BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT HƯNG YÊN

BÀI TẬP LỚN

XÂY DỰNG WEBSITE BÁN HÀNG THỜI TRANG TRỰC TUYẾN CHO HỆ THỐNG CỬA HÀNG HEATHER

MÔN HỌC : **THIẾT KẾ WEBSITE CƠ BẢN**

SINH VIÊN : **NGUYỄN THU HƯƠNG – 10124163**

NGUYỄN MẠNH QUYỀN – 10124271

MÃ LỚP : **125243**

HƯỚNG DẪN: **TRẦN ĐỖ THU HÀ**

GV CHẤM THI 1 **GV CHẤM THI 2**

HƯNG YÊN – 2025

NHẬN XÉT

Nhận xét của giảng viên hướng dẫn:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

GIẢNG VIÊN HƯỚNG DẪN

(Ký và ghi rõ họ tên)

LỜI CAM ĐOAN

Em xin cam đoan bài tập lớn ”xây dựng website bán hàng thời trang trực tuyến cho hệ thống cửa hàng Heather” là kết quả thực hiện của bản thân em dưới sự hướng dẫn của cô Trần Đỗ Thu Hà

Những phần sử dụng tài liệu tham khảo trong bài tập lớn đã được nêu rõ trong phần tài liệu tham khảo. Các kết quả trình bày trong bài tập lớn và chương trình xây dựng được hoàn toàn là kết quả do bản thân em thực hiện.

Nếu vi phạm lời cam đoan này, em xin chịu hoàn toàn trách nhiệm trước khoa và nhà trường.

Hưng Yên, ngày … tháng … năm…..

SINH VIÊN

(Ký, ghi rõ họ tên)

LỜI CẢM ƠN

Để có thể hoàn thành bài tập lớn này, lời đầu tiên em xin phép gửi lời cảm ơn tới bộ môn Công nghệ phần mềm, Khoa Công nghệ thông tin – Trường Đại học Sư phạm Kỹ thuật Hưng Yên đã tạo điều kiện thuận lợi cho em thực hiện bài tập lớn môn học này.

Đặc biệt em xin chân thành cảm ơn cô Trần Đỗ Thu Hà đã rất tận tình hướng dẫn, chỉ bảo em trong suốt thời gian thực hiện bài tập lớn vừa qua.

Em cũng xin chân thành cảm ơn tất cả các thầy/cô trong trường đã tận tình giảng dạy, trang bị cho em những kiến thức cần thiết, quý báu để giúp em thực hiện được bài tập lớn này.

Mặc dù em đã có cố gắng, nhưng với trình độ còn hạn chế, trong quá trình thực hiện đề tài không tránh khỏi những thiếu sót. Em hi vọng sẽ nhận được những ý kiến nhận xét, góp ý của các thầy/cô về những kết quả triển khai trong bài tập lớn.

Em xin trân trọng cảm ơn!

MỤC LỤC

DANH MỤC CÁC THUẬT NGỮ

STT	Từ viết tắt	Cụm từ tiếng anh	Diễn giải
1	HTML	Hypertext Markup Language	Ngôn ngữ đánh dấu siêu văn bản
2	CSS	Cascading Style Sheets	Định dạng các phần tử
3	JS	Java Script	Thêm các chức năng cho trang web

DANH MỤC CÁC BẢNG

DANH MỤC CÁC HÌNH VẼ, ĐỒ THỊ

CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI

1.1 Lý do chọn đề tài

Trong thời đại số hóa mạnh mẽ, việc sở hữu một website bán hàng chuyên nghiệp không chỉ là xu hướng mà còn là yếu tố sống còn đối với các doanh nghiệp, đặc biệt trong lĩnh vực thời trang – ngành hàng luôn biến động theo xu hướng và thị hiếu của người tiêu dùng. Đề tài "xây dựng website bán hàng thời trang trực tuyến cho hệ thống cửa hàng Heather" được lựa chọn với mục tiêu nâng cao trải nghiệm mua sắm cho khách hàng, đồng thời đem lại giải pháp số hóa quy trình bán hàng, giúp cửa hàng tiếp cận khách hàng nhanh chóng, hiệu quả và linh hoạt hơn.

Tầm quan trọng và vai trò của đề tài được thể hiện qua sự bùng nổ của thị trường thương mại điện tử tại Việt Nam. Theo báo cáo, thị trường này đã đạt giá trị 13,7 tỷ USD vào năm 2021 và dự kiến sẽ tăng lên 32 tỷ USD vào năm 2025, với tốc độ tăng trưởng trung bình hàng năm từ 16% trong năm 2020 lên 30% trong giai đoạn 2021–2025 [1].

Tính cấp thiết của đề tài càng được khẳng định khi hành vi tiêu dùng đang chuyển dịch mạnh mẽ sang kênh trực tuyến. Sự phát triển nhanh chóng của thương mại điện tử tại Việt Nam được thúc đẩy bởi nhiều yếu tố, bao gồm sự gia tăng của tầng lớp trung lưu và sự bùng nổ của nền kinh tế internet, với mức tăng trưởng dự kiến lên tới 43 tỷ USD vào năm 2025. Các nền tảng mua sắm trực tuyến như TikTok Shop, Shopee, Tiki, Lazada và Sendo đã góp phần đáng kể vào sự gia tăng mua sắm trực tuyến, đặc biệt trong các lĩnh vực như quần áo, điện tử, thiết bị gia dụng và sản phẩm chăm sóc cá nhân [1].

Những bất cập và hạn chế của hệ thống cũ – như việc chỉ bán hàng trực tiếp tại cửa hàng, thiếu kênh tương tác trực tuyến, không có công cụ quản lý sản phẩm, đơn hàng hay phản hồi khách hàng – đã gây khó khăn trong việc mở rộng quy mô kinh doanh và theo kịp với thói quen tiêu dùng mới. Việc xây dựng một website không chỉ khắc phục những hạn chế trên mà còn mở ra cơ hội phát triển lâu dài, bền vững cho hệ thống cửa hàng Heather.

1.2 Mục tiêu của đề tài

1.2.1 Mục tiêu tổng quát

Mục tiêu tổng quát của đề tài là thiết kế và triển khai một hệ thống website thương mại điện tử bán quần áo thời trang cho hệ thống cửa hàng Heather, đáp ứng nhu cầu kinh doanh thực tế, đồng thời mang lại trải nghiệm mua sắm tiện lợi, trực quan và hiệu quả cho người dùng.

Hệ thống website được xây dựng với hai phân hệ chính: trang bán hàng cho khách hàng và trang quản trị cho người quản lý, giúp số hóa toàn bộ quy trình từ giới thiệu sản phẩm, mua hàng đến quản lý đơn hàng và tồn kho – góp phần nâng cao hiệu quả hoạt động kinh doanh trong lĩnh vực thời trang bán lẻ.

1.2.2 Mục tiêu cụ thể

Trên cơ sở mục tiêu tổng quát, đề tài hướng đến việc hoàn thiện các mục tiêu cụ thể sau:

1. Phân tích nhu cầu sử dụng thực tế của hệ thống bán hàng và quản lý sản phẩm tại cửa hàng Heather, từ đó xác định yêu cầu về chức năng và thiết kế của website.
2. Thiết kế giao diện người dùng (UI) cho các trang chức năng chính của hệ thống bán hàng, bao gồm:
    - Trang chủ (home page)
    - Trang danh mục sản phẩm
    - Trang chi tiết sản phẩm
    - Trang giỏ hàng
    - Trang đặt hàng (checkout)
3. Thiết kế phân hệ quản trị, gồm:
    - Trang quản trị chính (dashboard)
    - Trang quản lý sản phẩm
    - Trang quản lý đơn hàng

4. Xây dựng toàn bộ giao diện bằng HTML và CSS, đảm bảo tương thích đa nền tảng (responsive) và mang tính thẩm mỹ, hiện đại, phù hợp với ngành thời trang.
5. Sử dụng JavaScript để:
    - Kiểm tra tính hợp lệ của dữ liệu đầu vào (form validation).
    - Tăng tính tương tác cho người dùng (ví dụ: cập nhật giỏ hàng, hiển thị thông tin chi tiết).
    - Thực hiện các chức năng động theo yêu cầu của giảng viên.
6. Kiểm thử website về chức năng, hiệu năng và khả năng tương tác, đảm bảo hệ thống hoạt động ổn định, chính xác.

1.3 Giới hạn và phạm vi của đề tài

1.3.1 Đối tượng nghiên cứu

- Đối tượng nghiên cứu: Thiết kế và xây dựng website bán hàng thời trang trực tuyến cho hệ thống cửa hàng Heather.
- Khách thể nghiên cứu: Quản lý hệ thống, nhân viên bán hàng, khách hàng.

1.3.2 Phạm vi nghiên cứu

- Phạm vị không gian: Áp dụng cho hệ thống cửa hàng Heather hoạt động trong khu vực nội thành Hà Nội.
- Phạm vi nghiên cứu: Nghiên cứu và phát triển website đuợc thực hiện từ tháng 5 đến tháng 6, dựa trên số và phản hồi thực tế từ cửa hàng Heather trong năm 2024 – 2025
- Ý nghĩa: Cung cấp giải pháp công nghệ tiên tiến, tối ưu, tiết kiệm nhân lực và tăng cao hiệu suất công việc.

1.4 Nội dung thực hiện

Đề tài sẽ triển khai các nội dung công việc chính sau:

1. Khảo sát hệ thống bán hàng hiện tại của cửa hàng Heather để hiểu rõ quy trình, điểm mạnh, hạn chế và nhu cầu thực tế.

2. Phân tích yêu cầu hệ thống, bao gồm yêu cầu chức năng và phi chức năng cho cả người dùng và quản trị viên.
3. Thiết kế giao diện người dùng (UI) cho các trang chính:
    - Trang chủ
    - Trang danh mục sản phẩm
    - Trang chi tiết sản phẩm
    - Trang giỏ hàng
    - Trang đặt hàng
    - Trang quản trị (dashboard, quản lý sản phẩm, đơn hàng)
4. Lập trình website bằng HTML, CSS và JavaScript, đảm bảo:
    - Hiển thị sản phẩm và thông tin chi tiết
    - Chức năng giỏ hàng, đặt hàng
    - Trang quản trị đơn giản, dễ sử dụng
5. Tích hợp kiểm tra dữ liệu đầu vào bằng JavaScript, đảm bảo các biểu mẫu (forms) hoạt động chính xác và thân thiện.
6. Kiểm thử hệ thống để đánh giá độ ổn định, hiệu năng và trải nghiệm người dùng.

1.5 Phương pháp tiếp cận

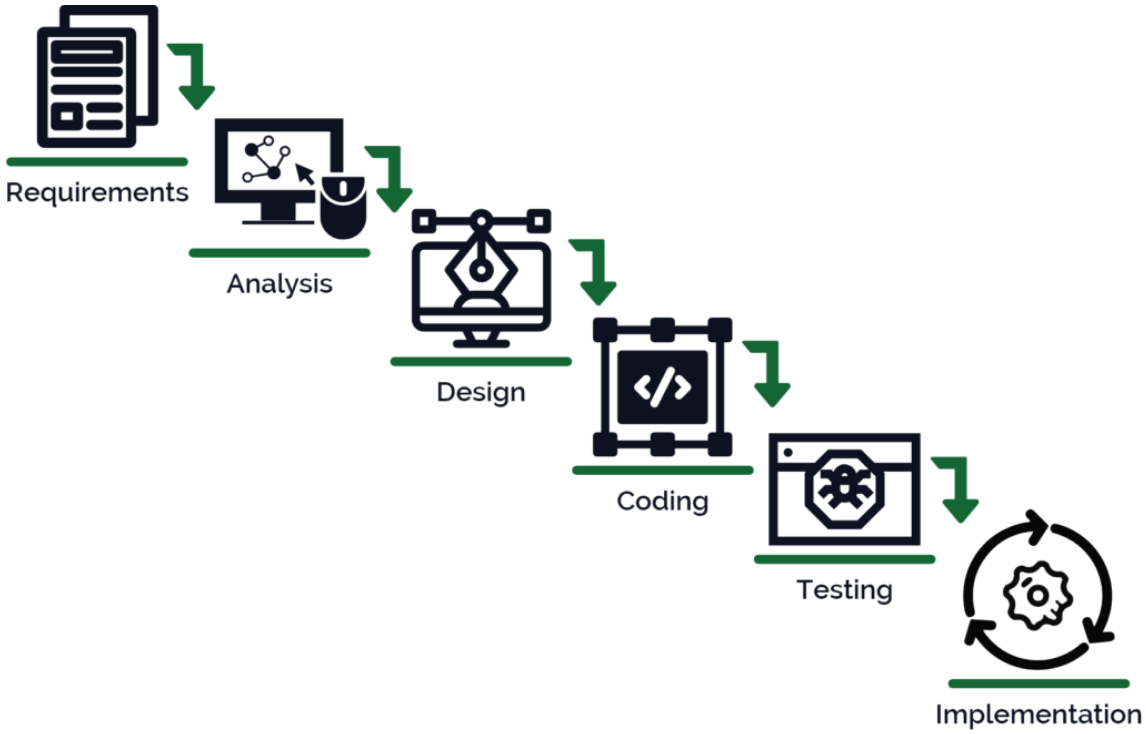
Đề tài sử dụng phương pháp tiếp cận trực tiếp với người sử dụng thông qua:

- Khảo sát thực tế tại cửa hàng Heather, tìm hiểu quy trình bán hàng hiện tại và các vấn đề thường gặp.
- Trao đổi với người quản lý và nhân viên bán hàng, ghi nhận yêu cầu và mong muốn về hệ thống mới.
- Đề xuất nhiều phương án thiết kế giao diện và chức năng, từ đó chọn lựa phương án phù hợp nhất dựa trên phản hồi từ phía cửa hàng.
- Phát triển và thử nghiệm hệ thống theo mô hình lặp (Iterative), liên tục lấy ý kiến phản hồi để cải tiến.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1 Quy trình phát triển phần mềm

Trong đề tài "Xây dựng website bán hàng thời trang trực tuyến cho hệ thống cửa hàng Heather", nhóm thực hiện áp dụng mô hình phát triển phần mềm thác nước (Waterfall Model). Đây là mô hình phát triển theo trình tự tuyến tính, mỗi giai đoạn cần được hoàn thành trước khi chuyển sang bước tiếp theo. Mô hình này phù hợp với các dự án có yêu cầu rõ ràng, ít thay đổi trong quá trình thực hiện – rất phù hợp với đề tài có tính học thuật và thực tiễn như hiện tại.



Hình 2.1: Mô Hình Waterfall

Quy trình gồm các bước chính như sau:

1. Khảo sát và thu thập yêu cầu
    - Tìm hiểu thực trạng kinh doanh, quy trình bán hàng của cửa hàng Heather.
    - Phỏng vấn người quản lý và khách hàng để xác định các yêu cầu cần thiết cho hệ thống website.
2. Phân tích yêu cầu

- Phân tích các chức năng chính mà hệ thống cần đáp ứng: quản lý sản phẩm, đơn hàng, giỏ hàng, đặt hàng, quản trị hệ thống,...
- Xác định các thành phần dữ liệu và luồng thông tin giữa người dùng – hệ thống – quản trị.

3. Thiết kế hệ thống
    - Thiết kế giao diện người dùng (UI) thân thiện, hiện đại.
    - Thiết kế kiến trúc tổng thể của website, bao gồm các trang chức năng chính cho cả người dùng và quản trị viên.
4. Lập trình (Xây dựng hệ thống)
    - Tiến hành viết mã bằng HTML, CSS và JavaScript.
    - Sử dụng JavaScript để xử lý các chức năng tương tác và kiểm tra tính hợp lệ của dữ liệu đầu vào.
5. Kiểm thử hệ thống
    - Thực hiện kiểm thử từng chức năng và toàn hệ thống để đảm bảo độ ổn định, chính xác, giao diện hiển thị tốt trên nhiều thiết bị.
6. Triển khai và đánh giá
    - Triển khai hệ thống demo để người dùng thử nghiệm.
    - Thu thập ý kiến phản hồi và đánh giá kết quả đạt được so với yêu cầu ban đầu.
7. Tài liệu hóa
    - Viết tài liệu hướng dẫn sử dụng và tài liệu kỹ thuật.
    - Hoàn tất báo cáo đề tài và chuẩn bị nội dung thuyết trình bảo vệ.

2.2 Thiết kế giao diện web với HTML, CSS

Hình 2.2: Ngôn ngữ HTML, CSS và Javascript

2.2.1 Cơ sở lý thuyết về HTML

HTML (HyperText Markup Language) là ngôn ngữ đánh dấu tiêu chuẩn để tạo ra các trang web. Nó mô tả cấu trúc của một trang web bằng cách sử dụng các thẻ (tags) để xác định các phần tử như tiêu đề, đoạn văn, hình ảnh, liên kết,... [2]. Một tài liệu HTML được định nghĩa bằng các thẻ (tag) nằm trong dấu ngoặc nhọn < >. Các thẻ này cung cấp thông tin cho trình duyệt về cách hiển thị và xử lý các phần tử trong trang web.

```
<!DOCTYPE html>
<html lang="vi">

<head>···
</head>

<body>···
</body>

</html>
```

Hình 2.3: Cơ sở lý thuyết cấu trúc HTML

Cấu trúc của một tài liệu HTML rất logic và đơn giản, tuân theo thứ tự từ trên xuống dưới, từ trái sang phải. Về cơ bản, một trang HTML được chia làm hai phần chính: phần đầu (head) và phần thân (body).

Phần <head> chứa các thông tin không trực tiếp hiển thị trên giao diện trình duyệt nhưng rất quan trọng như tiêu đề trang, định dạng ký tự, đường dẫn tới file CSS hoặc JavaScript, và các siêu dữ liệu khác.

Phần <body> là nơi chứa toàn bộ nội dung chính mà người dùng nhìn thấy trên trình duyệt, bao gồm văn bản, hình ảnh, liên kết, danh sách, bảng và các phần tử giao diện khác.

Mỗi tài liệu HTML đều cần khai báo <!DOCTYPE html> ngay từ dòng đầu tiên. Đây là khai báo bắt buộc nhằm xác định chuẩn văn bản mà trình duyệt cần tuân theo khi diễn giải mã HTML.

❖ **Một số lưu ý quan trọng:**

- Mỗi thẻ HTML nên được đóng đúng cách bằng thẻ đóng tương ứng để đảm bảo trình duyệt hiển thị chính xác và tránh lỗi không mong muốn.
- Một số thẻ HTML là thẻ tự đóng, không cần thẻ kết thúc, chẳng hạn như
, <img>, <input>, v.v.
- Khi sử dụng các thẻ lồng nhau, cần chú ý đóng thẻ theo đúng thứ tự đã mở để tránh lỗi cấu trúc.

- Việc viết mã HTML đúng chuẩn giúp trình duyệt hiển thị nội dung ổn định và hỗ trợ tốt hơn cho các công cụ SEO, truy cập hỗ trợ (accessibility), cũng như khả năng mở rộng với CSS và JavaScript.

❖ **Một số thẻ định dạng thường dùng:**

a) Thẻ định dạng văn bản (text)

Bảng 2.1: Một số thẻ định dnagj văn bản thường dùng

Thẻ	Chức năng
<b>	In đậm văn bản
	Đánh dấu văn bản quan trọng
<i>	In nghiêng văn bản
	Nhấn mạnh văn bản
<mark>	Đánh dấu văn bản
<small>	Hiển thị chữ nhỏ
<del>	Gạch bỏ văn bản
<ins>	Chèn thêm văn bản
<sub>	Hiển thị chỉ số dưới
<sup>	Hiển thị chỉ số trên

b) Thẻ định dạng bảng (table)

Bảng 2.2: Một số thẻ định dạng bảng

Thẻ	Chức năng
<table>	Khởi tạo bảng
<thead>	Phần tiêu đề bảng
<tbody>	Phần dữ liệu chính
<tfoot>	Phần tổng hợp, thống kê

<tr>	Dòng trong bảng
<th>	Ô tiêu đề trong bảng
<td>	Ô dữ liệu trong bảng

Lưu ý: Rằng về cú pháp, thẻ <th> và <td> được sử dụng tương tự nhau, tuy nhiên nên phân biệt rõ ràng về mặt ngữ nghĩa: <th> được dùng cho tiêu đề, còn <td> được dùng cho dữ liệu.

c) Thẻ định dạng danh sách

HTML hỗ trợ hai loại danh sách: danh sách có thứ tự và danh sách không có thứ tự. Danh sách có thứ tự sử dụng cặp thẻ <ol>..., trong khi danh sách không có thứ tự sử dụng cặp thẻ <ul>.... Mỗi mục trong danh sách được xác định bằng thẻ <li>.... Ngoài ra, một phần tử <li> có thể chứa một danh sách lồng bên trong.

d) Thẻ siêu liên kết

Thẻ siêu liên kết (<a>) cho phép điều hướng giữa các trang web hoặc các phần trong cùng một trang. Cú pháp cơ bản như sau:

Nội dung hiển thị

Các thuộc tính thường gặp của thẻ <a> bao gồm:

- href: xác định địa chỉ của liên kết.
- target: chỉ định nơi hiển thị trang liên kết, với các giá trị thông dụng:
- _self: mở trong tab hiện tại (mặc định).
- _blank: mở trong tab mới.
- _parent: mở trong frame cha.
- _top: mở trong khung lớn nhất (thường là toàn trang).

e) Thẻ xử lý khối và nội tuyến

Thẻ là phần tử khối, thường được sử dụng để nhóm các phần tử HTML lại với nhau và áp dụng các quy tắc định dạng chung bằng CSS. Thẻ này đóng vai trò lớn trong việc xây dựng bố cục trang web.

Ngược lại, thẻ là phần tử nội tuyến, dùng để nhóm các phần tử nhỏ (nội dung nằm trong một dòng) và cũng thường kết hợp với CSS để điều chỉnh định dạng.

f) Thể hiển hị hình ảnh

Trong HTML, thẻ <img> được sử dụng để hiển thị hình ảnh. Cú pháp cơ bản như sau:

<img src="link_ảnh" alt="nội_dung_thay_thế">

❖ **Các thuộc tính thường dùng bao gồm:**

- src: đường dẫn đến hình ảnh.
- alt: mô tả nội dung thay thế khi hình ảnh không thể hiển thị.
- width: độ rộng của ảnh (có thể là đơn vị px hoặc %).
- height: độ cao của ảnh (px hoặc %).

2.2.2 Cơ sở lý thuyết về CSS

CSS (Cascading Style Sheets) được sử dụng để kiểm soát cách trình bày của các phần tử HTML. Nó cho phép lập trình viên định nghĩa màu sắc, bố cục, phông chữ và cách giao diện hiển thị trên các loại thiết bị khác nhau [3].

❖ **Cú pháp và phương pháp áp dụng CSS:**

CSS có thể được áp dụng vào tài liệu HTML thông qua ba cách chính:

a) **Inline CSS:**

Là cách viết CSS trực tiếp trong thuộc tính style của từng thẻ HTML cụ thể. Cách này chỉ nên sử dụng khi cần định dạng nhanh một phần tử duy nhất và không áp dụng cho những tình huống cần tái sử dụng mã định dạng:

Ví dụ:

b) **Internal CSS:**

Được viết bên trong cặp thẻ <style> đặt trong phần <head> của tài liệu HTML. Phương pháp này cho phép định dạng toàn bộ trang web bằng một khối mã CSS nằm trong chính tài liệu HTML đó.

Ví dụ:

<head>

<style>

```
    body {
      font-family: Arial, sans-serif;
      background-color: #f4f4f4;
    }
  </style>
</head>
```

***c)* External CSS:**

Là phương pháp tối ưu và phổ biến nhất, cho phép người dùng tách mã CSS ra một tập tin riêng biệt với phần mở rộng .css. Sau đó, tập tin này sẽ được liên kết vào tài liệu HTML thông qua thẻ <link>. Việc tách riêng giúp tái sử dụng mã CSS cho nhiều trang và dễ dàng bảo trì hơn.

Ví dụ:

```
<head>
  <link rel="stylesheet" href="style.css">
</head>
```

❖ **Cấu trúc cơ bản của một đoạn CSS:**

Một khối mã CSS gồm ba thành phần chính:

```
vung-chon {
  thuoc-tinh-1: gia-tri-1;
  thuoc-tinh-2: gia-tri-2;
}
```

Giải thích các thành phần như sau:

- Vùng chọn (selector): Là thành phần xác định đối tượng HTML sẽ được áp dụng các thuộc tính CSS. Vùng chọn có thể là tên thẻ (ví dụ p, div), định danh (id) hoặc tên lớp (class), hay các dạng nâng cao như vùng chọn giả, thuộc tính, tổ hợp, v.v.

- Thuộc tính (property): Là tên của yếu tố muốn định dạng, ví dụ như color, font-size, margin, padding.
- Giá trị (value): Là giá trị gán cho thuộc tính, có thể là chuỗi văn bản, đơn vị đo lường (px, %, em, rem), mã màu (hex, rgb, rgba), từ khóa (auto, none), v.v.

Ví dụ:

```
h1 {
  color: blue;
  font-size: 24px;
}
```

Trong ví dụ trên, vùng chọn là h1, thuộc tính color có giá trị là blue, và thuộc tính font-size có giá trị là 24px.

❖ **Các loại bộ chọn (selectors) trong CSS:**

Bảng 2.3: Các loại bộ chọn trong CSS

Bộ chọn	Ví dụ	Mô tả các ví dụ	CSS
.class	.intro	Chọn tất cả các phần tử có class=”intro”	1
#id	#firstname	Chọn tất cả các phần tử có id=”firstname”	1
*	*	Chọn tất cả các phần tử	2
element	p	Chọn tất cả các phần tử <p>	1
element,element	div, p	Chọn tất cả các phần tử và phần tử <p>	1
element element	div p	Chọn tất cả các phần tử <p> và	1

		bên trong phần tử	
element>element	div > p	Chọn tất cả các phần tử <p> có phần tử cha là	2
element+element	div + p	Chọn tất cả các phần tử <p> được đặt phía sau phần tử	2
element1~element 2	p ~ ul	Chọn tất cả các phần tử được đặt trước bởi một phần tử <p>	3
[attribute]	[target]	Chọn tất cả các phần tử có cùng thuộc tính	2
[attribute=value]	[target=_blank]	Chọn tất cả các phần tử có thuộc tính bằng giá trị(target="_blank")	2
[attribute~=value]	[title~=flower]	Chọn tất cả các phần tử có tiêu đề của thuộc tính có chứa từ "flower"	2
[attribute|=value]	[lang|=en]	Chọn tất cả các phần tử có giá trị thuộc tính "lang" bắt đầu bằng "en"	2
[attribute^=value]	a[href^="https"]	Chọn tất cả các phần tử <a> có giá trị thuộc tính "href" bắt đầu bằng "https"	3

[attribute$=value]	a[href$=".pdf"]	Chọn tất cả các phần tử <a> có giá trị thuộc tính "href" kết thúc bằng".pdf"	3
[attribute*=value]	a[href*="timoday"]	Chọn tất cả các phần tử <a> có giá trị thuộc tính "href" chứa chuỗi"timoday"	3
:active	a:active	Chọn tất cả các liên kết được kích hoạt	1
::after	p::after	Chèn thêm nội dung ngay phía sau của các phần tử <p>	2
::before	p::before	Chèn thêm nội dung ngay phía trước của các phần tử <p>	2
:checked	input:checked	Chọn tất cả các phần tử <input> đang được chọn (selected)	3
:disabled	input:disabled	Chọn tất cả các phần tử <input> đang được vô hiệu hoá (disabled)	3
:empty	p:empty	Chọn tất cả các phần tử <p> không chứa phần tử con (bao gồm cả các nút văn bản)	3
:enabled	input:enabled	Chọn tất cả các phần tử <input> đang được kích hoạt	3

:last-child	p:last-child	Chọn tất cả các phần tử <p> là phần tử con cuối cùng của phần tử cha	3
:last-of-type	p:last-of-type	Chọn tất cả các phần tử <p> là thuộc tính cuối cùng của phần tử cha	3
:link	a:link	Chọn tất cả các liên kết khi chưa được click	1
:not(selector)	:not(p)	Chọn tất cả các phần tử không phải là một phần tử <p>	3
:nth-child(n)	p:nth-child(2)	Chọn tất cả các phần tử <p> là phần tử thứ hai của phần tử cha	3
:nth-last-child(n)	p:nth-last-child(2)	Chọn tất cả các phần tử <p> là phần tử con thứ hai của phần tử cha, tính từ phần tử con cuối cùng	3
:nth-last-of-type(n)	p:nth-last-of type(2)	Chọn tất cả các phần tử <p>là phần tử thuộc tính thứ hai của phần tử cha, tính từ phần tử thuộc tính con cuối cùng	3
:nth-of-type(n)	p:nth-of-type(2)	Chọn tất cả các phần tử <p> là phần tử thuộc tính con thứ hai của phần tử cha	3

:only-of-type	p:only-of-type	Chọn tất cả các phần tử <p> là thuộc tính duy nhất của phần tử cha	3
:only-child	p:only-child	Chọn tất cả các phần tử <p> là con duy nhất của phần tử cha	3
:optional	input:optional	Chọn tất cả các phần tử đầu vào không có thuộc tính "required"	3
:out-of-range	input:out-of-range	Chọn tất cả các phần tử đầu vào có giá trị ngoài một phạm vi nhất định	3
:read-only	input:read-only	Chọn tất cả các phần tử đầu vào có thuộc tính xác định "readonly"	3
:read-write	input:read-write	Chọn tất cả các phần tử đầu vào có thuộc tính không xác định "readonly"	3
:required	input:required	Chọn tất cả các phần tử đầu vào có thuộc tính "required" xác định	3
:root	:root	Chọn các phần tử gốc của văn bản	3
::selection	::selection	Chọn các phần tử được người dùng lựa chọn	

:target	#news:target	Chọn các phần tử đang hoạt động hiện tại (click trong các liên kết anchor name)	3
:valid	input:valid	Chọn tất cả các phần tử đầu vào có một giá trị hợp lệ	3
:visited	a:visited	Chọn tất cả các liên kết được truy cập	1

Trong dự án, nhóm sử dụng HTML để xây dựng khung sườn cho các trang như: trang chủ, danh mục sản phẩm, chi tiết sản phẩm, giỏ hàng, và quản trị. CSS được dùng để thiết kế giao diện theo xu hướng thời trang hiện đại, đồng thời áp dụng responsive design để website hiển thị tốt trên cả máy tính và thiết bị di động.

2.2.3 Cơ sở lý thuyết về Javascript

JavaScript là một ngôn ngữ lập trình thông dịch, chủ yếu được dùng để phát triển ứng dụng web (cả phía client và server với Node.js). Ngôn ngữ này được phát triển bởi Brendan Eich tại Netscape, ban đầu mang tên Mocha, sau đó đổi thành LiveScript, và cuối cùng là JavaScript.

JavaScript có cú pháp tương tự ngôn ngữ C, nhưng lại kế thừa ý tưởng từ Self. File JavaScript thường có phần mở rộng là .js.

Phiên bản chuẩn hóa hiện tại của JavaScript là ECMAScript 12. Theo W3Schools: "JavaScript is the programming language of the Web. It is used to make web pages dynamic and interactive by implementing custom client-side scripts." [3].

❖ **Các lệnh điều khiển**

a) **Câu lệnh if . . . else**

Câu lệnh này dùng để kiểm tra điều kiện, nó thực hiện việc tính toán trên một biểu thức. Nếu điều kiện là đúng (true) thì khối lệnh được thực thi.

```
if (condition) {
statements;
```

```
}
```

Ta cũng có thể chỉ ra khối lệnh cần thực hiện khi điều kiện là sai (false) bằng việc dùng mệnh đề else

```
if (condition)   {
    statements;
} else  {
statements2; }
```

Nếu điều kiện là sai khối lệnh sau else được thực thi.

***b)* Lệnh switch**

Khi ta có nhiều tùy chọn If...else thì tốt hơn nên sử dụng lệnh switch. Lệnh này còn được xem là lệnh case, lệnh switch thực thi một trong các khối lệnh tuỳ thuộc vào giá trị của biểu thức. Nếu không tìm thấy một giá trị nào trong danh sách các case của nó, khối lệnh trong phần default sẽ được thực hiện. Lệnh break dùng để thoát ra khỏi câu lệnh switch.

```
switch (expression) {
 case label:
 statement;
 break;
 case label:
 statement;
 break;
 ...
 default: statement;
}
```

***c)* Các lệnh vòng lặp**

Các cấu trúc điều khiển việc thực hiện lặp đi lặp lại trong một chương trình được gọi là vòng lặp. Có nhiều loại vòng lặp:

- Vòng lặp thực hiện lặp đi lặp lại các lệnh cho đến khi điều kiện là False
- Vòng lặp thực hiện lặp đi lặp lại các lệnh cho đến khi điều kiện là True
- Vòng lặp thực hiện lặp đi lặp lại các lệnh theo một số lần nhất định

Vòng lặp "for"

Vòng lặp for sẽ thực hiện lặp đi lặp lại khối lệnh cho đến khi điều kiện là false. Số lần thực hiện của vòng lặp thường được điều khiển thông qua một biến đếm.

Lệnh for bao gồm ba phần, cách nhau bởi dấu chấm phẩy. Cả ba phần đó đều không bắt buộc phải có, và chúng điều khiển việc thực hiện của vòng lặp for.

```
for (lệnh khởi tạo; điều kiện; lệnh tăng) {
statements;
}
```

Do . . . while

Vòng lặp Do...while được dùng để thực thi một khối lệnh cho đến khi điều kiên là false. Cú pháp là:

```
do {
statements;}
while (condition)
```

Lệnh while

Lệnh while là một cấu trúc lặp khác trong JavaScript. Nó được dùng để thực hiện một khối các câu lệnh chừng nào điều kiện là true. Nếu có nhiều câu lệnh thực hiện trong thân của vòng lặp chương trình phải sử dụng cặp dấu { và } để chứa các câu lệnh trong đó.

Khác biệt chính giữa vòng lặp while và do…while là các lệnh trong thân vòng lặp while có thể không được thực hiện một lần nào vì có thể ngay từ ban đầu điều kiện đã là false. Tuy nhiên vòng lặp do…while bao giờ cũng được thực hiện ít nhất một lần. Cú pháp là:

```
while (condition) {
statements;
```

}

Câu lệnh break & continue

Vòng lặp while loop và for sẽ kết thúc thực hiện khi điều kiện là false.Tuy nhiên ta cũng có thể kết thúc vòng lặp nếu muốn. Lệnh break dùng để kết thúc việc thực thi của một câu lênh. Khi được sử dụng trong một vòng lặp, lệnh break làm dừng ngay vòng lặp đó và không thực hiện thêm nữa.

Một lệnh đặc biệt khác cũng có thể được sử dụng trong vòng lặp là lệnh continue. Continue dừng ngay lần lặp hiện tại và quay lại kiểm tra điều kiện để thực hiện lần lặp tiếp theo.

1. for...in

Câu lệnh for . . in được dùng để duyệt các thuộc tính của một đối tượng hay các phần tử của một mảng. Ví dụ, chúng ta có thể muốn thực hiện một khối các câu lệnh cho mỗi phần tử của mảng. Chú pháp là:

for (variable in object) {

statements;

}

2. with

Câu lệnh "with" được dùng để thực thi một tập các lệnh cùng tham chiếu đến một đối tượng xác định. Đó là đối tượng được chỉ ra trong câu lệnh « with », Chú pháp:

with (object) {

statements;

}

❖ **Hàm và cách sử dụng hàm :**

Function hay còn gọi là hàm, là tập hợp một đoạn code dùng để xử lý một nhiệm vụ nào đó. Code bên trong function không được biên dịch cho tới khi được gọi đến. Chính vì vậy khi sử dụng function sẽ giúp chương trình được linh hoạt hơn. Cú pháp:

function name_of_function(var1, var2, var3, ...) {

// Some code

}

Trong đó:

- function: là từ khóa của javascript nên bắt buộc phải như vậy
- name_of_function: là tên của hàm, thông thường chúng ta tạo những tên có ý nghĩa như find_max, find_min, ...
- var1, var2, var3, ... là các tham số truyền vào hàm.

Để gọi hàm thì ta chỉ cần gọi đến tên hàm, sau đó truyền vào các tham số cần thiết. Hàm có return là hàm có sử dụng từ khóa **return** để đặt ở cuối hàm với mục đích trả kết quả về để sử dụng tiếp ở những đoạn code bên ngoài.

❖ **Một số đối tượng:**

***a)* Đối tượng DOM**

Trong một tài liệu HTML, tất cả các thẻ và nội dung đều được quản lý thông qua đối tượng đặc biệt có tên là document, thuộc về mô hình DOM (Document Object Model). DOM là cầu nối giữa JavaScript và trang HTML, cho phép lập trình viên truy xuất, thay đổi và tương tác với các thành phần của trang web.

Cấu trúc DOM được tổ chức dưới dạng một cây phân cấp, trong đó:

- Thẻ gốc cao nhất là <html>.
- Bên dưới thẻ <html> là hai nhánh chính: <head> và <body>.
- Phần <head> chứa các thẻ như <title>, <meta>, <style>, v.v.
- Phần <body> chứa nội dung hiển thị, bao gồm các phần tử như tiêu đề, đoạn văn, hình ảnh, liên kết, biểu mẫu,...

Từ cấu trúc này, có thể hiểu rằng để tương tác hoặc thao tác với các thẻ HTML bằng JavaScript, bắt buộc phải thông qua đối tượng document, tức là làm việc với DOM. Đây là nền tảng quan trọng giúp JavaScript tạo ra trang HTML động.

Thông qua DOM, JavaScript cung cấp đầy đủ sức mạnh để tương tác với trang web:

- Thay đổi nội dung hoặc thuộc tính của bất kỳ phần tử HTML nào trên trang.
- Thay đổi phong cách CSS của các phần tử, ví dụ như màu sắc, kích thước, bố cục,...
- Thêm mới các phần tử HTML hoặc thuộc tính vào tài liệu hiện có.

- Loại bỏ các phần tử hoặc thuộc tính không còn cần thiết.
- Gắn các xử lý sự kiện (event handler) để phản ứng với hành vi của người dùng như nhấp chuột, nhập liệu, di chuyển chuột,...
- Khởi tạo và phát sinh các sự kiện HTML mới thông qua JavaScript để mô phỏng tương tác người dùng hoặc thực hiện hành động tự động.

Về bản chất, Document Object Model (DOM) là một giao diện lập trình ứng dụng (API) cho các tài liệu HTML và XML. DOM mô hình hóa tài liệu như một cây dữ liệu, trong đó:

- Mỗi phần tử, thuộc tính và nội dung văn bản được biểu diễn bằng các node (nút).
- Các node được liên kết theo quan hệ cha – con, phản ánh cấu trúc lồng nhau của mã HTML.

DOM độc lập với hệ điều hành và được xây dựng dựa trên nguyên lý của lập trình hướng đối tượng, nhằm đảm bảo khả năng tương tác linh hoạt và nhất quán.

Trong giai đoạn đầu của sự phát triển web, mỗi trình duyệt triển khai DOM theo cách riêng, dẫn đến sự không đồng nhất và khó khăn trong phát triển ứng dụng web. Chính vì vậy, tổ chức World Wide Web Consortium (W3C) đã đưa ra các tiêu chuẩn cho DOM, nhằm thống nhất mô hình và đảm bảo khả năng tương thích giữa các trình duyệt.

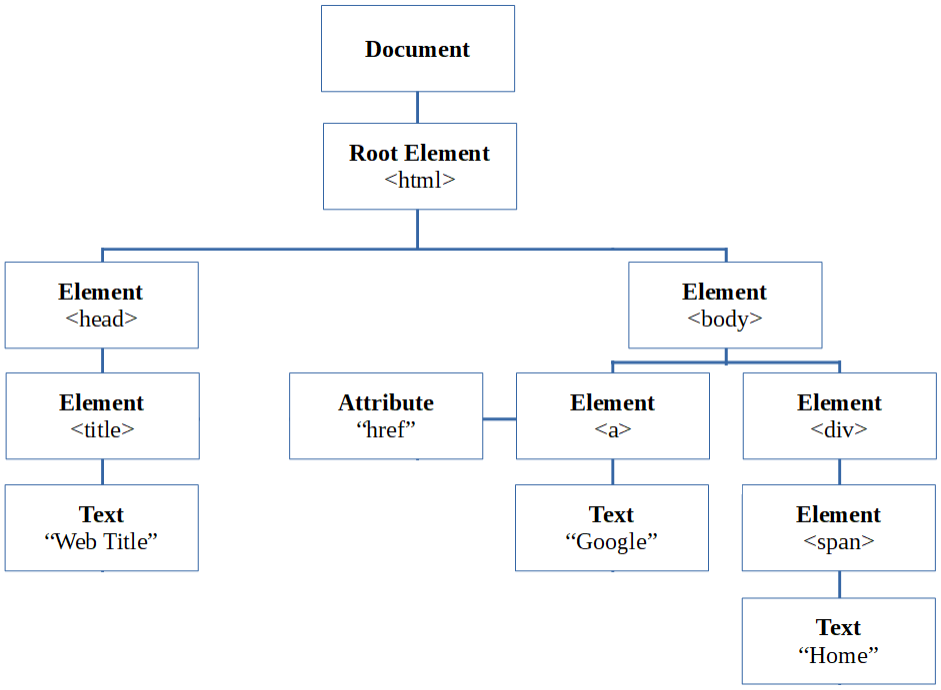
Hiểu rõ về DOM là điều kiện tiên quyết để xây dựng các ứng dụng web hiện đại, tương tác cao và linh hoạt.

b) Đối tượng JavaScript

Javascript có 5 kiểu dữ liệu Number, String, Boolean, Undefined và Null và còn 1 kiểu khác nữa đó là Object (kiểu dữ liệu phức hợp). Kiểu Object là kiểu được sử dụng nhiều nhất vì tính linh hoạt cực kỳ mạnh mẽ của nó trong việc xử lý dữ liệu

- **Document:**

Đối tượng document là thuộc tính của đối tượng window, vì vậy nó có thể được truy cập bằng window.document:



Hình 2.4: Các thuộc tính đuối tượng của Document

Chúng ta có thể truy cập và thay đổi nội dung của trang web bằng các phương thức của đối tượng document. Các phương thức quan trọng của đối tượng document như sau:

Bảng 2.4: Các phương thức quản trọng của đối tượng Document

Phương thức	Mô tả
write("string")	Viết chuỗi đã cho trên document.
writeln("string")	Viết chuỗi đã cho trên document với ký tự newline ở cuối.
getElementById()	Trả về phần tử có giá trị id đã cho.
getElementsByName()	Trả về tất cả các phần tử có giá trị name đã cho.
getElementsByTagName()	Trả về tất cả các phần tử có tên thẻ đã cho.

getElementsByClassName()	Trả về tất cả các phần tử có class đã cho.

Math:

Đối tượng Math là định nghĩa sẵn trong JS, nó chứa các thuộc tính và phương thức cho phép thi hành một số tác vụ về toán học. Đối tượng Math luôn có sẵn để sử dụng ngay mà không cần tạo mới. VD:

```
document.write(Math.PI);  //In ra 3.141592653589793
```

String:

Đối tượng **String** giúp bạn làm việc với một dãy các ký tự; nó giúp xử lý các kiểu dữ liệu chuỗi gốc trong JavaScript với một số phương thức giúp đỡ. Sử dụng cú pháp sau để tạo một đối tượng String:

```
var val = new String(string);
```

Array:

Bảng 2.5: Các phương thức, thuộc tính của Array

Phương thức/Thuộc tính	**Mô tả ngắn gọn**
concat()	Nối hai hoặc nhiều mảng lại với nhau.
includes()	Kiểm tra xem mảng có chứa giá trị chỉ định không.
copyWithin()	Sao chép một phần của mảng đến vị trí khác trong cùng mảng.
indexOf()	Trả về chỉ số đầu tiên của phần tử trong mảng.
entries()	Trả về một iterator gồm các cặp [index, value].
isArray()	Kiểm tra xem đối tượng có phải là mảng không.

every()	Kiểm tra tất cả phần tử có thỏa mãn điều kiện hay không.
fill()	Điền giá trị vào tất cả phần tử trong mảng.
keys()	Trả về một iterator với các chỉ số (keys) trong mảng.
filter()	Lọc các phần tử thỏa mãn điều kiện.
length	Độ dài của mảng.
find()	Trả về phần tử đầu tiên thỏa mãn điều kiện.
lastIndexOf()	Trả về chỉ số cuối cùng của phần tử trong mảng.
findIndex()	Trả về chỉ số đầu tiên của phần tử thỏa điều kiện.
map()	Tạo mảng mới bằng cách áp dụng hàm lên từng phần tử.
forEach()	Duyệt từng phần tử của mảng và thực thi hàm.
pop()	Loại bỏ phần tử cuối cùng khỏi mảng.
from()	Tạo mảng mới từ object hoặc iterable.
push()	Thêm một hoặc nhiều phần tử vào cuối mảng.

Thông qua JavaScript, trang web trở nên linh hoạt, thân thiện và đáp ứng được kỳ vọng về trải nghiệm người dùng trong lĩnh vực thương mại điện tử.

2.3 Lập trình phía front-end

Front End (còn được biết đến như client-side) là tất cả những gì liên quan đến những gì mà dùng nhìn thấy mỗi khi truy cập vào một trang web, bao gồm phạm trù thiết kế và các ngôn ngữ như HTML hay CSS.

Trong design software, front-end là một phần của hệ thống phần mềm, tương tác trực tiếp với người sử dụng. Cụ thể, đó là nền móng các giao diện user (GUI) và lập trình phía user.

Để trở thành một lập trình viên Front End thì 3 ngôn ngữ HTML, CSS, JavaScript là điều bắt buộc không thể không học. Ngoài ra lập trình viên cũng sẽ phải bổ sung các kiến thức khác liên quan:

- Bootstrap: Đây là Framework được viết bằng CSS và JavaScript. Ưu điểm là giúp xây dựng trang web nhanh chóng, tiết kiệm thời gian cho lập trình viên.
- Jquery: Thư viện được phát triển bởi JavaScript, có rất nhiều hiệu ứng sử dụng JQuery, vì vậy bạn cũng cần tìm hiểu qua nó.
- UX/UI: Tối ưu cho trải nghiệm người dùng cũng khá quan trọng. Với Frontend Developer cũng cần tìm hiểu về thiết kế giao diện làm thế nào để tăng trải nghiệm cho người dùng ghé thăm website.

CHƯƠNG 3: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

3.1 Phát biểu bài toán

Hiện nay, việc mua sắm trực tuyến ngày càng phổ biến nhờ tính tiện lợi, nhanh chóng và khả năng tiếp cận rộng rãi. Tuy nhiên, hệ thống cửa hàng Heather. một thương hiệu thời trang đang phát triển, vẫn chưa có nền tảng bán hàng trực tuyến chính thức, dẫn đến một số hạn chế như:

- Khách hàng không thể dễ dàng xem sản phẩm, thông tin giá cả, tình trạng hàng hóa từ xa.
- Việc đặt hàng chủ yếu diễn ra thủ công thông qua mạng xã hội hoặc tin nhắn, gây mất thời gian và thiếu tính chuyên nghiệp.
- Quản lý đơn hàng, sản phẩm và khách hàng còn thủ công, không có sự đồng bộ giữa các bộ phận.

***a)* Bài toán đặt ra:**

Xây dựng một website bán hàng thời trang trực tuyến hiện đại, trực quan và dễ sử dụng cho hệ thống cửa hàng Heather. Website này cần đảm bảo cung cấp đầy đủ chức năng dành cho người dùng và quản trị viên, cụ thể:

- Đối với người dùng (khách hàng):
    - Dễ dàng xem danh mục và chi tiết sản phẩm
    - Cho phép thêm vào giỏ hàng và đặt hàng trực tuyến
    - Giao diện thân thiện, dễ sử dụng trên mọi thiết bị
- Đối với người quản trị:
    - Quản lý sản phẩm, đơn hàng, trạng thái đặt hàng
    - Giao diện quản trị đơn giản, dễ thao tác
    - Có thể kiểm tra đơn hàng và cập nhật trạng thái theo thời gian thực

***b)* Mục tiêu cuối cùng:**

- thiết kế và phát triển hệ thống website có thể đưa vào ứng dụng thực tế, hỗ trợ tối ưu hóa hoạt động bán hàng và mở rộng khả năng tiếp cận khách hàng trên nền tảng số cho cửa hàng Heather.

3.2 Đặc tả yêu cầu phần mềm

3.2.1 Các yêu cầu chức năng

❖ **Chức năng của phân hệ quản trị nội dung (nếu có)**

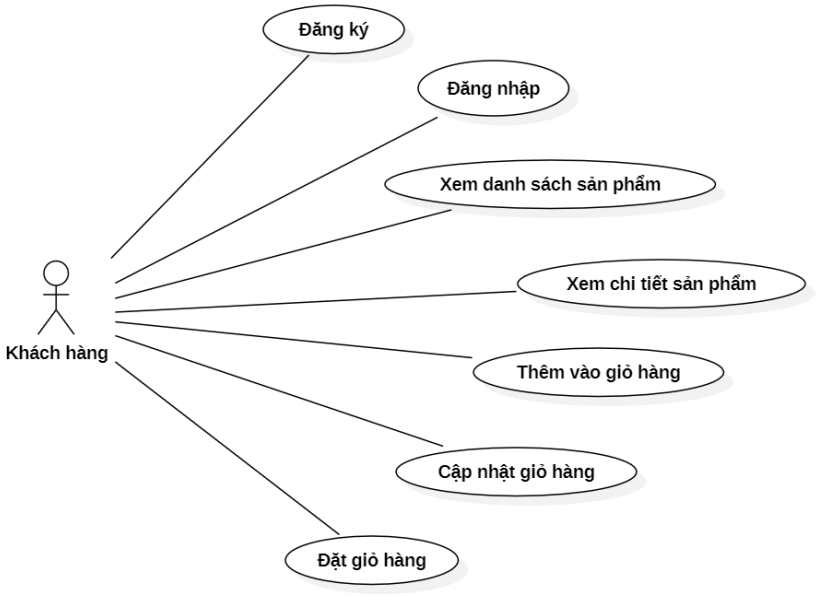
Phân hệ quản trị nội dung là khu vực dành riêng cho người quản trị hệ thống (admin), dùng để theo dõi và điều hành các hoạt động kinh doanh trên website. Các chức năng chính gồm:

Chức năng chính của phân hệ quản trị:

1. Đăng nhập quản trị viên: Xác thực người dùng là quản trị viên mới được truy cập hệ thống quản lý.
2. Quản lý sản phẩm
    - Thêm mới sản phẩm
    - Sửa thông tin sản phẩm
    - Xoá sản phẩm
    - Xem danh sách sản phẩm theo danh mục
3. Quản lý đơn hàng
    - Xem danh sách đơn hàng
    - Cập nhật trạng thái đơn hàng (đang xử lý, đã giao, đã hủy)
    - Xem chi tiết thông tin khách hàng và đơn hàng
4. Quản lý danh mục sản phẩm:
    - Tạo/sửa/xóa các danh mục như "Áo", "Quần", "Váy",...
5. Đăng xuất:
    - Kết thúc phiên làm việc, đảm bảo an toàn dữ liệu.

❖ **Chức năng của phân hệ người dùng**

a) Biểu đồ tổng quan ca sử dụng chính



Hình 3.1: Biểu đồ tổng quan ca sử dụng chính

b) Đặc tả chi tiết ca sử dụng

Bảng 3.1: Usecase đăng ký

Tên chức năng	Mô tả chức năng	Luồng sự kiện
Đăng ký	Cho phép người dùng tọa tài khoản mới trên hệ thống.	1. Người dùng chọn "Đăng ký". 2. Nhập thông tin: Họ tên, email, mật khẩu,... 3. Nhấn nút "Đăng ký". 4. Hệ thống kiểm tra dữ liệu (email trùng, định dạng sai, v.v.). 5. Nếu hợp lệ, hệ thống lưu thông tin vào CSDL và thông báo đăng ký thành công. 6. Điều hướng người

		dùng đến trang đăng nhập hoặc tự đăng nhập.

Bảng 3.2: Usecase đăng nhập

Tên chức năng	Mô tả chức năng	Luồng sự kiện
Đăng nhập	Cho phép người dùng đăng nhập vào hệ thống để mua hàng và quản lý tài khoản.	1. Người dùng vào trang "Đăng nhập". 2. Nhập email và mật khẩu. 3. Nhấn nút "Đăng nhập". 4. Hệ thống kiểm tra thông tin. 5. Nếu đúng, đăng nhập thành công và điều hướng về trang chủ hoặc trang tài khoản. 6. Nếu sai, hiển thị thông báo lỗi.

Bảng 3.3: Usecase xem danh sách sản phẩm

Tên chức năng	Mô tả chức năng	Luồng sự kiện
Xem danh mục sản phẩm	Hiển thị các danh mục sản phẩm theo loại: Quần áo, giày, trang sức, v.v.	1. Người dùng truy cập trang danh mục hoặc click từ menu. 2. Hệ thống truy vấn, lấy danh sách sản phẩm theo danh mục. 3. Hiển thị danh sách sản

		phẩm kèm hình ảnh, tên, giá,...

Bảng 3.4: Usecase xem chi tiết sản phẩm

Tên chức năng	Mô tả chức năng	Luồng sự kiện
Xem chi tiết sản phẩm	Hiển thị thông tin chi tiết của một sản phẩm	1. Người dùng click vào sản phẩm trong danh mục. 2. Hệ thống lấy thông tin chi tiết của sản phẩm. 3. Hiển thị chi tiết sản phẩm: hình ảnh lớn, mô tả, giá, size, màu,... 4. Người dùng có thể chọn số lượng, thêm vào giỏ.

Bảng 3.5: Usecase thêm sản phẩm vào giỏ hàng

Tên chức năng	Mô tả chức năng	Luồng sự kiện
Thêm sản phẩm vào giỏ hàng	Cho phép người dùng lưu sản phẩm muốn mua vào giỏ hàng tạm thời.	1. Tại trang chi tiết sản phẩm, người dùng chọn số lượng và nhấn "Thêm vào giỏ". 2. Hệ thống kiểm tra thông tin sản phẩm và lưu tạm vào giỏ hàng (session/localStorage hoặc DB). 3. Hiển thị thông báo hoặc chuyển hướng đến giỏ hàng.

Bảng 3.6: Usecase cập nhật giỏ hàng

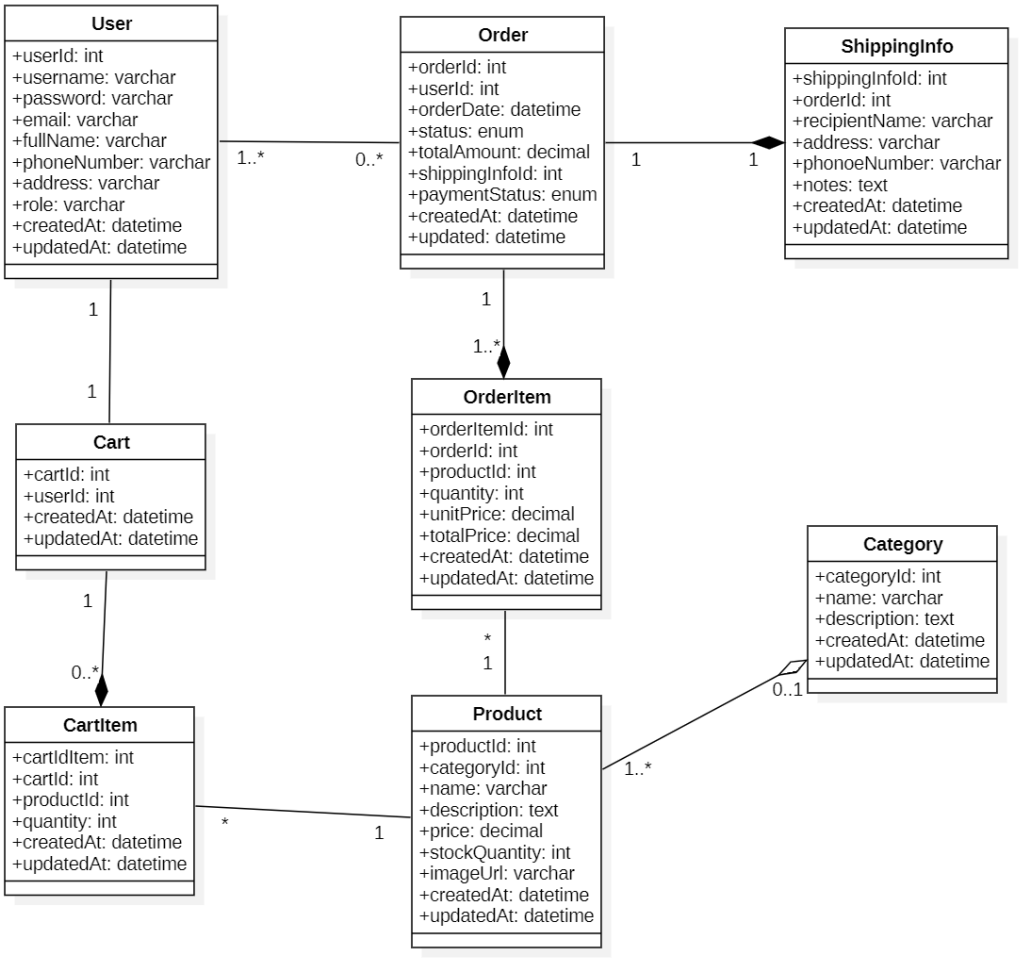
Tên chức năng	Mô tả chức năng	Luồng sự kiện
Cập nhật giỏ hàng	Người dùng có thể thay đổi số lượng sảnp hẩm hoặc xóa snar phẩm khỏi giỏ hàng	1. Người dùng truy cập trang "Giỏ hàng". 2. Tại đây có thể: thay đổi số lượng, xóa sản phẩm. 3. Nhấn "Cập nhật giỏ" hoặc hệ thống tự động tính lại tổng tiền. 4. Hệ thống cập nhật lại thông tin trong giỏ hàng.

Bảng 3.7: Usecase đặt hàng

Tên chức năng	Mô tả chức năng	Luồng sự kiện
Đặt hàng	Cho phép người dùng xác nhận đơn hàng và thanh toán	1. Người dùng vào giỏ hàng → nhấn "Tiến hành đặt hàng" 2. Nhập thông tin giao hàng (nếu chưa có) 3. Nhấn "Xác nhận đặt hàng" 4. Hiển thị thông báo đặt hàng thành công

3.2.2 Biểu đồ lớp thực thể

❖ **Biểu đồ lớp thực thể**



Hình 3.2: Biểu đồ lớp thực thể

❖ **Mô tả chi tiết lớp thực thể**

Bảng 3.8: Thực thể user

Thuộc tính	Kiểu dữ liệu	Mô tả
userId	INT / UUID	Khóa chính, định danh người dùng
username	VARCHAR	Tên đăng nhập
password	VARCHAR	Mật khẩu (đã mã hóa)
email	VARCHAR	Email người dùng
fullName	VARCHAR	Họ và tên
phoneNumber	VARCHAR	Số điện thoại

address	VARCHAR	Địa chỉ (mặc định)
role	ENUM('user','admin')	Phân quyền (khách, quản trị)
createdAt	DATETIME	Ngày tạo tài khoản
updatedAt	DATETIME	Ngày cập nhật thông tin

Bảng 3.9: Thực thể product

Thuộc tính	Kiểu dữ liệu	Mô tả
productId	INT / UUID	Khóa chính, định danh sản phẩm
categoryId	INT / UUID	Khóa ngoại, liên kết Category
name	VARCHAR	Tên sản phẩm
description	TEXT	Mô tả chi tiết sản phẩm
imageUrls	TEXT/ JSON	Đường dẫn hình ảnh sản phẩm
createdAt	DATETIME	Ngày tạo sản phẩm
updatedAt	DATETIME	Ngày cập nhật

Bảng 3.10: Thực thể productVariant

Thuộc tính	Kiểu dữ liệu	Mô tả
variantId	UUID / INT	Khóa chính
productId	UUID / INT	FK – Liên kết bảng Product
color	VARCHAR	Màu sắc (VD: red, blue)

size	VARCHAR	Kích cỡ (VD: S, M, L, XL)
price	DECIMAL(10,2)	Giá bán
discountPrice	DECIMAL(10,2)	Giá khuyến mãi (nếu có)
stockQuantity	INT	Số lượng tồn kho của biến thể này
imageUrl	VARCHAR	Ảnh đại diện riêng cho biến thể (nếu có)
sku	VARCHAR	Mã SKU riêng cho biến thể

Bảng 3.11: Thực thể category

Thuộc tính	Kiểu dữ liệu	Mô tả
categoryId	INT / UUID	Khóa chính, định danh danh mục
name	VARCHAR	Tên danh mục
description	TEXT	Mô tả ngắn về danh mục
createdAt	DATETIME	Ngày tạo danh mục
updatedAt	DATETIME	Ngày cập nhật

Bảng 3.12: Thực thể subcategory

Thuộc tính	Kiểu dữ liệu	Mô tả
subcategoryId	INT / UUID	Khóa chính, định danh danh mục
categoryId	INT/UUID (FK)	Khóa ngoại liên kết tới Category

name	VARCHAR	Tên danh mục
description	TEXT	Mô tả ngắn về danh mục
createdAt	DATETIME	Ngày tạo danh mục
updatedAt	DATETIME	Ngày cập nhật

Bảng 3.13: Thực thể cart

Thuộc tính	Kiểu dữ liệu	Mô tả
cartId	INT / UUID	Khóa chính, định danh giỏ hàng
userId	INT / UUID	Khóa ngoại, liên kết User
createdAt	DATETIME	Ngày tạo giỏ hàng
updatedAt	DATETIME	Ngày cập nhật

Bảng 3.14: Thực thể cartItem

Thuộc tính	**Kiểu dữ liệu**	**Mô tả**
cartItemId	INT / UUID	Khóa chính
cartId	INT / UUID	Khóa ngoại, liên kết Cart
productId	INT / UUID	Khóa ngoại, liên kết Product
quantity	INT	Số lượng sản phẩm trong giỏ
createdAt	DATETIME	Ngày thêm vào giỏ
updatedAt	DATETIME	Ngày cập nhật số lượng

Hình 3.2.2.7: Thực thể order

Bảng 3.15: Thực thể order

Thuộc tính	Kiểu dữ liệu	Mô tả
orderId	INT / UUID	Khóa chính
userId	INT / UUID	Khóa ngoại, liên kết User
orderDate	DATETIME	Ngày đặt hàng
status	ENUM	Trạng thái đơn hàng (Pending, Confirmed, Shipped, Delivered, Cancelled)
totalAmount	DECIMAL(10,2)	Tổng giá trị đơn hàng
shippingInfoId	INT / UUID	Khóa ngoại, liên kết ShippingInfo (nếu tách riêng)
paymentStatus	ENUM('Pending','Paid','Failed')	Trạng thái thanh toán
createdAt	DATETIME	Ngày tạo đơn
updatedAt	DATETIME	Ngày cập nhật

Bảng 3.16: Thực thể orderItem

Thuộc tính	**Kiểu dữ liệu**	**Mô tả**
orderItemId	INT / UUID	Khóa chính
orderId	INT / UUID	Khóa ngoại, liên kết Order
productId	INT / UUID	Khóa ngoại, liên kết Product
quantity	INT	Số lượng sản phẩm trong

		đơn
unitPrice	DECIMAL(10,2)	Giá sản phẩm tại thời điểm đặt
totalPrice	DECIMAL(10,2)	Tổng giá (unitPrice * quantity)
createdAt	DATETIME	Ngày tạo
updatedAt	DATETIME	Ngày cập nhật

Bảng 3.17: Thực thể shoppingInfo

Thuộc tính	Kiểu dữ liệu	Mô tả
shippingInfoId	INT / UUID	Khóa chính
orderId	INT / UUID	Khóa ngoại, liên kết Order
recipientName	VARCHAR	Tên người nhận
address	VARCHAR	Địa chỉ giao hàng
phoneNumber	VARCHAR	Số điện thoại người nhận
notes	TEXT	Ghi chú thêm
createdAt	DATETIME	Ngày tạo
updatedAt	DATETIME	Ngày cập nhật

3.2.3 Các yêu cầu phi chức năng

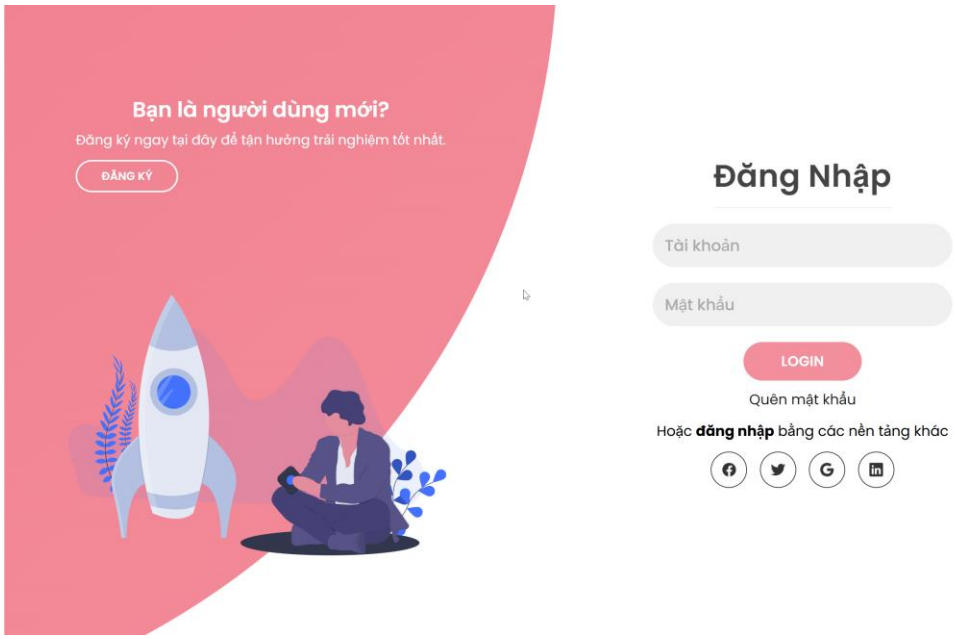
Bảng 3.18: Các yêu cầu phi chức năng

STT	Yêu cầu phi chức năng	Mô tả
1	Hiệu năng	Trang web phải tải trang chính trong vòng dưới 3 giây trên kết nối Internet phổ biến.

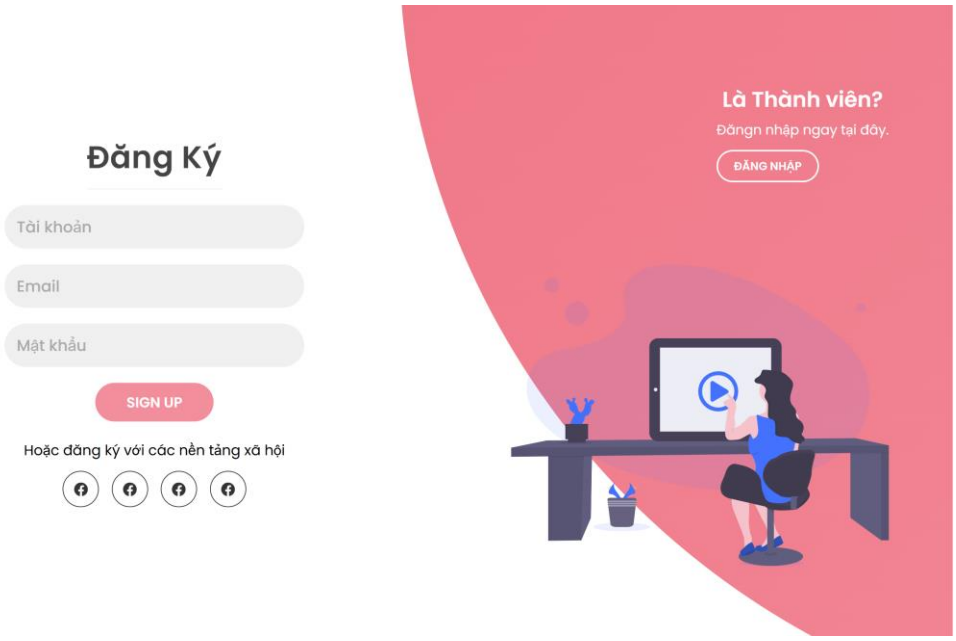
2	Tính sẵn sàng	Hệ thống hoạt động 99.9% thời gian trong ngày, đảm bảo truy cập liên tục cho người dùng.
3	Khả năng mở rộng	Hệ thống có thể mở rộng để xử lý tăng số lượng người dùng và đơn hàng trong tương lai.
4	Khả năng tương thích	Website hiển thị tốt và hoạt động trên các trình duyệt phổ biến (Chrome, Firefox, Safari, Edge).
5	Thân thiện thiết bị di động	Giao diện thích ứng tốt với các kích thước màn hình, từ điện thoại, máy tính bảng đến máy tính để bàn.
6	Khả năng bảo trì	Mã nguồn được tổ chức rõ ràng, dễ hiểu để thuận tiện sửa lỗi và nâng cấp trong tương lai.
7	Trải nghiệm người dùng	Giao diện đơn giản, dễ dùng, giúp người dùng dễ dàng tìm kiếm, mua hàng và quản lý tài khoản.
8	Sao lưu dữ liệu	Hệ thống tự động sao lưu dữ liệu định kỳ và có kế hoạch phục hồi khi xảy ra sự cố.
9	Tuân thủ luật pháp	Tuân thủ các quy định về bảo vệ dữ liệu cá nhân, bản quyền hình ảnh và các luật thương mại điện tử.

3.3 Thiết kế giao diện

3.3.1 Đăng nhập & đăng ký

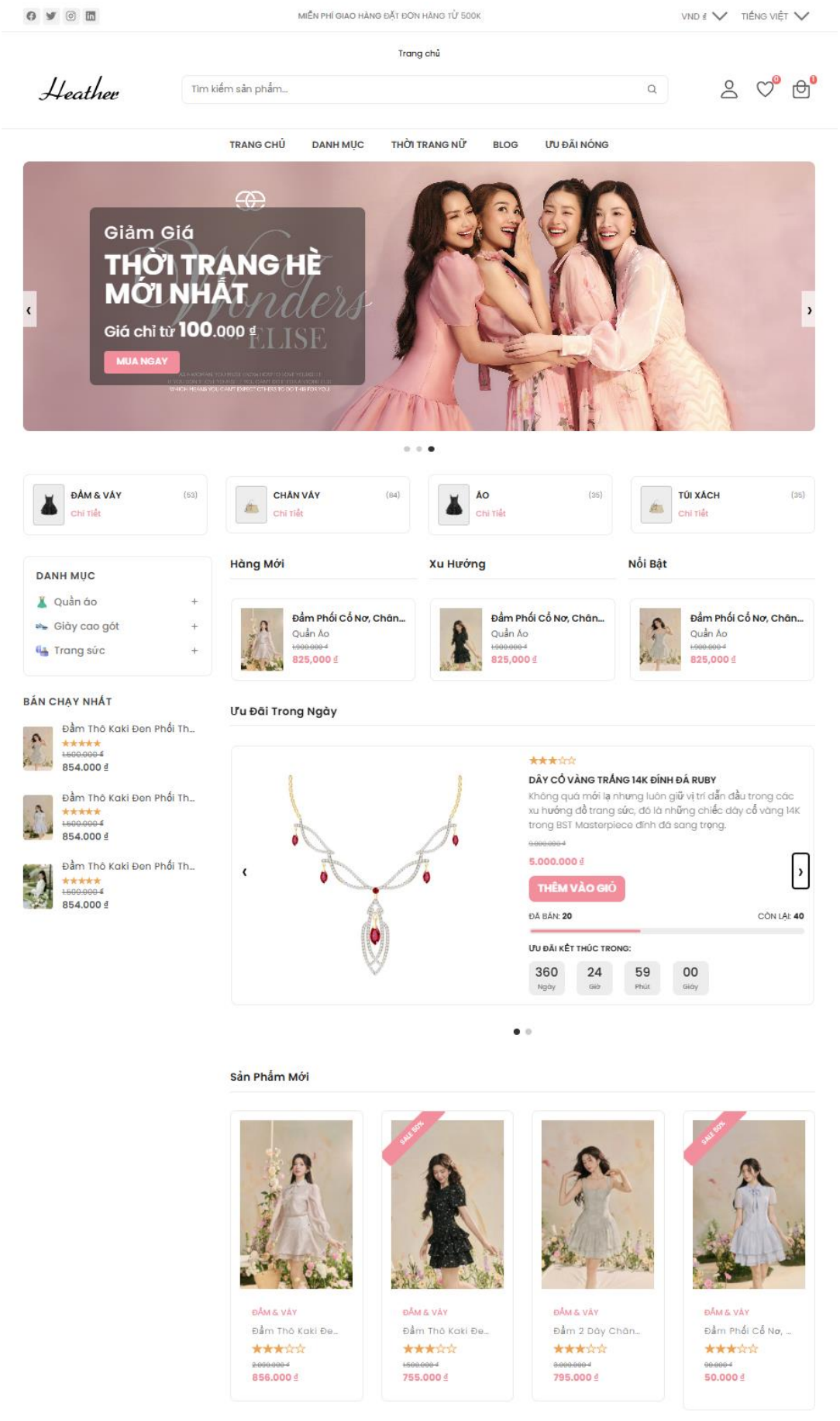


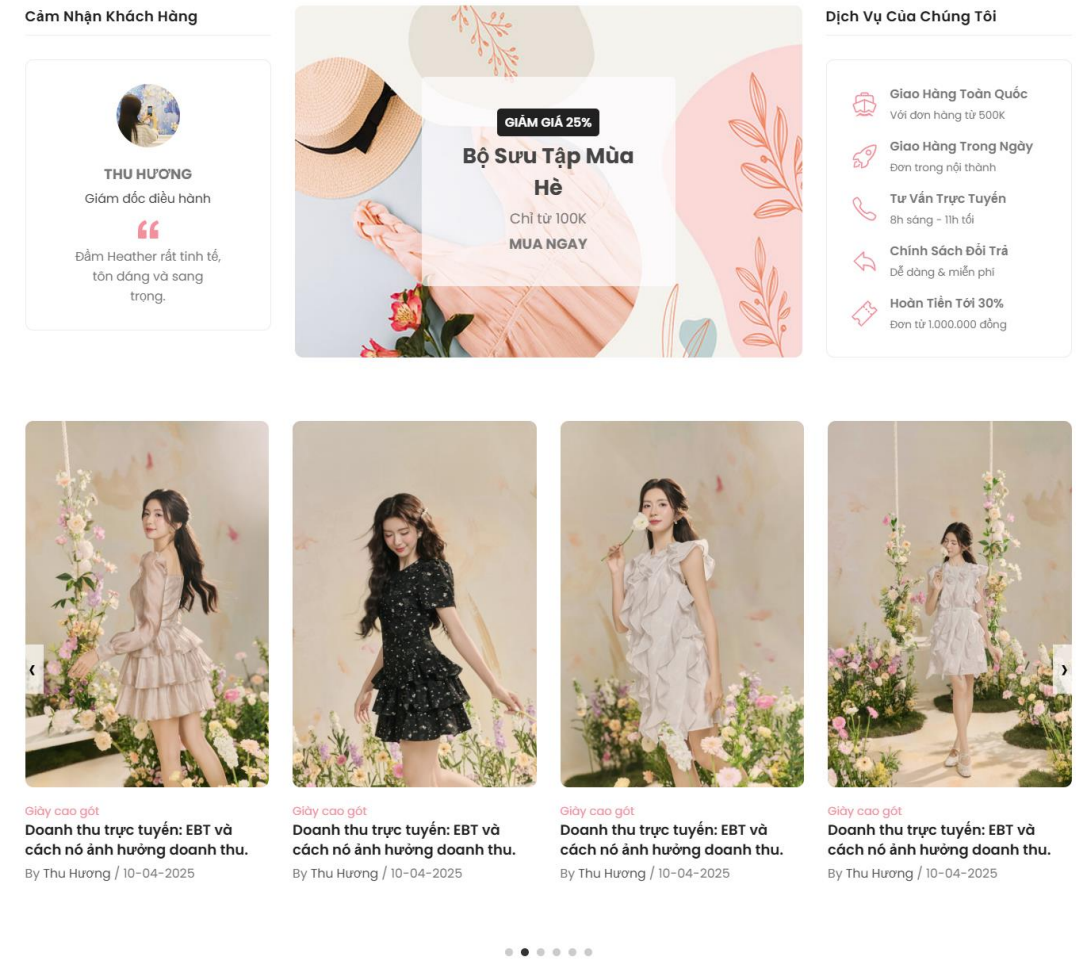
Hình 3.3: Giao diện trang đăng nhập



Hình 3.4: Giao diện trang đăng ký

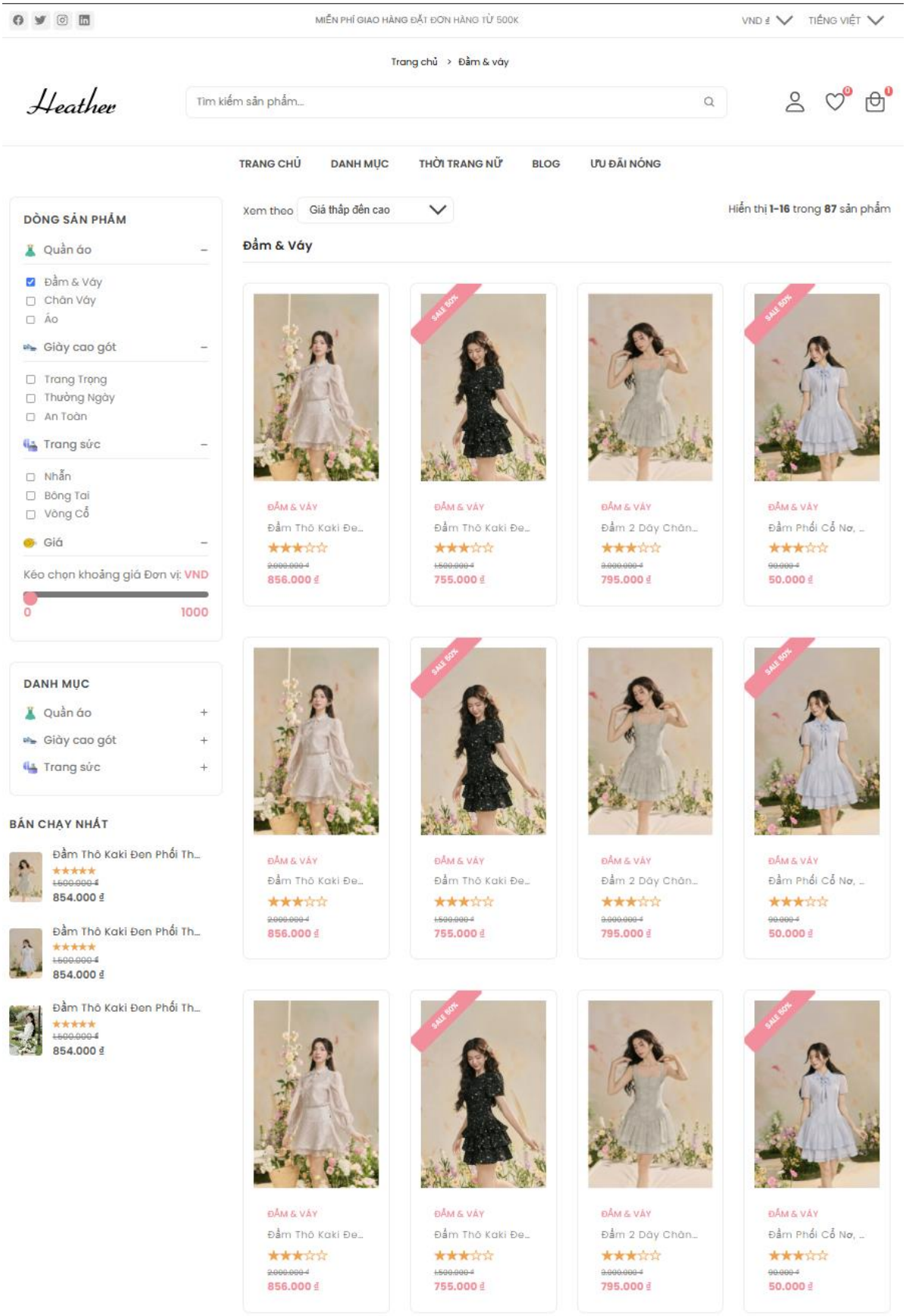
3.3.2 Trang chủ





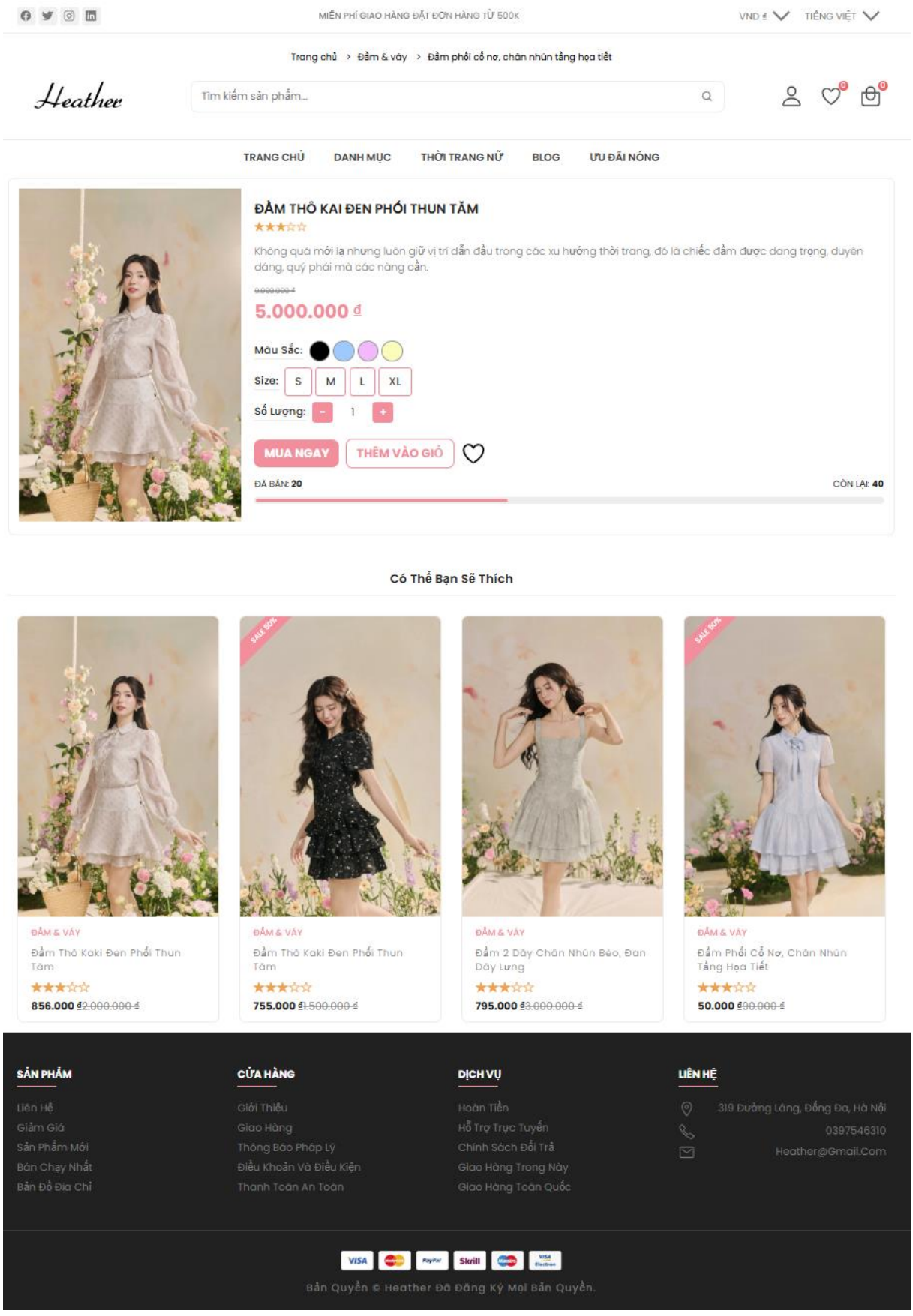
Hình 3.5: Giao diện trang chủ

3.3.3 Danh mục sản phẩm



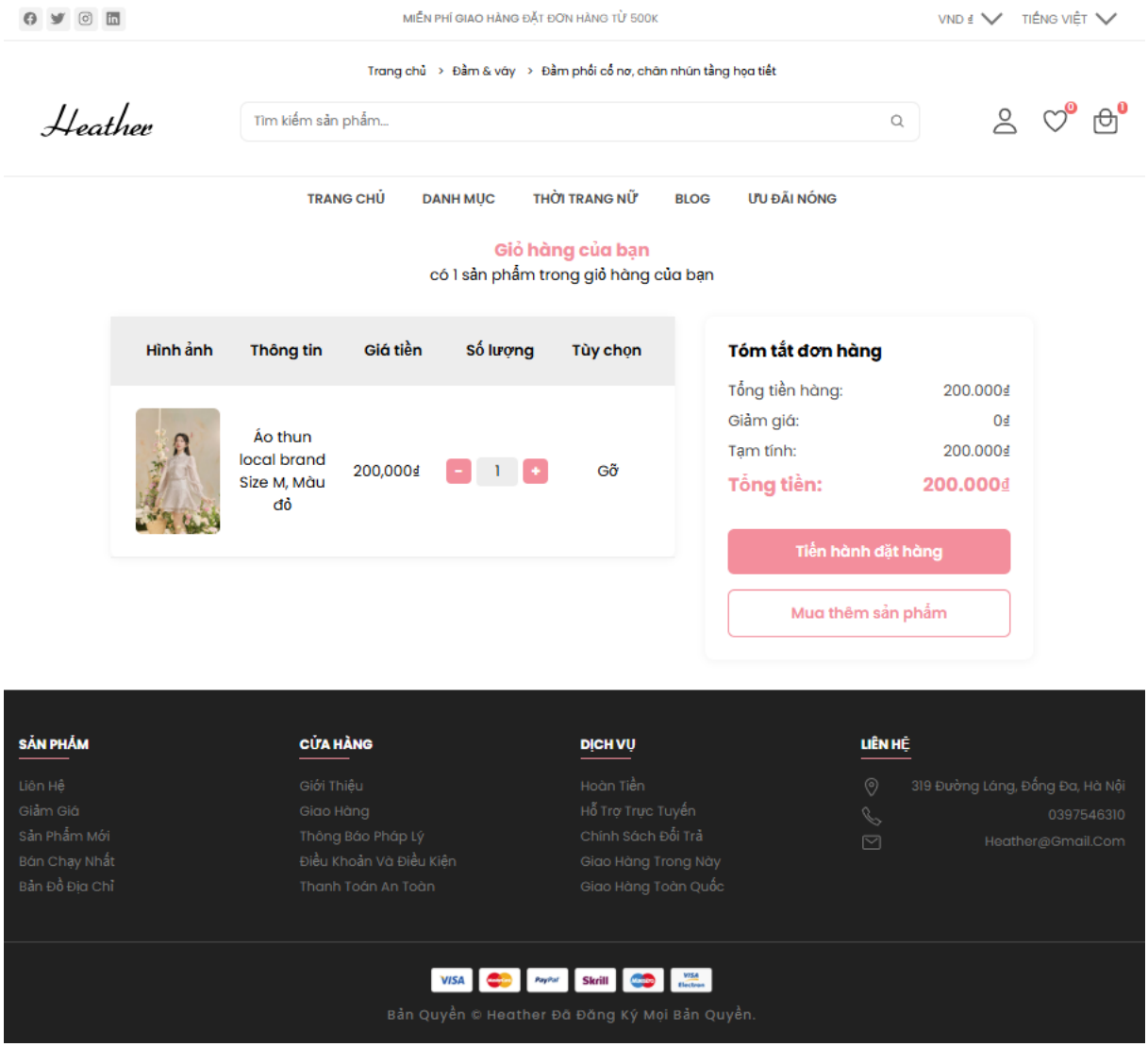
Hình 3.6: Giao diện danh mục sản phẩm

3.3.4 Trang chi tiết sản phẩm



Hình 3.7: Giao diện trang chi tiết sản phẩm

3.3.5 Giỏ hàng



Hình 3.8: giao diện trang giỏ hàng

CHƯƠNG 4: TRIỂN KHAI WEBSITE

4.1 Triển khai các chức năng cho phân hệ người dùng

Để xây dựng được các chức năng của trang người dùng theo thiết kế đã được trình bày ở chương 3, đề tài đã sử dụng HTML, CSS, và JavaScript để thiết kế giao diện và thao tác dữ liệu các trang theo yêu cầu. Tiếp theo, đề tài sẽ trình bày các kỹ thuật được sử dụng để xây dựng các trang.

4.1.1 Trang chủ

❖ **Xây dựng bố cục trang Home bằng các thẻ HTML:**

Trang chủ được cấu trúc bằng các thẻ HTML5 ngữ nghĩa để đảm bảo tính rõ ràng và khả năng mở rộng. Cách tổ chức bao gồm:

- <header>: Hiển thị thanh điều hướng trên cùng, chứa logo, tìm kiếm, biểu tượng giỏ hàng và menu.
- <main>: Phần nội dung chính gồm:
- Banner quảng cáo
- Danh mục sản phẩm
- Sản phẩm nổi bật
- Đánh giá khách hàng (testimonials)
- Blog
- <footer>: Chứa thông tin liên hệ, mạng xã hội và điều hướng bổ sung.

Cấu trúc HTML giúp chia nhỏ trang thành các khu vực chức năng rõ ràng, dễ bảo trì và tích hợp thêm tính năng trong tương lai:

```html
<!DOCTYPE html>
<html lang="vi">
<head>
  <!-- Thẻ meta, tiêu đề, liên kết CSS -->
</head>
<body>
  <!-- Cấu trúc bố cục chính -->
  <div class="overlay" data-overlay></div>

  <!-- Modal đăng ký nhận tin -->
  <div class="modal" data-modal>...</div>

  <!-- Thông báo dạng toast -->
  <div class="notification-toast" data-toast>...</div>

  <!-- Phần đầu trang -->
  <header>
    <div class="header-top">...</div>
    <div class="header-main">...</div>
    <nav class="desktop-navigation-menu">...</nav>
    <nav class="mobile-navigation-menu">...</nav>
  </header>

  <!-- Nội dung chính -->
  <main>
    <!-- Khu vực banner -->
    <div class="banner">...</div>

    <!-- Danh mục sản phẩm -->
    <div class="category">...</div>

    <!-- Sản phẩm nổi bật -->
    <div class="product-container">...</div>

    <!-- Đánh giá, CTA & dịch vụ -->
    <div class="testimonials-box">...</div>

    <!-- Bài viết blog -->
    <div class="blog">...</div>
  </main>

  <!-- Chân trang -->
  <footer>...</footer>
</body>
</html>
```

Hình 4.1: Cấu trúc HTML của trang chủ

❖ **Kỹ thuật định dạng bằng CSS:**

Đề tài sử dụng CSS thuần (Vanilla CSS) với các kỹ thuật hiện đại để định dạng giao diện theo đúng bố cục đã thiết kế:

- Biến CSS (Custom Properties) giúp tái sử dụng màu sắc và các thông số thiết kế thống nhất toàn bộ trang.
- Flexbox và Grid để bố cục linh hoạt và dễ căn chỉnh trong nhiều kích thước màn hình.
- Responsive Design (Mobile-First) bằng media queries để đảm bảo hiển thị tốt trên mọi thiết bị (máy tính bảng, điện thoại, desktop).

```css
/* Tùy biến CSS */
:root {
  --spanish-gray: hsl(0, 0%, 60%);
  --sonic-silver: hsl(0, 0%, 47%);
  --salmon-pink: hsl(353, 100%, 78%);
  /* Other color variables */
}

/* Flexbox */
.header-user-actions {
  display: flex;
  align-items: center;
  gap: 15px;
}

/* Responsive design */
.header-top,
.header-user-actions,
.desktop-navigation-menu {
  display: none;
}

@media (min-width: 570px) {
  .header-top .container {
    display: flex;
    justify-content: space-between;
    align-items: center;
  }
}
```

Hình 4.2: Cấu trúc CSS của trang chủ

❖ **Sử dụng Javascript để lập trình các chức năng:**

a) **Hiển thị Modal / Tosat thông báo**

Thông báo cho người dùng khi có hành động như đăng ký nhận bản tin hoặc thêm sản phẩm vào giỏ hàng.

Thuật toán xử lý:

- Bước 1: Lấy phần tử modal và nút đóng modal ([data-modal-close]) từ DOM.
- Bước 2: Gán sự kiện click cho nút đóng.
- Bước 3: Khi người dùng click, thêm class closed để ẩn modal.

```
const modal = document.querySelector('[data-modal]');
const modalCloseBtn = document.querySelector('[data-modal-close]');

const modalCloseFunc = function () {
  modal.classList.add('closed');
};

modalCloseBtn.addEventListener('click', modalCloseFunc);
```

Hình 4.3: Cấu trúc Javascript của chức năng hiển thị Modal

b) **Trình chiếu slider**

Hiển thị trình chiếu sản phẩm nổi bật hoặc banner tự động và điều hướng thủ công.

Thuật toán xử lý:

- Bước 1: Truy xuất các phần tử trong slider từ DOM (container, item, nút, dots).
- Bước 2: Tạo chấm tròn (dot) tương ứng với số lượng slide.
- Bước 3: Khi người dùng click vào dot hoặc nút trái/phải, thực hiện chuyển slide.
- Bước 4: Tự động chuyển slide sau khoảng thời gian delay.
- Bước 5: Mỗi lần người dùng thao tác thủ công, reset lại đồng hồ tự động chạy.

Cấu trúc HTML cần có cho chức năng slider:

```html
<!-- Nút trái -->
<button id="prevBtn" class="slider-arrow left">‹</button>

<!-- Nút phải -->
<button id="nextBtn" class="slider-arrow right">›</button>

<!-- Chấm điều hướng -->
<div class="slider-dots"></div>
```

Hình 4.4: Cấu trúc HTML cần có cho chức năng slider

```
function initSlider({
  containerSel,
  itemSel,
  prevBtnSel,
  nextBtnSel,
  dotsContainerSel,
  delay = 3000,
  step = 1,
}) {
  const container = document.querySelector(containerSel);
  const items = container.querySelectorAll(itemSel);
  const slideCount = items.length;
  let currentIndex = 0,
    autoSlide;

  // Tạo các dots
  const dotsContainer = document.querySelector(dotsContainerSel);
  dotsContainer.innerHTML = "";
  const dots = [];

  for (let i = 0; i < slideCount; i += step) {
    const dot = document.createElement("span");
    dot.classList.add("dot");
    if (i === 0) dot.classList.add("active");
    dot.addEventListener("click", () => {
      goToSlide(i);
      restart();
    });
    dotsContainer.appendChild(dot);
    dots.push(dot);
  }
```

```
  function updateDots(idx) {
    dots.forEach((d, i) => d.classList.toggle("active", i === idx));
  }

  function goToSlide(idx) {
    currentIndex = idx;
    container.scrollTo({
      left: container.clientWidth * idx,
      behavior: "smooth",
    });
    updateDots(idx);
  }

  function goToNext() {
    goToSlide((currentIndex + step) % slideCount);
  }

  function goToPrev() {
    goToSlide((currentIndex - step + slideCount) % slideCount);
  }

  function restart() {
    clearInterval(autoSlide);
    autoSlide = setInterval(goToNext, delay);
  }

  const prevBtn = document.querySelector(prevBtnSel);
  const nextBtn = document.querySelector(nextBtnSel);
  if (prevBtn)
    prevBtn.addEventListener("click", () => {
      goToPrev();
      restart();
    });
  if (nextBtn)
    nextBtn.addEventListener("click", () => {
      goToNext();
      restart();
    });

  restart();
  goToSlide(0);
}
```

Hình 4.5: Cấu trúc Javascript của chức năng slider

***c)* Hiển thị mobile menu**

Giúp người dùng mở menu điều hướng khi dùng thiết bị màn hình nhỏ như điện thoại.

Thuật toán xử lý:

- Bước 1: Lấy danh sách các nút mở menu ([data-mobile-menu-open-btn]) và danh sách các menu ([data-mobile-menu]).
- Bước 2: Gán sự kiện click cho mỗi nút mở.
- Bước 3: Khi người dùng click vào nút mở, thêm class active vào menu tương ứng và lớp phủ nền (overlay) để hiển thị menu di động.

```
const mobileMenuOpenBtn = document.querySelectorAll(
  "[data-mobile-menu-open-btn]"
);
const mobileMenu = document.querySelectorAll("[data-mobile-menu]");
const overlay = document.querySelector(".overlay"); // lớp phủ

for (let i = 0; i < mobileMenuOpenBtn.length; i++) {
  mobileMenuOpenBtn[i].addEventListener("click", function () {
    mobileMenu[i].classList.add("active");
    overlay.classList.add("active");
  });
}
```

Hình 4.6: Cấu trúc Javascript của chức năng hiển thị moblie menu

4.1.2 Trang chi tiết sản phẩm

❖ **Xây dựng bố cục trang Home bằng các thẻ HTML:**

Bố cục trang được chia làm 2 phần:

- <main class="product-detail">: Là thẻ bao ngoài nội dung chi tiết sản phẩm, giúp định danh nội dung chính của trang.
- <section class="product-showcase">: Hiển thị phần giới thiệu sản phẩm bao gồm:
- <section class="product-details">: Phần mô tả chi tiết sản phẩm bao gồm:

```
<main class="product-detail">
  <!-- Phần hình ảnh và thông tin cơ bản -->
  <section class="product-showcase">
    <div class="product-image">
      <div class="main-image">
        <img src="" alt="" id="mainImage" />
      </div>
      <div class="thumbnail-list">
        <!-- Các ảnh thumbnail -->
      </div>
    </div>

    <div class="product-info">
      <h1 class="product-title"></h1>
      <div class="product-price">
        <span class="current-price"></span>
        <span class="original-price"></span>
      </div>

      <div class="product-actions">
        <div class="quantity-selector">
          <button class="decrease">-</button>
          <input type="number" value="1" min="1" />
          <button class="increase">+</button>
        </div>
        <button class="add-to-cart">Thêm vào giỏ hàng</button>
      </div>
    </div>
  </section>

  <!-- Phần mô tả chi tiết -->
  <section class="product-details">
    <div class="tabs">
      <button class="tab active" data-tab="description">Mô tả</button>
      <button class="tab" data-tab="specifications">Thông số</button>
    </div>
    <div class="tab-content"></div>
  </section>
</main>
```

Hình 4.7: Cấu trúc HTML của trang xem sản phẩm

❖ **Kỹ thuật định dạng bằng CSS:**

- Flexbox và Grid để bố cục linh hoạt và dễ căn chỉnh trong nhiều kích thước màn hình.
- Responsive Design (Mobile-First) bằng media queries để đảm bảo hiển thị tốt trên mọi thiết bị (máy tính bảng, điện thoại, desktop).

```scss
.product-detail {
    max-width: 1200px;
    margin: 0 auto;
    padding: 2rem;
  }

  .product-showcase {
    display: grid;
    grid-template-columns: 1fr 1fr;
    gap: 2rem;
    margin-bottom: 3rem;
  }

  .product-image {
    .main-image {
      aspect-ratio: 1;
      border-radius: var(--border-radius-md);
      overflow: hidden;
      margin-bottom: 1rem;

      img {
        width: 100%;
        height: 100%;
        object-fit: cover;
      }
    }

    .thumbnail-list {
      display: flex;
      gap: 1rem;
    }
  }

  .product-actions {
    display: flex;
    gap: 1rem;
    margin-top: 2rem;

    .quantity-selector {
      display: flex;
      align-items: center;
      border: 1px solid #ddd;
      border-radius: var(--border-radius-md);
    }

    .add-to-cart {
      background: var(--main-color);
      color: white;
      padding: 0.5rem 2rem;
      border-radius: var(--border-radius-md);
    }
  }
```

Hình 4.8: Cấu trúc CSS của trang xem sản phẩm

❖ **Sử dụng Javascript để lập trình các chức năng:**

a) Chức năng chọn số lượng sản phẩm

Giúp người dùng chọn số lượng sản phẩm muốn mua một cách trực quan.

Thuật toán xử lý:

- Bước 1: Lắng nghe sự kiện click vào nút – và +.
- Bước 2: Lấy giá trị hiện tại từ ô nhập số lượng (quantityInput.value).
- Bước 3: Nếu click nút –, giảm số lượng nhưng không nhỏ hơn 1.
- Bước 4: Nếu click nút +, tăng số lượng.

```
document.querySelector('.decrease').onclick = () => {
  if (quantityInput.value > 1) quantityInput.value--;
};
document.querySelector('.increase').onclick = () => {
  quantityInput.value++;
};
```

Hình 4.9: Cấu trúc Javascript của chức năng chọn số lượng sản phẩm

b) Chức năng thêm giỏ hàng

Lưu sản phẩm vào giỏ hàng để người dùng có thể thanh toán sau.

Thuật toán xử lý:

- Bước 1: Khi click nút “Thêm vào giở hàng”, lấy số lượng người dùng đã chọn.
- Bước 2: Kiểm tra trong localStorage xem đã có sản phẩm đó chưa:
- Nếu đã có -> cộng thêm sản phẩm.
- Nếu chưa có -> thêm mới sản phẩm vòa giỏ hàng.
- Bước 3: Lưu lại giỏ hàng mới vào localStorage.
- Bước 4: Hiển thị thông báo cho người dùng “đã thêm vào giỏ hàng”.

```javascript
function addToCart(product, quantity) {
  let cart = JSON.parse(localStorage.getItem('cart') || '[]');

  const existingItem = cart.find(item => item.id === product.id);
  if (existingItem) {
    existingItem.quantity += quantity;
  } else {
    cart.push({...product, quantity});
  }

  localStorage.setItem('cart', JSON.stringify(cart));
  showNotification('Đã thêm sản phẩm vào giỏ hàng');
}
```

Hình 4.10: Cấu trúc Javascript của chức năng thêm giỏ hàng

4.1.3 Giỏ hàng

❖ **Xây dựng bố cục trang Home bằng các thẻ HTML:**

```
<main class="cart">
  <!-- Tiêu đề trang -->
  <section>
    <div class="container">
      <div class="title-wrapper center">
        <h3 class="title-head">Giỏ hàng của bạn</h3>
        <p class="title-main">
            có
            <span id="cart-count">0</span>
            sản phẩm</p>
      </div>
    </div>
  </section>

  <!-- Danh sách sản phẩm trong giỏ -->
  <section class="cart-list">
    <div class="cart-container">
      <div class="cart-layout">
        <!-- Bảng sản phẩm -->
        <div class="cart-table">
          <div class="cart-row cart-header">
            <div class="cart-col image">Hình ảnh</div>
            <div class="cart-col info">Thông tin</div>
            <div class="cart-col price">Giá tiền</div>
            <div class="cart-col quantity">Số lượng</div>
            <div class="cart-col option">Tùy chọn</div>
          </div>
          <!-- Các dòng sản phẩm -->
        </div>

        <!-- Tóm tắt đơn hàng -->
        <div class="order-summary">
          <h3>Tóm tắt đơn hàng</h3>
          <!-- Chi tiết thanh toán -->
        </div>
      </div>
    </div>
  </section>
</main>
```

Hình 4.11: Cấu trúc HTML của trang giỏ hàng

❖ **Kỹ thuật định dạng bằng CSS:**

```css
/* Container chính */
.cart-layout {
  display: flex;
  gap: 2rem;
  align-items: flex-start;
  max-width: 1100px;
  margin: 2rem auto;
  flex-wrap: wrap;
}

/* Bảng sản phẩm */
.cart-table {
  flex: 2;
  background-color: white;
  border-radius: var(--border-radius-md);
  box-shadow: 0 2px 10px rgba(0, 0, 0, 0.05);
  width: 100%;
  min-width: 600px;
}

/* Dòng sản phẩm */
.cart-row {
  display: flex;
  padding: 1rem;
  align-items: center;
  border-bottom: 1px solid var(--cultured);
}

/* Bộ chọn số lượng */
.number-picker {
  display: flex;
  align-items: center;
  gap: 6px;
}

/* Nút tăng/giảm số lượng */
.number-picker-btn {
  width: 30px;
  height: 30px;
  background-color: var(--salmon-pink);
  color: white;
  border-radius: 6px;
}
```

Hình 4.12: Cấu trúc CSS của trang giỏ hàng

❖ **Sử dụng Javascript để lập trình các chức năng:**

a) Quản lý giỏ hàng

Giúp người dùng thêm sản phẩm vào giỏ hàng và lưu trữ tạm thời trên trình duyệt.

Thuật toán xử lý:

- Bước 1: Lấy giỏ hàng từ localStorage, nếu chưa có thì khởi tạo mảng rỗng.
- Bước 2: Kiểm tra sản phẩm đã tồn tại trong giỏ hàng chưa bằng id.
- Bước 3: Nếu có → cập nhật số lượng; nếu không → thêm mới sản phẩm vào giỏ.
- Bước 4: Ghi lại giỏ hàng mới vào localStorage.
- Bước 5: Hiển thị thông báo đã thêm vào giỏ.

```
function addToCart(product, quantity) {
  let cart = JSON.parse(localStorage.getItem('cart') || '[]');

  // Kiểm tra sản phẩm đã tồn tại
  const existingItem = cart.find(item => item.id === product.id);
  if (existingItem) {
    existingItem.quantity += quantity;
  } else {
    cart.push({...product, quantity});
  }

  // Lưu vào localStorage
  localStorage.setItem('cart', JSON.stringify(cart));
  showNotification('Đã thêm sản phẩm vào giỏ hàng');
}
```

Hình 4.13: Cấu trúc JavaScript của chức năng quản lý giỏ hàng

b) Cập nhật tổng tiền

Tính và hiển thị lại tổng tiền mỗi khi số lượng sản phẩm thay đổi.

Thuật toán xử lý:

- Bước 1: Lấy tất cả các dòng sản phẩm trong giỏ (trừ dòng tiêu đề).
- Bước 2: Duyệt qua từng dòng → lấy giá và số lượng → tính tổng tiền.
- Bước 3: Tính các khoản khác nếu có (giảm giá, phí ship...).
- Bước 4: Cập nhật lại tổng tiền lên giao diện.

```
const updateSummary = () => {
  const rows = document.querySelectorAll(".cart-row:not(.cart-header)");
  let subtotal = 0;

  // Tính tổng tiền
  rows.forEach(row => {
    const price = parseInt(row.querySelector(".product-price").dataset.price);
    const quantity = parseInt(row.querySelector(".number-picker-input").value);
    subtotal += price * quantity;
  });

  // Cập nhật hiển thị
  const discount = 0;
  const tempTotal = subtotal - discount;
  const finalTotal = tempTotal;

  document.querySelector(".subtotal").textContent =
    subtotal.toLocaleString("vi-VN") + "đ";
  document.querySelector(".final-total").textContent =
    finalTotal.toLocaleString("vi-VN") + "đ";
};
```

Hình 4.14: Cấu trúc Javascript của chức năng cập nhật tổng tiền

c) Xử lý đặt hàng

Khi người dùng nhấn nút "Đặt hàng", hệ thống xác nhận và reset giỏ hàng.

Thuật toán xử lý:

- Bước 1: Lấy tất cả input số lượng sản phẩm trong giỏ.
- Bước 2: Duyệt qua các input để tính tổng số lượng sản phẩm.
- Bước 3: Hiển thị tổng sản phẩm đã đặt lên giao diện.
- Bước 4: Kích hoạt modal thông báo đặt hàng thành công.
- Bước 5: Xóa giỏ hàng khỏi localStorage.

```
document.getElementById("place-order").addEventListener("click", () => {
  const quantities = document.querySelectorAll(".number-picker-input");
  let totalItems = 0;

  // Tính tổng số lượng
  quantities.forEach(input => {
    totalItems += parseInt(input.value);
  });

  // Hiển thị thông báo thành công
  document.getElementById("totalItems").textContent = totalItems;
  document.getElementById("successModal").classList.add("active");

  // Xóa giỏ hàng
  localStorage.removeItem('cart');
});
```

Hình 4.15: Cấu trúc Javascript của chức năng xử lý đặt hàng

4.2 Triển khai các chức năng cho phân hệ quản trị nội dung

<Phần này trình bày các kết quả đã được triển khai cho phân hệ trang quản trị>

4.3 Kiểm thử và triển khai ứng dụng

4.3.1 Kiểm thử

Bảng 4.1: Danh sách kết quả kiểm thử

STT	Tên chức năng	Dữ liệu đầu vào	Hành động kiểm thử	Kết quả mong đợi	Trạng thái
1	Đăng ký	Tên, email, mật khẩu hợp lệ	Nhấn nút "Đăng ký"	Hệ thống lưu tài khoản và chuyển hướng đến trang đăng nhập	Pass
2	Đăng nhập	Email và mật khẩu đúng	Nhấn nút "Đăng nhập"	Hệ thống chuyển đến trang chủ và hiển thị tên người dùng	Pass
3	Xem danh mục sản	-	Truy cập trang danh mục sản	Hiển thị tất cả sản phẩm theo danh mục	Pass

	phẩm		phẩm		
4	Xem chi tiết sản phẩm	ID sản phẩm	Click vào "Xem chi tiết"	Hiển thị thông tin chi tiết đúng của sản phẩm	Pass
5	Thêm vào giỏ hàng	Chọn size, số lượng sản phẩm	Nhấn nút "Thêm vào giỏ hàng"	Sản phẩm được thêm đúng vào giỏ hàng	Pass
6	Cập nhật giỏ hàng	Thay đổi số lượng hoặc xóa sản phẩm	Click cập nhật/xóa	Giỏ hàng được cập nhật đúng theo hành động	Pass
7	Đặt hàng	Thông tin người nhận, phương thức thanh toán, sản phẩm giỏ hàng	Nhấn "Đặt hàng"	Đơn hàng được lưu, thông báo đặt hàng thành công	Pass

4.3.2 Đóng gói ứng dụng

Sau khi kiểm thử thành công, ứng dụng được đóng gói để chuẩn bị cho việc triển khai trên môi trường thực tế.

a) **Tổ chức mã nguồn theo cấu trúc thư mục chuẩn:**

- /assets/ (hình ảnh, CSS, JS)
- /pages/ (HTML từng trang)
- /scripts/ (logic JS)
- /styles/ (tệp CSS)

b) **Gộp và nén file:**

- Dùng công cụ Minify cho file CSS và JS.
- Gộp các tệp JS thành bundle.js, CSS thành style.min.css.
- Tạo tệp .zip hoặc thư mục build chứa toàn bộ source code và tài nguyên.

- Kiểm tra thử bằng cách chạy trên localhost hoặc dùng server giả lập Live Server để test lần cuối.

4.3.3 Triển khai ứng dụng

***a)* Yêu cầu phần cứng**

- Bộ vi xử lý: Tối thiểu Intel Core i3 hoặc tương đương.
- RAM: Từ 4GB trở lên.
- Dung lượng ổ cứng: Tối thiểu 200MB cho ứng dụng và dữ liệu.
- Trình duyệt hỗ trợ: Chrome, Firefox, Edge (các phiên bản từ năm 2020 trở lên).

***b)* Yêu cầu phần mềm**

- Hệ điều hành: Windows 10/11, macOS hoặc Linux.
- Trình biên tập mã nguồn: Visual Studio Code.
- Công nghệ frontend: HTML5, CSS3, JavaScript ES6+.
- Triển khai thực tế: Upload thư mục build lên Git.

KẾT LUẬN

Sau quá trình nghiên cứu và thực hiện, đề tài đã hoàn thiện cơ bản một website bán hàng trực tuyến đơn giản dành cho khách hàng. Các chức năng đã triển khai đều dựa trên HTML, CSS và JavaScript thuần, không sử dụng framework logic hay backend. Tuy nhiên vì kinh nghiệm còn thiếu nên vẫn sẽ còn có những hạn chế.

❖ **Kết quả cụ thể như sau:**

- Đã xây dựng được giao diện trang chủ, với bố cục hiện đại, hiển thị banner, danh mục sản phẩm, sản phẩm nổi bật, đồng thời có liên kết dẫn đến trang chi tiết sản phẩm.
- Hoàn thiện giao diện và chức năng xem danh mục sản phẩm, cho phép hiển thị sản phẩm theo từng loại, sử dụng kỹ thuật CSS Grid và Flexbox để căn chỉnh bố cục.
- Xây dựng trang chi tiết sản phẩm, có thể hiển thị đầy đủ thông tin sản phẩm như tên, giá, mô tả, hình ảnh, và nút "Thêm vào giỏ hàng".
- Cài đặt giỏ hàng sử dụng localStorage để lưu trữ sản phẩm, hỗ trợ tăng/giảm số lượng, xóa sản phẩm, và tính tổng giá trị đơn hàng.
- Giao diện được thiết kế responsive với CSS media queries, đảm bảo hiển thị tốt trên thiết bị di động và màn hình máy tính.
- Đã xử lý tương tác người dùng thông qua JavaScript: xử lý sự kiện khi click, nhập liệu, chuyển trang, cập nhật dữ liệu lên DOM…

Ngoài việc hoàn thiện sản phẩm, đề tài còn giúp em đạt được nhiều kiến thức mới về công nghệ web, cụ thể:

- Biết cách tổ chức và xây dựng một giao diện web bằng HTML/CSS/JAVASCRIPT
- Nắm được các kỹ thuật Javascript cơ bản trong việc thoa tác với DOM và xử lý dũ liệu trên trình duyệt.
- Đạt tới mức cơ bản về lưu trữ và truy xuất dữ liều với localStorage, qua đó giả lập tính năng giỏ hàng mà không cần cơ sở dữ liệu.
- Rèn luyện kỹ năng phân tích chức năng và mô hình hóa hệ thống qua việc thiết kế thực thể, luồng sự kiện và các trang chính.

❖ **Hạn chế của đề tài**

Bên cạnh những kết quả đã đạt được, đề tài cũng còn tồn tại một số hạn chế nhất định, chủ yếu xuất phát từ việc chỉ sử dụng công nghệ phía client (HTML, CSS, JS thuần), cụ thể:

- Chức năng đăng ký và đăng nhập hiện tại chỉ mang tính giao diện, chưa kết nối với bất kỳ hệ thống xác thực hoặc cơ sở dữ liệu nào.
- Website chưa thực sự liên kết phần quản trị viên, nên việc thêm, sửa, xóa sản phẩm hoàn toàn chưa thực hiện được.
- Dữ liệu của sản phẩm, giỏ hàng chỉ được lưu trữ cục bộ thông qua localStorage, không thể chia sẻ hoặc lưu trữ lâu dài.
- Website chưa có backend, không sử dụng cơ sở dữ liệu nên không thể xử lý các đơn hàng thực tế hoặc lưu thông tin người dùng thật.
- Thiếu các kỹ thuật như tối ưu hiệu năng, bảo vệ dữ liệu người dùng).

❖ **Hướng phát triển**

Trong tương lai, để nâng cao chất lượng sản phẩm và tiến gần hơn đến một hệ thống hoàn chỉnh, đề tài có thể được phát triển thêm theo các hướng sau:

- Tích hợp backend để xử lý đăng ký, đăng nhập, giỏ hàng, đơn hàng và xác thực người dùng.
- Kết nối cơ sở dữ liệu để lưu trữ thông tin người dùng, sản phẩm, đơn hàng và giỏ hàng một cách bền vững.
- Bổ sung liên kết phân hệ quản trị với trang web.
- Tích hợp cổng thanh toán như Momo, VNPAY hoặc Stripe để mô phỏng quy trình thanh toán thực tế.
- Tối ưu hiệu suất và SEO, giúp website tải nhanh hơn, thân thiện với công cụ tìm kiếm và trải nghiệm người dùng tốt hơn.
- Triển khai website lên môi trường thực tế như Netlify, Vercel hoặc Firebase Hosting để cho phép người thật sử dụng.
- Áp dụng thiết kế hướng đối tượng và mô hình module hóa để tách biệt logic xử lý, tái sử dụng mã nguồn tốt hơn.

TÀI LIỆU THAM KHẢO

[1] "Internet in Vietnam," Wikipedia, truy cập ngày 15 tháng 5 năm 2025.

[2] W3Schools. (n.d.). HTML Introduction.

[3] W3Schools. (n.d.). CSS Introduction.

[4] W3Schools. (n.d.). JavaScript Introduction.