TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI
PHÂN HIỆU TẠI TP. HỒ CHÍ MINH
BỘ MÔN CÔNG NGHỆ THÔNG TIN

HỌ VÀ TÊN SINH VIÊN: **TRẦN THẢO NƯƠNG**
HUỲNH GIA BẢO
LÊ QUANG CHÍ
NGUYỄN NGỌC NGÀ

TÊN ĐỀ TÀI:

THIẾT KẾ WEB THÔNG TIN DỰ BÁO THỜI TIẾT

BÁO CÁO MÔN HỌC: THIẾT KẾ WEB

TP.Hồ Chí Minh – 2025

TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI
PHÂN HIỆU TẠI TP. HỒ CHÍ MINH
BỘ MÔN CÔNG NGHỆ THÔNG TIN

HỌ VÀ TÊN SINH VIÊN: **TRẦN THẢO NƯƠNG**
HUỲNH GIA BẢO
LÊ QUANG CHÍ
NGUYỄN NGỌC NGÀ

TÊN ĐỀ TÀI:
THIẾT KẾ WEB THÔNG TIN DỰ BÁO THỜI TIẾT

Lớp: Công nghệ thông tin
Khóa:65
NGƯỜI HƯỚNG DẪN
1.ThS.NGUYỄN THIỆN DƯƠNG

TP.Hồ Chí Minh – Năm 2025

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Tp.HCM,ngày.....tháng ..... năm.....

Giảng viên hướng dẫn

ThS.Nguyễn Thiện Dương

NHẬN XÉT CỦA GIẢNG VIÊN PHẢN BIỆN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Nội dung đồ án đáp ứng yêu cầu của một đồ án tốt nghiệp cử nhân ngành CNTT.

Tp.HCM,ngày.....tháng ..... năm.....

Giảng viên phản biện

[Ký tên ghi rõ họ và tên]

LỜI CẢM ƠN

Lời đầu tiên, chúng em xin gửi đến Quý Thầy Cô Bộ môn Công nghệ Thông tin, Trường Đại học Giao thông Vận tải – Phân hiệu tại TP. Hồ Chí Minh, lời cảm ơn và kính chúc sức khỏe.

Đặc biệt, chúng em xin chân thành cảm ơn thầy Nguyễn Công Danh, giảng viên môn Thiết kế Web, đã tận tình hướng dẫn và truyền đạt cho chúng em nhiều kiến thức quý báu trong quá trình học tập và thực hiện đề tài *"Thiết kế Web thông tin dự báo thời tiết"*. Nhờ sự chỉ dạy tận tâm của thầy, chúng em đã củng cố được kỹ năng lập trình, thiết kế giao diện và tinh thần làm việc nhóm để hoàn thiện sản phẩm. Mặc dù đã cố gắng, nhưng do kiến thức còn hạn chế, bài làm khó tránh khỏi thiếu sót, mong thầy góp ý để chúng em hoàn thiện hơn.

Cuối cùng, chúng em xin kính chúc thầy nhiều sức khỏe, niềm vui và thành công trong sự nghiệp giảng dạy.

Chúng em xin chân thành cảm ơn!

TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

PHÂN HIỆU TẠI THÀNH PHỐ HỒ CHÍ MINH

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM

Độc lập – Tự do – Hạnh phúc

ĐỀ CƯƠNG CHI TIẾT

BỘ MÔN: **CÔNG NGHỆ THÔNG TIN**

-------***-------

Mã sinh viên: 6551071060 **Họ tên SV:** Trần Thảo Nương

6551071007 Huỳnh Gia Bảo

6551071010 Lê Quang Chí

6551071056 Nguyễn Ngọc Ngà

Khóa: 65 **Lớp:** Công nghệ thông tin

1. **Tên đề tài**

   Thiết kế web thông tin dự báo thời tiết.

2. **Mục đích thực hiện**

   Đề tài được thực hiện nhằm vận dụng các kiến thức đã học trong môn Thiết kế Web để xây dựng một trang web có khả năng hiển thị thông tin dự báo thời tiết theo thời gian thực.

   Về mặt lý thuyết, đề tài giúp nhóm củng cố kiến thức về HTML, CSS, JavaScript, Jquery và cấu trúc website.

   Về mặt thực nghiệm, nhóm mong muốn tạo ra một sản phẩm có giao diện thân thiện, dễ sử dụng, hỗ trợ người dùng tra cứu thông tin thời tiết nhanh chóng và trực quan.

3. **Mục tiêu thực hiện**

   - Xây dựng trang web hiển thị thông tin dự báo thời tiết theo vị trí người dùng.
   - Thiết kế giao diện hiện đại, trực quan và tương thích trên nhiều thiết bị.

- Tích hợp API để lấy dữ liệu thời tiết theo thời gian thực.
- Rèn luyện kỹ năng lập trình front-end và làm việc nhóm.

4. Nội dung và phạm vi đề tài

Đề tài tập trung vào xây dựng giao diện và xử lý dữ liệu dự báo thời tiết bằng HTML, CSS và JavaScript, jquery.

- **Input:** Tên địa điểm hoặc vị trí người dùng.
- **Output:** Thông tin dự báo thời tiết như nhiệt độ, độ ẩm, tốc độ gió, tình trạng mưa/nắng.
- Nội dung chính gồm:
- Thiết kế bố cục trang web.
- Kết nối và xử lý dữ liệu từ API (ví dụ: OpenWeatherMap).
- Hiển thị dữ liệu bằng biểu đồ và icon minh họa.
- Kiểm thử, tối ưu và hoàn thiện giao diện.

5. Phương pháp thực hiện

- **Ngôn ngữ:** HTML, CSS, JavaScript.
- **Công nghệ:** API OpenWeatherMap, Chart.js để hiển thị biểu đồ.
- **Công cụ:** Visual Studio Code, GitHub, trình duyệt Chrome.
- **Phương pháp:** Phân tích – thiết kế – lập trình – kiểm thử – hoàn thiện.

6. Các kết quả chính dự kiến sẽ đạt được và ứng dụng

- Hoàn thiện một trang web có khả năng hiển thị thời tiết theo địa điểm nhập vào.
- Giao diện thân thiện, dễ sử dụng, phản hồi nhanh.
- Ứng dụng có thể triển khai thực tế cho người dùng tra cứu thông tin thời tiết hằng ngày.
- Là nền tảng để mở rộng sang ứng dụng di động hoặc thêm các tính năng như cảnh báo thời tiết.

7. Kế Hoạch Thực Hiện:

Giai đoạn	Thời gian	Nội dung công việc	Người thực hiện
1.Tìm hiểu đề tà, thu thập tài liều	18/10 – 20/10	Nghiên cứu về API, xác định yêu cầu. Thống nhất giao diện cho web.	Huỳnh Gia Bảo Lê Quang Chí Nguyễn Ngọc Ngà Trần Thảo Nương
2.Thiết kế giao diện	20/10 – 5/11	Tạo bố cục, chọn màu sắc, font chữ.	Trần Thảo Nương Huỳnh Gia Bảo Nguyễn Ngọc Ngà
3. Lập trình chức năng chính	25/10 - 15/11	Kết nối API, xử lý dữ liệu.	Huỳnh Gia Bảo Lê Quang Chí Nguyễn Ngọc Ngà Trần Thảo Nương
4.Hoàn thiện, kiểm thử và chỉnh sửa	15/11 – 10/12	Kiểm tra lỗi, tối ưu hiển thị.	Huỳnh Gia Bảo Lê Quang Chí Nguyễn Ngọc Ngà Trần Thảo Nương
5. Viết báo cáo và nộp sản phẩm	22/11 – 11/12	Tổng hợp, hoàn thiện báo cáo.	Trần Thảo Nương Huỳnh Gia Bảo Lê Quang Chí Nguyễn Ngọc Ngà

8. Giảng viên hướng dẫn

Họ tên: Nguyễn Thiện Dương

Đơn vị công tác: Trường đại học giao thông vẩn tải phân hiệu thành phố Hồ Chí Minh

Điện thoại: 0978 720 901 Email: ntduong@st.utc2.edu.vn

Ngày … tháng … năm …	**Đã giao nhiệm vụ TKTN**
Trưởng BM Công nghệ Thông tin	**Giáo viên hướng dẫn**
(Ký xác nhận)	(Ký xác nhận)
ThS. Trần Phong Nhã	**ThS. Nguyễn Thiện Dương**

Đã nhận nhiệm vụ TKTN

Sinh viên: Trần Thảo Nương	Ký tên:
Điện thoại: 0976674955	Email: 6551071060@st.utc2.edu.vn
Sinh viên: Lê Quang Chí	Ký tên:
Điện thoại: 0364819217	Email: 6551071010@st.utc2.edu.vn
Sinh viên: Nguyễn Ngọc Ngà	Ký tên:
Điện thoại: 0843160561	Email: 6551071056@st.utc2.edu.vn
Sinh viên: Huỳnh Gia Bảo	Ký tên:
Điện thoại: 0365955132	Email: 6551071007@st.utc2.edu.vn

MỤC LỤC

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN i
NHẬN XÉT CỦA GIẢNG VIÊN PHẢN BIỆN ii
LỜI CẢM ƠN iii
DANH SÁCH HÌNH ẢNH..... xi
DANH SÁCH VIẾT TẮT xii
CHƯƠNG 1 MỞ ĐẦU 1
1.1 Giới thiệu đề tài 1
1.2 Mục tiêu đề tái 1
1.3 Mục tiêu nghiên cứu 2
1.4 Đối tượng nghiên cứu 2
1.5 Phạm vi nghiên cứu 2
1.6 Phương pháp nghiên cứu 3
1.7 Ý nghĩa thực tiễn của đề tài 3
1.8 Cấu trúc cuốn báo cáo đồ án 3
CHƯƠNG 2 CƠ SỞ LÝ THUYẾT VÀ THỰC NGHIỆM.....4
2.1 Tổng quan về lập trình web 4
2.2 Cơ sở lý thuyết.....5
2.2.1 HTML 5
2.2.2 CSS 14
2.2.3 JavaScript 19
2.2.4 API và dữ liệu JSON.....22
2.3 Thực nghiệm.....25
2.3.1 Mục tiêu thực nghiệm 25
2.3.2 Môi trường thực nghiệm 25
2.3.3 Quy trình thực nghiệm 25
2.3.4 Kết quả thực nghiệm 26
2.3.5 Đánh giá 26

CHƯƠNG 3: THIẾT KẾ VÀ TRIỂN KHAI WEBSITE 27
3.1 Thiết kế giao diện (UI Design) 27
3.2 Chức năng và tương tác (UX Design) 28
3.3 Mã nguồn và cấu trúc dự án 29
3.4 Triển khai website 31
3.4.1 Cấu trúc index.html 31
3.4.2 Cấu trúc intro.html 36
3.4.3 Cấu trúc Style.css 41
3.4.4 Cấu trúc Script.js 45
3.4.5 Cấu trúc Chart.js 50
CHƯƠNG 4: KIỂM THỬ, TỐI ƯU HÓA VÀ TRIỂN KHAI 56
4.1 Kiểm thử giao diện và chức năng 56
4.1.1. Kiểm tra từng chức năng 56
4.1.2. Kiểm thử responsive trên nhiều thiết bị 59
4.1.3. Kiểm tra tương thích trình duyệt 59
4.2. Tối ưu hóa hiệu suất 60
4.2.1. Tối ưu hình ảnh, CSS/JS 60
4.2.2. Đánh giá bằng Google PageSpeed / Lighthouse 60
4.2.3. SEO cơ bản: meta tags, alt text, semantic HTML 61
4.3. Triển khai website lên host và server 61
4.3.1. Giới thiệu hosting 61
4.3.2. Các bước triển khai website 62
4.3.3. Link truy cập website 62
4.3.4. Kiểm tra truy cập từ xa và độ ổn định 63
CHƯƠNG 5 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 64
5.1 Kết quả đạt được 64
5.2 Hạn chế 64
5.3 Hướng phát triển 65
TÀI LIỆU THAM KHẢO 66

PHỤ LỤC 67
Phụ lục A: Tóm tắt về web thông tin dự báo thời tiết 67
A.1 Ảnh chụp giao diện 67
A.2 Code chính 70
A.3 Bảng kiểm thử.....71
A.4 Link GitHub / Host 72

DANH SÁCH HÌNH ẢNH

Hình 2.1 Cấu trúc DOM 9
Hình 2.2 kiến trúc MVC/Client-Server 10
Hình 2.3 Cơ sở CSS 15
Hình 2.4 Nguyên tắc CSS 17
Hình 2.5 Cơ sở JavaScript 19
Hình 3.1Giao diện website hiển thị trên các thiết bị khác nhau (Responsive Design). . 28
Hình 3.2 Bố cục HTML 29
Hình 3.3 Giao diện sau khi thêm CSS 30
Hình 3.4 Cấu trúc thư mục 31
Hình A.1Ảnh giao diện hiển thị trên laptop 67
Hình A.2 Ảnh giao diện hiển thị trên mobile 68
Hình A.3 Sidebar hiển thị danh sách thành phố. 68
Hình A.4 Giao diện thông tin thời tiết và dự báo thời tiết 69

DANH SÁCH VIẾT TẮT

STT	Mô tả	Ý nghĩa	Ghi chú
1	HTML	HyperText Markup Language	
2	CSS	Cascading Style Sheets	
3	JS	JavaScript	
4	API	Application Programming Interface	OpenWeatherMap
5	DOM	Document Object Model	
6	UI	User Interface	
7	UX	User Experience	
8	AQI	Air Quality Index	

CHƯƠNG 1 MỞ ĐẦU

1.1 Giới thiệu đề tài

Trong bối cảnh biến đổi môi trường ngày càng nghiêm trọng, thời tiết trở nên thất thường và thay đổi nhanh chóng, nhu cầu cập nhật thông tin thời tiết một cách nhanh chóng, chính xác và thuận tiện trở nên vô cùng quan trọng. Các hoạt động sinh hoạt, lao động, du lịch, sản xuất nông nghiệp hay giao thông đều chịu ảnh hưởng trực tiếp từ điều kiện thời tiết, nên việc có một kênh thông tin đáng tin cậy là hết sức cần thiết.

Song song đó, công nghệ thông tin phát triển mạnh mẽ đã mở ra nhiều cơ hội để xây dựng các nền tảng trực tuyến giúp người dùng tiếp cận thông tin dễ dàng hơn. Tận dụng xu hướng này, việc phát triển một website dự báo thời tiết nhằm cung cấp dữ liệu thời gian thực, giao diện trực quan và thân thiện với người dùng là rất cần thiết và có tính ứng dụng cao.

Đề tài bọn em hướng đến việc xây dựng một trang web có khả năng hiển thị đầy đủ các thông tin thời tiết quan trọng như nhiệt độ, độ ẩm, tốc độ gió, tình trạng mưa, chất lượng không khí, cùng với dự báo theo giờ và theo ngày, nhằm hỗ trợ người dùng theo dõi tình hình thời tiết một cách tiện lợi và hiệu quả.

1.2 Mục tiêu đề tái

- Xây dựng một website cung cấp thông tin dự báo thời tiết nhanh chóng và chính xác.
- Áp dụng và thực hành các kiến thức về lập trình web như HTML, CSS, JavaScript, API.
- Hiển thị thông tin thời tiết theo vị trí người dùng hoặc vị trí tìm kiếm.
- Thiết kế giao diện trực quan, dễ sử dụng trên cả máy tính và điện thoại.
- Tối ưu trải nghiệm người dùng bằng cách cung cấp dữ liệu thời gian thực từ API thời tiết.

1.3 Mục tiêu nghiên cứu

- Tìm hiểu cách sử dụng API (OpenWeatherMap hoặc các dịch vụ tương tự) để lấy dữ liệu thời tiết.
- Nghiên cứu các kỹ thuật xử lý dữ liệu JSON và cập nhật giao diện động bằng JavaScript.
- Áp dụng các phương pháp thiết kế giao diện web hiện đại, có tính thẩm mỹ và dễ dùng.
- Nâng cao khả năng lập trình web, tích hợp API và xử lý dữ liệu thời gian thực.

1.4 Đối tượng nghiên cứu

- Các nguồn cung cấp dữ liệu thời tiết từ API quốc tế hoặc trong nước.
- Các phương pháp lập trình web front-end.
- Người dùng cần xem thông tin thời tiết nhanh chóng, chính xác và thuận tiện (học sinh, sinh viên, nhân viên văn phòng, người dân nói chung).

1.5 Phạm vi nghiên cứu

- Website chỉ tập trung vào tính năng **tra cứu và hiển thị thời tiết**, không bao gồm các chức năng nâng cao như bản đồ radar, cảnh báo thiên tai nâng cao, dự báo chuyên sâu.
- Dữ liệu thời tiết được lấy từ API miễn phí, do đó độ chính xác phụ thuộc vào nhà cung cấp.
- Website được xây dựng ở mức cơ bản đến trung bình (Frontend + API), chưa triển khai Backend hoặc cơ sở dữ liệu lớn.
- Phạm vi địa lý: toàn thế giới hoặc giới hạn theo từng API hỗ trợ.

1.6 Phương pháp nghiên cứu

- **Phương pháp thu thập dữ liệu:** Lấy dữ liệu thời tiết từ API như OpenWeatherMap.
- **Phương pháp phân tích – tổng hợp:** Phân tích dữ liệu JSON từ API, chọn lọc các thông tin cần thiết để hiển thị.
- **Phương pháp thực nghiệm:** Tự xây dựng, thử nghiệm và đánh giá giao diện web qua nhiều phiên bản.
- **Phương pháp so sánh:** Tham khảo các website dự báo thời tiết hiện có (AccuWeather, Weather.com) để cải tiến cách hiển thị thông tin.
- **Phương pháp lập trình:** Sử dụng HTML, CSS, JavaScript, thư viện fetch API để tương tác dữ liệu.

1.7 Ý nghĩa thực tiễn của đề tài

- Giúp người dùng dễ dàng tra cứu thông tin thời tiết phục vụ cho nhu cầu di chuyển, học tập, lao động.
- Cung cấp một công cụ trực quan cho sinh viên thực hành kỹ năng lập trình web và làm quen với APIs.
- Tăng khả năng ứng dụng thực tế của kiến thức trong nhà trường vào sản phẩm cụ thể.
- Có thể mở rộng thành ứng dụng di động hoặc website nâng cao trong tương lai.

1.8 Cấu trúc cuốn báo cáo đồ án

Báo cáo được chia làm 5 chương:

- Chương 1: Mở đầu
- Chương 2: Cơ sở lý thuyết và thực nghiệm
- Chương 3: Thiết kế và triển khai website
- Chương 4: Kết thử, tối ưu hóa và triển khai
- Chương 5: Kết luận và hướng phát triển

CHƯƠNG 2 CƠ SỞ LÝ THUYẾT VÀ THỰC NGHIỆM

Chương này trình bày cơ sở lý thuyết và thực nghiệm làm nền tảng cho nghiên cứu. Phần lý thuyết tổng hợp các khái niệm, nguyên lý và kết quả từ những công trình trước đây. Tiếp đó, phần thực nghiệm giới thiệu phương pháp, quy trình và điều kiện tiến hành nghiên cứu. Sự kết hợp giữa lý thuyết và thực nghiệm giúp đảm bảo tính khoa học, đồng thời tạo cơ sở cho việc phân tích và thảo luận ở các chương sau.

2.1 Tổng quan về lập trình web

Lập trình web là quá trình xây dựng và phát triển các ứng dụng hoặc trang web hoạt động trên môi trường Internet. Đây là một lĩnh vực quan trọng trong công nghệ thông tin, giữ vai trò kết nối người dùng với các dịch vụ trực tuyến thông qua trình duyệt web hoặc thiết bị di động.

Một trang web hoàn chỉnh thường được phát triển dựa trên hai phần chính: front-end và back-end. Phần front-end bao gồm giao diện người dùng, cách bố cục, màu sắc và cách thông tin được trình bày. Các công nghệ phổ biến trong front-end gồm HTML, CSS và JavaScript. Trong khi đó, phần back-end chịu trách nhiệm xử lý dữ liệu, tương tác với cơ sở dữ liệu và đảm bảo các chức năng hoạt động trơn tru; các ngôn ngữ thường được sử dụng là PHP, Node.js, Python, hoặc Java.

Bên cạnh đó, lập trình web hiện đại còn sử dụng các framework và thư viện như React, Angular, Laravel, Django… giúp giảm thời gian phát triển và tăng tính bảo mật, khả năng mở rộng của hệ thống. Ngoài ra, các công nghệ hỗ trợ như API, cơ sở dữ liệu (MySQL, MongoDB), điện toán đám mây, và bảo mật web cũng đóng vai trò quan trọng.

Nhờ sự phát triển nhanh chóng của Internet, lập trình web ngày nay không chỉ dừng lại ở việc hiển thị thông tin mà còn hỗ trợ các chức năng phức tạp như thương mại điện tử, bản đồ, chat thời gian thực, dự báo thời tiết, và nhiều ứng dụng tương tác khác. Điều này khiến lập trình web trở thành một kỹ năng thiết yếu trong lĩnh vực công nghệ và có tính ứng dụng cao trong thực tế.

2.2 Cơ sở lý thuyết

2.2.1 HTML

2.2.1.1 Giới thiệu chung

HTML (HyperText Markup Language) là ngôn ngữ đánh dấu chuẩn được sử dụng để tạo và cấu trúc nội dung trên World Wide Web. HTML không phải là ngôn ngữ lập trình mà là markup language (ngôn ngữ đánh dấu), sử dụng các thẻ (tags) để định nghĩa cấu trúc và ý nghĩa của nội dung web.

Đặc điểm chính của HTML:

- **Declarative Language**: Khai báo "cái gì" thay vì "làm thế nào".
- **Platform-independent**: Hoạt động trên mọi hệ điều hành và trình duyệt.
- **Human-readable**: Dễ đọc và viết bằng text editor thông thường.
- **Extensible**: Có thể mở rộng với custom attributes và components.

Trong dự án này, HTML đóng vai trò nền tảng, xây dựng khung sườn cho toàn bộ giao diện ứng dụng dự báo thời tiết. Ứng dụng được thiết kế với 2 trang HTML chính: **index.html** (trang chính hiển thị thông tin thời tiết) và **intro.html** (trang giới thiệu thành viên nhóm).

2.2.1.2 Cơ sở dữ liệu cho HTML

❖ **Cấu trúc cơ bản của HTML**

Một tài liệu HTML chuẩn có cấu trúc như sau:

```html
<!DOCTYPE html>
<html lang="vi">
  <head>
    <meta charset="UTF-8">
    <title>Tiêu đề</title>
  </head>
  <body>
    <!-- Nội dung hiển thị -->
  </body>
</html>
```

Giải thích các thành phần:

1. **DOCTYPE declaration**: <!DOCTYPE html>
   - Khai báo phiên bản HTML (HTML5)
   - Giúp trình duyệt render đúng chuẩn
   - Bắt buộc phải có ở đầu file
2. **HTML element**: <html lang="vi">
   - Root element (gốc) của document
   - Attribute lang xác định ngôn ngữ (vi = Vietnamese)
   - Tất cả elements khác phải nằm trong <html>
3. **Head section**: <head>
   - Chứa metadata (siêu dữ liệu)
   - Không hiển thị trực tiếp trên trang
   - Bao gồm: title, meta tags, links, scripts
4. **Body section**: <body>
   - Chứa nội dung hiển thị
   - Chỉ có 1 body trong document
   - Tất cả nội dung người dùng thấy nằm trong body

❖ **Elements và Attrributes**

HTML Element gồm 3 phần:

```
<tagname attribute="value">Content</tagname>
```

- **Opening tag:** <tagname> - Thẻ mở.
- **Content:** Nội dung bên trong.
- **Closing tag:** </tagname> - Thẻ đóng.

Một số thẻ tự đóng (self-closing):

```
<img src="image.jpg" />
<input type="text" />
<meta charset="UTF-8" />
```

HTML Attributes cung cấp thông tin bổ sung:

- Luôn nằm trong opening tag.
- Format: ‘name=”value”’.
- Ví dụ: `id=”cityName”`, `class=”card”`, `href=”index.html”`.

Phân loại attributes:

1. Global attributes: Dùng cho mọi element
    - `id`: Định danh duy nhất.
    - `class`: Phân loại, nhóm.
    - `style`: Inline CSS.
    - `title`: Tooltip text.
    - `data-*`: Custom data attributes.
2. Specific attributes: chỉ dùng cho element cụ thể
    - `src` (img, script): Nguồn file.
    - `href` (a): Link đích.
    - `type` (input): Kiểu input.
    - `placeholder` (input): Gợi ý nhập liệu.

❖ **HTML5 Semantic Elements**

HTML5 giới thiệu các semantic elements giúp cấu trúc trang web rõ ràng hơn:

<table>
<tr><td colspan="2"><header>
(Logo, Navigation, Search)</td></tr>
<tr><td><aside>
(Sidebar)</td><td><main>
<section>
<article>
<section></td></tr>
<tr><td colspan="2"><footer>
(Copyright, Links)</td></tr>
</table>

- **Các thẻ semantic chính:**

Thẻ	Mục đích	Ví dụ
<header>	Phần đầu trang/section.	Logo, menu navigation.
<nav>	Khối điều hướng.	Menu bar, breadcumbs.
<main>	Nội dung chính.	Main content area.
<aside>	Nội dung phụ.	Sidebar, related links.
<section>	Phần nội dung độc lập.	Weather card, forecast.
<article>	Nội dung độc lập, tái sử dụng.	Blog post, news item.
<footer>	Phần chân trang/section.	Copyright, contact info.

- **Lợi ích của semantic HTML:**

SEO (Search Engine Optimization) đóng vai trò quan trọng khi các công cụ tìm kiếm thường ưu tiên nội dung nằm trong thẻ . Việc sử dụng các thẻ ngữ nghĩa giúp công cụ hiểu rõ cấu trúc trang, từ đó cải thiện khả năng hiển thị và mang lại thứ hạng tốt hơn trên kết quả tìm kiếm.

Accessibility cũng chịu ảnh hưởng lớn bởi việc áp dụng thẻ ngữ nghĩa. Người dùng sử dụng trình đọc màn hình có thể dễ dàng nhận diện các mốc giới hạn nội dung, đồng thời người khuyết tật có thể tiếp cận thông tin thuận tiện hơn. Nội dung chính cũng có thể được bỏ qua hoặc truy cập nhanh thông qua tiêu đề, giúp trải nghiệm trở nên thân thiện hơn.

Maintainability hay khả năng bảo trì được nâng cao nhờ code mang tính tự giải thích, dễ đọc và có cấu trúc rõ ràng. Điều này không chỉ giúp lập trình viên mới nhanh chóng hiểu hệ thống mà còn tạo điều kiện cho sự cộng tác hiệu quả hơn giữa các thành viên trong nhóm.

Future-proof là một lợi ích quan trọng khi sử dụng chuẩn nền web APIs, vốn được tối ưu hóa cho các trình duyệt hiện đại. Nhờ đó, trang web dễ dàng thích nghi với các công nghệ hỗ trợ trong tương lai, đảm bảo tính bền vững và khả năng mở rộng lâu dài.

❖ **DOM (Document Object Model)**

- **Khái niệm:** DOM là giao diện lập trình cho HTML documents. Nó biểu diễn trang web dưới dạng cây cấu trúc (tree structure), mỗi node là một object đại diện cho một phần của document.
- **Cấu trúc DOM tree:**

```
Document
└── html
    ├── head
    │   ├── meta
    │   ├── title
    │   └── link
    └── body
        ├── header
        ├── main
        │   ├── section
        │   └── section
        └── footer
```

Hình 2.1 Cấu trúc DOM

Các loại node chính:

1. **Element nodes (Nút phần tử):** Đại diện cho HTML elements (, <p>).
2. **Text nodes (Nút văn bản):** Đại diện cho tex content.
3. **Attribute nodes (Nút thuộc tính):** Đại diện cho các thuộc tính (id, class).
4. **Comment nodes (Nút bình luận):** Đại diện cho comments (<!-- -->).

DOM APIs cho phép:

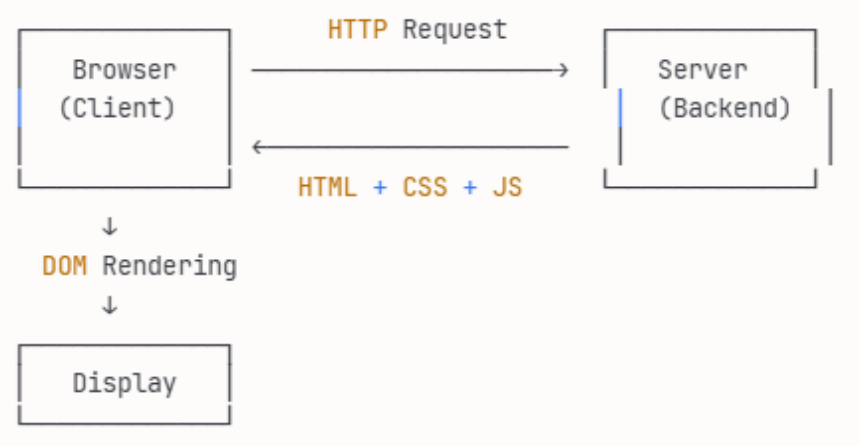
- Truy xuất elements: document.getElementById(), querySelector().
- Thay đổi nội dung: innerHTML, textContent.
- Thay đổi attributes: setAttribute(), getAttribute().
- Thêm/xóa elements: appendChild(), removeChild().
- Lắng nghe sự kiện: addEventListener().

Ví dụ:

```
// JavaScript có thể thao tác DOM
const temp = document.getElementById('temp');
temp.textContent = '28°C'; // Thay đổi nội dung
```

❖ **HTML trong kiến trúc MVC/Client-Server**

- **Vai trò của HTML trong web application:**



Hình 2.2 kiến trúc MVC/Client-Server

HTML trong dự án:

- **Static Structure (Cấu trúc tĩnh):** HTML cung cấp cấu trúc tĩnh ban đầu.
- **Dynamic Content (Nội dung động):** JavaScript điền dữ liệu từ API vào DOM.

- **Separation (Tách biệt):** HTML(Cấu trúc) → CSS (Trình bày) → JS (Xử lý).

❖ **Áp dụng trong dự án**

Trong dự án "Ứng dụng Dự báo Thời tiết Việt Nam", HTML đóng vai trò là nền tảng với 1 file chính là index.html và 1 file phụ là intro.html.

1. **index.html** – Trang chính:
   - Cấu trúc phức tạp với layout 2 cột (sidebar + main).
   - Sử dụng nhiều semantic elements (header, main, aside, section, footer).
   - Tích hợp thư viện (jQuery, Chart.js).
   - Chuẩn bị thùng chứa cho nội dung động.
   - Tổng số elements: ~40 + HTML elements.
2. **intro.html** – Trang giới thiệu:
   - Cấu trúc đơn giản, tập trung vào hiển thị thông tin.
   - Sử dụng table structure cho dữ liệu thành viên.
   - CSS độc lập – Viết thẳng vào trong thẻ <style> mà không dùng thư mục ngoài.
   - Không dùng đến JavaScript.
   - Tổng số elements: ~15 HTML elements.

Mục tiêu thiết kế:

- Tuân thủ với tiêu chuẩn HTML5.
- Dễ tiếp cận với người dùng.
- Đáp ứng được điều kiện xem về thời tiết.
- Giao diện thân thiện và ưa nhìn.
- Có thể duy trì ổn định và phát triển trang web.

2.2.1.3 Nguyên tắc thiết kế HTML trong dự án

➢ **Sử dụng Semantic HTML**

Dự án ưu tiên các thẻ semantic (có ý nghĩa) thay vì dùng :

- <header> : Phần đầu trang.
- <main> : Nội dung chính.
- <aside> : Nội dung phụ (sidebar).
- <section> : Các phần nội dung độc lập.
- <footer> : Phần cuối trang.

Lợi ích:

- Giúp công cụ tìm kiếm (Google, Bing) hiểu cấu trúc trang web tốt hơn (SEO).
- Công nghệ hỗ trợ người khuyết tật (screen reader) đọc nội dung chính xác hơn.
- Code dễ đọc, dễ duy trì hơn.

➢ **ID và Class hợp lý**

Mọi yếu tố quan trọng đều có ID hoặc class rõ ràng:

- **ID:** Dùng cho các yếu tố duy nhất cần JavaScript thao tác trực tiếp (#cityName, #temp, #searchInput).
- **Class:** Dùng cho styling và các nhóm yếu tố tương tự như (.card, .meta-item, .btn-ghost).

Quy ước đặt tên:

  - ID dùng camelCase: cityName, searchInput.
  - Class dùng kebab-case: current-card, meta-item, btn-ghost.

- **Accessibility (Khả năng tiếp cận)**

  Dự án chú trọng accessibility thông qua:

  - Thuộc tính aria-label: Mô tả cho screen reader ("Danh sách tỉnh thành", "Tìm tỉnh thành").
  - Thuộc tính role: Định nghĩa vai trò của element (role="list").
  - Thuộc tính title: Tooltip khi hover chuột ("Vị trí của tôi", "Làm mới dữ liệu").
  - Alt text ngâm định cho icon (sử dụng emoji có ý nghĩa).

- **Progressive Enhancement (Cải tiến dần)**

  HTML được thiết kế để:

  1. Hiển thị cấu trúc cơ bản ngay cả khi CSS/JavaScript bị tắt.
  2. Các giá trị placeholder ("—", "--") đảm bảo layout không bị vỡ trước khi JavaScript load dữ liệu.
  3. Phân tách rõ ràng: HTML lo cấu trúc, CSS lo giao diện, JavaScript lo tương tác.

- **Mobile-First Approach**

  Meta viewport width=device-width, initial-scale=1 đảm bảo:

  - Trang web tự động co giãn theo kích thước màn hình.
  - Không bị zoom out trên mobile.
  - Sẵn sàng cho responsive design (sẽ được CSS xử lý).

2.2.1.4 Ưu điểm của cấu trúc HTML

1. **Rõ ràng và logic:** Cấu trúc theo layout 2 cột, mỗi phần có vai trò riêng biệt.
2. **Dễ bảo trì:** ID/Class có tên mang tính mô tae, dễ tìm kiếm và chỉnh sửa.
3. **Tương thích tốt:** Sử dụng thẻ HTML5 chuẩn, hoạt động trên mọi trình duyệt hiện đại.

4. **Tách biệt concerns:** HTML chỉ tập trung vào cấu trúc, không lẫn lộn với CSS hay JavaScript.
5. **Sẵn sàng cho nội dung động:** Các thùng rỗng như #citiesList, #forecast dễ dàng để JavaScript điền nội dung.
6. **Accessibility:** Tuân thủ các chuẩn WCAF cơ bản, thân thiện với người khuyết tật.

2.2.1.5 Kết luận

Phần HTML trong dự án đã xây dựng được một nền tảng vững chắc cho ứng dụng dự báo thời tiết. Với 2 file HTML được thiết kế hợp lý, rõ ràng và tuân thủ các chuẩn web hiện đại, cấu trúc này tạo điều kiện thuận lợi cho việc styling bằng CSS và thêm tính tương tác bằng JavaScript. Đặc biệt, việc sử dụng semantic HTML và các thuộc tính accessibility không chỉ giúp ứng dụng thân thiện với người dùng mà còn tối ưu cho SEO và các công nghệ hỗ trợ.

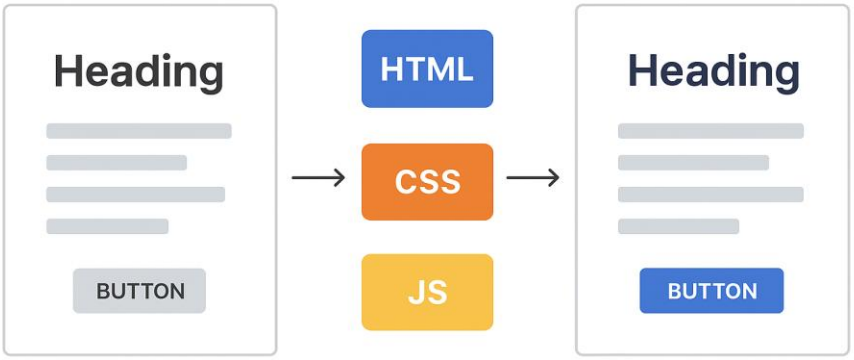
2.2.2 CSS

2.2.2.1 Cơ sở lý thuyết CSS

a. Khái niệm và vai trò

CSS (Cascading Style Sheets) là ngôn ngữ định kiểu cho các tài liệu HTML, cho phép tách biệt phần nội dung và phần trình bày. Nếu HTML đóng vai trò cấu trúc, thì CSS chính là lớp áo khoác giúp giao diện trở nên trực quan, đẹp mắt và dễ sử dụng. CSS giúp kiểm soát màu sắc, bố cục, phông chữ, hiệu ứng và khả năng tương tác của trang web.

Trong phát triển web hiện đại, CSS không chỉ đơn thuần là công cụ trang trí mà còn là yếu tố quyết định trải nghiệm người dùng (UX) và giao diện người dùng (UI). Một ứng dụng web có thể hoạt động tốt về mặt chức năng, nhưng nếu thiếu CSS, giao diện sẽ khô khan, khó tiếp cận và thiếu tính chuyên nghiệp.



Hình 2.3 Cơ sở CSS

b. Nguyên tắc cơ bản

CSS hoạt động dựa trên ba nguyên tắc chính:

- Cascade (xếp chồng): Khi nhiều quy tắc cùng áp dụng cho một phần tử, trình duyệt sẽ quyết định quy tắc nào được ưu tiên dựa trên thứ tự khai báo và mức độ cụ thể.
- Inheritance (kế thừa): Một số thuộc tính như màu chữ, font chữ có thể được kế thừa từ phần tử cha xuống phần tử con.
- Specificity (độ ưu tiên): Mức độ cụ thể của bộ chọn quyết định quy tắc nào được áp dụng. Ví dụ, bộ chọn #id có độ ưu tiên cao hơn .class và element.
- Ngoài ra, CSS còn hỗ trợ pseudo-class (:hover, :focus) và pseudo-element (::before, ::after) để tạo hiệu ứng động và bổ sung nội dung ảo.

c. Các thành phần chính

- Bộ chọn (Selectors)

Bộ chọn cho phép xác định phần tử HTML cần áp dụng kiểu. Trong code nhóm, ta thấy nhiều loại bộ chọn:

  - Bộ chọn thẻ: body, main, footer.
  - Bộ chọn lớp: .app, .sidebar, .search.
  - Bộ chọn id: #forecast.
  - Bộ chọn trạng thái: .city-item:hover, .city-item.active.

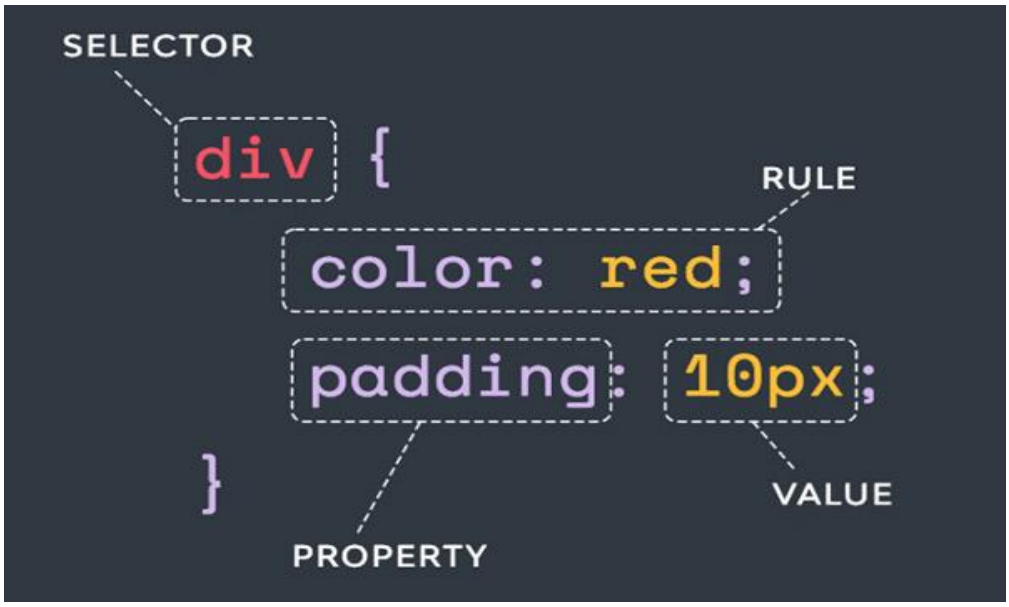
- Thuộc tính và giá trị (Properties & Values)

Mỗi bộ chọn đi kèm một tập hợp thuộc tính mô tả cách hiển thị. Ví dụ:

  - color: #111; để định màu chữ.
  - background-image: url(...); để hiển thị hình nền.
  - display: flex; để bố trí phần tử theo Flexbox.
  - transition: background-image 0.8s ease; để tạo hiệu ứng chuyển đổi mượt mà.

- Biến CSS (Custom Properties)

Trong :root, chúng em khai báo các biến như --glass-bg, --accent. Đây là cách quản lý giá trị toàn cục, giúp dễ dàng tái sử dụng và thay đổi giao diện mà không cần chỉnh sửa nhiều nơi.



Hình 2.4 Nguyên tắc CSS

- Responsive Design

CSS hỗ trợ media queries để giao diện thích ứng với nhiều kích thước màn hình.

```
@media (max-width: 880px) {
.app { flex-direction: column; }
.sidebar { width: 100%; display: flex; flex-direction: row; }
.chart-body { height: 240px; }
}
```

Điều này đảm bảo ứng dụng hiển thị tốt trên cả máy tính và thiết bị di động.

- Hiệu ứng và trực quan

CSS cung cấp các thuộc tính như transition, transform, box-shadow, backdrop-filter để tạo hiệu ứng động. Ví dụ:

  - .btn-intro:hover { transform: scale(1.02); } tạo hiệu ứng phóng to nhẹ khi rê chuột.
  - .cities .city-item.active { box-shadow: 0 6px 18px rgba(0,0,0,0.12); } làm nổi bật thành phố đang chọn.

2.2.2.2 Ứng dụng CSS trong báo cáo

a. Quản lý giao diện tổng thể

- Reset mặc định với * và body để loại bỏ margin/padding, thiết lập font chữ và màu nền.
- .app sử dụng Flexbox để chia giao diện thành thanh bên (.sidebar) và nội dung chính (main.main).

b. Thiết kế bố cục

- .meta-grid và .forecast-list dùng Grid để hiển thị thông tin thời tiết và dự báo theo dạng lưới.
- forecast-item hiển thị dự báo từng ngày với hình ảnh biểu tượng thời tiết (img.f-icon).

c. Trang trí và hiệu ứng

- Hình nền của main.main thay đổi theo điều kiện thời tiết (.weather-sunny, .weather-rain, .weather-cloud, .weather-snow, .weather-mist).
- Các nút (.btn-intro, .btn-ghost) có hiệu ứng hover, bo góc, màu sắc nổi bật và hiệu ứng mờ nền (backdrop-filter).
- Danh sách thành phố (.cities .city-item) có trạng thái hover, focus và active để người dùng dễ nhận biết lựa chọn.

d. Responsive design

Media query ở kích thước nhỏ hơn 880px giúp bố cục chuyển sang dạng cột, sidebar hiển thị ngang, và biểu đồ thu nhỏ chiều cao.

e. Thành phần cụ thể

- .card và .current-card sử dụng nền trong suốt, bo góc và shadow để tạo cảm giác nổi khối.
- .chart-body định nghĩa vùng hiển thị biểu đồ nhiệt độ với kích thước linh hoạt.

2.2.3 JavaScript

2.2.3.1 Khái niệm và vai trò

JavaScript là ngôn ngữ lập trình phía client dùng để xử lý các thao tác động trên trang web. Nếu HTML tạo cấu trúc và CSS tạo giao diện, thì JavaScript là phần "logic" giúp trang web hoạt động, tương tác và thay đổi theo hành vi người dùng.

JavaScript cho phép:

- Lấy dữ liệu từ API
- Xử lý, tính toán, cập nhật nội dung mà không cần tải lại trang
- Tạo hiệu ứng, thao tác DOM
- Kiểm soát biểu đồ, bản đồ, hoạt ảnh...

Trong một ứng dụng web hiện đại, JavaScript là yếu tố cốt lõi quyết định khả năng tương tác (interactivity) và tính thông minh của hệ thống.

Hình 2.5 Cơ sở JavaScript

2.2.3.2 Các khái niệm quan trọng

a) DOM Manipulation

JavaScript có thể truy cập và thay đổi phần tử HTML thông qua DOM:

- Thêm, sửa, xóa nội dung
- Thay đổi class, style
- Gán sự kiện (onclick, input…)

b) Event Handling

JavaScript bắt các hành động của người dùng:

- Nhập từ khóa
- Nhấn nút
- Chọn thành phố
- Cuộn trang

  Từ đó cập nhật giao diện ngay lập tức.

c) API & Fetch

JavaScript có thể gọi API bằng fetch():

- Gửi yêu cầu đến server
- Nhận dữ liệu JSON
- Xử lý và hiển thị thông tin thời tiết

Đây là phần quan trọng nhất trong các ứng dụng thời tiết.

d) Xử lý dữ liệu (Array, Object)

JS cung cấp nhiều hàm mạnh để xử lý dữ liệu:

- map() → chuyển đổi danh sách
- filter() → lọc dữ liệu
- reduce() → tính toán tổng, trung bình
- Truy cập object để lấy giá trị cần thiết từ API

e) Tạo biểu đồ

Khi kết hợp với thư viện như **Chart.js**, JavaScript có thể vẽ:

- Biểu đồ đường
- Biểu đồ cột
- Biểu đồ nhiệt độ theo ngày

2.2.3.3 Vai trò của JS trong dự án

- Lấy dữ liệu thời tiết từ API OpenWeather
- Xử lý và chuẩn hóa dữ liệu
- Hiển thị thông tin thời tiết hiện tại
- Hiển thị dự báo 3–7 ngày
- Cập nhật giao diện theo điều kiện thời tiết
- Vẽ biểu đồ nhiệt độ
- Xử lý sự kiện khi người dùng nhập thành phố hoặc chọn từ danh sách

2.2.4 API và dữ liệu JSON

2.2.4.1 API

a) Mục đích của việc nhúng API vào website

Trong đề tài *"Xây dựng website Dự báo thời tiết Việt Nam"*, mục tiêu chính của nhóm là xây dựng một website không chỉ có giao diện đẹp mà còn có khả năng hiển thị dữ liệu thời tiết theo thời gian thực. Để thực hiện được điều đó, nhóm đã tiến hành nhúng một dịch vụ dữ liệu thời tiết bên ngoài vào website thông qua API.

Sau khi tìm hiểu và so sánh nhiều dịch vụ khác nhau, nhóm quyết định lựa chọn OpenWeatherMap API vì những lý do sau:

- Đây là một API phổ biến, được sử dụng rộng rãi trên toàn thế giới
- Có phiên bản miễn phí, phù hợp với điều kiện của sinh viên
- Dữ liệu thời tiết chính xác, cập nhật liên tục
- Dễ tích hợp với website bằng JavaScript
- Có đầy đủ tài liệu hướng dẫn (documentation)

Việc nhúng API giúp website chuyển từ một trang web tĩnh (chỉ hiển thị giao diện) trở thành một ứng dụng web động, có khả năng cập nhật dữ liệu thời tiết trực tuyến và hiển thị cho người dùng theo thời gian thực.

b) Quá trình nhúng API vào code website (Script.js)

Việc nhúng API được thực hiện chủ yếu bằng ngôn ngữ JavaScript trong file Script.js. Nhóm sử dụng phương thức fetch() để gửi yêu cầu lấy dữ liệu và xử lý dữ liệu trả về từ OpenWeatherMap.

❖ **Nhúng API lấy thời tiết hiện tại**

Để lấy thông tin thời tiết hiện tại của một thành phố, nhóm sử dụng đường dẫn sau:

https://api.openweathermap.org/data/2.5/weather?q={city},VN&appid={API_KEY}&units=metric&lang=viTrong đó:

- {city}: Tên thành phố người dùng nhập
- {API_KEY}: Mã khóa do OpenWeatherMap cung cấp

- units=metric: Hiển thị nhiệt độ theo đơn vị độ C
- lang=vi: Hiển thị mô tả thời tiết bằng tiếng Việt

Khi người dùng nhập tên thành phố hoặc chọn thành phố từ danh sách, website sẽ tự động tạo đường dẫn API phù hợp và gửi yêu cầu bằng fetch().

Sau khi nhận dữ liệu trả về dưới dạng JSON, nhóm tiến hành trích xuất các thông tin quan trọng như:

- Nhiệt độ hiện tại: data.main.temp
- Nhiệt độ cảm nhận: data.main.feels_like
- Độ ẩm: data.main.humidity
- Tốc độ gió: data.wind.speed
- Mô tả thời tiết: data.weather[0].description
- Icon thời tiết: data.weather[0].icon

Những dữ liệu này sau đó được đưa lên giao diện thông qua các phần tử HTML như:

- Thẻ hiển thị nhiệt độ
- Thẻ hiển thị độ ẩm
- Thẻ hiển thị trạng thái thời tiết
- Biểu tượng thời tiết

❖ **Nhúng API lấy dự báo thời tiết 5 ngày**

Ngoài thời tiết hiện tại, nhóm còn sử dụng thêm một API khác của OpenWeatherMap để lấy dự báo thời tiết trong 5 ngày tiếp theo:

https://api.openweathermap.org/data/2.5/forecast?q={city},VN&appid={API_KEY}&units=metric&lang=vi

Dữ liệu trả về bao gồm thông tin dự báo theo từng mốc thời gian 3 giờ/lần trong 5 ngày. Từ dữ liệu này, nhóm đã:

- Lọc ra mỗi ngày 1 mốc thời gian tiêu biểu (12:00 trưa)
- Lấy nhiệt độ, trạng thái thời tiết và icon
- Hiển thị dự báo trong 5 ngày tiếp theo dưới dạng:

- Các khối thông tin 5 ngày
- Biểu đồ thể hiện nhiệt độ (sử dụng Chart.js)

- Điều này giúp người dùng không chỉ biết thời tiết hiện tại mà còn có thể theo dõi xu hướng thay đổi của thời tiết trong những ngày sắp tới.

c) Xử lý dữ liệu đầu vào – Tên thành phố

Một khó khăn trong quá trình nhúng API là việc người dùng có thể nhập tên thành phố theo nhiều cách khác nhau, ví dụ:

- Có dấu: "Hà Nội", "Đà Nẵng"
- Không dấu: "Ha Noi", "Da Nang"
- Viết tắt: "HCM", "HN"
- Tên cũ: "Sài Gòn"

Nhóm đã xây dựng các hàm xử lý và bảng ánh xạ để chuẩn hóa tên thành phố trước khi gửi đến API, tiêu biểu như:

- Hàm normalizeCityInput()
- Hàm getEnglishCityKey()
- Bảng VN_CITY_NAME_MAP

Nhờ đó, website có thể hiểu và chuyển đổi chính xác các cách nhập trên thành tên chuẩn tiếng Anh, giúp API nhận diện đúng thành phố.

Ví dụ:

- "tp hcm", "sài gòn" → "Ho Chi Minh"
- "hà nội" → "Hanoi"
- "đà nẵng" → "Da Nang"

Điều này giúp website thân thiện với người dùng Việt Nam và hạn chế lỗi "Không tìm thấy thành phố".

2.2.4.2 Dữ liệu JSON

2.3 Thực nghiệm

2.3.1 Mục tiêu thực nghiệm

Thực nghiệm được tiến hành nhằm:

Việc kiểm chứng tính đúng đắn của mã nguồn HTML, CSS và JavaScript đã xây dựng là bước quan trọng để đảm bảo ứng dụng hoạt động ổn định và không phát sinh lỗi trong quá trình triển khai. Song song đó, cần đánh giá khả năng hiển thị giao diện (UI) cũng như mức độ tương tác người dùng (UX) nhằm xác định xem trải nghiệm có trực quan, dễ sử dụng và đáp ứng nhu cầu thực tế hay không. Cuối cùng, việc xác định hiệu quả của thiết kế responsive khi ứng dụng chạy trên nhiều thiết bị khác nhau giúp bảo đảm tính linh hoạt, khả năng thích ứng và sự nhất quán của giao diện, từ đó nâng cao chất lượng tổng thể của sản phẩm.

2.3.2 Môi trường thực nghiệm

- **Phần mềm:** Visual Studio Code, trình duyệt Google Chrome, Mozilla Firefox.
- **Ngôn ngữ:** HTML5, CSS3, JavaScript ES6.
- **Thư viện hỗ trợ:** Chart.js để vẽ biểu đồ nhiệt độ.
- **Thiết bị kiểm thử:**
  - Máy tính để bàn với màn hình 24 inch.
  - Laptop 14 inch.
  - Điện thoại thông minh (Android, iOS).

2.3.3 Quy trình thực nghiệm

1. **Khởi tạo dự án:** Tạo cấu trúc thư mục gồm index.html, style.css, script.js và thư mục assets chứa hình ảnh.
2. **Triển khai giao diện:**
   - Sidebar hiển thị logo, ô tìm kiếm và danh sách thành phố.
   - Main hiển thị thông tin thời tiết hiện tại, dự báo, biểu đồ.
   - Footer hiển thị thông tin bản quyền và thời gian cập nhật.

3. **Áp dụng CSS:**
   - Kiểm tra hiệu ứng hover, transition, backdrop-filter.
   - Thay đổi hình nền theo điều kiện thời tiết (nắng, mưa, mây, tuyết, sương mù).
   - Kiểm tra responsive bằng cách thu nhỏ màn hình trình duyệt.
4. **Tích hợp JavaScript:**
   - Gọi API thời tiết để lấy dữ liệu theo thành phố.
   - Hiển thị dữ liệu lên giao diện và biểu đồ.
   - Kiểm tra chức năng chuyển đổi đơn vị nhiệt độ (°C ↔ °F).
5. **Kiểm thử người dùng:**
   - Nhập tên thành phố để tìm kiếm.
   - Chọn thành phố từ danh sách sidebar.
   - Quan sát sự thay đổi giao diện và dữ liệu.

2.3.4 Kết quả thực nghiệm

- **Giao diện:** Hiển thị đúng bố cục, màu sắc hài hòa, hình nền thay đổi theo thời tiết.
- **Tương tác:** Người dùng có thể tìm kiếm, chọn thành phố, chuyển đổi đơn vị nhiệt độ dễ dàng.
- **Responsive:** Ứng dụng hoạt động tốt trên cả máy tính và điện thoại, bố cục thay đổi hợp lý.
- **Biểu đồ:** Hiển thị trực quan sự thay đổi nhiệt độ, hỗ trợ người dùng theo dõi xu hướng.

2.3.5 Đánh giá

Thực nghiệm cho thấy ứng dụng đạt yêu cầu về cả giao diện và chức năng. CSS đóng vai trò quan trọng trong việc tạo hiệu ứng trực quan và responsive, trong khi JavaScript đảm bảo tính năng động và tương tác. Kết quả chứng minh sự kết hợp giữa lý thuyết và thực hành đã mang lại một sản phẩm hoàn chỉnh, có giá trị ứng dụng thực tế.

CHƯƠNG 3: THIẾT KẾ VÀ TRIỂN KHAI WEBSITE

3.1 Thiết kế giao diện (UI Design)

Giao diện người dùng (UI – User Interface) là yếu tố trực quan giúp người dùng tương tác với hệ thống. Trong dự án này, UI được thiết kế theo hướng **Responsive Web Design**, tức là giao diện có khả năng tự động thích ứng với nhiều kích thước màn hình khác nhau (máy tính, máy tính bảng, điện thoại di động).

- **Nguyên tắc thiết kế:**
  - Sử dụng Flexbox và CSS Grid để bố trí các thành phần linh hoạt.
  - Áp dụng media queries để thay đổi cách hiển thị khi màn hình nhỏ hơn 880px: sidebar chuyển sang dạng ngang, nội dung chính thu gọn, biểu đồ giảm chiều cao.
  - Màu sắc và hình nền thay đổi theo điều kiện thời tiết, tạo sự trực quan và sinh động.
- **Thành phần giao diện chính:**
  - Thanh bên (Sidebar): chứa logo, ô tìm kiếm, danh sách thành phố.
  - Nội dung chính (Main): hiển thị thông tin thời tiết hiện tại, dự báo, biểu đồ nhiệt độ.
  - Thẻ thông tin (Cards): sử dụng nền trong suốt, bo góc và shadow để tạo cảm giác nổi khối.
  - Nút và biểu tượng: thiết kế với hiệu ứng hover, màu sắc nổi bật, dễ nhận biết.

Nhờ thiết kế responsive, ứng dụng đảm bảo tính nhất quán và dễ sử dụng trên nhiều thiết bị, nâng cao trải nghiệm người dùng.

Hình 3.1Giao diện website hiển thị trên các thiết bị khác nhau (Responsive Design).

3.2 Chức năng và tương tác (UX Design)

Trải nghiệm người dùng (UX – User Experience) tập trung vào cách người dùng sử dụng và cảm nhận ứng dụng. Trong dự án này, UX được chú trọng thông qua các chức năng và tương tác sau:

- Tìm kiếm thành phố: Người dùng nhập tên thành phố vào ô tìm kiếm, hệ thống trả về thông tin thời tiết tương ứng.
- Danh sách thành phố: Cho phép chọn nhanh các thành phố đã lưu, với trạng thái active để dễ nhận biết.
- Hiển thị thông tin thời tiết:
    - Nhiệt độ hiện tại, cảm giác thực tế (feels like).
    - Các thông số bổ sung: độ ẩm, tốc độ gió, chỉ số AQI.
    - Biểu tượng thời tiết trực quan (nắng, mưa, mây, tuyết, sương mù).
- Dự báo nhiều ngày: Hiển thị danh sách dự báo theo dạng lưới, mỗi ô có nhiệt độ và biểu tượng thời tiết.
- Biểu đồ nhiệt độ: Cung cấp cái nhìn trực quan về sự thay đổi nhiệt độ theo thời

gian.

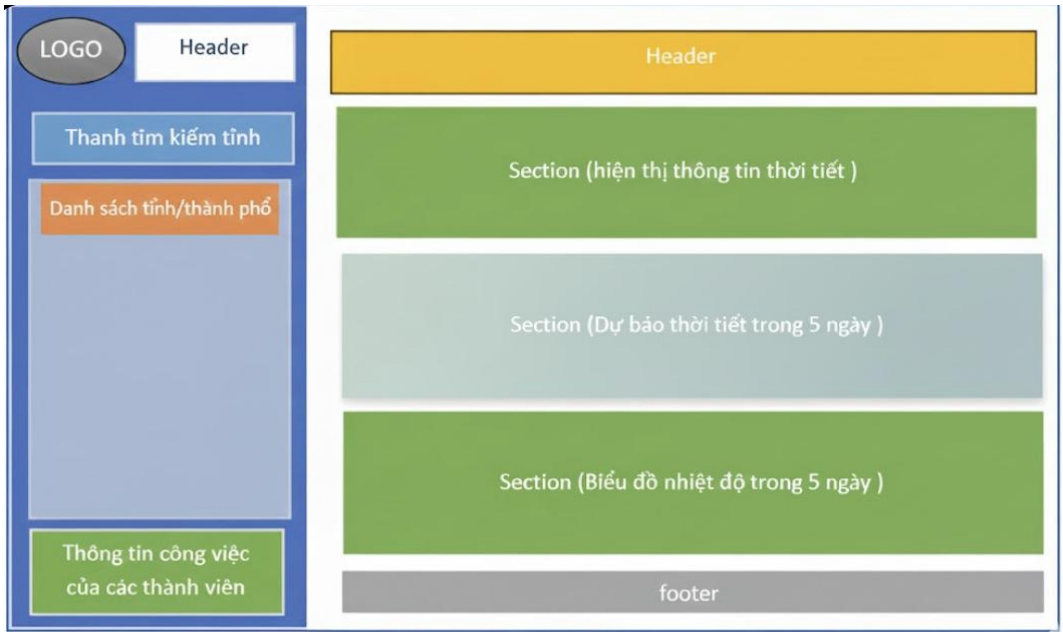
- Tương tác người dùng:
- Hiệu ứng hover, focus giúp giao diện sinh động.
- Nút chuyển đổi đơn vị nhiệt độ (°C ↔ °F).
- Thông báo thời gian cập nhật cuối cùng để người dùng nắm rõ dữ liệu mới nhất.

Nhờ các chức năng này, ứng dụng không chỉ cung cấp dữ liệu mà còn tạo ra trải nghiệm thân thiện, dễ sử dụng và hấp dẫn.

3.3 Mã nguồn và cấu trúc dự án

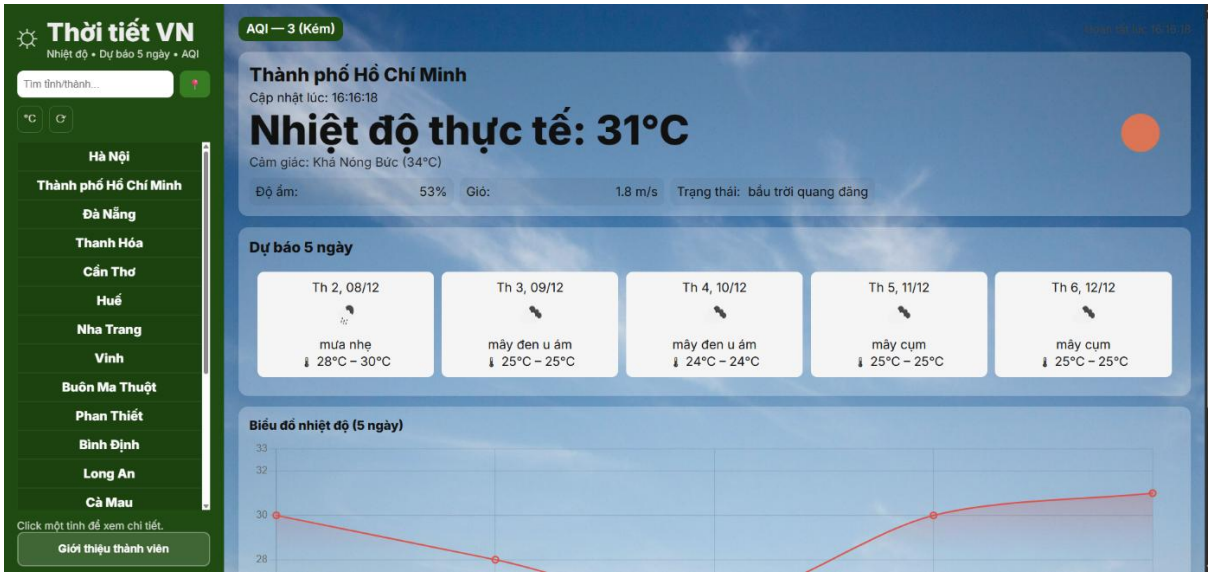
Mã nguồn của dự án được tổ chức theo cấu trúc rõ ràng, đảm bảo dễ bảo trì và mở rộng:

- HTML:
    - Định nghĩa cấu trúc cơ bản của ứng dụng: sidebar, main, footer.
    - Sử dụng thẻ semantic (header, main, footer) để tăng tính rõ ràng và hỗ trợ SEO.



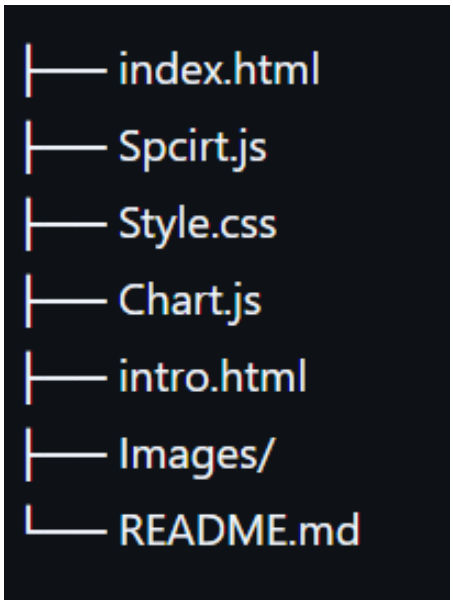
Hình 3.2 Bố cục HTML

- CSS:
    - Chứa toàn bộ định dạng giao diện, bao gồm biến toàn cục, bố cục Flexbox/Grid, hiệu ứng hover, responsive design.
    - Được chia thành các nhóm: reset, layout, component, responsive.



Hình 3.3 Giao diện sau khi thêm CSS

- JavaScript:
    - Xử lý logic ứng dụng: gọi API thời tiết, cập nhật dữ liệu, hiển thị biểu đồ.
    - Quản lý sự kiện người dùng: tìm kiếm, chọn thành phố, chuyển đổi đơn vị nhiệt độ.
    - Tích hợp thư viện vẽ biểu đồ (Chart.js).
- Cấu trúc thư mục:
- index.html: tệp chính chứa cấu trúc giao diện.
- intro.html: tệp Thông tin nhiệm vụ của các thành viên.
- Style.css: tệp định dạng giao diện.
- Script.js: tệp xử lý logic và tương tác.
- Chart.js: tệp xử lý biểu đồ.
- Thư mục Images/: chứa hình ảnh, biểu tượng, logo.



Hình 3.4 Cấu trúc thư mục

Cách tổ chức này giúp mã nguồn dễ đọc, dễ bảo trì, đồng thời thuận tiện cho việc mở rộng thêm chức năng trong tương lai.

3.4 Triển khai website

3.4.1 Cấu trúc index.html

File **index.html** là trang chính của ứng dụng, chứa đựng toàn bộ giao diện hiển thị thông tin thời tiết. Cấu trúc được chia thành các phần chính như sau:

- **Phần khai báo head**

```
<meta charset="UTF-8" />
<meta name="viewport" content="width=device-width,initial-scale=1" />
```

Phần <head> chứa các metadata và liên kết quan trọng:

- **Meta chaset="UTF-8":** Khai báo bảng mả UTF-8 để hiển thị đúng tiếng Việt và các ký tự đặc biệt.
- **Meta viewport:** Thiết lập để trang web hiển thị tốt trên mọi thiết bị, từ destop đến smartphone.

```
<link href="https://fonts.googleapis.com/css2?family=Inter:wght@300;400;600;800&display=swap" rel="stylesheet">

<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
```

- **Google Fonts:** Import font chữ Inter với nhiều độ đậm khác nhau (300, 400, 600, 800) để tạo giao diện hiện đại, dễ đọc.
- **Chart.js:** Thư viện JavaScript để vẽ biểu đồ nhiệt độ động, giúp người dùng dễ dàng theo dõi xu hướng thời tiết.

▪ **Cấu trúc layout tổng thể**

Toàn bộ nội dung được bọc trong thẻ , tạo container chính cho ứng dụng. Layout được chia thành 2 phần chính theo chiều ngang:

- **Sidebar (Thanh bên trái)**

Sidebar sử dụng thẻ semantic <aside> với thuộc tính aria-label="Danh sách tỉnh thành" để tăng khả năng truy cập cho người khuyết tật. Sidebar được chia thành 3 khu vực:

Khu vực brand (Thương hiệu):

```
<div class="brand">
  <div class="logo">☼</div>
  <div>
    <h1>Thời tiết VN</h1>
    <p class="subtitle">Nhiệt độ • Dự báo 5 ngày • AQI</p>
  </div>
</div>
```

▪ Logo mặt trời (☼) kết hợp với tiêu đề tạo nhận diện thương hiệu rõ ràng.

▪ Subtitle giới thiệu ngắn gọn các tính năng chính của ứng dụng.

Khu vực điều khiển (Controls):

```
<div class="search">
  <input id="searchInput" placeholder="Tìm tỉnh/thành..." aria-label="Tìm tỉnh thành" />
  <button id="locateBtn" title="Vị trí của tôi">📍</button>
</div>
```

▪ **Input tìm kiếm:** Cho phép người dùng tìm nhanh tỉnh thành mong muốn. Thuộc tính aria-label giúp screen reader đọc được mục đích của ô input.

- **Nút định vị:** Icon 📍 giúp người dùng nhanh chóng lấy thông tin thời tiết tại vị trí hiện tại.

```
<div class="unit-row">
  <button id="unitToggle" class="btn-ghost">°C</button>
  <button id="refreshBtn" class="btn-ghost" title="Làm mới dữ liệu">↻</button>
</div>
```

- **Nút chuyển đổi đơn vị:** Cho phép chuyển đổi giữa độ C (Celsius) và độ F (Fahrenheit).
- **Nút làm mới:** Cập nhật dữ liệu thời tiết mới nhất từ API.

Khu vực danh sách tỉnh:

```
<div id="citiesList" class="cities"
  role="list" aria-label="Danh sách tỉnh thành">
</div>
```

- Container rỗng này sẽ được JavaScript điền dữ liệu động.
- Thuộc tính role="list" giúp công nghệ hỗ trợ nhận biết đây là danh sách.

Khu vực footer sidebar:

```
<div class="sidebar-bottom">
  <small>Click một tỉnh để xem chi tiết.</small>
  <a href="intro.html" class="intro-link">
  <button class="btn-intro">Giới thiệu thành viên</button>
  </a>
</div>
```

- Hướng dẫn người dùng cách sử dụng.
- Link điều hướng đến trang giới thiệu nhóm.

- **Main (Khu vực chính – Bên phải)**

Sử dụng thẻ semantic <main> chứa toàn bộ thông tin thời tiết chi tiết.

Header:

```
<header class="main-header">

  <div id="aqiBox" class="aqi-badge">AQI — --</div>
  <div id="lastAction" class="last-action">Sẵn sàng</div>

</header>
```

- **AQI badge:** Hiển thị chỉ số chất lượng không khí, thông số quan trọng cho sức khỏe.
- **Last action:** Thông báo trạng thái hoạt động gần nhất của ứng dụng (ví dụ: "Đã tải Hà Nội", "Đang làm mới…").

Card thời tiết hiện tại:

```
<section id="currentCard" class="card current-card">
  <div class="left">

    <div class="city-and-time">
      <h2 id="cityName">—</h2>
      <div id="updated" class="updated">—</div>
    </div>
    <div id="desc" class="desc">—</div>

    <div class="main-temp-row">

      <div class="temp-wrap">
        <div id="temp" class="temp">--°</div>
        <div id="feels" class="feels">Feels: --</div>
      </div>

      <div class="meta-grid">
        <div class="meta-item">
          <span class="label">Độ ẩm:</span>
          <span id="humidity" class="value">--%</span>
        </div>
        <div class="meta-item">
          <span class="label">Gió:</span>
          <span id="wind" class="value">-- m/s</span>
        </div>

        <div class="meta-item">
          <span class="label">Trạng thái:</span>
          <span id="status" class="value">--</span>
        </div>
      </div>

    </div>

  </div>

  <div class="right">
    <div id="icon" class="weather-icon">🌞</div>
  </div>
</section>
```

Card này được chia làm 2 cột:

- **Cột trái:**
  - Tên thành phố (cityName) và thời gian cập nhật (updated).
  - Mô tả thời tiết (desc) như "Trời quang đãng", "Nhiều mây".
  - Nhiệt độ chính (temp) hiển thị to và rõ ràng.
  - Nhiệt độ cảm nhận (feels) – nhiệt độ mà cơ thể thực sự cảm nhận được.
  - Grid 3 thông số phụ: độ ẩm, tốc độ gió, và trạng thái thời tiết.
- **Cột phải:** Icon thời tiết thay đổi theo điều kiện thời tiết.

Tất cả các giá trị ban đầu đều là placeholder ("—", "--") và sẽ được JavaScript cập nhật khi tải dữ liệu.

Section dự báo 5 ngày:

```
<section class="forecast card">
  <h3>Dự báo 5 ngày</h3>

  <div id="forecast" class="forecast-list"></div>

</section>
```

- Tiêu đề rõ ràng để người dùng biết đây là phần dự báo dài hạn.
- Container forecast-list sẽ được JavaScript render thành các card nhỏ cho từng ngày.

Section biểu đồ nhiệt độ:

```
<section class="chart card">
  <div class="chart-head">
    <h4>Biểu đồ nhiệt độ (5 ngày)</h4>
  </div>
  <div class="chart-body">
    <canvas id="tempChart" height="140"></canvas>
  </div>
</section>
```

- Thẻ <canvas> là element đặc biệt của HTML5 cho phép vẽ đồ họa 2D.
- Chart.js sẽ được sử dụng canvas này để vẽ biểu đồ đường (line chart) thể hiện xu hướng nhiệt độ.

Footer:

```
<footer class="footer">
  <div>© 2025 – Thời tiết VN</div>
</footer>
```

Thông tin bản quyền đơn giản.

- **Tích hợp thư viện và script**

```
<script src="Chart.js"></script>
<script src="https://code.jquery.com/jquery-3.7.1.min.js"></script>
<script src="Script.js"></script>
```

Cuối file HTML, trước thẻ đóng </body>, có các dòng import JavaScript:

- **Chart.js:** File cấu hình riêng cho biểu đồ.
- **jQuery 3.7.1:** Thư viện giúp đơn giản hóa thao tác DOM và xử lý sự kiện.
- **Script.js:** File JavaScript chính chứa logic nghiệp vụ của ứng dụng.

Thứ tự import rất quan trọng: jQuery phải được load trước Script.js vì Script.js sử dụng jQuery.

3.4.2 Cấu trúc intro.html

File **intro.html** có mục đích đơn giản hơn: giới thiệu thông tin các thành viên nhóm.

➢ **Đặc điểm cấu trúc**

Khác với index.html, trang này:

- Không cần sidebar, chỉ tập trung vào nội dung chính.
- CSS được nhúng trực tiếp trong thẻ <style> thay vì file riêng, giúp trang độc lập.
- Không phụ thuộc vào JavaScript, mọi thứ đều là nội dung tĩnh.

- **Các phần chính**

Container chính và nút quay lại:

```
<div class="container">
  <a href="index.html" class="back-btn">
    ← Quay lại trang chính
  </a>
```

Một div duy nhất bọc toàn bộ nội dung, được căn giữa màn hình. Và nút quay lại để điều hướng về trang chủ với icon mũi tên trái, giúp điều hướng rõ ràng.

Header giới thiệu:

```
<div class="header">
  <h1>Giới thiệu thành viên</h1>
  <p>Bài Tập Lớn: Ứng dụng Dự báo Thời tiết Việt Nam</p>
</div>
```

Tiêu đề và mô tả ngắn gọn về mục đích của trang.

Bảng thông tin thành viên:

```html
<table>
  <thead>
    <tr>
      <th>Họ và Tên</th>
      <th>MSSV</th>
      <th>Nhiệm vụ chính</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Huỳnh Gia Bảo</td>
      <td>6551071007</td>
      <td>JavaScript - Báo cáo - Thuyết trình</td>
    </tr>
    <tr>
      <td>Lê Quang Chí</td>
      <td>6551071010</td>
      <td>JavaScript - Báo cáo</td>
    </tr>
      <tr>
        <td>Nguyễn Ngọc Ngà</td>
        <td>6551071056</td>
        <td>HTML - Báo cáo</td>
      </tr>
      <tr>
        <td>Trần Thảo Nương</td>
        <td>6551071060</td>
        <td>CSS - Báo cáo</td>
      </tr>
    </tbody>
  </table>
```

Sử dụng thẻ <table> - cấu trúc chuẩn HTML để hiển thị dữ liệu dạng bảng:

- <thead> : Phần header chứa tiêu đề các cột.
- <tbody> : Phần body chứa dữ liệu thực tế.
- Mỗi <tr> (table row) đại diện cho một thành viên.

- Mỗi <td> (table data) chứa một thông tin cụ thể.

Footer:

```
<div class="footer">
  © 2025 Thời tiết VN – Môn Thiết Kế Web
</div>
```

Thông tin bản quyền và tên môn học.

Kết luận : Nguyên tắc thiết kế HTML trong dự án

➢ **Sử dụng Semantic HTML**

Dự án ưu tiên các thẻ semantic (có ý nghĩa) thay vì dùng :

- <header> : Phần đầu trang.
- <main> : Nội dung chính.
- <aside> : Nội dung phụ (sidebar).
- <section> : Các phần nội dung độc lập.
- <footer> : Phần cuối trang.

Lợi ích:

- Giúp công cụ tìm kiếm (Google, Bing) hiểu cấu trúc trang web tốt hơn (SEO).
- Công nghệ hỗ trợ người khuyết tật (screen reader) đọc nội dung chính xác hơn.
- Code dễ đọc, dễ duy trì hơn.

➢ **ID và Class hợp lý**

Mọi yếu tố quan trọng đều có ID hoặc class rõ ràng:

- **ID:** Dùng cho các yếu tố duy nhất cần JavaScript thao tác trực tiếp (#cityName, #temp, #searchInput).
- **Class:** Dùng cho styling và các nhóm yếu tố tương tự như (.card, .meta-item, .btn-ghost).

Quy ước đặt tên:

- ID dùng camelCase: cityName, searchInput.
- Class dùng kebab-case: current-card, meta-item, btn-ghost.

➢ **Accessibility (Khả năng tiếp cận)**

Dự án chú trọng accessibility thông qua:

- Thuộc tính aria-label: Mô tả cho screen reader ("Danh sách tỉnh thành", "Tìm tỉnh thành").
- Thuộc tính role: Định nghĩa vai trò của element (role="list").
- Thuộc tính title: Tooltip khi hover chuột ("Vị trí của tôi", "Làm mới dữ liệu").
- Alt text ngâm định cho icon (sử dụng emoji có ý nghĩa).

➢ **Progressive Enhancement (Cải tiến dần)**

HTML được thiết kế để:

4. Hiển thị cấu trúc cơ bản ngay cả khi CSS/JavaScript bị tắt.
5. Các giá trị placeholder ("—", "--") đảm bảo layout không bị vỡ trước khi JavaScript load dữ liệu.
6. Phân tách rõ ràng: HTML lo cấu trúc, CSS lo giao diện, JavaScript lo tương tác.

➢ **Mobile-First Approach**

Meta viewport width=device-width, initial-scale=1 đảm bảo:

- Trang web tự động co giãn theo kích thước màn hình.
- Không bị zoom out trên mobile.
- Sẵn sàng cho responsive design (sẽ được CSS xử lý).

3.4.3 Cấu trúc Style.css

3.4.3.1 Biến toàn cục (CSS Variables)

```
:root {
  --glass-bg: rgba(255, 255, 255, 0.8);
  --glass-dark: rgba(0, 0, 0, 0.45);
  --accent: #0078d7;
}
```

Định nghĩa biến màu và hiệu ứng dùng chung:

- **--glass-bg:** nền trong suốt trắng.
- **--glass-dark:** nền tối trong suốt.
- **--accent:** màu nhấn (#0078d7).

3.4.3.2 Thiết lập cơ bản

```
* {
  box-sizing: border-box;
  margin: 0;
  padding: 0;
}
body {
  font-family: Inter, system-ui, Arial, sans-serif;
  color: #111;
  min-height: 100vh;
  display: flex;
  justify-content: center;
}
```

- Reset margin/padding toàn bộ.
- Font chữ: Inter, system-ui, Arial.
- Body: căn giữa nội dung, chiều cao tối thiểu toàn màn hình.

3.4.3.3 Cấu trúc chính

```
.app {
  display: flex;
  overflow: hidden;
  box-shadow: 0 10px 40px rgba(2, 6, 23, 0.15);
  width: 100%;
  height: auto;
}
.sidebar {
  width: 280px;
  padding: 18px;
  background-color: rgb(6, 76, 3);
  color: #eeeded;
  height: auto;
}
main.main {
  flex: 1;
  padding: 18px;
  background-size: cover;
  background-position: center;
  transition: background-image 0.8s ease;
}
```

.app

- Flex container, có bóng đổ, chiếm toàn bộ chiều ngang.

.sidebar

- Rộng 280px, nền xanh đậm rgb(6, 76, 3), chữ sáng.
- Dùng cho menu bên trái.

main.main

- Flex: chiếm phần còn lại.
- Có padding, background động theo thời tiết:
  - weather-sunny: ảnh nắng.
  - weather-rain: ảnh mưa.
  - weather-cloud: ảnh mây.
  - weather-snow: ảnh tuyết.
  - weather-mist: ảnh sương mù.
- Hiệu ứng chuyển nền mượt (transition 0.8s).

3.4.3.4 Thành phần giao diện

Brand & Logo

- .brand: flex, căn giữa logo và subtitle.
- .logo: font-size 28px.
- .subtitle: font nhỏ, mờ nhẹ.

Search

- .search: flex, có input và button.
- Input: bo tròn, không viền.
- Button: nền xanh lá, chữ trắng, bo tròn.

Nút giới thiệu .btn-intro

- Nền trong suốt, viền trắng mờ, chữ trắng.
- Hiệu ứng hover: sáng hơn, phóng to nhẹ.

Nút phụ .btn-ghost

- Nền trong suốt, viền mờ, chữ sáng.
- Hover: nền sáng nhẹ.

3.4.3.5 Danh sách thành phố

- .cities: danh sách cuộn dọc.
- .c: item cơ bản, nền mờ, hover sáng hơn.
- .city-item: nổi bật hơn:
- Nền tối mờ, chữ trắng, font đậm.
- Hover: sáng hơn, dịch chuyển nhẹ.
- Active: nền trắng, chữ xanh lá, có bóng đổ.

3.4.3.6 Thông tin thời tiết

Header

- .main-header: flex, căn đều, có badge AQI.
- .aqi-badge: nền xanh lá, chữ trắng, bo tròn.

Card

- .card: nền trong suốt, bo tròn, bóng nhẹ.
- .current-card: flex, hiển thị thông tin hiện tại.
- .temp: font-size lớn (52px).
- .feels: chữ mờ nhẹ.

Meta Grid

- .meta-grid: grid 3 cột.
- .meta-item: nền mờ, bo tròn, căn giữa.

Forecast

- .forecast-list: grid tự động, item bo tròn.
- .forecast-item: nền sáng, text căn giữa.
- Icon thời tiết: kích thước 100px.

3.4.3.7 Footer

- Chữ nhỏ, màu xám, căn giữa.

3.4.3.8 Biểu đồ

- .chart-body: khung chứa chart, cao 400px.
- Canvas chiếm toàn bộ chiều rộng/chiều cao.

3.4.3.9 Responsive Design

Tablet (<880px)

- .app: chuyển sang dạng cột.
- .sidebar: ngang, cuộn ngang.
- .chart-body: giảm chiều cao xuống 240px.

  Mobile (<768px)
- .mobile-menu-btn: hiện nút menu.
- .sidebar: chuyển thành menu trượt (slide-in).
- .main.dimmed: làm mờ khi menu mở.
- Forecast: wrap, item nhỏ hơn.
- Icon thời tiết: 60px.
- Nhiệt độ: font-size giảm xuống 42px.

3.4.4 Cấu trúc Script.js

3.4.4.1 Cấu hình ban đầu của ứng dụng

Ngay đầu file, chương trình khai báo các biến cấu hình dùng xuyên suốt:

```
// 🔑 API Key và Thành phố mặc định
const API_KEY = "6065e876077b800805afb4908e593bfa";
const DEFAULT_CITY = "Ho Chi Minh";
let lastSuccessfulCity = DEFAULT_CITY; // <-- THÊM BIẾN NÀY để sửa lỗi Refresh
```

Ý nghĩa:

- API_KEY → key để gọi OpenWeatherMap API.
- DEFAULT_CITY → thành phố mặc định khi mở trang.
- lastSuccessfulCity → lưu thành phố cuối cùng lấy thành công dùng cho nút Refresh.

3.4.4.2 Ánh xạ tên tỉnh/thành Việt Nam

API trả tên không dấu (Hanoi, Danang).

```
// 🌍 Bảng ánh xạ tên tỉnh/thành Việt Nam
const VN_CITY_NAME_MAP = {
  "Ho Chi Minh": "Hồ Chí Minh",
  Hanoi: "Hà Nội",
  "Da Nang": "Đà Nẵng",
```

Để hiển thị tiếng Việt chuẩn, dùng bảng ánh xạ:

Ý nghĩa: giao diện luôn hiển thị đúng tiếng Việt.

3.4.4.3 Danh sách tỉnh/thành để hiển thị trong sidebar

```
//Tạo danh sách tên tỉnh để dễ nhấn chọn
const cities = [
    "Hà Nội",
    "Thành phố Hồ Chí Minh",
    "Đà Nẵng",
```

3.4.4.4 Đóng sidebar trên mobile sau khi chọn thành phố

```
// Đóng sidebar trên mobile sau khi chọn thành phố
const sidebar = document.querySelector(".sidebar");
const main = document.querySelector(".main");
if (sidebar && main) {
    sidebar.classList.remove("open");
    main.classList.remove("dimmed");
}
```

Đoạn code này dùng để:

✓ Khi người dùng chọn 1 thành phố trên mobile → sidebar tự đóng lại

✓ Giao diện trở về trạng thái bình thường

✓ Tránh phải tự tay nhấn đóng menu

Nó là phần quan trọng trong UI responsive giống các app thời tiết hiện nay.

3.4.4.5 Tự động đóng sidebar khi người dùng thao tác

```
main.addEventListener("click", () => {
  if (main.classList.contains("dimmed")) {
    sidebar.classList.remove("open");
    main.classList.remove("dimmed");
  }
});

document.querySelectorAll(".cities .city-item").forEach(item => {
  item.addEventListener("click", () => {
    sidebar.classList.remove("open");
    main.classList.remove("dimmed");
  });
});
```

3.4.4.6 Hàm mô tả cảm giác nhiệt độ

```
// 🌡️ Hàm mô tả cảm giác nhiệt độ
function getFeelStatus(feelsLikeC) {
  if (feelsLikeC >= 35) return "Rất Nóng (Khó Chịu)";
  if (feelsLikeC >= 30) return "Khá Nóng Bức";
  if (feelsLikeC >= 25) return "Ấm Áp";
  if (feelsLikeC >= 20) return "Mát Mẻ Dễ Chịu";
  if (feelsLikeC >= 15) return "Se Lạnh";
  if (feelsLikeC >= 10) return "Khá Lạnh";
  return "Rét Đậm";
}
```

Ý nghĩa: hiển thị trạng thái như app thời tiết thật.

3.4.4.7 Đổi màu nền theo trạng thái thời tiết

```
function updateWeatherBackground(weatherText) {  ... }
```

Ý nghĩa:

- Tự đổi màu nền theo trạng thái:
  - Trời nắng → vàng sáng
  - Trời mưa → xám xanh
  - Nhiều mây → ghi
  - Sương mù → trắng mờ
  - Tuyết → xanh nhạt

3.4.4.8 Hàm chính gọi API thời tiết

```
function fetchAndUpdateWeather(city) { ... }
```

Chức năng:

- Chuẩn hóa tên thành phố.
- Gọi API /weather.
- Xử lý lỗi:
  - Thành phố không tồn tại
  - Lỗi API Key
  - Mất mạng
- Cập nhật tất cả UI:
  - Tên thành phố
  - Nhiệt độ
  - Cảm giác
  - Độ ẩm, gió
  - Icon thời tiết
  - Thời điểm cập nhật
- Lưu lại thành phố thành công → tránh lỗi khi refresh.

3.4.4.9 Hàm lấy dữ liệu dự báo 5 ngày – 3 giờ/lần

function fetchForecast(city) { ... }

Chức năng:

- Gọi API /forecast.
- Lọc dữ liệu theo mốc 12:00 mỗi ngày.
- Tạo HTML card dự báo (icon, nhiệt độ min-max, mô tả).
- Vẽ biểu đồ dự báo (gọi renderForecastChart()).

3.4.4.10 Các hiệu ứng giao diện bằng jQuery

Trong quá trình xây dựng giao diện, hệ thống đã áp dụng một số hiệu ứng bằng jQuery nhằm nâng cao trải nghiệm người dùng. Cụ thể: danh sách tỉnh được thiết kế với hiệu ứng mở/đóng giúp thao tác thuận tiện và gọn gàng; các meta-item có hiệu ứng hover để tăng tính trực quan khi người dùng di chuột qua; trong quá trình làm mới dữ liệu, hệ thống hiển thị loading spinner nhằm thông báo trạng thái xử lý.

Bên cạnh đó, các thông báo lỗi được trình bày với hiệu ứng fade in/out, vừa đảm bảo tính thẩm mỹ vừa dễ nhận biết. Phần dự báo thời tiết 5 ngày được hiển thị bằng hiệu ứng slide, tạo cảm giác mượt mà khi chuyển đổi thông tin. Các icon thời tiết cũng được bổ sung hiệu ứng động, giúp giao diện sinh động và hiện đại hơn. Cuối cùng, hệ thống có cơ chế reset dữ liệu khi khởi động, đảm bảo trạng thái ban đầu ổn định và sẵn sàng cho người dùng.

3.4.5 Cấu trúc Chart.js

Đoạn mã JavaScript dưới đây có chức năng xử lý dữ liệu thời tiết và trực quan hóa chúng dưới dạng biểu đồ đường bằng thư viện Chart.js. Hệ thống gồm hai hàm chính:

- renderTemperatureChart() → dùng để vẽ biểu đồ nhiệt độ
- renderForecastChart() → xử lý dữ liệu dự báo thô rồi gửi cho hàm trên

Nội dung dưới đây phân tích chi tiết toàn bộ hoạt động theo đúng quá trình của đoạn code bạn cung cấp.

3.4.5.1 Biến lưu trữ biểu đồ

```
let tempChart = null;
```

Biến này giữ phiên bản biểu đồ hiện tại. Chart.js không cho phép tạo nhiều biểu đồ chồng lên cùng một <canvas>, vì vậy đoạn code sử dụng nó để hủy biểu đồ cũ trước khi tạo biểu đồ mới, tránh lỗi vẽ đè hoặc rò rỉ bộ nhớ.

3.4.5.2 Hàm renderTemperatureChart(dailyData)

Đây là hàm trung tâm đảm nhiệm việc vẽ biểu đồ.

2.1 Kiểm tra dữ liệu đầu vào

```
if (!dailyData || dailyData.length === 0) return;
```

Nếu dữ liệu không tồn tại hoặc rỗng, hàm dừng ngay để tránh lỗi.

2.2 Tạo nhãn ngày (labels)

Biểu đồ cần các nhãn trục X để hiển thị ngày và thứ.

```
const labels = dailyData.map((item) => {
  const date = new Date(item.dt * 1000);
  const weekday = date
    .toLocaleDateString("vi-VN", { weekday: "short" })
    .replace(".", "");
  const day = date.getDate().toString().padStart(2, "0");
  const month = (date.getMonth() + 1).toString().padStart(2, "0");
  return `${weekday} ${day}/${month}`;
});
```

API cung cấp thời gian UNIX (tính theo giây) nên phải nhân 1000.

Chuyển sang thứ ngắn theo tiếng Việt: T2, T3, T4…

Ghép thành chuỗi dạng: "T3 07/12" – ngắn gọn, đẹp, phù hợp giao diện thời tiết.

2.3 Tách dữ liệu nhiệt độ

```
const tempsMin = dailyData.map((item) => Math.round(item.main.temp_min));
const tempsMax = dailyData.map((item) => Math.round(item.main.temp_max));
```

Tạo 2 mảng:

- tempsMin → nhiệt độ thấp nhất trong ngày
- tempsMax → nhiệt độ cao nhất trong ngày

Việc làm tròn (Math.round) giúp biểu đồ sạch và dễ đọc hơn.

2.4. Lấy canvas và chuẩn bị context

```
const canvas = document.getElementById("tempChart");
if (!canvas) return;
const ctx = canvas.getContext("2d");
```

canvas là nơi biểu đồ sẽ được vẽ lên.

ctx là đối tượng để Chart.js thao tác đồ họa.

2.5. Hủy biểu đồ cũ nếu tồn tại

if (tempChart) tempChart.destroy();

Nếu không hủy, mỗi lần cập nhật biểu đồ sẽ bị:

- chồng lên nhau
- giật, nhảy
- lỗi "Canvas is already in use"

Đây là cách đúng chuẩn để cập nhật biểu đồ trong Chart.js.

2.6. Tạo hiệu ứng màu gradient

Gradient màu xanh (dành cho nhiệt độ thấp)

```
const gradientBlue = ctx.createLinearGradient(0, 0, 0, canvas.height || 300);
gradientBlue.addColorStop(0, "rgba(59,130,246,0.45)");
gradientBlue.addColorStop(1, "rgba(59,130,246,0)");
```

Gradient màu đỏ (dành cho nhiệt độ cao)

```
const gradientRed = ctx.createLinearGradient(0, 0, 0, canvas.height || 300);
gradientRed.addColorStop(0, "rgba(239,68,68,0.45)");
gradientRed.addColorStop(1, "rgba(239,68,68,0)");
```

Hiệu ứng:

- Phần phía trên đậm → phía dưới mờ dần
- Giúp biểu đồ hiện đại, giống các ứng dụng thời tiết chuyên nghiệp

2.7. Tạo biểu đồ Chart.js

```
tempChart = new Chart(ctx, {
  type: "line",
  data: {
    labels: labels,
    datasets: [
      { ... },
      { ... }
```

```
    ]
  },
  options: { ... }
});
```

Phân tích datasets:

Dataset nhiệt độ thấp:

- borderColor: "#3B82F6" → màu xanh
- backgroundColor: gradientBlue → có hiệu ứng đổ màu
- tension: 0.3 → đường cong mượt
- pointRadius: 4 → điểm nhỏ vừa đủ

Dataset nhiệt độ cao:

Tương tự dataset 1 nhưng màu đỏ.

2.8. Cấu hình Tooltip và Legend

```
  plugins: {
  tooltip: {
    callbacks: {
      label: (ctx) => `${ctx.dataset.label}: ${ctx.raw}°C`,
    },
  },
  legend: {
    display: false
  }
}
```

tooltip hiển thị thông tin chính xác khi rê chuột lên điểm dữ liệu.

legend bị tắt để bố cục gọn và đẹp hơn trên giao diện nhỏ.

2.9. Chỉnh trục Y để biểu đồ cân đối

```
const yMin = Math.min(...tempsMin) - 2;
const yMax = Math.max(...tempsMax) + 2;
tempChart.options.scales.y.min = Math.floor(yMin);
tempChart.options.scales.y.max = Math.ceil(yMax);
tempChart.update();
```

Ý nghĩa:

- Tránh cho đường biểu đồ chạm sát mép trên/dưới
- Giúp biểu đồ thoáng, nhìn dễ hơn
- Tự động theo dữ liệu, không cố định

3.4.5.3 Hàm renderForecastChart(forecastData)

API dự báo 5 ngày trả dữ liệu mỗi 3 giờ, tức là một ngày có 8 mẫu.

Hàm này xử lý dữ liệu thô và gom lại thành 5 ngày cần thiết.

a) Kiểm tra dữ liệu

if (!forecastData || !forecastData.list) return;

b) Nhóm dữ liệu theo ngày

const groups = {};

Mỗi phần tử trong forecastData.list chứa:

- thời gian dự báo
- nhiệt độ tại thời điểm đó

Code gom những điểm cùng ngày vào cùng một nhóm.

Cách nhóm theo key:

const key = d.toISOString().split("T")[0]; // YYYY-MM-DD

c) Tính nhiệt độ thấp nhất và cao nhất thật sự trong ngày

groups[key].min = Math.min(groups[key].min, item.main.temp_min);

groups[key].max = Math.max(groups[key].max, item.main.temp_max);

→ Đây là bước quan trọng vì API trả theo giờ, không phải theo ngày!

d) Lấy ra 5 ngày đầu tiên

```
const days = Object.values(groups)
  .sort((a, b) => a.date - b.date)
  .slice(0, 5)
  .map((g) => ({
    dt: g.date.getTime() / 1000,
    main: { temp_min: g.min, temp_max: g.max }
  }));
```

e) Gửi dữ liệu đã xử lý sang hàm vẽ biểu đồ

```
renderTemperatureChart(days);
```

CHƯƠNG 4: KIỂM THỬ, TỐI ƯU HÓA VÀ TRIỂN KHAI

4.1 Kiểm thử giao diện và chức năng

Sau khi hoàn thành xây dựng website *"Dự báo thời tiết Việt Nam"*, nhóm tiến hành giai đoạn kiểm thử nhằm đánh giá mức độ hoạt động chính xác của các chức năng, sự ổn định của giao diện, khả năng tương thích với thiết bị và trình duyệt khác nhau. Đây là bước quan trọng để đảm bảo website đạt yêu cầu về kỹ thuật và trải nghiệm người dùng trước khi triển khai chính thức.

4.1.1. Kiểm tra từng chức năng

Website được xây dựng với nhiều chức năng khác nhau, vì vậy nhóm đã tiến hành kiểm thử theo từng chức năng cụ thể:

a) Chức năng tìm kiếm thành phố

Người dùng có thể nhập tên tỉnh/thành phố vào ô tìm kiếm ở thanh bên trái (Sidebar). Website có khả năng nhận diện tên thành phố:

- Có dấu: *"Hồ Chí Minh", "Hà Nội", "Đà Nẵng"...*
- Không dấu: *"Ho Chi Minh", "Ha Noi"...*
- Các cách viết viết tắt: *"HCM", "SG", "Sai Gon"...*

Sau khi nhấn Enter hoặc nhấn nút 📍, website sẽ gửi yêu cầu đến API OpenWeatherMap. Kết quả kiểm thử:

- Nhiệt độ hiện tại được hiển thị chính xác
- Thông tin độ ẩm, tốc độ gió, trạng thái thời tiết cập nhật đầy đủ
- Icon thời tiết thay đổi phù hợp với trạng thái thực tế (nắng, mưa, nhiều mây,…)

→ Chức năng hoạt động ổn định, phản hồi nhanh.

b) Chức năng chọn thành phố từ danh sách (Sidebar)

Danh sách các tỉnh thành được hiển thị ở thanh bên trái dưới dạng các nút bấm. Khi người dùng click vào từng tên thành phố như:

- Hà Nội
- Thành phố Hồ Chí Minh
- Đà Nẵng
- Cần Thơ
- Nha Trang
- Cà Mau,…

Hệ thống ngay lập tức:

- Cập nhật tên thành phố
- Cập nhật nhiệt độ mới
- Cập nhật thông tin thời tiết hiện tại
- Cập nhật dự báo 5 ngày và biểu đồ nhiệt độ

Kết quả kiểm thử cho thấy việc chuyển đổi giữa các thành phố diễn ra nhanh và ổn định, không bị lag hoặc treo trang.

c) Chức năng làm mới dữ liệu (Refresh)

Nút làm mới dữ liệu "↻" được thiết kế để tải lại thông tin thời tiết của thành phố được xem gần nhất. Nhóm đã sử dụng biến lastSuccessfulCity trong JavaScript để lưu lại thành phố cuối cùng hoạt động thành công.

Kết quả kiểm thử:

- Khi nhấn Refresh, website không bị trở về thành phố mặc định
- Dữ liệu được cập nhật lại đúng thành phố đang xem
- Không bị lỗi "City not found" hoặc lỗi API

→ Chức năng này giúp website hoạt động chuyên nghiệp và tiện lợi hơn cho người dùng.

d) Hiển thị dự báo thời tiết 5 ngày

Website sử dụng API /forecast của OpenWeatherMap để lấy dữ liệu dự báo 5 ngày tiếp theo. Kết quả được hiển thị bao gồm:

- Thứ / Ngày
- Biểu tượng thời tiết
- Mô tả: (mưa nhẹ, nhiều mây, nắng, …)
- Nhiệt độ thấp nhất – cao nhất

Dữ liệu này được hiển thị dưới dạng các khung nhỏ giúp người dùng dễ quan sát. Khi thay đổi thành phố, dữ liệu dự báo cũng đổi theo tương ứng.

e) Biểu đồ nhiệt độ (Chart.js)

Website sử dụng thư viện Chart.js để vẽ biểu đồ đường thể hiện sự thay đổi nhiệt độ theo thời gian. Biểu đồ có tác dụng:

- Tăng tính trực quan cho thông tin dự báo
- Người dùng dễ so sánh nhiệt độ các ngày
- Nâng cao tính hiện đại của website

Kết quả kiểm thử cho thấy biểu đồ hoạt động tốt, không xảy ra lỗi giao diện hoặc lỗi dữ liệu.

f) Hiệu ứng và trải nghiệm người dùng (UI/UX)

Trang web còn được bổ sung nhiều hiệu ứng giúp tăng trải nghiệm như:

- Hiệu ứng hover khi rê chuột vào nút và icon
- Hiệu ứng ẩn/hiện khi nhấn vào tiêu đề dự báo

- Hiệu ứng nền thay đổi theo thời tiết (nắng – mưa – mây – sương)
- Hiệu ứng loading (spinner) khi làm mới dữ liệu

Những hiệu ứng này hoạt động tốt, không ảnh hưởng đến hiệu suất và giúp website trở nên sinh động hơn.

4.1.2. Kiểm thử responsive trên nhiều thiết bị

Website được thiết kế theo hướng đáp ứng (Responsive Web Design) với việc sử dụng nhiều đoạn mã @media query trong CSS để tự động thay đổi bố cục theo kích thước màn hình.

Nhóm đã tiến hành kiểm thử trên:

- Laptop màn hình 15.6 inch
- Điện thoại Android

Kết quả:

- Laptop giao diện rộng rãi, đầy đủ thông tin.
- Điện thoại gaio diện thu gọn, dễ thao tác.

Một số thay đổi khi ở trên thiết bị nhỏ (mobile):

- Thanh menu chuyển thành menu trượt
- Icon và chữ được thu nhỏ lại
- Dự báo 5 ngày chuyển thành dạng danh sách 2 cột
- Biểu đồ giảm chiều cao để phù hợp màn hình

Nhờ đó, trải nghiệm người dùng trên các thiết bị khác nhau vẫn được đảm bảo.

4.1.3. Kiểm tra tương thích trình duyệt

Website đã được mở và sử dụng thử trên nhiều trình duyệt khác nhau gồm:

- Google Chrome

- Microsoft Edge
- Mozilla Firefox
- Trình duyệt điện thoại (Chrome Mobile)

Kết quả:

- Giao diện hiển thị giống nhau
- Không có lỗi font chữ
- Không có lỗi JavaScript
- Không xảy ra tình trạng vỡ layout

Điều này cho thấy website tương thích tốt với chuẩn trình duyệt hiện đại.

4.2. Tối ưu hóa hiệu suất

4.2.1. Tối ưu hình ảnh, CSS/JS

Trong quá trình xây dựng, nhóm đã chú trọng đến vấn đề hiệu suất:

- Các file CSS và JavaScript được tách riêng giúp trình duyệt tải nhanh hơn.
- Sử dụng thư viện từ CDN (Chart.js, jQuery) để giảm tải cho server.
- Hạn chế số lượng hình ảnh không cần thiết.
- Thiết kế giao diện đơn giản, không quá nặng.

Ngoài ra, JavaScript được viết theo hướng tối ưu:

- Chỉ gọi API khi người dùng yêu cầu
- Không dùng vòng lặp dư thừa
- Tái sử dụng hàm khi cần thiết

- Điều này giúp website hoạt động mượt mà hơn.

4.2.2. Đánh giá bằng Google PageSpeed / Lighthouse

Nhóm đã sử dụng công cụ Lighthouse tích hợp trong trình duyệt Google Chrome để đánh giá website. Kết quả cho thấy:

- Website có tốc độ tải nhanh
- Điểm hiệu suất nằm ở mức khá tốt
- Tối ưu tốt cho trải nghiệm người dùng

Mặc dù còn một số hạn chế đến từ ảnh nền và API bên ngoài, nhưng nhìn chung website hoàn toàn phù hợp với mục tiêu học tập và đồ án sinh viên.

4.2.3. SEO cơ bản: meta tags, alt text, semantic HTML

Website có áp dụng các yếu tố SEO cơ bản gồm:

- Thẻ <title> rõ ràng: *Dự báo thời tiết Việt Nam*
- Có thẻ <meta charset="UTF-8">
- Có thẻ <meta name="viewport">
- Sử dụng đúng các tiêu đề: h1, h2, h3, h4
- Icon thời tiết có thuộc tính alt
- Cấu trúc HTML rõ ràng, dễ đọc

Nhờ đó, website dễ dàng được các công cụ tìm kiếm như Google nhận diện và thu thập dữ liệu.

4.3. Triển khai website lên host và server

4.3.1. Giới thiệu hosting

Nhóm sử dụng dịch vụ InfinityFree để triển khai website. Đây là một nền tảng hosting miễn phí, phù hợp cho sinh viên và người mới học làm web.

Một số ưu điểm của InfinityFree:

- Miễn phí hoàn toàn
- Có dung lượng lưu trữ
- Có tên miền phụ
- Quản lý bằng giao diện cPanel

Việc sử dụng InfinityFree giúp nhóm tiết kiệm chi phí nhưng vẫn đảm bảo website có thể truy cập online.

4.3.2. Các bước triển khai website

Quá trình triển khai gồm các bước sau:

1. Tạo tài khoản trên InfinityFree
2. Tạo một website mới
3. Upload các file:
    - index.html
    - Style.css
    - Script.js
    - intro.html
    - Chart.js
4. Kiểm tra liên kết giữa các file
5. Kiểm tra lỗi đường dẫn
6. Chạy thử website

Sau khi hoàn thành, website có thể truy cập trên Internet như một trang web thực thụ.

4.3.3. Link truy cập website

Website được triển khai thành công tại địa chỉ: https://thoietvn.infinityfreeapp.com

Chỉ cần một thiết bị có kết nối Internet và trình duyệt, người dùng có thể truy cập và sử dụng website.

4.3.4. Kiểm tra truy cập từ xa và độ ổn định

Nhóm đã kiểm tra truy cập website từ:

- Máy tính cá nhân
- Điện thoại di động
- Mạng Wi-Fi và mạng 4G

Kết quả:

- Website tải nhanh
- Không bị treo hoặc lỗi
- API vẫn hoạt động ổn định
- Trải nghiệm người dùng tốt

Sau nhiều lần truy cập liên tục, website vẫn giữ được độ ổn định cao, không xảy ra lỗi sập trang.

CHƯƠNG 5 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1 Kết quả đạt được

- Nhóm đã hoàn thành đầy đủ các tiêu chí theo rubric của môn học: giao diện, chức năng, báo cáo và demo sản phẩm.
- Xây dựng được một website dự báo thời tiết hoạt động ổn định, có khả năng tìm kiếm thành phố, lấy vị trí hiện tại, xử lý API và hiển thị dữ liệu theo thời gian thực.
- Giao diện web được thiết kế hiện đại, responsive tốt trên cả máy tính và điện thoại.
- Tối ưu trải nghiệm người dùng: menu trượt, trạng thái loading, nền động theo điều kiện thời tiết.
- Hiểu và áp dụng thành công các kiến thức HTML, CSS, JavaScript, API, biểu đồ, tổ chức thư mục.
- Tự triển khai website lên GitHub Pages và hosting riêng, chứng minh khả năng đưa sản phẩm vào thực tế.
- Mức độ hoàn thành đáp ứng tốt mục tiêu ban đầu của đề tài: tạo ra một website thời tiết trực quan, dễ dùng, nhanh và chính xác.

5.2 Hạn chế

- API miễn phí có giới hạn số lượng request, dẫn đến đôi lúc tải chậm hoặc bị từ chối truy cập.
- Một số thành phố nhỏ, thị xã chưa hiển thị dữ liệu đầy đủ do phía API không cung cấp.
- Chưa có backend để ẩn API Key hoàn toàn → dễ bị lộ key khi public.
- Chưa có hệ thống lưu lịch sử tìm kiếm hoặc gợi ý thông minh.
- Biểu đồ chưa hỗ trợ nhiều loại dữ liệu như lượng mưa, cảm giác nhiệt, dự báo nhiều ngày.

- Giao diện chưa có dark mode và animation mượt như sản phẩm thương mại.
- Hiệu năng trên điện thoại cấu hình thấp đôi lúc giảm khi load nhiều ảnh nền.

5.3 Hướng phát triển

- Dark/Light Mode tự động theo hệ thống.
- Animation mềm mại (fade-in, slide-up, loading).
- Bản đồ thời tiết (Mapbox / Leaflet).
- Lịch sử tìm kiếm + gợi ý thông minh.
- PWA (Progressive Web App) → cho phép cài như app mobile.
- Chuyển sang Backend nhỏ (Node.js) để ẩn API KEY.
- Tối ưu SEO + performance theo Lighthouse.
- Thiết kế UI Giáng Sinh / Tết / Chủ đề mùa (theo event).

TÀI LIỆU THAM KHẢO

[1] https://developer.mozilla.org/ “MDN Web Docs” Truy cập vào ngày 2/11/2025]

[2] https://www.w3schools.com/ “W3Schools HTML/CSS/JS” [Truy cập vào ngày 20/10/2025]

[3] https://tailwindcss.com/ “TailwindCSS” [Truy cập vào ngày 20/11/2025]

[4] https://openweathermap.org/api “OpenWeatherMap Docs” [Truy cập vào ngày 11/11/2-25]

[5] https://dribbble.com / https://uiverse.io “UI Inspiration” [Truy cập vào ngày 15/11/2025]

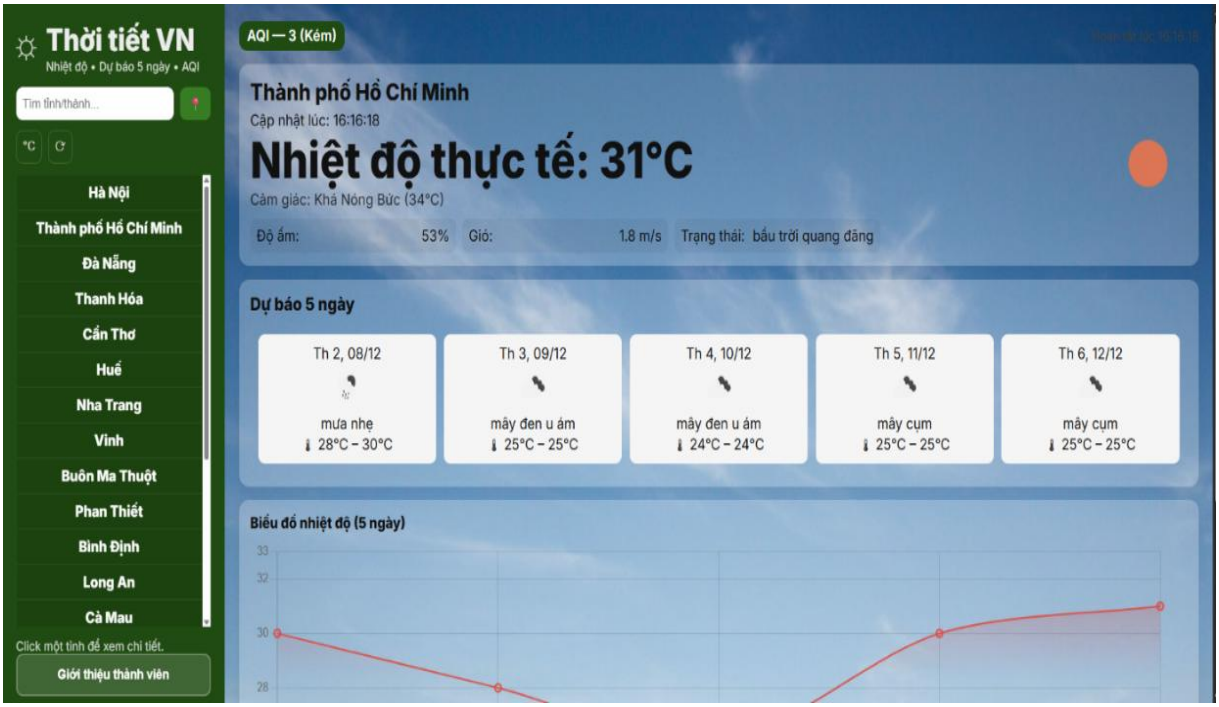
[6] https://responsivedesign.is/ “Responsive Design” [Truy cập vào ngày 15/11/2025]

[7] https://javascript.info/async “Async/Fetch Basics” [Truy cập vào ngày 2/11/2025]

PHỤ LỤC

Phụ lục A: Tóm tắt về web thông tin dự báo thời tiết

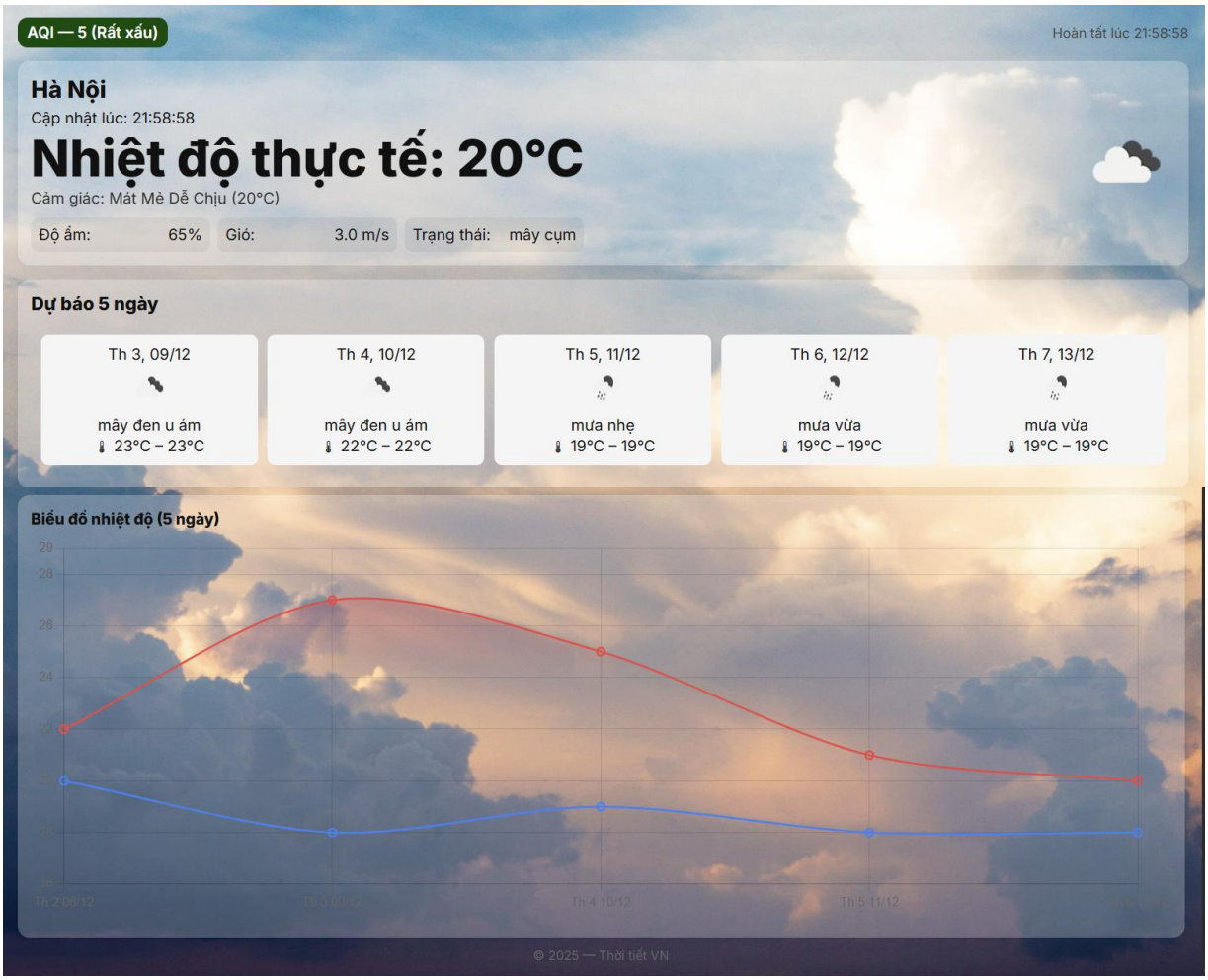
A.1 Ảnh chụp giao diện



Hình A.1 Ảnh giao diện hiển thị trên laptop

Hình A.2 Ảnh giao diện hiển thị trên mobile

Hình A.3 Sidebar hiển thị danh sách thành phố.



Hình A.4 Giao diện thông tin thời tiết và dự báo thời tiết

A.2 Code chính

- **File CSS:** style.css
  - Chứa toàn bộ cấu trúc style (đã phân tích ở phần báo cáo).
- **File HTML:**
- index.html
  - Khung giao diện chính, liên kết với CSS, Javascript và intro.html.
- Intro.html
  - Khung giao diện giới thiệu thành viên công việc.
- **File Javascript:**
  - Script.js
    - Xử lý logic thời tiết, API, sự kiện click, responsive menu.
  - Chart.js
    - Xử lý biểu đồ dự báo nhiệt độ.

A.3 Bảng kiểm thử

STT	Nhóm chức năng	Mô tả kiểm thử	Đầu vào	Kết quả mong đợi	Kết quả thực tế	Trạng thái
1	Giao diện	Tải trang chính	URL index.html	Hiển thị đầy đủ, không lỗi CSS/JS	Đúng	✅
2	Giao diện	Tải trang intro	URL intro.html	Hiển thị thông tin + nút Back hoạt Đúng động	Đúng	✅
3	Giao diện	Responsive mobile	Thu nhỏ < 768px	Đúng Layout không vỡ, menu thu gọn	Đúng	✅
4	Tìm kiếm	Tìm thành phố hợp lệ	"Ha noi"	Hiển thị đúng thời tiết	Đúng	✅
5	Tìm kiếm	Tìm thành phố không tồn tại	"abcxyz"	Báo "Không tìm thấy vị trí"	Đúng	✅
6	Tìm kiếm	Reset input	Xóa chữ	Placeholder xuất hiện	Đúng	✅
7	API	Tải mặc định	DEFAULT_CITY_API	Nhận dữ liệu JSON đúng	Đúng	✅
8	API	API lỗi	Tắt mạng	Hiện thông báo lỗi	Đúng	✅

9	Biểu đồ	Hiện thị		Hiện thị bình thường	Đúng	✅
10	Menu	Mở menu	Click icon	Menu mở ra	Đúng	✅
11	Menu	Đóng menu	Click ra ngoài	Menu đóng	Đúng	✅
12	Định vị	Lấy GPS	Click nút vị trí	Lấy tọa độ → hiển thị thời tiết	Đúng	✅
13	Màu nền	Biến đổi them thời tiết	"trời mưa"	Nền đổi sang nền trời mưa	Đúng	✅
14	Hiệu năng	Thời gian phản hồi API		< 2 giây	Đúng	✅
15	Hiệu năng	Load ảnh nền		< 1.5 giây	Đúng	✅
16	Tổng hợp	Tìm kiếm liên tục 10 lần		Hoạt động bình thường	Đúng	✅
17	Tổng hợp	Dùng menu + API + biểu đồ		Hoạt động ổn định	Đúng	✅

A.4 Link GitHub / Host

GitHub Repository: https://github.com/ttn238/web_du_bao_thoi_tiet/

Host (Demo): https://ttn238.github.io/web_du_bao_thoi_tiet/

Domain: https://thoitietvn.infinityfreeapp.com