**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

**NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG**

*Trà Vinh, ngày ….. tháng …… năm ……*

**Thành viên hội đồng**

*(Ký tên và ghi rõ họ tên)*

LỜI CẢM ƠN

Được sự phân công của Khoa Kỹ thuật và Công nghệ, Trường Đại học Trà Vinh, và dưới sự hướng dẫn của cô Phạm Thị Trúc Mai giảng viên Bộ môn Công nghệ thông tin của trường Đại học Trà Vinh, em đã hoàn thành đề tài “Tìm hiểu Tailwind CSS framework và xây dựng ứng dụng web minh họa”.

Để hoàn thành đề tài này, em xin chân thành cảm ơn các thầy cô giảng viên đã tận tình hướng dẫn, giảng dạy trong suốt quá trình học tập, nghiên cứu và rèn luyện ở Trường Đại học Trà Vinh. Đặc biệt xin gửi lời cảm ơn chân thành tới giảng viên hướng dẫn cô Phạm Thị Trúc Mai đã tận tình, chu đáo hướng dẫn em thực hiện đề tài này.

Mặc dù đã có nhiều cố gắng để thực hiện đề tài một cách hoàn chỉnh nhất. Song do thời gian có hạn và còn hạn chế về kiến thức cho nên trong đồ án không thể tránh khỏi những thiếu sót. Em rất mong nhận được sự đóng góp ý kiến của thầy cô và bạn bè để em có thể hoàn thiện đồ án này tốt hơn. Một lần nữa cho em xin gửi lời cảm ơn đến quý thầy cô, chúc quý thầy cô thật nhiều sức khỏe, thành công và hạnh phúc.

|  |  |
| --- | --- |
|  | **Sinh viên thực hiện** |

**MỤC LỤC**

[LỜI CẢM ƠN iii](#_Toc155509684)

[DANH MỤC HÌNH ẢNH – BẢNG BIỂU vi](#_Toc155509685)

[DANH MỤC TỪ VIẾT TẮT vii](#_Toc155509686)

[TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH viii](#_Toc155509687)

[MỞ ĐẦU ix](#_Toc155509688)

[CHƯƠNG 1 TỔNG QUAN 1](#_Toc155509689)

[CHƯƠNG 2 NGHIÊN CỨU LÝ THUYẾT 2](#_Toc155509690)

[2.1 Tailwind CSS Framework 2](#_Toc155509693)

[2.1.1 Tùy chỉnh cấu hình 4](#_Toc155509694)

[2.1.2 Xử lý Hover, Focus và những trạng thái khác 5](#_Toc155509695)

[2.1.3 Responsive (tương thích nhiều thiết bị) 6](#_Toc155509696)

[2.1.4 Dark mode (chế độ tối) 7](#_Toc155509697)

[2.1.5 Thêm các kiểu tùy chỉnh 7](#_Toc155509698)

[2.1.6 Container (phần tử chứa) 9](#_Toc155509699)

[2.1.7 Columns (cột) 10](#_Toc155509700)

[2.1.8 Flexbox (hộp linh hoạt) 10](#_Toc155509701)

[2.1.9 Typography (kiểu chữ) 15](#_Toc155509702)

[2.1.10 Effect (hiệu ứng) 16](#_Toc155509703)

[2.1.11 Animation (chuyển động) 16](#_Toc155509704)

[2.2 Javascript 17](#_Toc155509705)

[2.2.1 DOM Manipulation (Thao tác với DOM) 17](#_Toc155509706)

[2.2.2 Asynchronous Javascript (Javascript không đồng bộ) 19](#_Toc155509707)

[2.2.3 Local Storage (Lưu trữ cục bộ) 19](#_Toc155509708)

[CHƯƠNG 3 THỰC HIỆN HÓA NGHIÊN CỨU 20](#_Toc155509709)

[3.1 Mô tả bài toán 20](#_Toc155509712)

[3.2 Đặc tả về yêu cầu giao diện 20](#_Toc155509713)

[3.2.1 Yêu cầu chức năng 20](#_Toc155509714)

[3.2.2 Yêu cầu phi chức năng 20](#_Toc155509715)

[3.3 Thiết kế giao diện 21](#_Toc155509716)

[3.3.1 Home (trang chủ) 21](#_Toc155509717)

[3.3.2 Courses (các khóa học) 21](#_Toc155509718)

[3.3.3 My courses (các khóa học theo người dùng) 22](#_Toc155509719)

[3.3.4 Courses detail (chi tiết khóa học khi chưa tham gia) 22](#_Toc155509720)

[3.3.5 My courses detail (chi tiết khóa học khi đã tham gia) 23](#_Toc155509721)

[CHƯƠNG 4 KẾT QUẢ NGHIÊN CỨU 24](#_Toc155509722)

[4.1 Giao diện người dùng 24](#_Toc155509724)

[4.2 Giao diện trang quản trị 28](#_Toc155509725)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 29](#_Toc155509726)

[4.3 Kết luận 29](#_Toc155509727)

[4.3.1 Các vấn đề đã đạt được 29](#_Toc155509728)

[4.3.2 Các vấn đề chưa làm được 29](#_Toc155509729)

[4.4 Hướng phát triển 29](#_Toc155509730)

[DANH MỤC TÀI LIỆU THAM KHẢO 31](#_Toc155509731)

**DANH MỤC HÌNH ẢNH – BẢNG BIỂU**

[Hình 2.1 Bảng màu của Tailwind CSS. 5](#_Toc155509732)

[Hình 2.2 Kết quả hiển thị khi sử dụng `columns-{số lượng}`. 10](#_Toc155509733)

[Hình 2.3 Kết quả hiện thị khi sử dụng `basis-{kích thước}` . 11](#_Toc155509734)

[Hình 2.4 Kết quả hiện thị khi sử dụng `flex-row`. 11](#_Toc155509735)

[Hình 2.5 Kết quả hiển thị khi sử dụng `flex-row-reverse`. 11](#_Toc155509736)

[Hình 2.6 Kết quả hiển thị khi sử dụng `flex-col`. 12](#_Toc155509737)

[Hình 2.7 Kết quả hiển thị khi sử dụng `flex-col-reverse`. 12](#_Toc155509738)

[Hình 2.8 Kết quả hiển thị khi dùng `grid-cols-{n}`. 13](#_Toc155509739)

[Hình 2.9 Kết quả hiển thị khi sử dụng `col-span-{n}`. 13](#_Toc155509740)

[Hình 2.10 Kết quả hiển thị khi sử dụng `grid-rows-{n}`. 14](#_Toc155509741)

[Hình 2.11 Kết quả hiển thị khi sử dụng box shadow color. 16](#_Toc155509742)

[Hình 2.12 Mô hình các đối tượng trong HTML. 18](#_Toc155509743)

[Hình 3.1 Sơ đồ website. 21](#_Toc155509744)

[Hình 3.2 Thiết kế giao diện home. 21](#_Toc155509745)

[Hình 3.3 Thiết kế giao diện hiển thị các khóa học. 22](#_Toc155509746)

[Hình 3.4 Thiết kế giao diện hiển thị các khóa học theo người dùng. 22](#_Toc155509747)

[Hình 3.5 Thiết kế giao diện chi tiết khóa học khi chưa tham gia. 23](#_Toc155509748)

[Hình 3.6 Thiết kế giao diện chi tiết khóa học khi đã tham gia. 23](#_Toc155509749)

[Hình 4.1 Giao diện trang chủ. 24](#_Toc155509750)

[Hình 4.2 Giao diện trang đăng nhập. 24](#_Toc155509751)

[Hình 4.3 Giao diện các khóa học. 25](#_Toc155509752)

[Hình 4.4 Giao diện chi tiết khóa học khi chưa tham gia. 26](#_Toc155509753)

[Hình 4.5 Giao diện các khóa học theo người dùng. 26](#_Toc155509754)

[Hình 4.6 Giao diện chi tiết khóa học khi đã đăng ký khóa học. 27](#_Toc155509755)

[Hình 4.7 Giao diện chi tiết khóa học khi đã hoàn thành khóa học. 28](#_Toc155509756)

[Hình 4.8 Giao diện dashboard cho trang quản trị. 28](#_Toc155509757)

[Bảng 2.1 Danh sách những breakpoints mặc định. 6](#_Toc155509758)

[Bảng 2.2 Bảng những utilities classes flexbox. 12](#_Toc155509759)

[Bảng 2.3 Bảng utilities classes áp dụng cho bố cục dạng lưới. 14](#_Toc155509760)

[Bảng 2.4 Bảng các họ phông chữ. 15](#_Toc155509761)

[Bảng 2.5 Bảng các utilities classes điều chỉnh độ dày phông chữ. 15](#_Toc155509762)

[Bảng 2.6 Bảng utilities classes để căn chỉnh văn bản. 15](#_Toc155509763)

[Bảng 2.7 Bảng các animation thông dụng của Tailwind CSS. 16](#_Toc155509764)

[Bảng 2.8 Bảng các thuộc tính thao tác với DOM thường gặp. 18](#_Toc155509765)

[Bảng 2.9 Bảng các phương thức thao tác với DOM thường gặp. 18](#_Toc155509766)

DANH MỤC TỪ VIẾT TẮT

|  |  |
| --- | --- |
| **Từ viết tắt** | **Ý nghĩa** |
| CLI | Command line interface |
| CSS | Cascading Style Sheets |
| DOM | Document Object Model |
| HTML | HyperText Markup Language |
| NPM | Node package manager |
| UI | User Interface |
| XML | Extensible Markup Language |

TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH

Vấn đề nghiên cứu: Đề tài tập trung vào nghiên cứu và áp dụng Tailwind CSS framework vào việc xây dựng giao diện một trang web dạy và học trực tuyến (E-leaning platform). Tailwind CSS là một CSS framework hiện đại, cho phép phát triển giao diện người dùng một cách nhanh chóng, linh hoạt và dễ dàng tùy chỉnh.

Các hướng tiếp cận: Nghiên cứu sâu về cách hoạt động, cú pháp và cách sử dụng của Tailwind CSS framework. Tìm hiểu các yêu cầu cụ thể về giao diện người dùng, trải nghiệm người dùng của một nền tảng E-learning từ đó áp dụng Tailwind CSS framework để tạo ra một giao diện minh họa về cách nó hoạt động trong thực tế.

Cách giải quyết vấn đề: Tìm hiểu cú pháp, các class CSS được framework cung cấp như bố cục, màu sắc, kích thước,… Để tạo giao diện linh hoạt và tương thích với nhiều thiết bị.

Kết quả đạt được: Nắm vững kiến thức về Tailwind CSS framework và khả năng sử dụng hiệu quả nó trong các dự án phát triển trang web tương tự. Trải qua quá trình nghiên cứu, tạo nên một nền tảng vững chắc hơn về kiến thức, có cơ hội nâng cao kỹ năng thiết kế giao diện. Đồng thời, thành công việc áp dụng Tailwind CSS framework tạo một giao diện linh hoạt, tương thích với nhiều thiết bị và dễ bảo trì.

MỞ ĐẦU

Trang web E-learning thường có các thành phần giao diện đa dạng như trang chủ, trang khóa học, trang tìm kiếm, và các trang chi tiết khóa học. Điều này sẽ cung cấp cơ hội thực hành và thử nghiệm các tính năng của Tailwind CSS framework trên nhiều trường hợp khác nhau.

Một trang web E-learning thường yêu cầu sự linh hoạt và tùy chỉnh cao để tạo ra giao diện thân thiện và dễ sử dụng cho người dùng. Tailwind CSS, với tính linh hoạt trong việc tạo giao diện, sẽ là lựa chọn rất phù hợp để minh họa và thấy rõ tính ứng dụng trong việc tạo ra các thành phần giao diện phức tạp.

Mục đích nghiên cứu: Hiểu rõ cách thức hoạt động, cú pháp và lợi ích của Tailwind CSS framework trong việc xây dựng trang web. Áp dụng Tailwind CSS framework vào việc xây dựng giao diện trang web để thấy rõ được sức mạnh và hạn chế của framework này.

Đối tượng nghiên cứu: Đề tài tập trung nghiên cứu Tailwind CSS framework, cách sử dụng, ứng dụng các class utilities mà nó cung cấp để xây dựng giao diện web.

Phạm vi nghiên cứu: Thiế kế front-end cho trang web học trực tuyến (E-learning) để minh họa cho kết quả của quá trình tìm hiểu về các cú pháp, cách sử dụng class utilities của Tailwind CSS framework, các cách ứng dụng chúng trong thực tế, và các tính năng đặc biệt để thấy rõ ưu điểm và nhược điểm của framework trong môi trường thực tế.

# TỔNG QUAN

Trước Tailwind CSS có các thư viện thành phần như Bootstrap và Material UI [1]. Các thư viện này đi kèm với một tập hợp các thành phần giao diện được xây dựng sẵn một cách đầy đủ như nút, menu, menu thả, thanh điều hướng,… Thay vào đó Tailwind CSS cung cấp các lớp tiện ích để tạo các thành phần bằng cách phối hợp các lớp tiện ích này và những thành phần này có thể tái sử dụng.

Vì lý do đó mà Tailwind CSS cung cấp sự linh hoạt và kiểm soát nhiều hơn so với các CSS framework khác. Điều này cho phép người sử dụng tạo ra một ứng dụng trang web mang tính thương hiệu hóa, cá nhân hóa nhanh chóng và thuận tiện hơn.

Tailwind CSS đã thu hút sự chú ý trong cộng đồng phát triển web với cách tiếp cận mới mẻ, tính tới ngày 9/11/2023, Tailwind CSS có hơn 75.000 sao trên Github [8]. Tailwind CSS cung cấp tập hợp toàn diện các lớp tiện ích bao gồm mọi thứ từ kiểu chữ và khoảng cách đến màu sắc và bố cục được responsive. Tailwind CSS khuyến khích khả năng tái sử dụng theo thiết kế. Thay vì tạo tùy chỉnh cho mọi phần tử. Tailwind CSS có khả năng tùy biến cao. Có thể điều chỉnh framework này để phù hợp với nhu cầu thiết kế cụ thể của dự án bằng cách sửa đổi tệp cấu hình của nó. Điều này cho phép xác định màu sắc, phông chữ tùy chỉnh và hơn thế nữa.

Mặc dù Tailwind CSS cung cấp nhiều lợi ích và thuận lợi trong việc xây dựng giao diện người dùng, nhưng cũng có một số hạn chế. Sử dụng Tailwind CSS có thể làm cho HTML trở nên lớn hơn và khó đọc hơn, đặc biệt là khi áp dụng nhiều class cho mỗi phần tử HTML.

# NGHIÊN CỨU LÝ THUYẾT



## Tailwind CSS Framework

Tailwind CSS được xây dựng dựa trên một kỹ thuật viết CSS là Atomic CSS. Atomic CSS là cách tiếp cận với kiến trúc CSS ưu tiên các class nhỏ, đơn mục đích với tên class dựa trên chức năng trực quan [2]. Thuật ngữ "Atomic CSS" được đặt ra bởi Thierry Koblenz trong một bài viết của ông "Challenging CSS Best Practices" vào tháng 10 năm 2013 [3]. Nó giống như việc tạo ra các class để đạt được một mục đích duy nhất. Ví dụ tạo một class `bg-red`

.bg-red {

background-color: rgb(239 68 68);

}

Tailwind CSS là một CSS framework được xây dựng dựa trên các biến thể của Atomic CSS [2]. Tailwind CSS tập trung chủ yếu vào việc cung cấp một tập hợp các lớp tiện ích (utility classes) [4]. Mỗi lớp đại diện cho một thuộc tính cụ thể như margin, padding, font size, color, border, và nhiều thuộc tính CSS khác. Thay vì viết CSS tùy chỉnh lập trình viên có thể thêm các lớp tiện ích này trực tiếp vào HTML để áp dụng các thuộc tính.

Tailwind CSS được phát triển bằng Javascript, chạy qua thông qua Node.js và cài đặt bằng các trình quản lý gói môi trường như NPM (Node package manager) hoặc Yarn [5].

Cách đơn giản nhất và nhanh nhất để thiết lập Tailwind CSS là sử dụng công cụ NPM của Nodejs.

Vào thư mục lưu trữ web cục bộ trên máy tính, sau đó chạy lệnh sau trong Terminal để cài đặt `taiwindcss` bằng npm và tạo file thiết lập `tailwind.config.js`.

npm install -D tailwindcss

npx tailwindcss init

Tailwind CSS hoạt động bằng cách quét qua tất cả các file HTML, Javascript hoặc bất kì file mẫu nào khác để tìm những class tiện ích của nó cung cấp rồi viết thành những thuộc tính CSS tương ứng và ghi vào file CSS tĩnh.

Thêm đường dẫn đến tất cả các file mẫu này trong file `tailwind.config.js`

module.exports = {

content: ["./src/\*\*/\*.{html,js}"],

theme: {

extend: {},

},

plugins: [],

}

Trước phiên bản Tailwind v3, nó thường tạo ra tất cả các thuộc tính CSS sau đó mới loại bỏ các thuộc tính không được sử dụng, bằng cách này thì nó giữ cho file CSS do Tailwind CSS tạo ra luôn ở kích thước nhỏ nhất có thể [6, 9].

Theo Tailwind CSS, file CSS được tạo ra có dung lượng từ 5-10KB. Nhưng trong quá trình phát triển trang web thì file CSS này có thể có dung lượng rất lớn.

Từ phiên bản v3 trở lên, Tailwind CSS đã phát hành một tính năng mới gọi là trình biên dịch Just-in-Time. Trình biên dịch này tránh việc tạo ra tất cả các thuộc tính CSS ngay từ đầu mà chỉ biên dịch CSS khi lập trình viên viết chúng vào HTML trong thời gian thực [4].

Khởi chạy Tailwind CSS bằng cách sử dụng công cụ CLI (Command line interface) của Tailwind CSS để tìm những class và xây dựng CSS tương ứng.

npx tailwindcss -i ./src/input.css -o ./dist/output.css --watch

Khi này Tailwind CSS sẽ quét qua các file mã đã cấu hình đường dẫn ở trên, và những CSS tùy chỉnh mà lập trình viên đã viết bên trong file `input.css` sẽ được Tailwind CSS biên dịch rồi tạo ra file `output.css`.

Sau đó lập trình viên chỉ cần liên kết file CSS đã được biên dịch này vào HTML.

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8" />

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

<link href="/dist/output.css" rel="stylesheet" />

</head>

<body>

<h1 class="text-3xl font-bold underline">Hello world!</h1>

</body>

</html>

Tailwind CSS, theo trang web chính thức của họ được gọi là “utility-first CSS framework” [7], nó cung cấp các utility classes, mà lập trình viên có thể sử dụng trực tiếp bên trong HTML. Tailwind CSS cung cấp rất nhiều các utility classes, các cấu hình cho việc thiết kế tạo và tạo ra một giao diện website. Tuy nhiên trong phạm vi đề tài chỉ đề cập một số cốt lõi và được sử dụng nhiều nhất.

### Tùy chỉnh cấu hình

Tailwind CSS là một framework để xây dựng giao diện người dùng, nên từ đầu nó được thiết kế để có thể được tùy chỉnh cho phù hợp với từng dự án. Mặc định Tailwind CSS sẽ tìm file `tailwind.config.js` ở thư mục gốc của dự án để xác định những cấu hình được tùy chỉnh, nếu người dùng không cấu hình trong file `tailwind.config.js` thì sẽ trở về cấu hình mặc định của Tailwind CSS.

Những cấu hình mặc định của Tailwind CSS rất mạnh mẽ, có thể kể đến như những breakpoints cho responsive:

/\*\* @type {import('tailwindcss').Config} \*/

module.exports = {

theme: {

screens: {

'sm': '640px',

// => @media (min-width: 640px) { ... }

'md': '768px',

// => @media (min-width: 768px) { ... }

'lg': '1024px',

// => @media (min-width: 1024px) { ... }

'xl': '1280px',

// => @media (min-width: 1280px) { ... }

'2xl': '1536px',

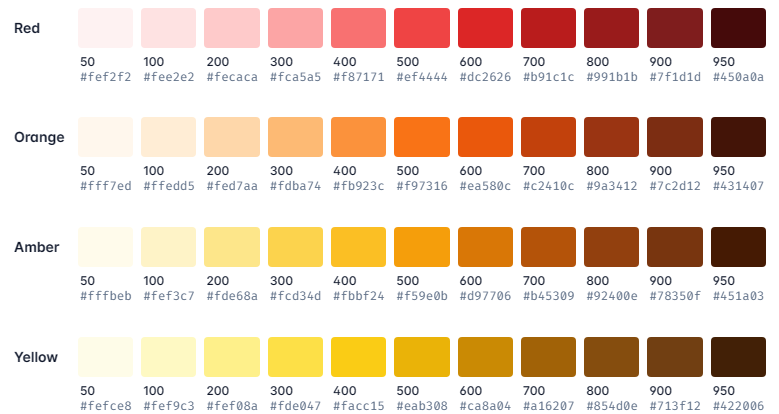
// => @media (min-width: 1536px) { ... }

}

}

}

Tailwind CSS cung cấp một bảng màu mặc định với đầy đủ tất cả màu sắc được thiết kế sẵn một cách chuyên nghiệp.



Hình 2.1 Bảng màu của Tailwind CSS.

Tailwind CSS còn cung cấp những thang đo khoảng cách mặc định và tỷ lệ kích thước mặc định.

/\*\* @type {import('tailwindcss').Config} \*/

module.exports = {

theme: {

spacing: {

'1': '8px',

'2': '12px',

'3': '16px',

'4': '24px',

'5': '32px',

'6': '48px',

}

}

}

Mặc định thang đo này sẽ được kế thừa bởi các thuộc tính như `padding`, `width`, `height`, `gap`,…

### Xử lý Hover, Focus và những trạng thái khác

Mọi utility class nào của Tailwind CSS đều có thể được áp dụng một cách có điều kiện bằng cách thêm tiền tố vào tên class mà người dùng muốn áp dụng khi th mãn điều kiện.

Ví dụ một nút có màu nền là `bg-sky-500` khi di chuột vào thì thay đổi màu nền thành `bg-sky-700`:

<button class="bg-sky-500 hover:bg-sky-700 ...">

Save changes

</button>

Đối với các trạng thái của một phần tử HTML, Tailwind CSS cung cấp đầy đủ các điều khiển. Tuy nhiên, thay vì các trạng thái này được hiểu như các hậu tố trong CSS truyền thống thì đối với Tailwind CSS trở thành các định dạng tiền tố như `hover:`, `focus:`, `first:`, ` before:`, ` after:`,…

Những trạng thái này có thể xếp chồng lên nhau để nhắm tới một điều kiện cụ thể hơn.

<button class="dark:md:hover:bg-fuchsia-600 ...">

Save changes

</button>

### Responsive (tương thích nhiều thiết bị)

Mọi utility class của Tailwind CSS đều có thể áp dụng những điều kiện breakpoints khác nhau.

Tailwind CSS cung cấp sẵn năm breakpoints mặc định từ độ phân giải phổ biến của các thiết bị.

Bảng 2.1 Danh sách những breakpoints mặc định.

|  |  |  |
| --- | --- | --- |
| **Tiền tố breakpoints** | **Chiều rộng tối thiểu** | **CSS** |
| sm | 640px | @media (min-width: 640px) { ... } |
| md | 768px | @media (min-width: 768px) { ... } |
| lg | 1064px | @media (min-width: 1024px) { ... } |
| xl | 1280px | @media (min-width: 1280px) { ... } |
| 2xl | 1536px | @media (min-width: 1536px) { ... } |

Để áp dụng một thuộc tính CSS chỉ có hiệu lực khi ở một breakpoints nhất định, cần thêm một tiền tố breakpoints và dấu `:`.

<!—Mặc định chiều rộng là 16, 32 trên màn hình trung bình, 48 trên màn hình lớn-->

<img class="w-16 md:w-32 lg:w-48" src="...">

### Dark mode (chế độ tối)

Giờ đây chế độ tối là tính năng hàng đầu của các hệ điều hành, việc thiết kế phiên bản tối của website đi cùng với thiết kế mặc định của website đang ngày càng phổ biến hơn.

Trong Tailwind CSS lập trình viên có thể sử dụng tiền tố `dark:` để thay đổi màu sắc của website khi trình duyệt người dùng đang bật chế độ tối.

Mặc định, Tailwind CSS thực hiện điều này bằng cách sử dụng `prefers-color-scheme` một Media Query trong CSS được sử dụng để kiểm tra xem người dùng đang sử dụng chủ đề màu sắc gì. Nhưng lập trình viên có thể xây dựng website hỗ trợ việc chuyển đổi chế độ tối thủ công bằng cách sử dụng các class.

<div class="bg-white dark:bg-black...">

{...}

</div>

### Thêm các kiểu tùy chỉnh

Thông thường khó khăn lớn nhất khi làm việc với một framework là nó không cung cấp đủ các kiểu tùy chỉnh. Tailwind CSS được thiết kế để có thể mở rộng và tùy chỉnh. Do đó, dù người sử dụng đang xây dựng ứng dụng gì thì cũng không mang lại cảm giác đang ghi đè lên chính framework đó.

Ngoài việc lập trình viên có thể thay đổi những thứ như bảng màu, tỉ lệ giãn cách, tỉ lệ kiểu chữ hoặc breakpoints vào phần `theme` trong file `tailwind.config.js`. Để có thể sử dụng các giá trị tùy ý (arbitrary values).

Để sử dụng các giá trị tùy ý bằng cách bỏ các hậu tố vào dấu ngoặc vuông `[…]`.

<div class="top-[117px]">

<!-- ... -->

</div>

Và nó hoạt động với hầu hết mọi class của framework, bao gồm cả màu nền, font-size, nội dung của pseudo-element.

<div class="bg-[#bada55] text-[22px] before:content-['Festivus']">

<!-- ... -->

</div>

Lập trình viên cũng có thể sử dụng những giá trị tùy chỉnh này với các biến của CSS mà không cần bao biến đó bởi `var(…)`.

<div class="bg-[--my-color]">

<!-- ... -->

</div>

Ngoài ra có thể sử dụng các thuộc tính CSS mà framework chưa hỗ trợ vào ngoặc vuông.

<div class="[mask-type:luminance] hover:[mask-type:alpha]">

<!-- ... -->

</div>

Đối với các giá trị tùy chỉnh có chứa khoảng trắng lập trình viên cần thay dấu khoảng trắng thành dấu gạch dưới (`\_`). Tailwind CSS sẽ tự động chuyển đổi dấu gạch dưới thành dấu khoảng trắng.

<div class="grid grid-cols-[1fr\_500px\_2fr]">

<!-- ... -->

</div>

Trong trường hợp việc chuyển đổi không hợp lệ thì Tailwind CSS sẽ giữ nguyên dấu gạch dưới, ví dụ trường hợp trong URL.

<div class="bg-[url('/what\_a\_rush.png')]">

<!-- ... -->

</div>

Trong trường hợp lập trình viên cần sử dụng dấu gạch dưới nhưng khi chuyển đổi sang dấu khoảng trắng vẫn hợp lệ thì cần thêm dấu gạch chéo ngược `\` và Tailwind CSS sẽ không chuyển đổi nó sang dấu khoảng trắng.

<div class="before:content-['hello\\_world']">

<!-- ... -->

</div>

Nếu như lập trình viên muốn đặt một số giá trị mặc định cho trang (như màu chữ, màn nền hoặc phông chữ), thì cách đơn giản nhất chỉ cần thêm classes vào thẻ `html` hoặc `body`.

<!doctype html>

<html lang="en" class="text-gray-900 bg-gray-100 font-serif">

<!-- ... -->

</html>

### Container (phần tử chứa)

Class `container` được Tailwind CSS cung cấp sẵn để giới hạn độ rộng tối đa của một phần tử với độ rộng tối thiểu của từng breakpoints. Việc này có thể giúp lập trình viên thiết kế giao diện cho một nhóm các kích thước màn hình nhất định thay vì cố gắng đáp ứng toàn bộ kích thước màn hình.

Lưu ý rằng container của Tailwind CSS không giống ở những framework khác khi nó không tự động căn giữa hoặc có những thuộc tính `padding` hay `margin`. Để căn giữa các containers theo mặc định thì đặt tùy chọn `center` thành `true` trong phần `theme.container` trong file cấu hình.

/\*\* @type {import('tailwindcss').Config} \*/

module.exports = {

theme: {

container: {

center: true,

},

},

}

Ta cũng có thể thêm phần đệm theo mặc định theo từng breakpoints khác nhau.

/\*\* @type {import('tailwindcss').Config} \*/

module.exports = {

theme: {

container: {

padding: {

DEFAULT: '1rem',

sm: '2rem',

lg: '4rem',

xl: '5rem',

'2xl': '6rem',

},

},

},

};

### Columns (cột)

Sử dụng `columns-{số lượng}` để đặt số lượng cột để tạo nội dung trong một phần tử. Độ rộng của cột sẽ được tự động điều chỉnh để phù hợp với số lượng cột.

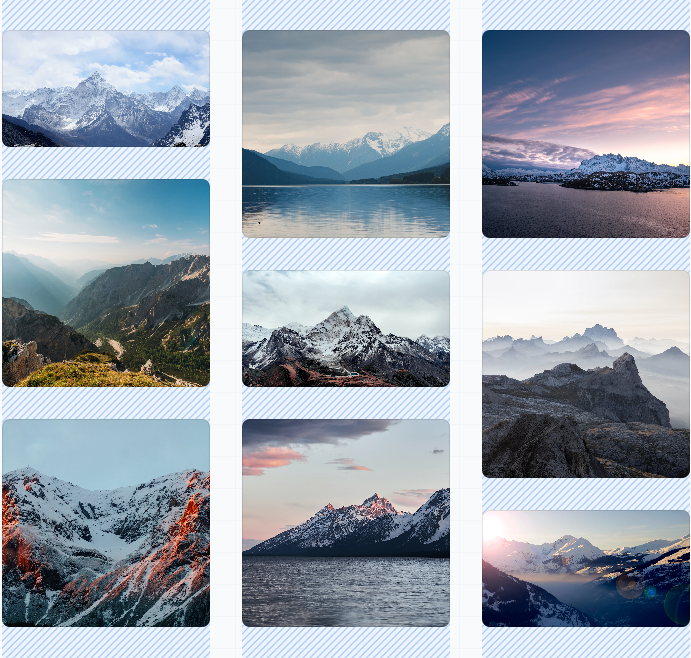
<div class="columns-3 ...">

<img class="w-full aspect-video ..." src="..." />

<img class="w-full aspect-square ..." src="..." />

<!-- ... -->

</div>



Hình 2.2 Kết quả hiển thị khi sử dụng `columns-{số lượng}`.

### Flexbox (hộp linh hoạt)

Class `basis-{kích thước}` dùng để kiểm soát kích thước ban đầu của các flex items.

<div class="flex flex-row">

<div class="basis-1/4">01</div>

<div class="basis-1/4">02</div>

<div class="basis-1/2">03</div>

</div>



Hình 2.3 Kết quả hiện thị khi sử dụng `basis-{kích thước}` .

Sử dụng `flex-row` để điều chỉnh vị trí các flex items theo chiều ngang.

<div class="flex flex-row ...">

<div>01</div>

<div>02</div>

<div>03</div>

</div>



Hình 2.4 Kết quả hiện thị khi sử dụng `flex-row`.

Sử dụng `flex-row-reverse` để điều chỉnh vị trí các flex items theo hướng ngược lại.

<div class="flex flex-row-reverse ...">

<div>01</div>

<div>02</div>

<div>03</div>

</div>



Hình 2.5 Kết quả hiển thị khi sử dụng `flex-row-reverse`.

Sử dụng `flex-col` để điều chỉnh các flex items theo chiều dọc.

<div class="flex flex-col ...">

<div>01</div>

<div>02</div>

<div>03</div>

</div>



Hình 2.6 Kết quả hiển thị khi sử dụng `flex-col`.

Sử dụng `flex-col-reverse` để điều chỉnh các flex items dọc và theo chiều ngược lại.

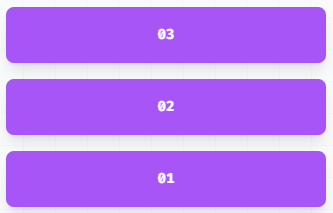
<div class="flex flex-col-reverse ...">

<div>01</div>

<div>02</div>

<div>03</div>

</div>



Hình 2.7 Kết quả hiển thị khi sử dụng `flex-col-reverse`.

Ngoài ra còn những utilities classes flexbox khác:

Bảng 2.2 Bảng những utilities classes flexbox.

|  |  |
| --- | --- |
| **Class** | **Properties** |
| flex-wrap | flex-wrap: wrap; |
| flex-wrap-reverse | flex-wrap: wrap-reverse; |
| flex-nowrap | flex-wrap: nowrap; |
| flex-1 | flex: 1 1 0%; |
| flex-auto | flex: 1 1 auto; |
| flex-initial | flex: 0 1 auto; |
| flex-none | flex: none; |
| grow | flex-grow: 1; |
| grow-0 | flex-grow: 0; |
| shrink | flex-shrink: 1; |
| shrink-0 | flex-shrink: 0; |

Trong bố cục dạng lưới (grid) lập trình viên có thể dùng `grid-cols-{n}` để tạo lưới với n cột với kích thước bằng nhau.

<div class="grid grid-cols-4 gap-4">

<div>01</div>

<!-- ... -->

<div>09</div>

</div>



Hình 2.8 Kết quả hiển thị khi dùng `grid-cols-{n}`.

Sử dụng `col-span-{n}` để tạo một phần tử chiếm n cột.

<div class="grid grid-cols-3 gap-4">

<div class="...">01</div>

<div class="...">02</div>

<div class="...">03</div>

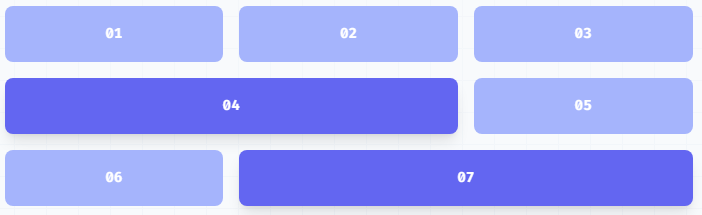
<div class="col-span-2 ...">04</div>

<div class="...">05</div>

<div class="...">06</div>

<div class="col-span-2 ...">07</div>

</div>



Hình 2.9 Kết quả hiển thị khi sử dụng `col-span-{n}`.

Ta có thể dùng `grid-rows-{n}` để tạo lưới với n hàng có kích thước bằng nhau.

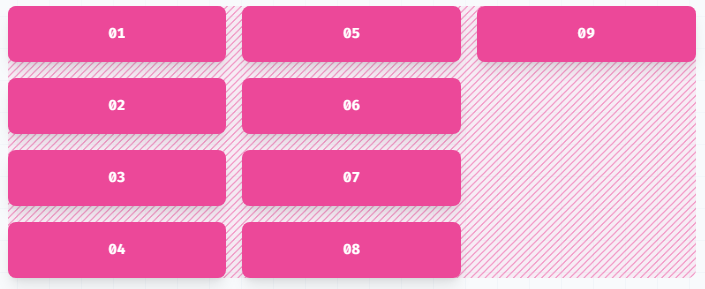
<div class="grid grid-rows-4 grid-flow-col gap-4">

<div>01</div>

<!-- ... -->

<div>09</div>

</div>



Hình 2.10 Kết quả hiển thị khi sử dụng `grid-rows-{n}`.

Ngoài ra còn nhiều utilities classes khác áp dụng cho bố cục dạng lưới như:

Bảng 2.3 Bảng utilities classes áp dụng cho bố cục dạng lưới.

|  |  |
| --- | --- |
| **Class** | **Properties** |
| row-auto | grid-row: auto; |
| row-span-1 | grid-row: span 1 / span 1; |
| row-span-full | grid-row: 1 / -1; |
| row-start-1 | grid-row-start: 1; |
| row-start-auto | grid-row-start: auto; |
| row-end-1 | grid-row-end: 1; |
| row-end-auto | grid-row-end: auto; |
| grid-flow-row | grid-auto-flow: row; |
| grid-flow-col | grid-auto-flow: column; |
| grid-flow-dense | grid-auto-flow: dense; |
| grid-flow-row-dense | grid-auto-flow: row dense; |
| grid-flow-col-dense | grid-auto-flow: column dense; |
| auto-cols-auto | grid-auto-columns: auto; |
| auto-cols-min | grid-auto-columns: min-content; |
| auto-cols-max | grid-auto-columns: max-content; |
| auto-cols-fr | grid-auto-columns: minmax(0, 1fr); |
| auto-rows-auto | grid-auto-rows: auto; |
| auto-rows-min | grid-auto-rows: min-content; |
| auto-rows-max | grid-auto-rows: max-content; |
| auto-rows-fr | grid-auto-rows: minmax(0, 1fr); |

### Typography (kiểu chữ)

Tailwind CSS cũng cung cấp sẵn các utilities classes để kiểm soát kiểu chữ, như là họ phông chữ của một phần tử.

Bảng 2.4 Bảng các họ phông chữ.

|  |  |
| --- | --- |
| **Class** | **Properties** |
| font-sans | font-family: ui-sans-serif, system-ui, -apple-system, BlinkMacSystemFont, "Segoe UI", Roboto, "Helvetica Neue", Arial, "Noto Sans", sans-serif, "Apple Color Emoji", "Segoe UI Emoji", "Segoe UI Symbol", "Noto Color Emoji"; |
| font-serif | font-family: ui-serif, Georgia, Cambria, "Times New Roman", Times, serif; |
| font-mono | font-family: ui-monospace, SFMono-Regular, Menlo, Monaco, Consolas, "Liberation Mono", "Courier New", monospace; |

Tailwind CSS cũng cung cấp cho người dùng các utilities classes để kiểm soát độ dày phông chữ của một phần tử.

Bảng 2.5 Bảng các utilities classes điều chỉnh độ dày phông chữ.

|  |  |
| --- | --- |
| **Class** | **Properties** |
| font-thin | font-weight: 100; |
| font-extralight | font-weight: 200; |
| font-light | font-weight: 300; |
| font-normal | font-weight: 400; |
| font-medium | font-weight: 500; |
| font-semibold | font-weight: 600; |
| font-bold | font-weight: 700; |
| font-extrabold | font-weight: 800; |
| font-black | font-weight: 900; |
| font-thin | font-weight: 100; |

Các utilities classes để căn chỉnh văn bản.

Bảng 2.6 Bảng utilities classes để căn chỉnh văn bản.

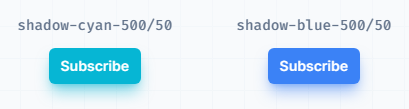
|  |  |
| --- | --- |
| **Class** | **Properties** |
| text-left | text-align: left; |
| text-center | text-align: center; |
| text-right | text-align: right; |
| text-justify | text-align: justify; |
| text-start | text-align: start; |
| text-end | text-align: end; |

### Effect (hiệu ứng)

Đặc biệt Tailwind CSS còn cung cấp các utilities classes có sẵn để tạo hiệu ứng như box shadow hay box shadow color.

<button class="bg-cyan-500 shadow-lg shadow-cyan-500/50 ...">Subscribe</button>

<button class="bg-blue-500 shadow-lg shadow-blue-500/50 ...">Subscribe</button>



Hình 2.11 Kết quả hiển thị khi sử dụng box shadow color.

### Animation (chuyển động)

Tailwind CSS cung cấp hàng loạt các animation thông dụng như là spin (chuyển động xoay tròn), ping (chớp), pulse (mờ dần), bounce (nảy lên xuống), lập trình viên chỉ cần thêm utilities `animate-{kiểu chuyển động}` vào mà không cần viết CSS tùy chỉnh rất dài.

Bảng 2.7 Bảng các animation thông dụng của Tailwind CSS.

|  |  |
| --- | --- |
| **Class** | **Properties** |
| animate-none | animation: none; |
| animate-spin | animation: spin 1s linear infinite;  @keyframes spin {  from {  transform: rotate(0deg);  }  to {  transform: rotate(360deg);  }  } |
| animate-ping | animation: ping 1s cubic-bezier(0, 0, 0.2, 1) infinite;  @keyframes ping {  75%, 100% {  transform: scale(2);  opacity: 0;  }  } |
| animate-pulse | animation: pulse 2s cubic-bezier(0.4, 0, 0.6, 1) infinite;  @keyframes pulse {  0%, 100% {  opacity: 1;  }  50% {  opacity: .5;  }  } |
| animate-bounce | animation: bounce 1s infinite;  @keyframes bounce {  0%, 100% {  transform: translateY(-25%);  animation-timing-function: cubic-bezier(0.8, 0, 1, 1);  }  50% {  transform: translateY(0);  animation-timing-function: cubic-bezier(0, 0, 0.2, 1);  }  } |

## Javascript

Javascript là một ngôn ngữ lập trình phổ biến được sử dụng chủ yếu trong lập trình web. Nó là một ngôn ngữ kịch bản (scripting language) có khả năng tương tác với HTML để thao tác được với các thẻ HTML thì nó phải thông qua một cơ chế ta gọi là DOM.

### DOM Manipulation (Thao tác với DOM)

DOM (Document Object Model) là mô hình các đối tượng trong HTML. Trong mỗi thẻ HTML sẽ có những thuộc tính (Properties) và có phân cấp cha - con với các thẻ HTML khác. Sự phân cấp và các thuộc tính của thẻ HTML này được gọi là selector và trong DOM sẽ có nhiệm vụ xử lý các vấn đề như đổi thuộc tính của thẻ, đổi cấu trúc HTML của thẻ,… DOM được dùng để truy xuất các HTML và XML, có dạng một cây cấu trúc dữ liệu [10].



Hình 2.12 Mô hình các đối tượng trong HTML.

DOM cho phép các ngôn ngữ lập trình như Javascript thao tác với các phần tử HTML và XML trên trang web. Thông qua DOM, lập trình viên có thể thay đổi cấu trúc, nội dung của trang web.

Các thuộc tính và phương thức thường gặp:

Bảng 2.8 Bảng các thuộc tính thao tác với DOM thường gặp.

|  |  |
| --- | --- |
| **Properties** | **Diễn giải** |
| id | Định danh – là duy nhất cho mỗi phần tử nên thường được dùng để truy xuất DOM trực tiếp và nhanh chóng. |
| **class** | Tên lớp – Cũng dùng để truy xuất trực tiếp như id, nhưng 1 className có thể dùng cho nhiều phần tử. |
| **tagName** | Tên thẻ HTML. |
| **innerHTML** | Trả về mã HTML bên trong phần tử hiện tại. Đoạn mã HTML này là chuỗi kí tự chứa tất cả phần tử bên trong, bao gồm các nút phần tử và nút văn bản. |
| **outerHTML** | Trả về mã HTML của phần tử hiện tại |
| **textContent** | Trả về 1 chuỗi kí tự chứa nội dung của tất cả nút văn bản bên trong phần tử hiện tại. |
| **attributes** | Tập các thuộc tính như id, name, class, href, title… |
| **style** | Tập các định dạng của phần tử hiện tại |
| **value** | Lấy giá trị của thành phần được chọn thành một biến. |

Bảng 2.9 Bảng các phương thức thao tác với DOM thường gặp.

|  |  |
| --- | --- |
| **Method** | **Diễn giải** |
| getElementById(id) | Tham chiếu đến một nút duy nhất có thuộc tính id giống với id cần tìm. |
| **getElementsByTagName(tagname)** | Tham chiếu đến tất cả các nút có thuộc tính tagName giống với tên thẻ cần tìm. |
| **getElementsByName(name)** | Tham chiếu đến tất cả các nút có thuộc tính name cần tìm. |
| getAttribute(attributeName) | Lấy giá trị của thuộc tính. |
| **setAttribute(attributeName, value)** | Sửa giá trị của thuộc tính. |
| **appendChild(node)** | Thêm một nút con vào nút hiện tại. |
| **removeChild(node)** | Xóa một nút con khỏi nút hiện tại. |

### Asynchronous Javascript (Javascript không đồng bộ)

Asynchronous Javascript là cách thức thực hiện các tác vụ mà không chờ đợi các tác vụ khác hoàn thành trước. Nó sử dụng các đối tượng như Promises, async/await để thực hiện các tác vụ mà không chặn luồng thực thi chính của chương trình. Điều này giúp tránh việc chương trình bị ngưng đột ngột khi thực hiện các tác vụ mất thời gian [11].

Sử dụng async và await để xử lý các hàm bất đồng bộ, cho phép gọi các hàm trả về Promise một cách đồng bộ hơn.

export default async () => {

try {

const data = await getData()

//Xử lý dữ liệu nhận được

} catch (error) {

console.log(error)

}

}

### Local Storage (Lưu trữ cục bộ)

Sử dụng `localStorage.getItem` và `localStorage.setItem` để lưu trữ và truy xuất dữ liệu trong local storage của trình duyệt.

//tạo một biến lưu trữ cục bộ vào trình duyệt web

localStorage.setItem(keyName, keyValue)

//nhận biến lưu trữ cục bỗ đã lưu

localStorage.getItem(keyName, keyValue)

# THỰC HIỆN HÓA NGHIÊN CỨU



## Mô tả bài toán

Yêu cầu đặt ra của đề tài là: Xây dựng một trang web E-learning trong đó sử dụng Tailwind CSS và Javascipt để hiển thị danh sách các khóa học, bao gồm tên khóa học và tùy chọn xem chi tiết về khóa học. Người dùng đã đăng nhập vào tài khoản của mình trên trang web, người dùng có xem chi tiết về khóa học mà họ muốn tham gia, hệ thống sẽ hiển thị thông tin về khóa học bao gồm hình ảnh minh họa (thumbnail), giới thiệu về khóa học và tùy chọn tham gia khóa học. Sau khi người dùng đã chọn tham gia khóa học, nội dung của khóa học gồm video bài giảng, giới thiệu về khóa học, thanh tiến độ hoàn thành khóa học và phần nội dung lý thuyết khóa học. Sau khi người dùng hoàn thành khóa học hệ thống sẽ cấp một giấy chứng nhận đã hoàn thành khóa học. Người dùng có thể xem lại các khóa học đang tham gia hoặc đã hoàn thành.

## Đặc tả về yêu cầu giao diện

### Yêu cầu chức năng

**Hiển thị danh sách các khóa học:** Hiển thị danh sách các khóa học và tùy chọn xem chi tiết thông tin của từng khóa học.

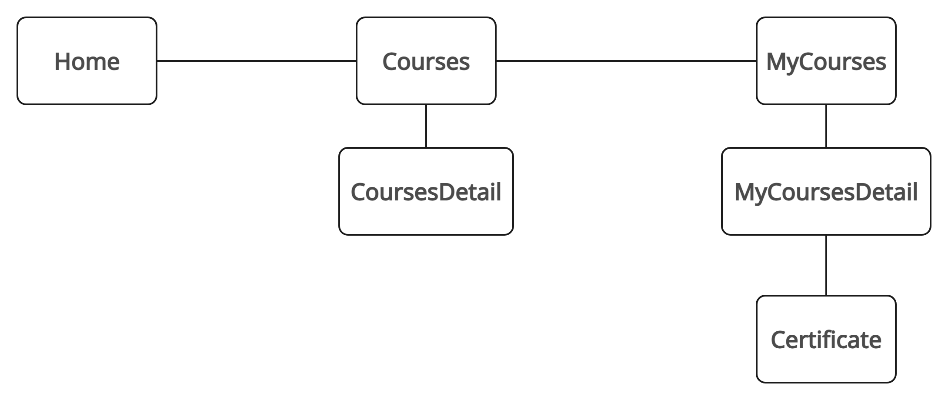
**Hiển thị danh sách các khóa học theo người dùng:** Hiển thị danh sách các khóa học mà người dùng đã tham gia hoặc đã hoàn thành và tùy chọn xem chi tiết thông tin của từng khóa học.

### Yêu cầu phi chức năng

Thiết kế giao diện thân thiện với người dùng: Giao diện dễ đọc, có chế độ tối thân thiện với mắt, dễ sử dụng. Các nút điều hướng cần được thiết kế một cách rõ ràng và dễ sử dụng để người dùng có thể dễ dàng điều hướng giữa các trang.

Tính tương thích: Đảm bảo giao diện hiển thị đẹp và đầy đủ chức năng trên mọi thiết bị khác nhau, đảm bảo đồng nhất về kiểu chữ, màu sắc.

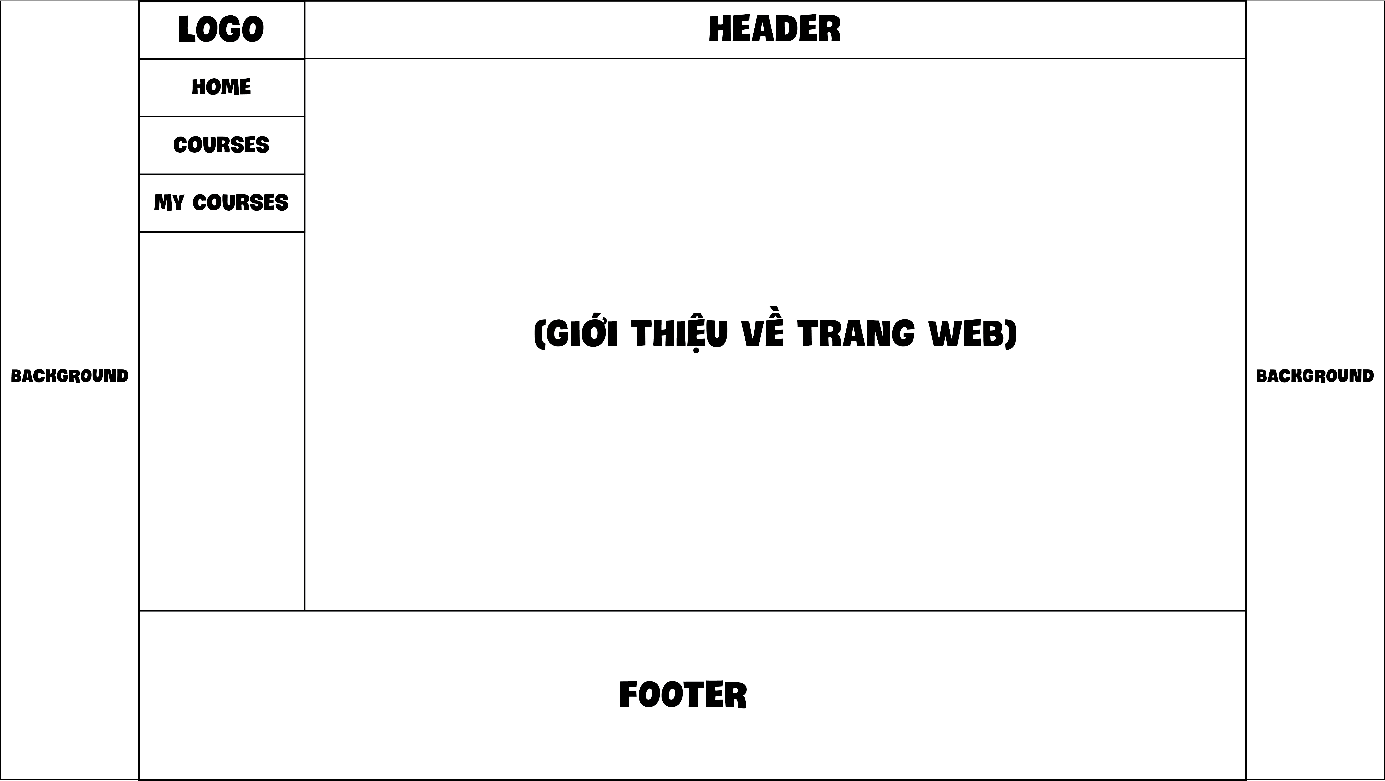
## Thiết kế giao diện



Hình 3.1 Sơ đồ website.

### Home (trang chủ)

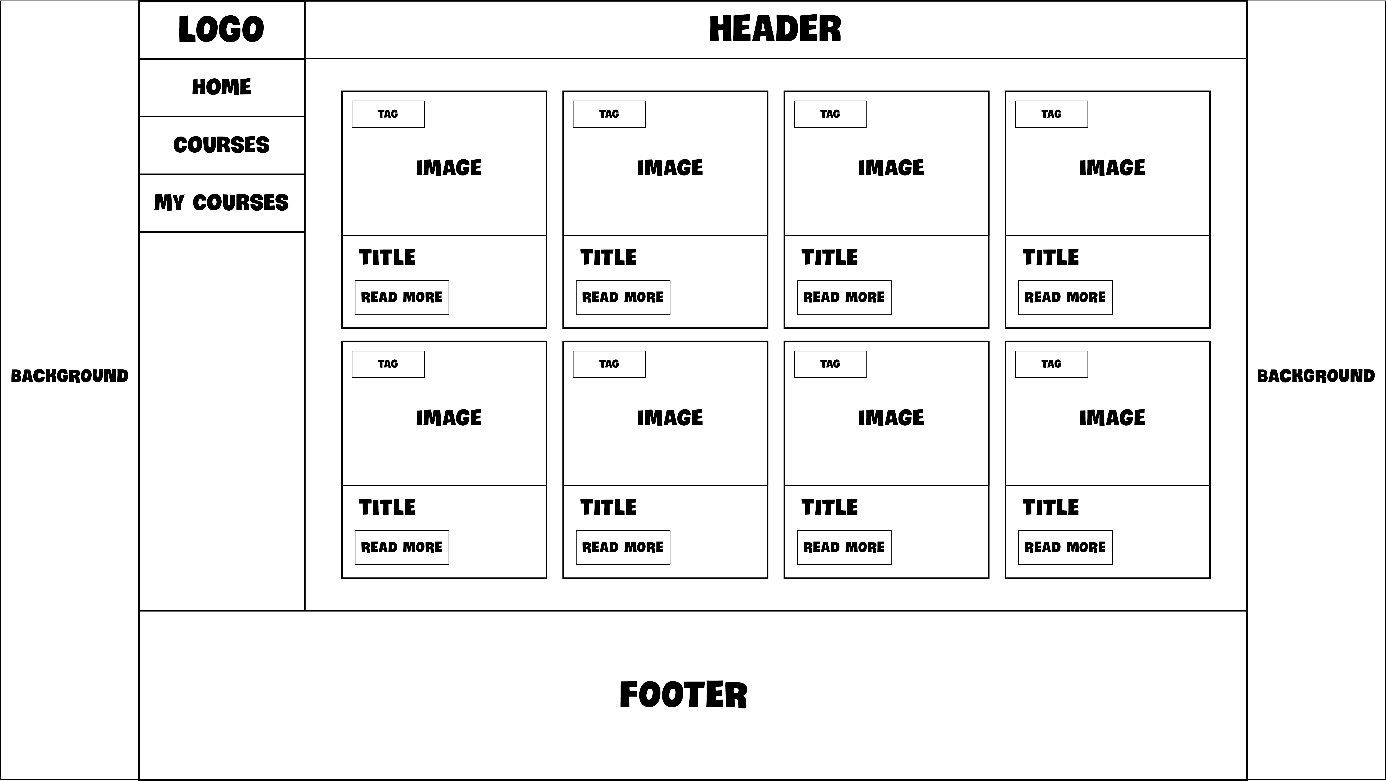
Mô tả: Trang home có mục giới thiệu về trang web bao gồm chào mừng người dùng khi người dùng truy cập vào, giới thiệu về số lượng khóa học, số lượng học viên, …



Hình 3.2 Thiết kế giao diện home.

### Courses (các khóa học)

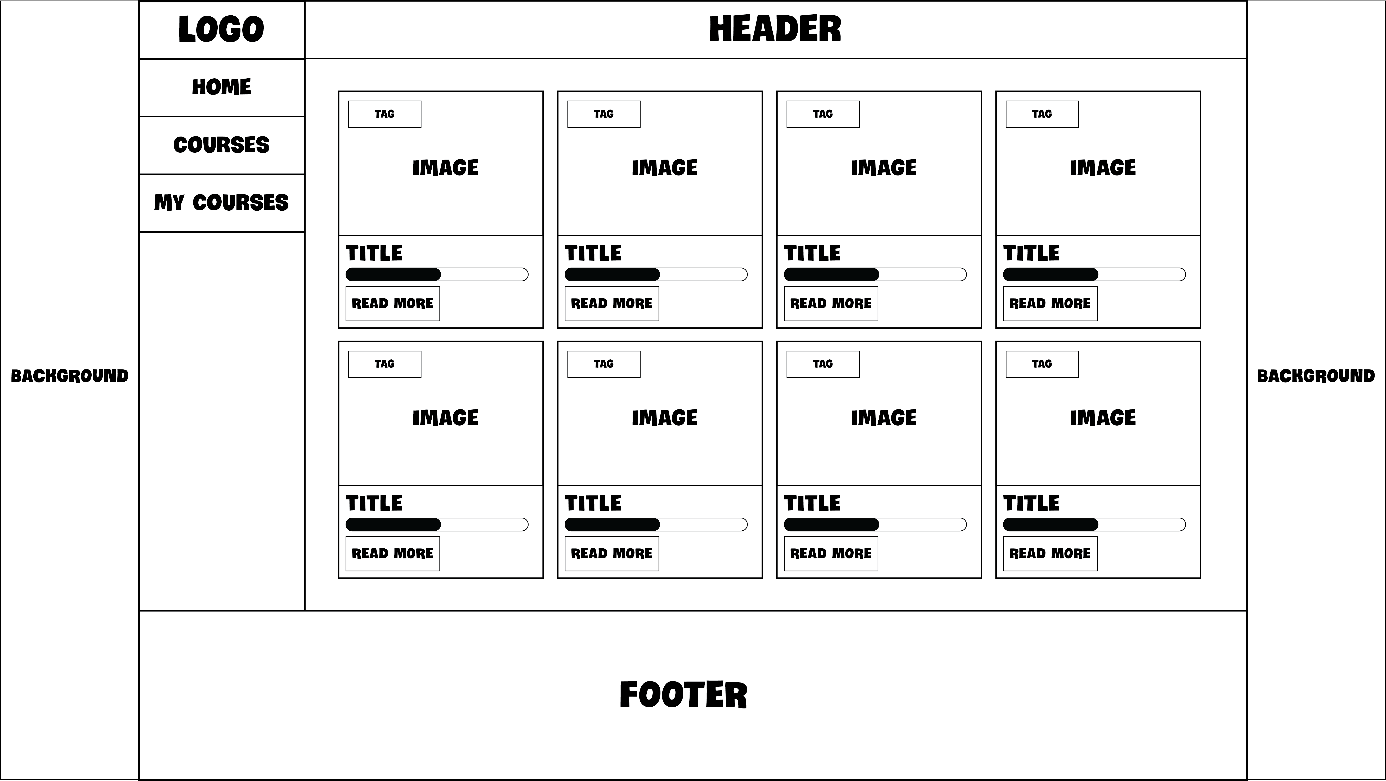
Mô tả: Trang courses hiển thị danh sách các khóa học gồm hình ảnh minh họa về khóa học, chủ đề khóa học, tiêu đề và nút xem chi tiết khóa học.



Hình 3.3 Thiết kế giao diện hiển thị các khóa học.

### My courses (các khóa học theo người dùng)

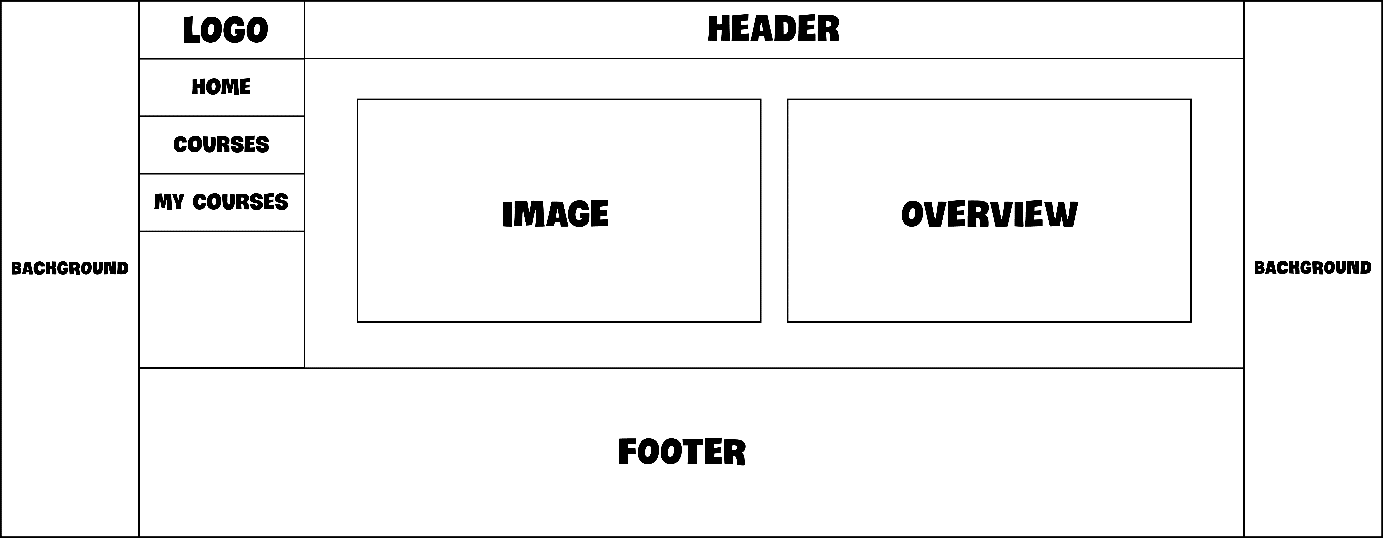
Mô tả: Trang courses hiển thị danh sách các khóa học mà người dùng đã tham gia gồm hình ảnh minh họa về khóa học, chủ đề khóa học, tiêu đề, thanh tiến độ và nút xem chi tiết khóa học.



Hình 3.4 Thiết kế giao diện hiển thị các khóa học theo người dùng.

### Courses detail (chi tiết khóa học khi chưa tham gia)

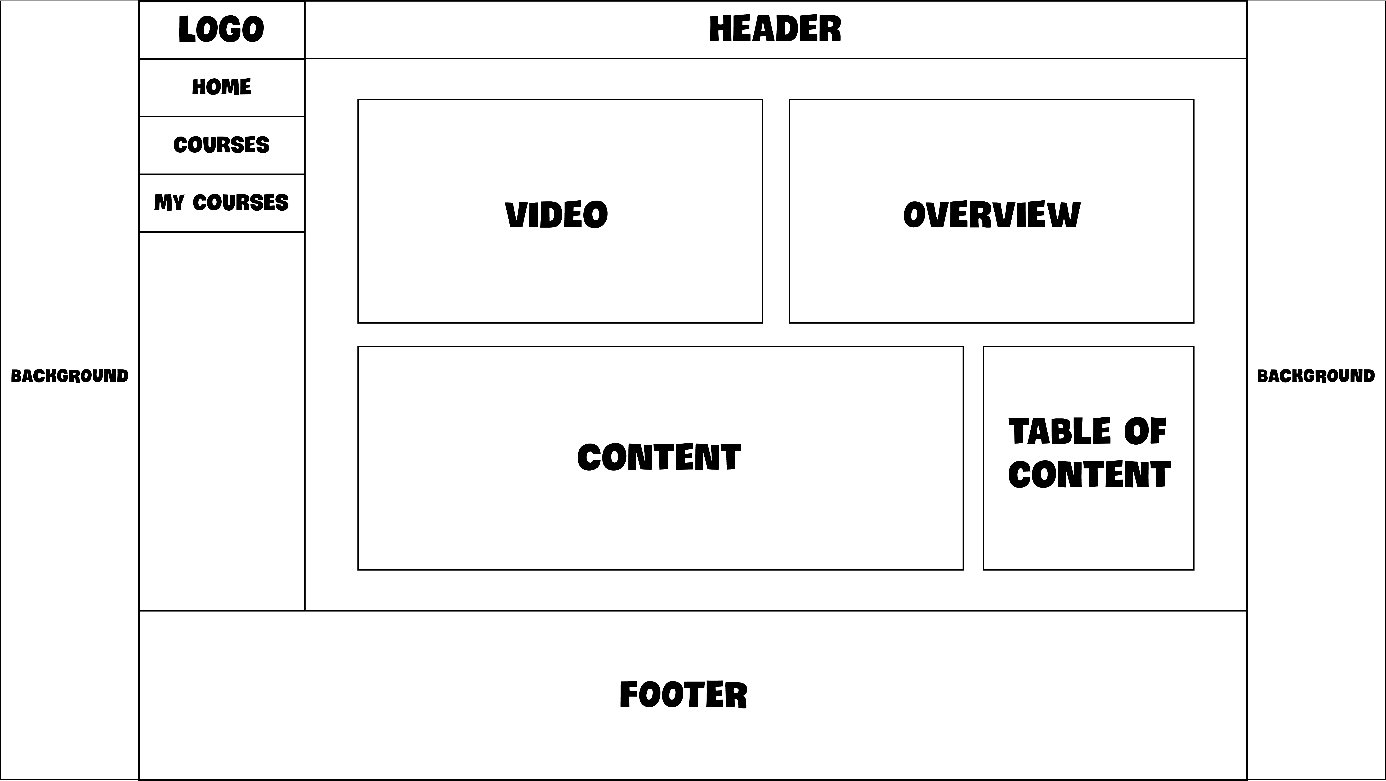
Mô tả: Trang courses detail hiển thị hình ảnh minh họa cho khóa học và phần giới thiệu khóa học kèm theo tùy chọn tham gia khóa học.



Hình 3.5 Thiết kế giao diện chi tiết khóa học khi chưa tham gia.

### My courses detail (chi tiết khóa học khi đã tham gia)

Mô tả: Trang my courses detail được hiển thị khi người dùng đã đăng ký khóa học này, bao gồm video giảng dạy, giới thiệu về khóa học, nội dung khóa học và bảng mục lục cho các nội dung khóa học.



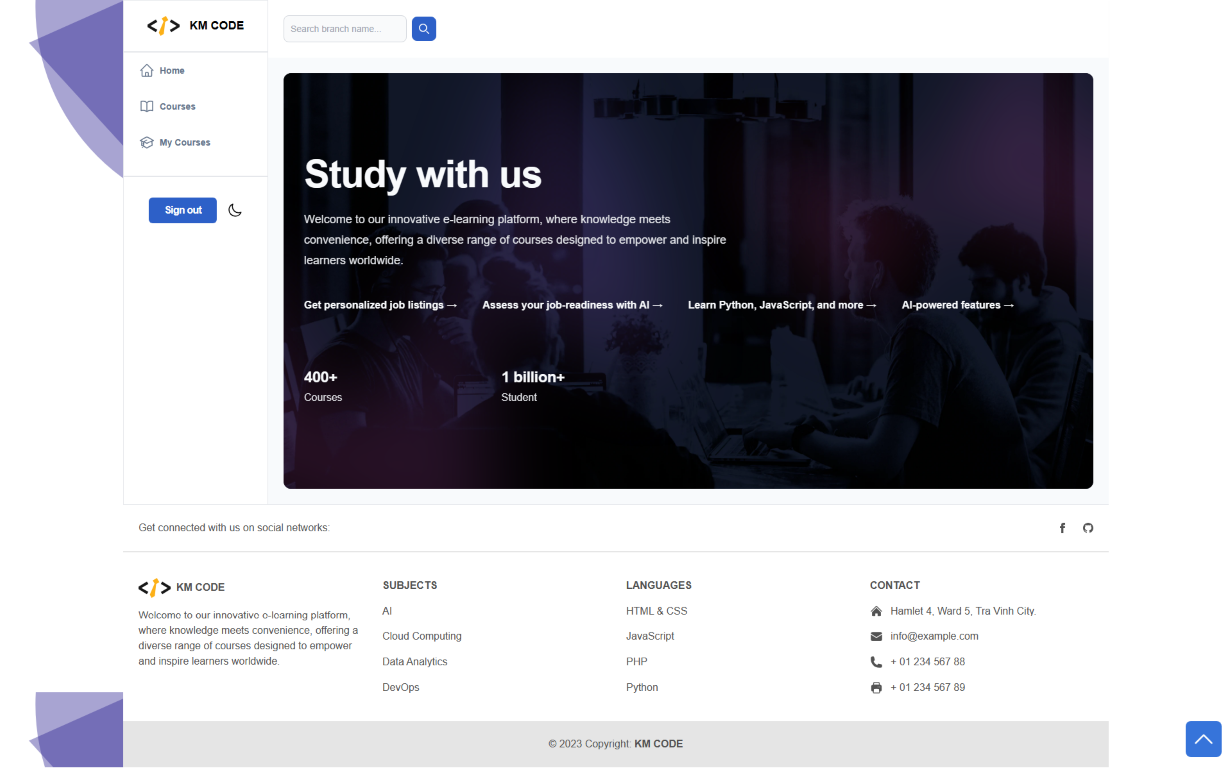
Hình 3.6 Thiết kế giao diện chi tiết khóa học khi đã tham gia.

# KẾT QUẢ NGHIÊN CỨU



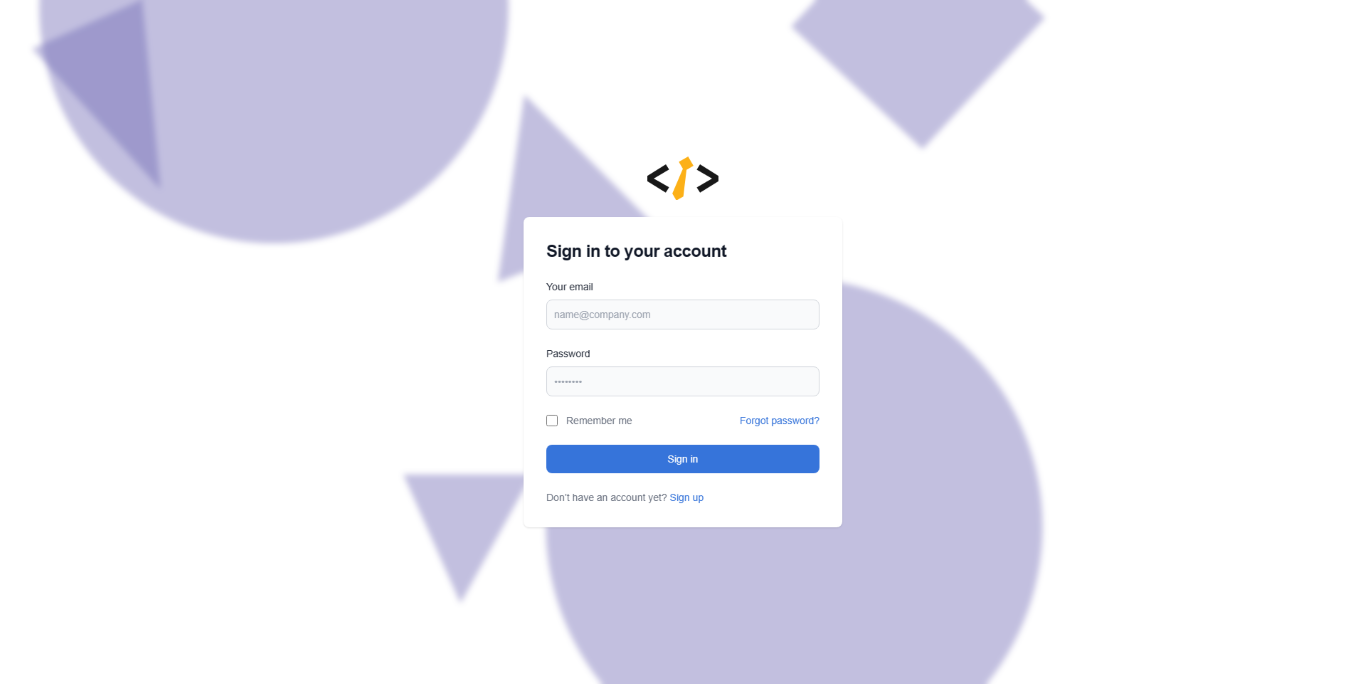
## Giao diện người dùng

Giao diện trang chủ của website:



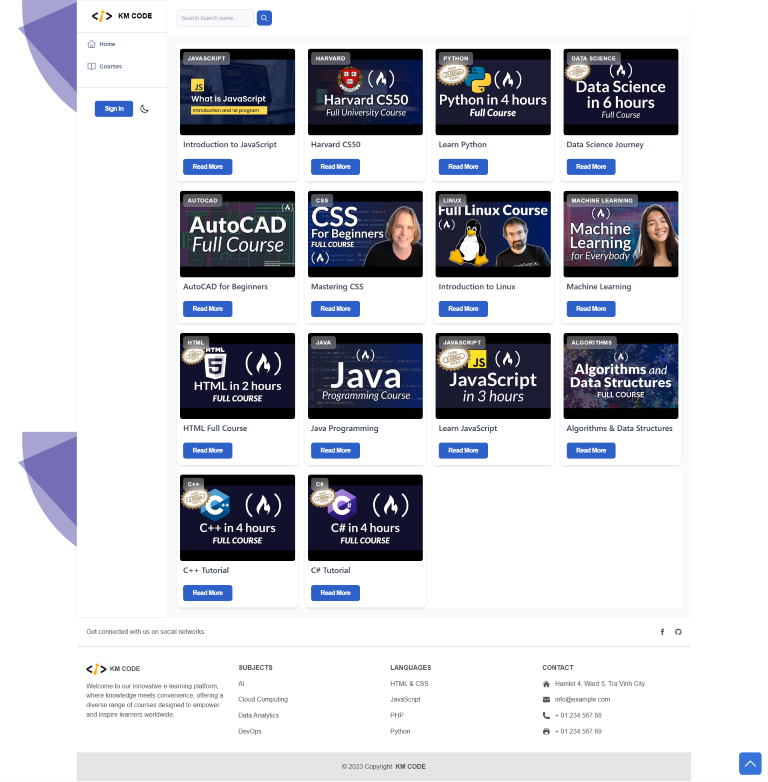
Hình 4.1 Giao diện trang chủ.

Giao diện trang đăng nhập:



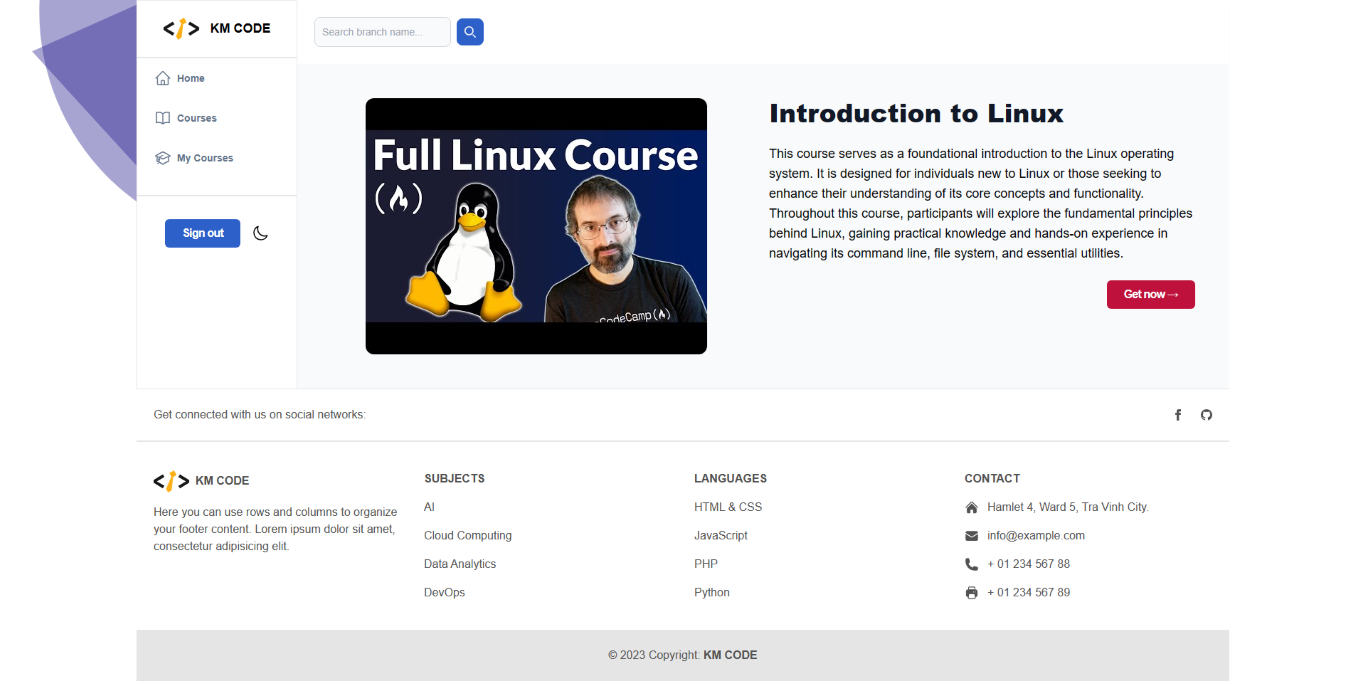
Hình 4.2 Giao diện trang đăng nhập.

Giao diện hiện thị danh sách các khóa học:



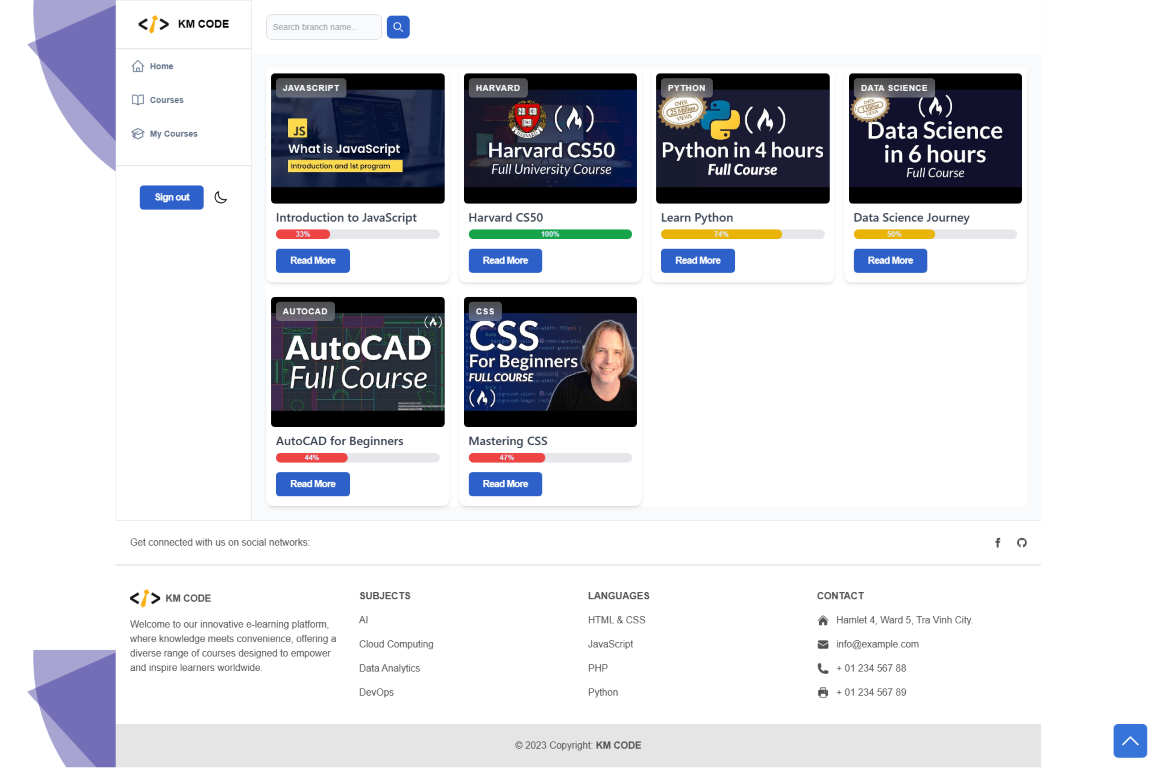
Hình 4.3 Giao diện các khóa học.

Giao diện hiển thị chi tiết về khóa học khi người dùng chưa đăng nhập tài khoản hoặc đã đăng nhập nhưng chưa tham gia khóa học:



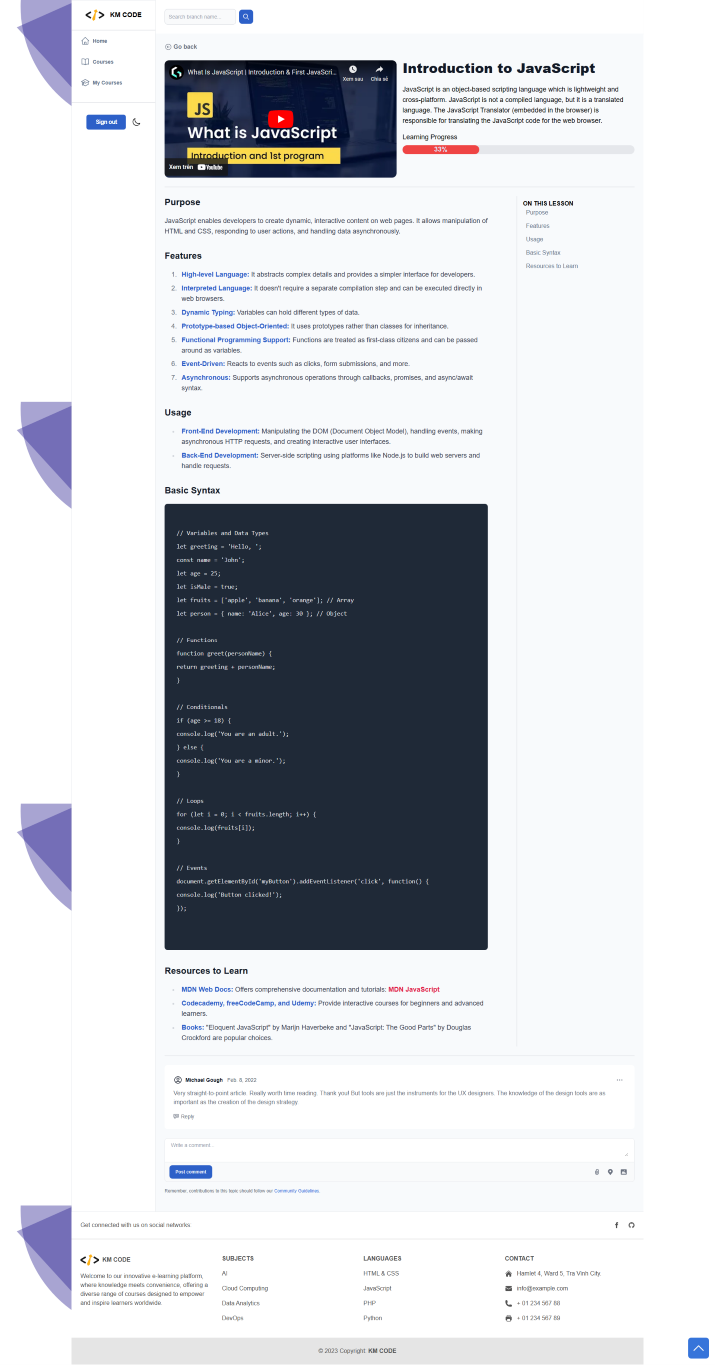
Hình 4.4 Giao diện chi tiết khóa học khi chưa tham gia.

Ở giao diện danh sách các khóa học theo người dùng, tôi sử dụng dữ liệu demo gồm tên đăng nhập là “billpham@gmail.com” và mật khẩu là “123”. Do phạm vi của đề tài này ở mức thiết kế front-end nên các dữ liệu này được tôi lưu trữ dưới dạng biếng Javascript.



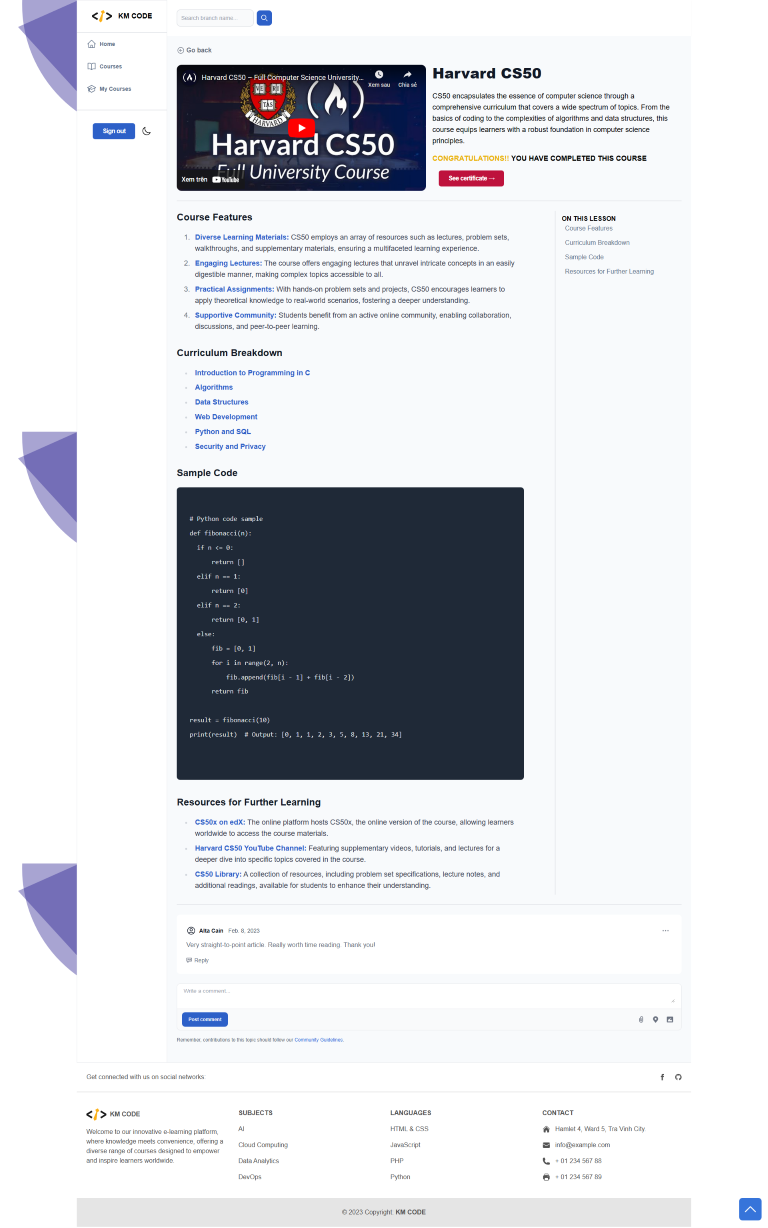
Hình 4.5 Giao diện các khóa học theo người dùng.

Giao diện chi tiết khóa học sau khi người dùng đã đăng nhập và tùy chọn tham gia khóa học:



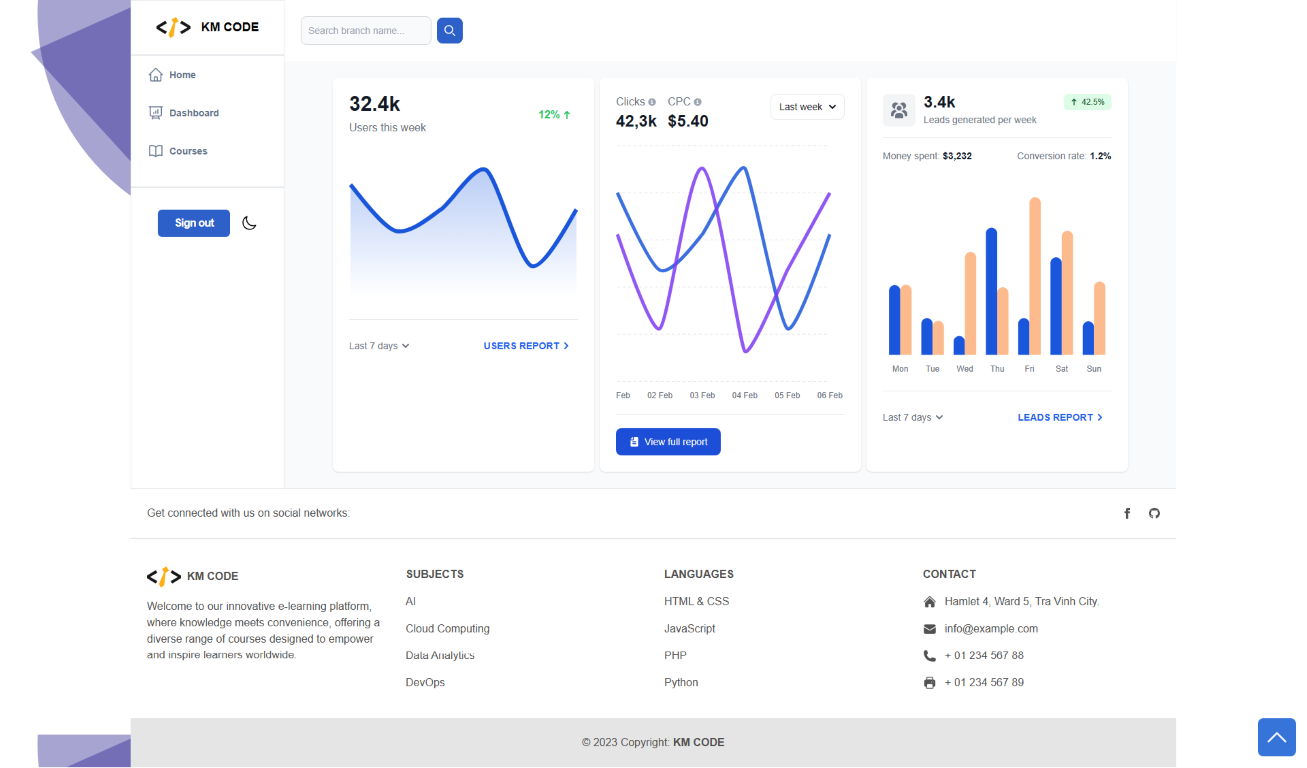
Hình 4.6 Giao diện chi tiết khóa học khi đã đăng ký khóa học.

Giao diện chi tiết khóa học sau khi người dùng đã hoàn thành khóa học:



Hình 4.7 Giao diện chi tiết khóa học khi đã hoàn thành khóa học.

## Giao diện trang quản trị



Hình 4.8 Giao diện dashboard cho trang quản trị.

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận

### Các vấn đề đã đạt được

**Nghiên cứu sâu về Tailwind CSS:** Đã tiến hành nghiên cứu kỹ về Tailwind CSS framework, bao gồm các class và tính năng nổi bật. So sánh và đánh giá được ưu điểm và nhược điểm của Tailwind CSS so với các phương pháp truyền thống và các framework khác.

**Xây dựng ứng dụng web minh họa:** Thành công trong việc áp dụng Tailwind CSS để xây dựng giao diện cho một ứng dụng web nền tảng học tập trực tuyến (e-learning platform). Ứng dụng có thể hoạt động và thể hiện được tính linh hoạt cũng như khả năng tùy chỉnh cao của Tailwind CSS.

**Giao diện đẹp:** Sử dụng Tailwind CSS để thiết kế giao diện trực quan, hấp dẫn với các thành phần và màu sắc tương thích.

**Dark mode và Responsive Design:** Thêm tính năng dark mode và đảm bảo trang web phản hồi mượt mà trên nhiều thiết bị và kích thước màn hình khác nhau.

**Tối ưu hóa trải nghiệm người dùng:** Tập trung vào việc cải thiện trải nghiệm người dùng thông qua việc sử dụng Tailwind CSS để tối ưu hóa cách hiển thị và tương tác trên trang web.

### Các vấn đề chưa làm được

**Thiếu cơ sở dữ liệu:** Trang web chỉ sử dụng dữ liệu dưới dạng biếng Javascript và chưa tích hợp cơ sở dữ liệu thực.

**Chưa có nhiều chức năng:** Trang web chưa phát triển nhiều tính năng, tập trung chủ yếu vào giao diện và trải nghiệm người dùng.

## Hướng phát triển

Tiếp tục nghiên cứu về Tailwind CSS và mở rộng phạm vi ứng dụng của nó trong các ngữ cảnh khác nhau để đem lại giá trị cao hơn trong việc phát triển ứng dụng web.

Nghiên cứu chi tiết về cách tích hợp cơ sở dữ liệu vào trang web, tăng cường tính tương tác và chức năng.

Phát triển và mở rộng các chức năng mới như hệ thống đăng nhập, quản lý người dùng, khóa học và bài giảng.

Phát triển các chức năng gợi ý các khóa học tương tự, các khóa học theo lộ trình, hoặc các khóa học theo chủ đề.

Phát triển các tính năng phản hồi của người dùng, tích hợp các bài tập, bài thi trắc nghiệm liên quan khi người dùng hoàn thành khóa học.

DANH MỤC TÀI LIỆU THAM KHẢO

1. Dillion Megida, "How to Get Started with TailwindCSS", 03/08/2020. [Trực tuyến]. URL: https://www.freecodecamp.org/news/get-started-with-tailwindcss. Truy cập [09/11/2023].
2. John Polacek, "Let’s Define Exactly What Atomic CSS is", 10/04/2017 (cập nhật 13/08/2021). [Trực tuyến]. URL: https://css-tricks.com/lets-define-exactly-atomic-css. Truy cập [09/11/2023].
3. Thierry Koblentz, "Challenging CSS Best Practices", 21/10/2013. [Trực tuyến]. URL: https://www.smashingmagazine.com/2013/10/challenging-css-best-practices-atomic-approach. Truy cập [09/11/2023].
4. "Utility-First Fundamentals", URL: https://tailwindcss.com/docs/utility-first. Truy cập [11/11/2023].
5. "Get started with Tailwind CSS", URL: https://tailwindcss.com/docs/installation. Truy cập [11/11/2023].
6. Soham De Roy, "What is Tailwind CSS? A Beginner's Guide", 12/09/2022. [Trực tuyến]. URL: https://www.freecodecamp.org/news/what-is-tailwind-css-a-beginners-guide. Truy cập [11/11/2023].
7. "Rapidly build modern websites without ever leaving your HTML", URL: https://tailwindcss.com. Truy cập [11/11/2023].
8. "Github: Tailwind Labs", URL: https://github.com/tailwindlabs/tailwindcss. Truy cập [11/11/2023].
9. "Github: Tailwind CSS v3.0.0-alpha.1",

URL: https://github.com/tailwindlabs/tailwindcss/releases/tag/v3.0.0-alpha.1. Truy cập [11/11/2023].

1. W3schools, "Javascript HTML DOM",

URL: https://www.w3schools.com/js/js\_htmldom.asp. Truy cập [11/11/2023].

1. "Asynchronous Javascript",

URL: https://developer.mozilla.org/en-US/docs/Learn/Javascript/Asynchronous. Truy cập [11/11/2023].