền tảng khác nhau. Điều này đảm bảo rằng trang web xây dựng bằng HTML có thể hiển thị chính xác trên hầu hết các thiết bị và trình duyệt khác nhau.
* Tích hợp dễ dàng với các công nghệ khác: HTML có thể dễ dàng tích hợp với CSS (Cascading Style Sheets) để định dạng và tạo kiểu cho trang web, JavaScript để tạo ra tương tác và các ngôn ngữ lập trình phía máy chủ khác để xử lý logic phức tạp.
* Tính mở rộng và linh hoạt: HTML cho phép bạn tạo ra các trang web đơn giản hoặc phức tạp tùy thuộc vào yêu cầu của dự án. Bạn có thể thêm, chỉnh sửa và mở rộng các phần tử HTML để đáp ứng nhu cầu cụ thể của trang web.
* Tính khả năng tìm kiếm (SEO): HTML cung cấp các thẻ và cấu trúc phù hợp để tối ưu hóa trang web cho các công cụ tìm kiếm. Điều này giúp trang web dễ dàng được tìm thấy và xếp hạng cao trên các kết quả tìm kiếm.
* Nhược điểm
* Thiếu tính tương tác: HTML là một ngôn ngữ đánh dấu tĩnh, có nghĩa là nó không thể xử lý sự kiện và tương tác trực tiếp với người dùng. Để có được tương tác phức tạp, bạn cần sử dụng JavaScript hoặc các công nghệ khác.
* Thiếu khả năng định dạng và tạo kiểu phức tạp: Mặc dù HTML có thể sử dụng CSS để định dạng và tạo kiểu cho trang web, nhưng nó hạn chế so với các ngôn ngữ đánh dấu khác như XML hoặc Markdown.
* Không thể lưu trữ dữ liệu: HTML không cung cấp các cơ chế tích hợp để lưu trữ dữ liệu trên máy chủ hoặc trên máy khách. Để lưu trữ và xử lý dữ liệu, bạn cần sử dụng các công nghệ bổ sung như JavaScript hoặc ngôn ngữ lập trình phía máy chủ.
* Khả năng bảo mật hạn chế: HTML không cung cấp các cơ chế bảo mật tích hợp để bảo vệ trang web khỏi các cuộc tấn công như tấn công XSS (Cross-Site Scripting) hoặc tấn công CSRF (Cross-Site Request Forgery). Để đảm bảo bảo mật, bạn cần thực hiệncác biện pháp bảo mật bổ sung như sử dụng phiên bản HTML an toàn hơn, kiểm tra đầu vào người dùng và xác thực dữ liệu đầu vào.



Hình 2.1:HTML & HTML5

Nguồn: Internet

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

***2.2.1. Khai báo DOCTYPE:***

* Đây là phần đầu tiên của tài liệu HTML và nó khai báo loại tài liệu HTML mà trang web sử dụng. Ví dụ: <!DOCTYPE html>.
* Thẻ <html>: Thẻ <head> chứa thông tin meta-data và các liên kết tới các tệp tin CSS hoặc JavaScript. Các thông tin này không được hiển thị trực tiếp trên trình duyệt. Thẻ <title>: Thẻ <title> định nghĩa tiêu đề của trang web, nó hiển thị trên thanh tiêu đề của trình duyệt.
* Thẻ <head>: Thẻ <head> chứa thông tin meta-data và các liên kết tới các tệp tin CSS hoặc JavaScript. Các thông tin này không được hiển thị trực tiếp trên trình duyệt.
* Thẻ <title>: Thẻ <title> định nghĩa tiêu đề của trang web, nó hiển thị trên thanh tiêu đề của trình duyệt.
* Thẻ <body>: Thẻ <body> chứa toàn bộ nội dung hiển thị trên trang web, bao gồm văn bản, hình ảnh, đoạn mã JavaScript và các phần tử khác.

**2.3. Các thẻ HTML thông dụng**

* Các thẻ xử lý văn bản:
* <h1> - <h6>: Định dạng tiêu đề với mức độ ưu tiên từ cao đến thấp.
* <p>: Định dạng đoạn văn bản.
* <strong> hoặc <b>: In đậm văn bản.
* <em> hoặc <i>: Làm nghiêng (in nghiêng) văn bản.
* <u>: Gạch chân văn bản.
* <s> hoặc <del>: Gạch ngang văn bản.
* <sub>: Hiển thị văn bản dưới đáy.
* <sup>: Hiển thị văn bản trên đỉnh.
* <blockquote>: Định dạng văn bản là một khối trích dẫn.
* <cite>: Định dạng văn bản là một nguồn trích dẫn.
* <q>: Trích dẫn ngắn của một bài viết hay một trình duyệt nào đó.
* <br>: Tạo một dòng mới.
* <hr>: Tạo một đường ngang ngăn cách các phần trong trang.
* <mark>: Đánh dấu văn bản với màu nổi bật.
* <small>: Định dạng văn bản với kích thước nhỏ hơn.
* <span>: Định dạng một phần của văn bản hoặc một phần tử nhỏ hơn không ảnh hưởng đến cấu trúc trang.
* Các thẻ tạo bảng
* <table>: Thẻ gốc để bắt đầu và kết thúc một bảng.
* <tr>: Đại diện cho một hàng trong bảng.
* <th>: Đại diện cho một ô tiêu đề trong bảng. Thường được sử dụng cho hàng đầu tiên hoặc cột đầu tiên của bảng.
* <td>: Đại diện cho một ô dữ liệu trong bảng.
* <caption>: Được sử dụng để thêm tiêu đề cho bảng.
* <thead>, <tbody>, <tfoot>: Các phần tử này được sử dụng để nhóm các phần tử <tr> thành các phần tử tiêu đề (<thead>), phần tử thân (<tbody>) và phần tử chân (<tfoot>) của bảng.
* Thẻ liên kết
* <a>: Đây là thẻ chính để tạo liên kết và có thuộc tính href để chỉ định đích của liên kết. Ví dụ: <a href="https://www.example.com">Liên kết</a>. Thẻ <a> có thể được sử dụng để tạo liên kết đến các trang web khác, các tệp tin, các vị trí trong cùng một trang,và nhiều hơn nữa.
* <link>: Thẻ <link> được sử dụng để liên kết với các tệp tin bên ngoài, chẳng hạn như các tệp CSS hoặc các biểu tượng trang web. Ví dụ: <link rel="stylesheet" href="styles.css"> sẽ liên kết với tệp CSS để áp dụng các kiểu cho trang web.
* <area>: Thẻ <area> được sử dụng trong kết hợp với một bản đồ hình ảnh (<map>) để tạo ra các liên kết vùng trên hình ảnh. Với <area>, bạn có thể định nghĩa các khu vực nhất định trên hình ảnh và liên kết mỗi khu vực đó với một URL cụ thể.
* <nav>: Thẻ <nav> được sử dụng để nhóm các liên kết dẫn đến các trang hoặc phần của trang web. Ví dụ: <nav><a href="page1.html">Trang 1</a> | <a href="page2.html">Trang 2</a></nav>.
* <base>: Thẻ <base> được sử dụng để thiết lập một URL cơ sở cho tất cả các liên kết trong trang. Nó giúp giảm thiểu việc lặp lại việc ghi địa chỉ URL đầy đủ cho từng liên kết trong trang.
* Các thẻ đa phương tiện
* <img>: Thẻ <img> được sử dụng để chèn và hiển thị hình ảnh trên trang web. Nó có một thuộc tính src để chỉ định đường dẫn đến hình ảnh. Ví dụ: <img src="image.jpg" alt="Hình ảnh">.
* <video>: Thẻ <video> được sử dụng để chèn và phát video trên trang web. Nó hỗ trợ nhiều định dạng video và cung cấp các thuộc tính như src (đường dẫn video), controls (hiển thị các điều khiển phát), và autoplay (tự động phát video). Ví dụ: <video src="video.mp4" controls></video>.
* <audio>: Thẻ <audio> được sử dụng để chèn và phát âm thanh trên trang web. Nó hỗ trợ nhiều định dạng âm thanh và tương tự như thẻ <video>, nó có các thuộc tính như src, controls, và autoplay. Ví dụ: <audio src="audio.mp3" controls></audio>.
* <iframe>: Thẻ <iframe> được sử dụng để nhúng một trang web hoặc tài liệu từ một nguồn khác vào trong trang web hiện tại. Nó có thuộc tính src để chỉ định đường dẫn đến tài liệu được nhúng. Ví dụ: <iframe src="https://www.example.com"></iframe>.
* <embed>: Thẻ <embed> được sử dụng để nhúng nội dung đa phương tiện, như video, âm thanh hoặc ứng dụng, vào trong trang web. Ví dụ: <embed src="video.swf">.
* <object>: Thẻ <object> được sử dụng để nhúng một đối tượng media như Flash, Java applet hoặc ứng dụng khác vào trang web. Ví dụ:<object data="applet.jar"></object>
* Thẻ tạo khung
* <div>: tạo khung gốc cho phần tử

Sử dụng CSS để chia khung thành các phần mà bản thân muốn.

**2.4. Các thẻ tạo biểu mẫu (form)**

* <form>: Đây là thẻ gốc để bắt đầu và kết thúc một biểu mẫu. Nó định nghĩa một khu vực trong trang web để chứa các phần tử đầu vào và các phần tử điều khiển của biểu mẫu.
* <input>: Thẻ <input> được sử dụng để tạo các trường đầu vào trong biểu mẫu, như hộp văn bản, nút radio, nút checkbox, nút gửi (submit), vv. Thuộc tính type của thẻ <input> xác định loại trường đầu vào. Ví dụ: <input type="text">, <input type="radio">, <input type="checkbox">, <input type="submit">.
* <label>: Thẻ <label> được sử dụng để định nghĩa nhãn cho các trường đầu vào trong biểu mẫu. Nó cung cấp một mô tả hoặc nhãn cho trường đầu vào liên quan. Thẻ <label> thường được kết hợp với các trường đầu vào bằng cách sử dụng thuộc tính for hoặc bọc trực tiếp nội dung của trường đầu vào. Ví dụ: <label for="username">Tên người dùng:</label> <input type="text" id="username">.
* <textarea>: Thẻ <textarea> được sử dụng để tạo một ô văn bản đa dòng trong biểu mẫu, cho phép người dùng nhập văn bản dài. Ví dụ: <textarea rows="4" cols="50"></textarea>.
* <select> và <option>: Thẻ <select> được sử dụng để tạo một danh sách thả xuống (dropdown) trong biểu mẫu, cho phép người dùng chọn một lựa chọn từ danh sách. Thẻ <option> được sử dụng để định nghĩa các tùy chọn trong danh sách.

**2.5. Một số thẻ HTML đặc biệt**

* <doctype>: Thẻ <!DOCTYPE> không phải là một thẻ HTML truyền thống, nhưng nó được sử dụng để khai báo kiểu tài liệu HTML được sử dụng trong trang web. Ví dụ: <!DOCTYPE html>.
* <html>: Thẻ <html> là thẻ gốc của mọi tài liệu HTML và định nghĩa phần tử HTML chính của trang web.
* <head>: Thẻ <head> chứa các thông tin meta-data và các liên kết tới các tệp tin CSS hoặc JavaScript. Nó không hiển thị trực tiếp nội dung trên trình duyệt.
* <title>: Thẻ <title> định nghĩa tiêu đề của trang web và hiển thị trên thanh tiêu đề của trình duyệt.
* <body>: Thẻ <body> chứa toàn bộ nội dung hiển thị trên trang web, bao gồm văn bản, hình ảnh, đoạn mã JavaScript và các phần tử khác.
* <script>: Thẻ <script> được sử dụng để nhúng mã JavaScript vào trong trang web. Nó có thể được đặt trong thẻ <head> hoặc thẻ <body>.
* <style>: Thẻ <style> được sử dụng để định nghĩa kiểu CSS trực tiếp trong tài liệu HTML. Nó thường được đặt trong thẻ <head>.
* <meta>: Thẻ <meta> được sử dụng để định nghĩa các thông tin meta-data về trang web, chẳng hạn như mã ngôn ngữ, bộ mã hóa, mô tả trang, từ khóa, và nhiều hơn nữa. Thẻ <meta> thường được đặt trong thẻ <head>.
* <comment>: Thẻ <!-- --> được sử dụng để chú thích (comment) trong mã HTML. Nội dung được đặt trong các dấu chú thích sẽ không được hiển thị trên trình duyệt và chỉ có tác dụng cho việc đọc và hiểu mã nguồn.
* <nav>, <header>, <footer>, <section>, <article>: Các thẻ này là các thẻ đặc biệt được sử dụng để định dạng và nhóm các phần tử trong trang web để tạo cấu trúc và ý nghĩa semantically. Chúng giúp máy chủ và công cụ tìm kiếm hiểu cấu trúc trang web một cách tốt hơn.

**2.6. HTML5**

* + Khái niệm: HTML5 là phiên bản mới nhất của ngôn ngữ đánh dấu siêu văn bản (HTML), được phát triển bởi W3C (World Wide Web Consortium) và WHATWG (Web Hypertext Application Technology Working Group). Nó là một tiêu chuẩn web mạnh mẽ và linh hoạt, đưa ra nhiều cải tiến so với các phiên bản HTML trước đó.
  + ƯU điểm và Nhược điểm
* Ưu điểm
* Hỗ trợ đa phương tiện tích hợp: HTML5 cung cấp các thẻ như <video>, <audio>, <canvas> và <svg> cho phép tích hợp trực tiếp video, âm thanh, đồ họa vector và các phương tiện tương tác khác vào trang web. Điều này loại bỏ sự phụ thuộc vào các công nghệ bên thứ ba như Flash và Silverlight, giúp tăng cường trải nghiệm đa phương tiện và giảm độ phức tạp.
* Cấu trúc trang web rõ ràng hơn: HTML5 giới thiệu các thẻ ngữ nghĩa mới như <header>, <nav>, <section>, <article>, <footer>,... giúp xác định rõ ràng cấu trúc và ý nghĩa của các phần tử trên trang web. Điều này cải thiện khả năng đọc và hiểu của người đọc cũng như khả năng tìm kiếm và SEO.
* Hỗ trợ thiết bị di động: HTML5 được thiết kế để tương thích tốt trên các thiết bị di động. Nó cung cấp các tính năng như responsive design, hỗ trợ cảm ứng, định vị địa lý và khả năng lưu trữ ngoại tuyến. Điều này cho phép xây dựng trang web linh hoạt và tương thích trên nhiều loại thiết bị di động khác nhau.
* Tích hợp các công nghệ tiên tiến: HTML5 kết hợp với CSS3 và JavaScript để tạo ra các trang web mạnh mẽ và tương tác. Nó hỗ trợ các tính năng như CSS3 animations, transitions, transforms và hỗ trợ JavaScript mạnh mẽ, cho phép phát triển viên tạo ra các trải nghiệm web đa phương tiện và tương tác phức tạp.
* Hỗ trợ các API và tính năng tiên tiến: HTML5 cung cấp nhiều API và tính năng tiên tiến như Geolocation, Drag and Drop, Web Storage, Web Workers, WebRTC, v.v. Điều này mở ra các khả năng mới cho việc phát triển ứng dụng web phức tạp và tương tác.
* Nhược điểm
* Tương thích trình duyệt: Mặc dù HTML5 được hỗ trợ rộng rãi trên các trình duyệt hiện đại, nhưng việc hỗ trợ không đồng đều trên các phiên bản trình duyệt cũ và thiết bị di động cũ hơn có thể gây ra sự không tương thích và khó khăn trong việc phát triển đa nền tảng.
* Việc triển khai chưa hoàn chỉnh: Mặc dù HTML5 đã được công bố là một tiêu chuẩn, nhưng việc triển khai đầy đủ của các tính năng và API của nó trên các trình duyệt vẫn đang trong quá trình tiến hành. Điều này có thể làm cho việc sử dụng các tính năng mới và tiên tiến của HTML5 gặp khó khăn và phụ thuộc vào trình duyệt cụ thể.
* Hiệu suất: Một số tính năng tiên tiến của HTML5 có thể đòi hỏi tài nguyên và xử lý nặng hơn, làm tăng tải trang và ảnh hưởng đến hiệu suất. Ví dụ, sử dụng video và âm thanh trong trang web có thể tạo ra lưu lượng dữ liệu lớn và yêu cầu xử lý mạnh mẽ từ phía máy khách.
* Bảo mật: Một số tính năng của HTML5 có thể tạo ra các lỗ hổng bảo mật và khả năng tấn công. Ví dụ, sử dụng các tính năng như geolocation hoặc local storage đòi hỏi sự chú ý đặc biệt để đảm bảo rằng dữ liệu người dùng được xử lý và lưu trữ an toàn.
* Sự phụ thuộc vào JavaScript: HTML5 phụ thuộc mạnh mẽ vào JavaScript để triển khai các tính năng và tương tác phức tạp. Điều này có nghĩa là việc phát triển và duy trì các ứng dụng HTML5 sẽ yêu cầu kiến thức và kỹ năng về JavaScript.
  + Cú pháp của HTML5

<!DOCTYPE html>

<html lang=””en>

<head>

<meta charset=””UTF-8>

<meta name=”viewport” content=”width=device-width, initial-scale=1.0”>

<title>Document</title>

</head>

<body>

* + Khai báo DOCTYPE trong HTML5

<!DOCTYPE html>

* + Thẻ ngữ nghĩa

Dưới đây là một số thẻ ngữ nghĩa quan trọng trong HTML5:

* <header>: Định nghĩa phần đầu trang của tài liệu hoặc một phần của nó.
* <nav>: Định nghĩa một khu vực chứa các liên kết điều hướng.
* <article>: Định nghĩa một phần tử độc lập có thể tồn tại và được sử dụng riêng lẻ trong tài liệu.
* <section>: Định nghĩa một phần tử có nghĩa ngữ nghĩa, thường là một phần hoặc một chương của tài liệu.
* <aside>: Định nghĩa một phần tử bên lề chứa thông tin bổ sung mà không cần phụ thuộc vào nội dung chính.
* <footer>: Định nghĩa phần cuối trang của tài liệu hoặc một phần của nó.
* <main>: Định nghĩa phần nội dung chính của tài liệu, chỉ có một phần tử <main> được phép trong mỗi trang.
* <figure> và <figcaption>: <figure>định nghĩa một hình ảnh hoặc sơ đồ, và<figcaption> định nghĩa chú thích cho hình ảnh hoặc sơ đồ đó.
  + Webform
* Webform (còn được gọi là form) trong HTML5 là một phần tử cho phép người dùng nhập dữ liệu và gửi nó đến máy chủ để xử lý. Webform cho phép người dùng tương tác với trang web bằng cách nhập thông tin, chẳng hạn như tên, địa chỉ email, số điện thoại, v.v., và gửi dữ liệu đến máy chủ để xử lý hoặc lưu trữ.

<form action="/process-form" method="post">

<label for="name">Tên:</label>

<input type="text" id="name" name="name" required>

<label for="email">Email:</label>

<input type="email" id="email" name="email" required>

<label for="message">Tin nhắn:</label>

<textarea id="message" name="message" rows="4" required></textarea>

<input type="submit" value="Gửi">

</form>

**KẾT LUẬN CHƯƠNG 2**

Trên hành trình phát triển web, HTML và HTML5 đã chơi một vai trò quan trọng trong việc xây dựng và định dạng các trang web. HTML đã là ngôn ngữ đánh dấu cơ bản cho việc hiển thị nội dung trên web từ những ngày đầu tiên. Từ đó, HTML5 đã mang lại một cuộc cách mạng trong công nghệ web. HTML5 đã giới thiệu nhiều tính năng mới, cải tiến và tiêu chuẩn hóa những gì trước đây chỉ có thể được thực hiện bằng các công nghệ phụ trợ như Flash và Silverlight. Với HTML5, chúng ta có thể tích hợp video, âm thanh, đồ họa vector, tạo ra các trải nghiệm tương tác đa phương tiện mạnh mẽ hơn và tạo ra các ứng dụng web phong phú. Ngoài ra, HTML5 cũng mang đến các thẻ ngữ nghĩa mới, giúp cấu trúc trang web trở nên rõ ràng hơn và dễ hiểu hơn cho cả người đọc và máy chủ. Điều này hỗ trợ việc tạo nội dung phong phú và tăng cường khả năng tìm kiếm và khả năng tương tác của trang web. Ngoài những ưu điểm bên trên đã nói còn có một số nhược điểm cần phải chú ý như: Thiếu tính tương tác, Thiếu khả năng định dạng và tạo kiểu phức tạp, Khả năng bảo mật hạn chế, Hạn chế hỗ trợ phiên bản cũ, Qúa trình chuyển đổi.

**CHƯƠNG III: CSS và CSS3**

 (Cascading Style Sheets)

**3.1. CSS là gì?**

CSS là viết tắt của cụm từ Cascading Style Sheets, đây là ngôn ngữ dùng để tìm và định dạng miêu tả lại các phần tử được tạo ra bởi ngôn ngữ Markup cấu trúc HTML. Để nói một cách ngắn gọn, CSS là ngôn ngữ dùng để tạo nên phong cách cho trang web.

**3.2. Cú pháp CSS**

7 loại cú pháp:

* Loại 1: chọn phần tử để áp dụng CSS:

selector {

property: value;

}

* Loại 2: sử dụng ID để chọn phần tử:

#id {

property: value;

}

* Loại 3: Sử dụng lớp để chọn phần từ:

.class {

property: value;

}

* Loại 4: Sử dụng thuộc tính:

selector {

property1: value1;

property2: value2;

}

* Loại 5: Kết hợp các lựa chọn:

selector1, selector2 {

property: value;

}

* Loại 6: Sử dụng các giá trị cụ thể:

selector {

property: value1 value2;

}

* Loại 7: Sử dụng các giá trị có đơn vị:

selector {

property: value;

}

**3.3. Áp dụng CSS vào trang HTML**

* Inline CSS: Ta có thể thêm các thuộc tính CSS trực tiếp vào các phần tử HTML bằng cách sử dụng thuộc tính style. Cú pháp:

<h1 style="color: blue;">Tiêu đề</h1>

* Internal CSS: Ta có thể đặt các quy tắc CSS trong thẻ <style> bên trong phần <head> của tệp HTML. Ví dụ:

<head>

<style>

h1 {

color: blue;

}

</style>

</head>

<body>

<h1>Tiêu đề</h1>

</body>

* External CSS: Ta có thể tạo một tệp CSS riêng biệt và sau đó liên kết nó với tệp HTML bằng cách sử dụng thẻ <link> trong phần <head>. Ví dụ:

Tệp CSS:

h1 {

color: blue;

}

Tệp HTML:

<head>

<link rel="stylesheet" href="style.css">

</head>

<body>

<h1>Tiêu đề</h1>

</body>

**3.4. Các chức năng và cách hoạt động của CSS**

***3.4.1. Universal selector***

Universal selector là một chọn lựa trong CSS cho phép bạn áp dụng một quy tắc CSS cho tất cả các phần tử trên trang web. Nó được biểu diễn bằng ký hiệu dấu sao (\*). Khi bạn sử dụng universal selector, các quy tắc CSS sẽ được áp dụng cho mọi phần tử trên trang, bao gồm cả phần tử gốc <html> và <body>. Ví dụ:

\* {

margin: 0;

padding: 0;

}

***3.4.2. Type selector***

Type selector (còn được gọi là element selector) là một loại chọn lựa trong CSS cho phép bạn áp dụng một quy tắc CSS cho một loại phần tử cụ thể trên trang web. Nó được biểu diễn bằng tên của phần tử HTML. Ví dụ:

h1 {

color: blue;

}

***3.4.3. Identity selector***

ID selector: Được biểu diễn bằng ký hiệu dấu thăng (#), ID selector cho phép bạn chọn một phần tử duy nhất trên trang bằng cách sử dụng giá trị của thuộc tính id của phần tử đó. Ví dụ: #myElement { color: red; } sẽ áp dụng màu đỏ cho phần tử có id là "myElement".Ví dụ:

<style>

#myElement {

color: red;

}

</style>

<div id="myElement">Đây là một phần tử có id "myElement"</div>

***3.4.4. Class selector***

Class selector là một loại chọn lựa trong CSS cho phép bạn áp dụng một quy tắc CSS cho tất cả các phần tử có cùng giá trị của thuộc tính class. Được biểu diễn bằng ký hiệu dấu chấm (.), ví dụ .myClass { color: blue; } sẽ áp dụng màu xanh cho tất cả các phần tử có class là "myClass". Ví dụ:

<style>

.myClass {

color: blue;

}

</style>

<div class="myClass">Đây là một div có class "myClass"</div>

<p class="myClass">Đây là một đoạn văn có class "myClass"</p>

***3.4.5. Descendant selector***

Descendant selector (còn được gọi là space combinator) cho phép bạn áp dụng quy tắc CSS cho phần tử con nằm trong một phần tử cha. Được biểu diễn bằng khoảng trắng, ví dụ div p { color: red; } sẽ áp dụng màu đỏ cho tất cả các phần tử <p> mà là con của một phần tử <div>. Ví dụ:

<style>

div p {

color: red;

}

</style>

<div>

<p>Đây là một đoạn văn nằm trong một div</p>

</div>

***3.4.6. Child selector***

Child selector (còn được gọi là greater than combinator) cho phép bạn áp dụng quy tắc CSS cho phần tử con trực tiếp của một phần tử cha. Được biểu diễn bằng dấu mũ (^), ví dụ div > p { font-weight: bold; } sẽ áp dụng đậm cho tất cả các phần tử <p> là con trực tiếp của phần tử <div>.

<style>

div > p {

font-weight: bold;

}

</style>

<div>

<p>Đây là một đoạn văn nằm trong một div</p>

<span>

<p>Đây là một đoạn văn khác không được áp dụng quy tắc</p>

</span>

</div>

***3.4.7. Adjacent sibling selector***

Adjacent sibling selector cho phép bạn áp dụng quy tắc CSS cho phần tử anh em kế tiếp của một phần tử. Được biểu diễn bằng dấu cộng (+), ví dụ h1 + p { margin-top: 20px; } sẽ áp dụng độ. Ví dụ:

<style>

h1 + p {

margin-top: 20px;

}

</style>

<h1>Đây là tiêu đề</h1>

<p>Đây là một đoạn văn nằm sau tiêu đề</p>

***3.4.8. Attribute selector*** Attribute selector trong CSS cho phép bạn chọn các phần tử dựa trên thuộc tính và giá trị của chúng. Có hai loại attribute selector: attribute presence selector và attribute value selector. Ví dụ:

* Attribute presence selector: input[type] chọn tất cả các phần tử <input> có thuộc tính type.
* Attribute value selector: a[href="https://www.example.com"] chọn tất cả các phần tử <a> có thuộc tính href với giá trị là "https://www.example.com".

Ví dụ:

<style>

/\* Attribute presence selector \*/

input[type] {

background-color: yellow;

}

/\* Attribute value selector \*/

a[href="https://www.example.com"] {

color: blue;

}

</style>

<input type="text" value="Attribute presence selector">

<a href="https://www.example.com">Attribute value selector</a>

***3.4.9. Pseudo class selector***

Pseudo class selector trong CSS cho phép bạn chọn các phần tử dựa trên trạng thái hoặc vị trí của chúng. Pseudo class selector được biểu diễn bằng dấu hai chấm (::) hoặc dấu hai chấm đơn (:). Ví dụ:

* a:hover chọn các phần tử <a> khi con trỏ chuột hover lên.
* input:focus chọn các phần tử <input> khi được focus.

Ví dụ:

<style>

/\* Hover pseudo class \*/

a:hover {

color: red;

}

/\* Focus pseudo class \*/

input:focus {

border: 2px solid blue;

}

</style>

<a href="#">Hover over me</a>

<input type="text" value="Click to focus">

***3.4.10. Group selector***

Group selector trong CSS cho phép bạn chọn nhiều phần tử cùng một lúc và áp dụng các quy tắc CSS chung cho chúng. Group selector được biểu diễn bằng dấu phẩy (,). Ví dụ:

* h1, h2, h3 chọn tất cả các phần tử <h1>, <h2>, và <h3>.

**3.5. Đơn vị đo lường CSS**

CSS hỗ trợ nhiều đơn vị đo lường để xác định kích thước, khoảng cách và các thuộc tính khác. Một số đơn vị phổ biến bao gồm:

* px (pixels): đơn vị đo lường dựa trên pixel, ví dụ 10px.
* % (percent): đơn vị đo lường dựa trên phần trăm của phần tử cha, ví dụ 50%.
* em: đơn vị đo lường dựa trên kích thước của phông chữ hiện tại, ví dụ 1.5em.
* rem: đơn vị đo lường dựa trên kích thước của phông chữ gốc (root font size), ví dụ 2rem.
* vw (viewport width): đơn vị đo lường dựa trên chiều rộng viewport, ví dụ 10vw.

## vh (viewport height): đơn vị đo lường dựa trên chiều cao viewport, ví dụ 20vh.

<style>

.box {

width: 200px;

height: 150px;

font-size: 16px;

margin-bottom: 20px;

}

</style>

<div class="box">This is a box</div>

**3.6. Kế thừa thuộc tính**

Trong CSS, một số thuộc tính được kế thừa từ phần tử cha đến phần tử con mặc định. Điều này có nghĩa là các thuộc tính đó sẽ được áp dụng cho phần tử con mà không cần khai báo lại. Một số thuộc tính kế thừa phổ biến bao gồm màu chữ (color), phông chữ (font-family), kích thước phông chữ (font-size).

Ví dụ:

<style>

.parent {

color: blue;

font-size: 20px;

}

.child {

/\* Kế thừa thuộc tính color từ .parent \*/

}

</style>

<div class="parent">

This is the parent element

<div class="child">

This is the child element

</div>

</div>

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

***3.7.1. Thuộc tính định dạng font chữ, văn bản (font, text)***

* font-family: Xác định kiểu font chữ được sử dụng cho văn bản. Bạn có thể chỉ định một danh sách các font chữ phụ fallback, nếu font chữ chính không khả dụng trên máy tính của người dùng.

p {

font-family: Arial, sans-serif;

}

* font-size: Xác định kích thước của font chữ.

h1 {

font-size: 24px;

}

* font-weight: Xác định độ đậm của font chữ.

strong {

font-weight: bold;

}

* text-align: Xác định căn chỉnh ngang của văn bản.

p {

text-align: center;

}

* text-align: Xác định căn chỉnh ngang của văn bản.text-decoration: Xác định kiểu trang trí cho văn bản, chẳng hạn như gạch chân, gạch ngang, hoặc gạch trên.

a {

text-decoration: none;

}

* line-height: Xác định khoảng cách giữa các dòng văn bản.

p {

line-height: 1.5;

}

Và còn một số loại thuộc tính khác…,

***3.7.2. Thuộc tính định dạng nền (background)***

* background-color: Xác định màu nền cho phần tử.

body {

background-color: #f2f2f2;

}

* background-image: Xác định hình ảnh nền cho phần tử.

div {

background-image: url('background.jpg');

}

* background-repeat: Xác định cách lặp lại hình ảnh nền.

div {

background-image: url('background.jpg');

background-repeat: repeat-x;

}

* background-position: Xác định vị trí bắt đầu của hình ảnh nền.

div {

background-image: url('background.jpg');

background-position: center;

}

* background-size: Xác định kích thước của hình ảnh nền.

div {

background-image: url('background.jpg');

background-size: cover;

}

* background-attachment: Xác định cách hình ảnh nền di chuyển khi cuộn trang.

body {

background-image: url('background.jpg');

background-attachment: fixed;

}

Và một số loại thuộc tính khác để xác định dạng nền…,

***3.7.3. Mô hình hộp (box model)***

Mô hình hộp (box model) là một khái niệm quan trọng trong CSS, mô tả cách mà các phần tử HTML được hiển thị và tương互作用 với nhau trong một trang web. Mỗi phần tử HTML được coi là một hộp, và mô hình hộp xác định các thành phần cấu thành hộp đó.

* Mô hình hộp bao gồm các thành phần sau:
* Content (Nội dung): Đây là phần bên trong của hộp, chứa nội dung thực tế của phần tử HTML, chẳng hạn như văn bản, hình ảnh, hoặc các phần tử con bên trong.
* Padding (Khoảng đệm): Là khoảng cách giữa nội dung và viền (border) của hộp. Khoảng đệm có thể được định dạng bằng thuộc tính padding trong CSS.
* Border (Viền): Là viền xung quanh hộp, được đặt quanh phần nội dung và khoảng đệm. Viền có thể có độ dày, kiểu và màu sắc khác nhau, và có thể được định dạng bằng thuộc tính border trong CSS.
* Margin (Khoảng cách): Là khoảng cách giữa viền của hộp và các phần tử khác xung quanh. Khoảng cách này được sử dụng để tạo ra không gian giữa các phần tử. Khoảng cách có thể được định dạng bằng thuộc tính margin trong CSS.

***3.7.4. Các thuộc tính định margin, padding, border***

* margin: Thuộc tính margin được sử dụng để xác định không gian xung quanh các phần tử và tạo ra khoảng cách giữa chúng và các phần tử khác. Nó có thể được đặt riêng cho mỗi cạnh của phần tử (trên, dưới, trái và phải) hoặc đồng thời cho tất cả các cạnh. Giá trị được sử dụng trong thuộc tính margin có thể là số nguyên dương (px, em, rem, v.v.), phần trăm (%), hoặc các giá trị khác như auto.
* padding: Thuộc tính padding được sử dụng để định nghĩa khoảng cách giữa nội dung bên trong phần tử và viền xung quanh. Giống như thuộc tính margin, padding cũng có thể được đặt riêng cho từng cạnh hoặc đồng thời cho tất cả các cạnh. Các giá trị của thuộc tính padding cũng có thể là số nguyên dương, phần trăm hoặc các đơn vị đo lường khác.
* border: Thuộc tính border được sử dụng để xác định đường viền xung quanh phần tử. Nó có thể được tạo ra bằng cách sử dụng các giá trị như solid, dashed, dotted, và nhiều giá trị khác để xác định kiểu viền. Bạn cũng có thể xác định màu sắc và độ dày của đường viền bằng cách sử dụng các thuộc tính border-color và border-width tương ứng.

***3.7.5. Thuộc tính định dạng cách hiển thị (display)***

* block: Phần tử được hiển thị như một khối và chiếm hết chiều ngang của phần tử cha. Mỗi phần tử block sẽ xuống dòng mới. Ví dụ: <div>, <p>, <h1>.
* inline: Phần tử được hiển thị như một đoạn văn bản và chỉ chiếm không gian cần thiết xung quanh nội dung của nó. Các phần tử inline không tạo ra xuống dòng mới. Ví dụ: <span>, <a>, <strong>.
* inline-block: Kết hợp tính chất của block và inline. Phần tử được hiển thị như một đoạn văn bản và cũng có thể có chiều rộng và chiều cao được định rõ. Các phần tử inline-block không tạo ra xuống dòng mới. Ví dụ: <img>, <button>.
* none: Phần tử không được hiển thị và sẽ bị ẩn đi. Nó không chiếm không gian trên trang. Tuy nhiên, phần tử vẫn tồn tại trong cây DOM và có thể được hiển thị lại bằng cách thay đổi giá trị của thuộc tính display. Ví dụ: <script>.
* flex: Phần tử được hiển thị theo mô hình linh hoạt (flexbox). Nó cho phép bạn kiểm soát sắp xếp, căn chỉnh và kích thước của các phần tử con bên trong. Ví dụ: <div> với thuộc tính display: flex;.
* grid: Phần tử được hiển thị theo mô hình lưới (grid). Nó cho phép bạn tạo ra các khu vực lưới và kiểm soát vị trí và kích thước của các phần tử con bên trong. Ví dụ: <div> với thuộc tính display: grid;.

***3.7.6. Thuộc tính xác định vị trí (position)***

Trong CSS, thuộc tính position được sử dụng để xác định vị trí của một phần tử HTML trong không gian trình duyệt. Thuộc tính position có các giá trị sau:

* static (mặc định): Phần tử được hiển thị theo vị trí mặc định của nó trong luồng tài liệu. Thuộc tính top, right, bottom, và left không có hiệu lực với giá trị static.
* relative: Phần tử được hiển thị theo vị trí tương đối so với vị trí ban đầu của nó. Bạn có thể sử dụng thuộc tính top, right, bottom, và left để di chuyển phần tử từ vị trí ban đầu. Phần tử vẫn chiếm không gian ban đầu của nó trong luồng tài liệu.
* absolute: Phần tử được hiển thị theo vị trí tuyệt đối so với phần tử cha gần nhất có thuộc tính position khác static. Các thuộc tính top, right, bottom, và left được sử dụng để xác định vị trí chính xác của phần tử. Phần tử không chiếm không gian trong luồng tài liệu và sẽ chồng lên các phần tử khác.
* fixed: Phần tử được hiển thị theo vị trí tuyệt đối so với cửa sổ trình duyệt. Nó sẽ giữ vị trí ngay cả khi trang được cuộn. Các thuộc tính top, right, bottom, và left được sử dụng để xác định vị trí chính xác của phần tử. Phần tử không chiếm không gian trong luồng tài liệu và sẽ chồng lên các phần tử khác.
* sticky: Phần tử được hiển thị theo vị trí tương đối ban đầu của nó trong phần tử cha, cho đến khi đạt đến một ngưỡng cuộn đã định trước. Khi vượt qua ngưỡng cuộn, phần tử sẽ bị "dán" vào vị trí đã định trước và di chuyển theo cuộn trang. Bạn có thể sử dụng thuộc tính top, right, bottom, và left để xác định vị trí dán của phần tử.

**3.8. Float & Clear**

Trong CSS, thuộc tính float được sử dụng để xác định cách một phần tử được đặt trong dòng chữ và quyết định phần tử khác có thể nằm bên trái hoặc bên phải của nó hay không. Khi một phần tử được thiết lập với giá trị float, nó rời khỏi luồng bình thường của tài liệu và được đặt về cạnh bên trái hoặc bên phải của phần tử cha.

Cú pháp:

* float: left|right|none|initial|inherit;
* left: Phần tử được đặt bên trái và phần tử khác có thể nằm bên phải của nó.
* right: Phần tử được đặt bên phải và phần tử khác có thể nằm bên trái của nó.
* none (mặc định): Phần tử không được đặt dọc theo bất kỳ hướng nào và nằm trong luồng bình thường của tài liệu.
* initial: Thiết lập giá trị mặc định cho thuộc tính.
* inherit: Kế thừa giá trị của phần tử cha.
* Để xử lý các vấn đề liên quan đến phần tử có thuộc tính float và đảm bảo rằng các phần tử sau đó không chồng lên phần tử đó, ta sử dụng thuộc tính clear.
* Thuộc tính clear được sử dụng để xác định xem phần tử có thể nằm bên trái hoặc bên phải của các phần tử trước nó có thuộc tính float hay không. Nó chỉ định rằng phần tử phải bắt đầu trên một dòng mới hoặc không được phép nằm bên cạnh phần tử nào có thuộc tính float.

Cú pháp:

* clear: none|left|right|both|initial|inherit;
* none: Phần tử có thể nằm bên trái hoặc bên phải của các phần tử trước đó có thuộc tính float.
* left: Phần tử không được phép nằm bên trái của phần tử trước đó có thuộc tính float.
* right: Phần tử không được phép nằm bên phải của phần tử trước đó có thuộc tính float.
* both: Phần tử không được phép nằm bên trái hoặc bên phải của phần tử trước đó có thuộc tính float.
* initial: Thiết lập giá trị mặc định cho thuộc tính.
* inherit: Kế thừa giá trị của phần tử cha.

Các giá trị clear thường được sử dụng để đảm bảo rằng các phần tử sau phần tử có thuộc tính float không chồng lên phần tử đó và bắt đầu trên một dòng mới.

**3.9. Flex**

Flexbox là một module trong CSS3 được sử dụng để tạo ra bố cục linh hoạt và điều chỉnh các phần tử trong một container. Nó cung cấp một cách mạnh mẽ để xây dựng các giao diện linh hoạt và phản ứng.

Để sử dụng Flexbox, bạn cần thiết lập thuộc tính display của container thành flex hoặc inline-flex. Khi thuộc tính display được thiết lập như vậy, các phần tử con bên trong container sẽ trở thành các phần tử linh hoạt và có thể được điều chỉnh theo các quy tắc Flexbox.

Các thuộc tính chính trong Flexbox bao gồm:

* flex-direction: Xác định hướng của các phần tử con trong container. Có các giá trị sau:
* row: Các phần tử con được xếp theo hàng ngang từ trái sang phải (mặc định).
* row-reverse: Các phần tử con được xếp theo hàng ngang từ phải sang trái.
* column: Các phần tử con được xếp theo cột từ trên xuống dưới.
* column-reverse: Các phần tử con được xếp theo cột từ dưới lên trên.
* flex-wrap: Xác định cách các phần tử con phân bố trên nhiều dòng hoặc một dòng duy nhất. Có các giá trị sau:
* nowrap: Các phần tử con được xếp trên một dòng duy nhất (mặc định).
* wrap: Các phần tử con được phân bố trên nhiều dòng.
* wrap-reverse: Các phần tử con được phân bố trên nhiều dòng theo chiều ngược lại.
* justify-content: Xác định cách các phần tử con được căn chỉnh theo chiều ngang (theo hướng của flex-direction). Có các giá trị sau:
* flex-start: Các phần tử con được căn chỉnh về phía đầu container (mặc định).
* flex-end: Các phần tử con được căn chỉnh về phía cuối container.
* center: Các phần tử con được căn chỉnh vào giữa container.
* space-between: Các phần tử con được căn chỉnh sao cho có khoảng cách đều nhau giữa chúng.
* space-around: Các phần tử con được căn chỉnh sao cho có khoảng cách đều nhau xung quanh chúng.
* align-items: Xác định cách các phần tử con được căn chỉnh theo chiều dọc (ngược với hướng của flex-direction). Có các giá trị sau:
* flex-start: Các phần tử con được căn chỉnh về phía đầu container.
* flex-end: Các phần tử con được căn chỉnh về phía cuối container.
* center: Các phần tử con được căn chỉnh vào giữa container.
* baseline: Các phần tử con được căn chỉnh dựa trên dòng cơ sở của chúng.
* stretch: Các phần tử con được kéo dãn để lấp đầy container.
* align-content: Xác định cách các hàng hoặc cột của các phần tử con được căn chỉnh theo chiều dọc khi chúng được phân bố trên nhiều dòng. Có các giá trị sau:
* flex-start: Các hàng hoặc cột được căn chỉnh về phía đầu container.
* flex-end: Các hàng hoặc cột được căn chỉnh về phía cuối container.
* center: Các hàng hoặc cột được căn chỉnh vào giữa container.
* space-between: Các hàng hoặc cột được căn chỉnh sao cho có khoảng cách đều nhau giữa chúng.
* space-around: Các hàng hoặc cột được căn chỉnh sao cho có khoảng cách đều nhau xung quanh chúng.
* stretch: Các hàng hoặc cột được kéo dãn để lấp đầy container.

**3.10. Grid**

Grid là một module trong CSS3 được sử dụng để tạo ra bố cục lưới (grid layout) cho các phần tử trong một container. Nó cung cấp một cách mạnh mẽ để xây dựng các giao diện phức tạp và điều chỉnh vị trí và kích thước của các phần tử.

Để sử dụng Grid, bạn cần thiết lập thuộc tính display của container thành grid. Khi thuộc tính display được thiết lập như vậy, container và các phần tử con bên trong nó sẽ trở thành một lưới, trong đó bạn có thể xác định các hàng (rows) và cột (columns) để sắp xếp các phần tử.

Các thuộc tính chính trong Grid bao gồm:

* grid-template-rows và grid-template-columns: Xác định số lượng và kích thước của các hàng và cột trong lưới. Bạn có thể sử dụng các giá trị như đơn vị pixel, đơn vị phần trăm, tỷ lệ, hoặc auto.
* grid-gap (hoặc gap): Xác định khoảng cách giữa các hàng và cột trong lưới.
* grid-template-areas: Xác định các vùng (areas) trong lưới và xác định vị trí của các phần tử bằng cách gán tên cho các vùng. Bạn có thể sử dụng tên vùng để xác định vị trí của các phần tử.
* grid-row và grid-column: Xác định vị trí của một phần tử trong lưới bằng cách xác định hàng và cột mà phần tử đó chiếm.
* justify-items và align-items: Xác định cách các phần tử con được căn chỉnh trong ô của mình trong lưới.
* justify-content và align-content: Xác định cách các hàng và cột của lưới được căn chỉnh theo chiều ngang và chiều dọc.

**3.11. CSS3**

***3.11.1. CSS3 là gì?***

CSS3 là phiên bản thứ ba của Cascading Style Sheets (CSS), một ngôn ngữ đánh dấu sử dụng để mô tả cách trình bày và thiết kế các tài liệu HTML và XML trên web. CSS3 là một bản cập nhật của CSS2, và nó bổ sung nhiều tính năng mới và cải tiến so với phiên bản trước đó.

CSS3 cung cấp một loạt các tính năng mới, bao gồm các thuộc tính, lựa chọn và hiệu ứng mới, giúp các nhà phát triển web tạo ra các trang web và ứng dụng web đa dạng và hấp dẫn hơn. Một số tính năng chính của CSS3 bao gồm:

* Border-radius: Cho phép làm tròn các góc của các phần tử.
* Box-shadow: Cho phép thêm bóng đổ cho các phần tử.
* Gradient: Cho phép tạo ra các hiệu ứng màu sắc gradient cho nền và vùng bôi đen.
* Transition và Animation: Cho phép tạo các hiệu ứng chuyển động và hoạt hình cho các phần tử.
* Flexbox và Grid: Cung cấp các công cụ mạnh mẽ để tạo bố cục linh hoạt và lưới cho các phần tử trên trang.
* Media Queries: Cho phép điều chỉnh giao diện dựa trên các thiết bị và kích thước màn hình khác nhau.

***3.11.2. CSS3 Selectors***

CSS3 cung cấp nhiều loại Selectors (bộ chọn) để chọn và áp dụng các quy tắc CSS cho các phần tử cụ thể trên trang web. Dưới đây là một số Selectors phổ biến của CSS3:

* Element Selector (Bộ chọn phần tử): Chọn tất cả các phần tử có tên thẻ tương ứng. Ví dụ: p chọn tất cả các phần tử <p>.
* Class Selector (Bộ chọn lớp): Chọn tất cả các phần tử có cùng giá trị lớp. Ví dụ: .my-class chọn tất cả các phần tử có lớp là "my-class".
* ID Selector (Bộ chọn ID): Chọn phần tử duy nhất có cùng giá trị ID. Ví dụ: #my-id chọn phần tử có ID là "my-id".
* Attribute Selector (Bộ chọn thuộc tính): Chọn tất cả các phần tử có thuộc tính và giá trị tương ứng. Ví dụ: [type="text"] chọn tất cả các phần tử có thuộc tính type có giá trị là "text".
* Pseudo-class Selector (Bộ chọn giả lập): Chọn các phần tử trong trạng thái xác định hoặc vị trí xác định. Ví dụ: :hover chọn phần tử khi con trỏ của chuột đang trỏ vào phần tử đó.
* Pseudo-element Selector (Bộ chọn giả lập phần tử): Chọn các phần tử ảo được tạo ra trên một phần tử. Ví dụ: ::before chọn phần tử ảo được tạo ra trước nội dung của một phần tử.
* Combinators (Bộ chọn kết hợp): Kết hợp các Selectors với nhau để chọn các phần tử dựa trên mối quan hệ giữa chúng. Ví dụ: elementA elementB chọn các phần tử B nằm trong phần tử A.

***3.11.3. CSS3 Rounded Corners***

Trong CSS3, bạn có thể sử dụng thuộc tính border-radius để làm tròn các góc của các phần tử trên trang web. Thuộc tính này cho phép bạn tạo ra các góc bo tròn, tạo hiệu ứng mềm mại và thẩm mỹ.

* Cú pháp sử dụng thuộc tính border-radius như sau:

selector {

border-radius: value;

}

Giá trị value có thể là một giá trị duy nhất hoặc một danh sách giá trị được phân tách bằng dấu cách. Dưới đây là một số ví dụ:

* Giá trị duy nhất: border-radius: 10px;
* Làm tròn tất cả các góc của phần tử thành bán kính 10px.
* Hai giá trị: border-radius: 10px 20px;
* Làm tròn góc trên trái và góc dưới phải thành bán kính 10px.
* Làm tròn góc trên phải và góc dưới trái thành bán kính 20px.
* Ba giá trị: border-radius: 10px 20px 30px;
* Làm tròn góc trên trái thành bán kính 10px.
* Làm tròn góc trên phải và góc dưới trái thành bán kính 20px.
* Làm tròn góc dưới phải thành bán kính 30px.
* Bốn giá trị: border-radius: 10px 20px 30px 40px;
* Làm tròn góc trên trái thành bán kính 10px.
* Làm tròn góc trên phải thành bán kính 20px.
* Làm tròn góc dưới phải thành bán kính 30px.
* Làm tròn góc dưới trái thành bán kính 40px.

***3.11.4. CSS3 Border Images***

Trong CSS3, bạn có thể sử dụng thuộc tính border-image để áp dụng hình ảnh làm viền cho các phần tử trên trang web thay vì sử dụng các màu sắc hoặc đường viền thông thường. Thuộc tính này cho phép bạn tạo ra hiệu ứng viền độc đáo và tùy chỉnh theo ý muốn.

* Cú pháp sử dụng thuộc tính border-image như sau:

selector {

border-image: source slice width outset repeat;

}

* source: Đường dẫn tới hình ảnh được sử dụng làm viền.
* slice: Khoảng cách cắt hình ảnh (slice) từ các cạnh của viền.
* width: Độ rộng của viền.
* outset: Khoảng cách giữa hình ảnh viền và phần tử.
* repeat: Xác định cách lặp lại hình ảnh viền.

Ví dụ minh họa:

.example {

border-image: url(border.png) 30 fill / 10px 20px;

}

Trong ví dụ:

* url(border.png) chỉ định đường dẫn tới hình ảnh border.png được sử dụng làm viền.
* 30 xác định khoảng cách cắt (slice) hình ảnh.
* fill xác định cách lấp đầy hình ảnh viền vào phần tử.
* 10px 20px xác định độ rộng của viền trên và dưới, và độ rộng của viền trái và phải.

***3.11.5. CSS3 Backgrounds***

Trong CSS3, có nhiều thuộc tính để tùy chỉnh nền (background) của các phần tử trên trang web. Các thuộc tính này cho phép bạn áp dụng màu sắc, hình ảnh, gradient và các hiệu ứng nền khác cho phần tử.

* Dưới đây là một số thuộc tính CSS3 được sử dụng để tùy chỉnh nền:
* background-color: Đặt màu sắc nền cho phần tử. Ví dụ: background-color: #ff0000; sẽ đặt màu sắc nền là đỏ.
* background-image: Đặt hình ảnh làm nền cho phần tử. Ví dụ: background-image: url(image.jpg); sẽ đặt hình ảnh image.jpg làm nền.
* background-repeat: Xác định cách lặp lại hình ảnh nền. Các giá trị phổ biến bao gồm repeat (lặp lại theo chiều ngang và dọc), no-repeat (không lặp lại), và repeat-x (lặp lại theo chiều ngang).
* background-position: Xác định vị trí bắt đầu hiển thị hình ảnh nền. Ví dụ: background-position: center top; sẽ đặt hình ảnh nền ở giữa phần tử và trên cùng.
* background-size: Xác định kích thước hiển thị của hình ảnh nền. Ví dụ: background-size: cover; sẽ điều chỉnh kích thước hình ảnh để nó che phủ toàn bộ phần tử.
* background-attachment: Xác định liệu hình ảnh nền có di chuyển khi cuộn trang hay không. Các giá trị phổ biến bao gồm scroll (di chuyển theo trang) và fixed (không di chuyển).
* background-gradient: Tạo gradient làm nền cho phần tử. Bạn có thể sử dụng các hàm linear-gradient() hoặc radial-gradient() để tạo gradient tuyến tính hoặc gradient hình tròn.

Ví dụ:

.example {

background-color: #ff0000;

background-image: url(image.jpg);

background-repeat: no-repeat;

background-position: center top;

background-size: cover;

background-attachment: fixed;

}

Trong ví dụ này, phần tử có lớp .example sẽ có nền màu đỏ (#ff0000), hình ảnh image.jpg làm nền (không lặp lại), được căn giữa phần tử và nằm trên cùng, có kích thước bao phủ toàn bộ phần tử, và hình ảnh nền không di chuyển khi cuộn trang.

***3.11.6. CSS Gradients***

CSS Gradients là một tính năng trong CSS3 cho phép bạn tạo ra hiệu ứng gradient trên các phần tử trên trang web. Gradient là sự kết hợp màu sắc từ một màu đến màu khác, tạo ra một hiệu ứng màu chuyển đổi mượt mà.

Có hai loại gradient phổ biến trong CSS3: linear gradient (gradient tuyến tính) và radial gradient (gradient hình tròn).

* Linear Gradient:

Để tạo gradient tuyến tính, bạn có thể sử dụng hàm linear-gradient() và xác định các điểm dừng (color stops) trên gradient. Cú pháp của hàm này như sau:

background: linear-gradient(direction, color-stop1, color-stop2, ...);

* direction xác định hướng của gradient, ví dụ: to right (từ trái qua phải), to bottom (từ trên xuống dưới).
* color-stop xác định màu sắc và vị trí của các điểm dừng trên gradient. Ví dụ: red 0%, blue 100% nghĩa là gradient bắt đầu từ màu đỏ ở vị trí 0% và kết thúc bằng màu xanh dương ở vị trí 100%.

Dưới đây là một ví dụ cách tạo gradient tuyến tính từ màu đỏ đến màu xanh dương từ trên xuống dưới:

.example {

background: linear-gradient(to bottom, red, blue);

}

* Radial Gradient:

Để tạo gradient hình tròn, bạn có thể sử dụng hàm radial-gradient() và xác định các điểm dừng trên gradient. Cú pháp của hàm này như sau:

.example {

background: linear-gradient(to bottom, red, blue);

}

* shape size xác định hình dạng và kích thước của gradient, ví dụ: circle closest-side (hình tròn với bán kính bằng cạnh gần nhất).
* at position xác định vị trí bắt đầu của gradient, ví dụ: center center (tại trung tâm của phần tử).
* start-color và last-color xác định màu sắc tại các điểm dừng trên gradient.

Dưới đây là một ví dụ cách tạo gradient hình tròn từ màu đỏ đến màu xanh dương:

.example {

background: radial-gradient(circle closest-side at center center, red, blue);

}

***3.11.7. CSS3 Shadows***

CSS3 Shadows (bóng đổ) là một tính năng trong CSS3 cho phép bạn thêm hiệu ứng bóng đổ cho các phần tử trên trang web. Bóng đổ giúp tạo ra chiều sâu và mức độ thị giác của các phần tử, tạo hiệu ứng 3D và làm nổi bật chúng.

Có hai loại bóng đổ phổ biến trong CSS3: bóng đổ text-shadow (bóng đổ cho văn bản) và box-shadow (bóng đổ cho hộp).

* Text Shadow:

Để tạo bóng đổ cho văn bản, bạn có thể sử dụng thuộc tính text-shadow. Cú pháp của thuộc tính này như sau:

text-shadow: h-shadow v-shadow blur-radius color;

* h-shadow là giá trị của bóng đổ theo chiều ngang.
* v-shadow là giá trị của bóng đổ theo chiều dọc.
* blur-radius là bán kính của hiệu ứng mờ.
* color là màu sắc của bóng đổ.

Dưới đây là một ví dụ cách tạo bóng đổ cho hộp:

.example {

box-shadow: 2px 2px 4px rgba(0, 0, 0, 0.5);

}

Trong ví dụ này, phần tử có lớp .example sẽ có bóng đổ với giá trị 2px theo chiều ngang, 2px theo chiều dọc, bán kính mờ là 4px, và màu sắc là đen với độ trong suốt 50%.

***3.11.8. CSS3 Fonts***

CSS3 Fonts là một tính năng trong CSS3 cho phép bạn tùy chỉnh và điều khiển các phông chữ trên trang web của bạn. CSS3 cung cấp nhiều thuộc tính và khả năng mới để làm việc với phông chữ, bao gồm font family, font size, font style, font weight và font loading.

Dưới đây là một số thuộc tính phông chữ phổ biến trong CSS3:

* Font Family:

Thuộc tính font-family xác định loại phông chữ mà bạn muốn sử dụng cho văn bản. Bạn có thể chỉ định một danh sách các phông chữ được ưu tiên theo thứ tự, để trình duyệt sẽ sử dụng phông chữ đầu tiên có sẵn trên hệ thống. Ví dụ:

body {

font-family: Arial, sans-serif;

}

Trong ví dụ này, trình duyệt sẽ hiển thị phông chữ Arial nếu có, nếu không nó sẽ sử dụng một phông chữ sans-serif mặc định.

* Font Size:

Thuộc tính font-size xác định kích thước của phông chữ. Bạn có thể sử dụng các đơn vị như pixels (px), ems (em), rems (rem) hoặc phần trăm (%). Ví dụ:

p {

font-size: 16px;

}

Trong ví dụ này, văn bản trong các phần tử <p> sẽ có kích thước font là 16 pixels.

* Font Style:

Thuộc tính font-style xác định kiểu dáng của phông chữ. Các giá trị phổ biến bao gồm normal (bình thường), italic (nghiêng) và oblique (nghiêng). Ví dụ:

h1 {

font-style: italic;

}

Trong ví dụ này, phông chữ trong tiêu đề <h1> sẽ được hiển thị dưới dạng nghiêng.

* Font Weight:

Thuộc tính font-weight xác định độ đậm của phông chữ. Các giá trị phổ biến bao gồm normal (bình thường), bold (đậm), lighter (nhẹ hơn) và bolder (đậm hơn). Ví dụ:

h2 {

font-weight: bold;

}

Trong ví dụ này, phông chữ trong tiêu đề <h2> sẽ được hiển thị đậm.

* Font Loading:

CSS3 cũng cung cấp các khả năng mới để tải và quản lý phông chữ từ các nguồn bên ngoài. Bằng cách sử dụng @font-face rule, bạn có thể nhúng phông chữ tùy chỉnh từ các file font trên máy chủ hoặc từ các dịch vụ như Google Fonts. Ví dụ:

@font-face {

font-family: 'CustomFont';

src: url('path/to/font.woff2') format('woff2'),

url('path/to/font.woff') format('woff');

}

body {

font-family: 'CustomFont', sans-serif;

}

Trong ví dụ này, phông chữ tùy chỉnh được nhúng từ file font.woff2 và font.woff và được sử dụng cho nội dung trong trang web.

***3.11.9. CSS3 2D Transforms***

CSS3 2D Transforms là một tính năng trong CSS3 cho phép bạn thay đổi và biến đổi các phần tử trong mặt phẳng 2D trên trang web. CSS3 cung cấp nhiều thuộc tính và khả năng mới để thực hiện các biến đổi như di chuyển, xoay, thu phóng và nghiêng các phần tử.

Dưới đây là một số thuộc tính biến đổi 2D phổ biến trong CSS3:

* Translate (Di chuyển):

Thuộc tính translate() được sử dụng để di chuyển một phần tử theo các giá trị x và y. Nó sử dụng hàm translate() và chấp nhận giá trị trong đơn vị độc lập như pixels (px), ems (em) hoặc phần trăm (%). Ví dụ:

.box {

transform: translate(50px, 100px);

}

Trong ví dụ này, phần tử có class "box" sẽ được di chuyển 50 pixels sang phải và 100 pixels xuống dưới.

* Rotate (Xoay):

Thuộc tính rotate() được sử dụng để xoay phần tử theo một góc quay cụ thể. Nó sử dụng hàm rotate() và chấp nhận giá trị góc trong đơn vị độ như degrees (deg), radians (rad) hoặc turns (turn). Ví dụ:

.box {

transform: rotate(45deg);

}

* Scale (Thu phóng):

Thuộc tính scale() được sử dụng để thu phóng phần tử theo các tỷ lệ x và y. Nó sử dụng hàm scale() và chấp nhận các giá trị tỷ lệ như 1 (giữ nguyên kích thước ban đầu), 2 (phóng to gấp đôi), 0.5 (co lại gấp nửa) và âm (đảo ngược). Ví dụ:

.box {

transform: scale(2, 1.5);

}

Trong ví dụ này, phần tử có class "box" sẽ được thu phóng gấp đôi theo chiều ngang và 1.5 lần theo chiều dọc.

* Skew (Nghiêng):

Thuộc tính skew() được sử dụng để nghiêng phần tử theo các góc nghiêng x và y. Nó sử dụng hàm skew() và chấp nhận giá trị góc nghiêng trong đơn vị độ như degrees (deg), radians (rad) hoặc turns (turn). Ví dụ:

.box {

transform: skew(30deg, -15deg);

}

Trong ví dụ này, phần tử có class "box" sẽ được nghiêng 30 độ theo chiều ngang và -15 độ theo chiều dọc.

* Transform Origin (Gốc biến đổi):

Thuộc tính transform-origin xác định điểm gốc của các biến đổi 2D. Nó xác định điểm trung tâm hoặc điểm neo của phần tử, từ đó các biến đổi được áp dụng. Ví dụ:

.box {

transform-origin: 50% 50%;

}

Trong ví dụ này, phần tử có class "box" sẽ có điểm gốc ở trung tâm của nó.

***3.11.10. CSS3 3D Transforms***

CSS3 3D Transforms là một tính năng trong CSS3 cho phép bạn thực hiện các biến đổi và biến đổi phần tử trong không gian 3D trên trang web. Nó cung cấp các thuộc tính và khả năng mới để di chuyển, xoay, thu phóng và nghiêng các phần tử trong không gian 3 chiều.

Dưới đây là một số thuộc tính biến đổi 3D phổ biến trong CSS3:

* Translate3d (Di chuyển 3D):

Thuộc tính translate3d() được sử dụng để di chuyển một phần tử theo các giá trị x, y và z. Nó sử dụng hàm translate3d() và chấp nhận giá trị trong đơn vị độc lập như pixels (px), ems (em) hoặc phần trăm (%). Ví dụ:

.box {

transform: translate3d(50px, 100px, 0);

}

Trong ví dụ này, phần tử có class "box" sẽ được di chuyển 50 pixels sang phải và 100 pixels xuống dưới trong không gian 3D.

* Rotate3d (Xoay 3D):

Thuộc tính rotate3d() được sử dụng để xoay phần tử theo một trục xác định trong không gian 3D. Nó sử dụng hàm rotate3d() và chấp nhận giá trị cho trục x, y, z và góc xoay. Ví dụ:

.box {

transform: rotate3d(1, 1, 0, 45deg);

}

Trong ví dụ này, phần tử có class "box" sẽ được xoay 45 độ quanh trục (1, 1, 0) trong không gian 3D.

* Scale3d (Thu phóng 3D):

Thuộc tính scale3d() được sử dụng để thu phóng phần tử theo các tỷ lệ x, y và z trong không gian 3D. Nó sử dụng hàm scale3d() và chấp nhận các giá trị tỷ lệ như 1 (giữ nguyên kích thước ban đầu), 2 (phóng to gấp đôi), 0.5 (co lại gấp nửa) và âm (đảo ngược). Ví dụ:

.box {

transform: scale3d(2, 1.5, 1);

}

Trong ví dụ này, phần tử có class "box" sẽ được thu phóng gấp đôi theo chiều ngang, 1.5 lần theo chiều dọc và không thay đổi theo chiều z trong không gian 3D.

* Perspective (Góc nhìn):

Thuộc tính perspective xác định góc nhìn và sự phóng đại của không gian 3D. Nó xác định khoảng cách từ mắt của người xem đến mặt phẳng 3D. Ví dụ:

.container {

perspective: 1000px;

}

Trong ví dụ này, phần tử có class "container" sẽ có góc nhìn 3D với khoảng cách 1000 pixels.

* Transform-style (Kiểu biến đổi):

Thuộc tính transform-style xác định cách các phần tử con bên trong một phần tử cha được biến đổi trong không gian 3D. Nó có hai giá trị: flat (phần tử con không được biến đổi theo không gian 3D) và preserve-3d (phần tử con được biến đổi riThe CSS3 3D Transforms is a feature in CSS3 that allows you to perform transformations and manipulate elements in a 3D space on a webpage. It provides new properties and capabilities to move, rotate, scale, and skew elements in 3-dimensional space.

***3.11.11. CSS3 Transitions***

CSS3 Transitions là một tính năng trong CSS3 cho phép bạn tạo hiệu ứng mượt mà khi thay đổi thuộc tính của phần tử trong một khoảng thời gian nhất định. Bằng cách sử dụng CSS3 Transitions, bạn có thể điều chỉnh giá trị của các thuộc tính CSS, chẳng hạn như màu sắc, kích thước, vị trí và độ trong suốt, và tạo ra hiệu ứng chuyển động trơn tru và hấp dẫn.

Để sử dụng CSS3 Transitions, bạn cần xác định thuộc tính transition trong CSS cho phần tử mà bạn muốn tạo hiệu ứng chuyển động. Thuộc tính transition chấp nhận một chuỗi giá trị, bao gồm:

* Property (Thuộc tính): Xác định thuộc tính CSS mà bạn muốn tạo hiệu ứng chuyển động. Ví dụ: width, height, color, opacity, và nhiều thuộc tính khác.
* Duration (Thời gian): Xác định thời gian mà hiệu ứng chuyển động sẽ diễn ra. Ví dụ: 0.5s (0.5 giây), 1s (1 giây), 200ms (200 mili giây).
* Timing Function (Hàm thời gian): Xác định cách hiệu ứng chuyển động thay đổi theo thời gian. CSS3 cung cấp các hàm như linear (tuyến tính), ease (mượt mà), ease-in (mượt mà từ đầu), ease-out (mượt mà từ cuối), ease-in-out (mượt mà từ đầu đến cuối), và nhiều hàm khác.
* Delay (Độ trễ): Xác định thời gian trễ trước khi hiệu ứng chuyển động bắt đầu. Ví dụ: 0.2s (0.2 giây), 500ms (500 mili giây).

Ví dụ:

.box {

width: 100px;

height: 100px;

background-color: blue;

transition: width 1s ease-in-out, height 0.5s linear, background-color 0.3s ease;

}

.box:hover {

width: 200px;

height: 200px;

background-color: red;

}

Trong ví dụ này, phần tử có class "box" sẽ có hiệu ứng chuyển động mượt mà khi chuột di chuyển qua phần tử đó. Khi chuột hover vào phần tử, chiều rộng (width) sẽ tăng từ 100px lên 200px trong 1 giây, chiều cao (height) sẽ tăng từ 100px lên 200px trong 0.5 giây, và màu nền (background-color) sẽ thay đổi từ màu xanh (blue) sang màu đỏ (red) trong 0.3 giây.

***3.11.12. CSS3 Multiple Columns***

CSS3 Multiple Columns là một tính năng trong CSS3 cho phép bạn chia nội dung của một phần tử thành nhiều cột, tạo ra một bố cục đa cột. Bằng cách sử dụng CSS3 Multiple Columns, bạn có thể tạo ra các bố cục tương tự như báo, tạp chí hoặc trình đọc sách, nơi nội dung được tổ chức thành các cột song song.

Để sử dụng CSS3 Multiple Columns, bạn cần áp dụng thuộc tính column-count hoặc column-width vào phần tử chứa nội dung muốn chia thành cột. Dưới đây là một số thuộc tính chính liên quan:

* column-count: Xác định số lượng cột bạn muốn tạo. Bạn có thể đặt giá trị là một số nguyên dương, ví dụ: 3 để tạo ra 3 cột.
* column-width: Xác định chiều rộng của mỗi cột. Bạn có thể đặt giá trị là một đơn vị đo lường, ví dụ: 200px, 20%, hoặc auto (chiều rộng tự động).
* column-gap: Xác định khoảng cách giữa các cột. Bạn có thể đặt giá trị là một đơn vị đo lường, ví dụ: 20px, 2em, hoặc normal (khoảng cách mặc định).

Ví dụ:

.column-container {

column-count: 3;

column-gap: 20px;

}

.column-container p {

break-inside: avoid; /\* Đảm bảo không bị phân cắt giữa các cột \*/

}

Trong ví dụ này, phần tử với class "column-container" sẽ được chia thành 3 cột, với khoảng cách 20px giữa các cột. Nội dung trong các cột sẽ được tự động phân chia sao cho không bị cắt giữa các cột, bằng cách sử dụng thuộc tính break-inside: avoid.

***3.11.13. CSS3 Animations***

CSS3 Animations là một tính năng trong CSS3 cho phép bạn tạo các hiệu ứng chuyển động phức tạp và động trong trang web của bạn. Bằng cách sử dụng CSS3 Animations, bạn có thể tạo ra các hiệu ứng như quay, phóng to/thu nhỏ, di chuyển, mờ dần và nhiều hiệu ứng khác mà có thể thay đổi thuộc tính của phần tử trong khoảng thời gian nhất định.

Để sử dụng CSS3 Animations, bạn cần định nghĩa một keyframe (khung chính) được xác định bằng thuộc tính @keyframes. Keyframe định nghĩa các trạng thái của phần tử trong quá trình thay đổi. Chi tiết của keyframe sẽ xác định các thuộc tính CSS và giá trị tương ứng mà phần tử sẽ chuyển đổi qua.

Ví dụ:

@keyframes slide-in {

0% {

transform: translateX(-100%);

}

100% {

transform: translateX(0);

}

}

.box {

width: 200px;

height: 200px;

background-color: red;

animation-name: slide-in;

animation-duration: 1s;

animation-delay: 0.5s;

animation-fill-mode: forwards;

}

**3.12. SCSS**

SCSS (Sassy CSS) là một dạng mở rộng của CSS, cung cấp các tính năng và cú pháp mạnh mẽ hơn để viết mã CSS. SCSS giúp tăng tính module, khả năng tái sử dụng và dễ bảo trì của mã CSS thông qua việc sử dụng biến, nested rules, mixins, và nhiều tính năng khác.

Dưới đây là một số tính năng chính của SCSS:

* Biến (Variables): Bạn có thể định nghĩa biến trong SCSS để lưu trữ các giá trị như màu sắc, kích thước, hoặc bất kỳ giá trị nào khác. Biến giúp tái sử dụng và dễ dàng thay đổi các giá trị trong toàn bộ mã CSS.
* Nesting: Bạn có thể lồng các quy tắc CSS bên trong nhau trong SCSS, giúp mã CSS trở nên rõ ràng hơn và dễ đọc hơn. Thay vì lặp lại các tên lớp cha trong CSS, bạn có thể sử dụng nesting để chỉ định quy tắc cho các phần tử con.
* Mixins: Mixins cho phép bạn định nghĩa một tập hợp các quy tắc CSS và tái sử dụng chúng ở nhiều nơi trong mã SCSS của bạn. Điều này giúp giảm lặp lại mã và tạo ra các khối CSS tương tự nhau một cách dễ dàng.
* Import: SCSS cho phép bạn chia nhỏ mã CSS thành các file nhỏ hơn và import chúng vào một file SCSS chính. Điều này giúp tổ chức mã CSS một cách tốt hơn và tạo điều kiện cho việc quản lý mã dễ dàng hơn.
* Operators: SCSS hỗ trợ các phép tính toán số học và chuỗi trong quy tắc CSS. Bạn có thể sử dụng các toán tử như +, -, \*, / để thực hiện các phép tính trong biểu thức CSS.

**3.13. SASS**

SASS (Syntactically Awesome Style Sheets) là một ngôn ngữ mở rộng của CSS. Nó cung cấp cú pháp mạnh mẽ hơn và có thêm các tính năng đáng chú ý để viết mã CSS dễ dàng hơn. SASS được biên dịch thành CSS chuẩn để sử dụng trong các trình duyệt web.

Dưới đây là một số tính năng chính của SASS:

* Biến (Variables): Bạn có thể định nghĩa biến trong SASS để lưu trữ các giá trị như màu sắc, kích thước, hoặc bất kỳ giá trị nào khác. Biến giúp tái sử dụng và dễ dàng thay đổi các giá trị trong toàn bộ mã CSS.
* Nesting: Bạn có thể lồng các quy tắc CSS bên trong nhau trong SASS, giúp mã CSS trở nên rõ ràng hơn và dễ đọc hơn. Thay vì lặp lại các tên lớp cha trong CSS, bạn có thể sử dụng nesting để chỉ định quy tắc cho các phần tử con.
* Mixins: Mixins cho phép bạn định nghĩa một tập hợp các quy tắc CSS và tái sử dụng chúng ở nhiều nơi trong mã SASS của bạn. Điều này giúp giảm lặp lại mã và tạo ra các khối CSS tương tự nhau một cách dễ dàng.
* Import: SASS cho phép bạn chia nhỏ mã CSS thành các file nhỏ hơn và import chúng vào một file SASS chính. Điều này giúp tổ chức mã CSS một cách tốt hơn và tạo điều kiện cho việc quản lý mã dễ dàng hơn.
* Operators: SASS hỗ trợ các phép tính toán số học và chuỗi trong quy tắc CSS. Bạn có thể sử dụng các toán tử như +, -, \*, / để thực hiện các phép tính trong biểu thức CSS.

**KẾT LUẬN CHƯƠNG 3**

Tổng kết lại, CSS là một ngôn ngữ đánh dấu được sử dụng để định dạng và trình bày các phần tử trên trang web. CSS3 là phiên bản mới nhất của CSS và cung cấp nhiều tính năng mạnh mẽ hơn cho việc tạo ra các hiệu ứng chuyển động, đổ bóng, gradient và nhiều tính năng khác. Cả hai đều là công cụ quan trọng trong quá trình phát triển web và giúp tạo ra các trang web đẹp và tương tác.

CSS và CSS3 đều có ưu điểm và nhược điểm riêng. CSS cung cấp khả năng tách rời nội dung và kiểu dáng, tái sử dụng mã và quản lý dễ dàng. CSS3 mang lại tính năng mạnh mẽ và tương thích trình duyệt tốt hơn. Tuy nhiên, cả hai đều đối mặt với các vấn đề về tương thích trình duyệt, quản lý phức tạp và hiệu năng trang web.

**CHƯƠNG 4: THIẾT KẾ WEBISTE CỬA HÀNG BÁN ĐỒ ĂN NHANH BẰNG HTML5 & CSS5, JAVASCRIPT**

**4.1. Ý tưởng của Website**

***4.1.1. Mục tiêu và đối tượng***

* Mục tiêu của trang web là tạo ra một trang thực phẩm có thể đặt và mua hàng, tạo ra một không gian để khám phá những thức ăn mới lạ . Đối tượng chính có thể là cả người dân khắp cả nước quan tâm đến đồ ăn nhanh.

***4.1.2. Nội dung***

* Trang web có thể cung cấp thông tin chi tiết về các loại thức ăn nhanh, và ngoài mặt là số thông tin của cửa hàng cùng với sản phẩm mới ở phần “News”.
* Trang web cung cấp thông tin của cửa hàng ở phần “Thông tin của hàng”.
* Trang web có mục thực đơn cho khách hàng lựa chọn thực phẩm muốn mua “Menu”.

***4.1.3 Giao diện***

* Thiết kế giao diện của trang web có thể lấy cảm hứng từ kiến trúc truyền thống của nhật bản và một chút đặc sắc văn hóa của Việt Nam. Sử dụng màu sắc nhẹ nhàng, hình ảnh đẹp và các yếu tố liên quan tới thức ăn để tạo ra một trải nghiệm vị giác nhẹ nhàng mà chìm đắm trong hương vị của đồ ăn.

***4.1.4. Chức năng***

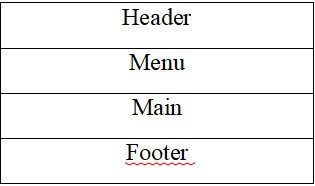
* Trang web có thể cung cấp thông tin về các lễ hội và sự kiện diễn ra tại nhà Hàng, đồng thời cung cấp lịch trình và thông tin sản phẩm mới.
* Tìm kiếm thông tin sản phẩm và nguồn gốc nguyên liệu một cách chính xác.

***4.1.5. Trải nghiệm người dùng***

* Trang web nên được thiết kế để tạo ra một trải nghiệm người dùng dễ sử dụng và thân thiện. Cung cấp hướng dẫn rõ ràng và dễ hiểu về việc điều hướng trang web, cung cấp thông tin cần thiết một cách dễ dàng và tạo ra một không gian trực tuyến yên tĩnh và tĩnh lặng tương tự như khi thăm Chùa Hương.

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

* Có <header> : là phần chứa menu điều hướng chính của trang web,giúp dễ dàng dẫn người trùy cập tới trang mong muốn.
* Có <menu>: là phần menu chứa các liên kết danh sách tùy chọn cho người dùng truy cập
* Có <main>: Là nội dung chính của trang web thức ăn nhanh
* Chân trang (Footer): Chân trang nằm ở cuối trang và chứa thông tin liên hệ, thông tin chủ nhà hàng.



Hình 4:Bố cục trang web

**4.3. Thiết kế trang Web bằng HTML và CSS, JavaScript.**

Sử dụng các loại thẻ để căn chỉnh nội dung phù hợp với màn hình.

* Mục tiêu là giúp cho người dùng tìm kiếm được thông tin họ muốn chỉ bằng cái nhấp chuột.
* Thiết kế giao diện: Sử dụng CSS để thiết kế giao diện các trang web. Điều này bao gồm xác định các màu sắc, phông chữ, kích thước và vị trí của các phần tử. Sử dụng các thuộc tính CSS như color, font-family, font-size, background-color, padding, và margin để điều chỉnh giao diện.
* Tạo đường dẫn và liên kết: Sử dụng thẻ <a> để tạo liên kết và đường dẫn trong trang web. Đảm bảo các liên kết hoạt động chính xác và dẫn đúng đến các trang hoặc vị trí khác trên trang web hoặc ngoài trang web.
* Định dạng văn bản: Sử dụng các thẻ HTML như <h1>, <p>, <span>, <nav> và <div> để định dạng văn bản trên trang web. Sử dụng CSS để điều chỉnh kích thước, kiểu chữ, khoảng cách dòng và các thuộc tính khác của văn bản.
* Tạo hình ảnh và phương tiện: Sử dụng thẻ <img> để chèn hình ảnh và sử dụng CSS để tùy chỉnh kích thước, vị trí và hiển thị của hình ảnh.
* Tạo mảng và 1 số từ khóa giúp chúng ta khi đăng ký hay đăng nhập đều có thể:
* Lưu trữ dữ liệu ( tài khoản vừa đăng nhập)
* Khi đặt hàng sẽ có dòng thông báo đã đặt hàng thành công
* Tăng giảm số lượng hàng muốn đặt
* Kiểm tra điều kiện của tài khoản đăng nhập & đăng ký có đúng với yêu cầu không!, nếu không nó sẽ báo lỗi và phải nhập lại.

**4.5. Kết quả đạt được**

* Tạo ra giao diện trang web đơn giản, sử dụng màu sắc vừa phải, không quá nổi bật.
* Giúp người dùng dễ dàng truy cập và sử dụng thông tin trong trang web, bố cục logic, nội dung được sắp xếp một cách hợp lí, các liên kết chính xác giúp người dùng dễ dàng tìm kiếm được thông tin họ muốn.
* Tóm lại, khi thiết kế trang web bằng HTML5 và CSS5, bạn có thể đạt được một giao diện hấp dẫn hoặc đơn giản tùy vào cách bạn bố trí bố cục của trang web, trải nghiệm người dùng tốt, tối ưu hóa tìm kiếm, dễ dàng duy trì và cập nhật.

**KẾT LUẬN CHƯƠNG IV**

Thiết kế trang web bằng HTML và CSS cung cấp khả năng tạo ra một giao diện hấp dẫn và chuyên nghiệp để giới thiệu về Nhà hàng bán đồ ăn nhanh. Bằng cách sử dụng các yếu tố HTML như tiêu đề, đoạn văn bản, hình ảnh và liên kết, bạn có thể tạo ra một trang web có cấu trúc rõ ràng và dễ đọc. HTML và CSS cũng cho phép bạn tạo ra trang web dễ dàng duy trì và cập nhật. Với cấu trúc rõ ràng của HTML và phân chia giữa HTML và CSS, bạn có thể dễ dàng thực hiện các chỉnh sửa, thay đổi và bảo trì trang web giới thiệu về Nhà hàng bán đồ ăn nhanh một cách hiệu quả. Sử dụng CSS, bạn có thể thiết lập kiểu dáng và bố cục của trang web giới thiệu về Nhà hàng bán đồ ăn nhanh. Bằng cách áp dụng các quy tắc CSS cho các yếu tố HTML, bạn có thể tạo ra giao diện trực quan, sử dụng các màu sắc phù hợp, định dạng văn bản và hình ảnh để tạo nên một trang web hấp dẫn.

Khả năng mở rộng hạn chế có thể gặp khó khăn khi muốn mở rộng chức năng hoặc tính năng phức tạp hơn, giới hạn độ tương thích trình duyệt xuất hiện sai sót hoặc không tương thích trên một số trình duyệt cụ thể, khó khăn trong việc xử lý dữ liệu động là việc hiển thị thông tin từ cơ sở dữ liệu hoặc tương tác người dùng động yêu cầu sử dụng ngôn ngữ lập trình phía máy chủ.

**KẾT LUẬN**

Việc thiết kế trang web bằng HTML và CSS có nhiều lợi ích như cho phép bạn tạo ra giao diện bắt mắt và chuyên nghiệp, dễ dàng thu hút người dùng bằng các màu sắc hay hình ảnh. Giúp trải nghiệm người dùng tốt hơn sử dụng các liên kết một cách linh hoạt nhất. việc thiết kế trang web bằng HTML và CSS mang lại nhiều lợi ích, bao gồm giao diện hấp dẫn, trải nghiệm người dùng tốt, tương thích trên nhiều thiết bị, tối ưu hóa tìm kiếm, dễ dàng duy trì và cập nhật.

Nhưng bên cạnh đó còn một số những nhược điểm cần được chú ý như Khả năng mở rộng hạn chế, thiết kế trang web chỉ sử dụng HTML và CSS có thể gặp khó khăn khi muốn mở rộng chức năng hoặc tính năng phức tạp hơn. Khả năng tương thích trình duyệt Mỗi trình duyệt có thể hiểu và hiển thị HTML và CSS một cách khác nhau, dẫn đến khả năng xuất hiện sai sót hoặc không tương thích trên một số trình duyệt cụ thể. Khó khăn trong việc bảo mật, việc xử lý dữ liệu động, quản lý phiên, xác thực người dùng và các yếu tố bảo mật khác yêu cầu sự hỗ trợ của các ngôn ngữ lập trình phía máy chủ và các công nghệ bảo mật khác.

Chức năng của trang web giúp người dùng có thể tìm kiếm tới nội dung cũng như thông tin mà họ muốn tìm, thiết kế phù hợp với mọi người ,đơn giản không quá cầu kì bắt mắt, tối ưu hóa hiệu suất và tốc độ tải trang của trang web giúp cho người dùng có một trải nghiệm thật tốt.

**DANH MỤC TÀI LIỆU THAM KHẢO**

1. Jon Duckett (2011), *HTML and CSS:Design and Build Websites,* NXB Wiley. Hoa Kỳ
2. Jennifer Robbins (2007), *Learning Web Design,* NXB O'Reilly Media. Hoa Kỳ