

**PHIẾU HỌC TẬP CSS CƠ BẢN**

**Họ và tên:** Trịnh Xuân Duy

**Mã sinh viên:** 2451271104

**Lớp:** 66ANM1

**Ngày làm:** 23/02/2026



**PHIẾU HỌC TẬP --- CÚ PHÁP CSS & BỘ CHỌN (SELECTORS)**

**Tài liệu đã xem (Link):**

**1. Kiến thức cốt lõi (Tóm tắt bằng lời của bạn)**

**1.1. CSS là gì? (Viết bằng lời riêng của bạn)**

CSS là viết tắt của **Cascading Style Sheets**

Nó có vai trò gì trong 1 trang web?

* **HTML** chịu trách nhiệm: **Định nghĩa cấu trúc và nội dung cốt lõi của trang web** (như văn bản, hình ảnh, các tiêu đề, bảng và danh sách).
* **CSS** chịu trách nhiệm: **Định dạng lớp vỏ bên ngoài, màu sắc và bố cục** (làm cho trang web trở nên đẹp mắt, chuyên nghiệp và có thể hiển thị tốt trên nhiều thiết bị khác nhau).
* **JavaScript** chịu trách nhiệm: **Xử lý các tương tác và hành vi động** (như phản hồi khi người dùng click chuột, kiểm tra dữ liệu form, cập nhật nội dung mà không cần tải lại trang).

* **1.2. Cú pháp cơ bản của CSS**

Một quy tắc CSS (CSS Rule) gồm 2 phần:

selector {  
 property: value;  
}

**Giải thích:**

* **Selector (bộ chọn):** Là thành phần dùng để **xác định hoặc "trỏ" đến phần tử HTML** mà bạn muốn áp dụng quy tắc định dạng (ví dụ: thẻ h1, lớp .box, hoặc định danh #header).
* **Property (thuộc tính):** Là **khía cạnh thiết kế** mà bạn muốn thay đổi của phần tử đó (ví dụ: color để đổi màu chữ, font-size để chỉnh cỡ chữ, margin để chỉnh khoảng cách).
* **Value (giá trị):** Là **thông số hoặc lựa chọn cụ thể** được gán cho thuộc tính (ví dụ: red, 20px, center).

**1.3. Ba cách áp dụng CSS vào HTML**

Hoàn thành bảng sau:

| **Cách áp dụng** | **Vị trí code** | **Ưu điểm** | **Nhược điểm** |
| --- | --- | --- | --- |
| **Inline CSS** | Viết trực tiếp trong thuộc tính style của thẻ HTML cụ thể. | Áp dụng nhanh cho một phần tử duy nhất, có độ ưu tiên cao nhất. | Code bị rối, khó bảo trì và không thể tái sử dụng cho các phần tử khác. |
| **Internal CSS** | Viết bên trong cặp thẻ <style> đặt trong phần <head> của file HTML. | Quản lý phong cách của một trang web duy nhất một cách tập trung, không cần file rời. | Chỉ có tác dụng trên một trang, làm tăng dung lượng file HTML. |
| **External CSS** | Viết trong một file riêng biệt có đuôi .css và liên kết bằng thẻ <link>. | Tách biệt hoàn toàn nội dung và giao diện; một file CSS có thể dùng cho nhiều trang web khác nhau. | Cần thêm một yêu cầu tải file từ trình duyệt, có thể làm chậm tốc độ hiển thị ban đầu một chút. |

**2. Các loại Selector cơ bản**

Hoàn thành bảng sau:

| **Loại Selector** | **Cú pháp** | **Ví dụ code** | **Ý nghĩa** |
| --- | --- | --- | --- |
| **Universal Selector** | \* | \* { margin: 0; } | Chọn tất cả các phần tử trong trang HTML. |
| **Type Selector** | element | p { color: blue; } | Chọn tất cả các phần tử theo tên thẻ HTML. |
| **Class Selector** | .className | .btn { padding: 10px; } | Chọn các phần tử có thuộc tính class tương ứng. |
| **ID Selector** | #idName | #header { background: gray; } | Chọn duy nhất một phần tử có thuộc tính id tương ứng. |
| **Descendant Selector** | parent child | nav a { text-decoration: none; } | Chọn các phần tử child nằm bên trong parent. |
| **Group Selector** | selector1, selector2 | h1, h2, h3 { font-family: Arial; } | Chọn nhiều selector cùng lúc để áp dụng chung một kiểu CSS. |
| **Pseudo-class Selector** | selector:state | a:hover { color: red; } | Chọn phần tử ở trạng thái đặc biệt (hover, focus, active, visited,...). |



**3. Bài tập thực hành 1 (20 phút): Áp dụng CSS & Selector**

**Mục tiêu:**

* Tạo file style.css và liên kết vào HTML
* Dùng các loại selector: type, class, id, group, descendant, pseudo-class
* Thực hành styling cơ bản cho menu và header

**Cho trước file HTML (không sửa cấu trúc):**

<!DOCTYPE html>  
<html lang="vi">  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport" content="width=device-width, initial-scale=1.0">  
 <title>Lab CSS Cơ Bản</title>  
 <link rel="stylesheet" href="style.css">  
</head>  
<body>  
 <header id="main-header">  
 <h1>My Blog</h1>  
 </header>  
   
 <nav>  
 <a href="#" class="nav-link">Home</a>  
 <a href="#" class="nav-link">About</a>  
 <a href="#" class="nav-link">Contact</a>  
 </nav>  
   
 <main>  
 <section class="intro">  
 <h2>Chào mừng đến với Blog</h2>  
 <p>Đây là trang blog về lập trình web.</p>  
 </section>  
 </main>  
   
 <footer>  
 <p>&copy; 2026 My Blog. All rights reserved.</p>  
 </footer>  
</body>  
</html>

**Yêu cầu CSS (viết vào file style.css):**

**Bước 1:** Liên kết file CSS vào HTML bằng thẻ <link> trong <head>.

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

**Bước 2:** Viết CSS cho các yêu cầu sau:

/\* TODO 1: Universal selector \*/

\* {

margin: 0;

padding: 0;

}

/\* TODO 2: Type selector cho body \*/

body {

font-family: Arial;

background-color: #f5f5f5;

}

/\* TODO 3: ID selector cho #main-header \*/

#main-header {

background-color: #333;

color: white;

text-align: center;

padding: 20px;

}

/\* TODO 4: Class selector cho .nav-link \*/

.nav-link {

color: #333;

text-decoration: none;

padding: 10px 15px;

}

/\* TODO 5: Group selector cho h1, h2, h3 \*/

h1, h2, h3 {

font-family: 'Georgia', serif;

}

/\* TODO 6: Descendant selector cho nav a \*/

nav a {

display: inline-block;

margin: 0 10px;

}

/\* TODO 7: Pseudo-class selector cho a:hover \*/

a:hover {

color: #ff6600;

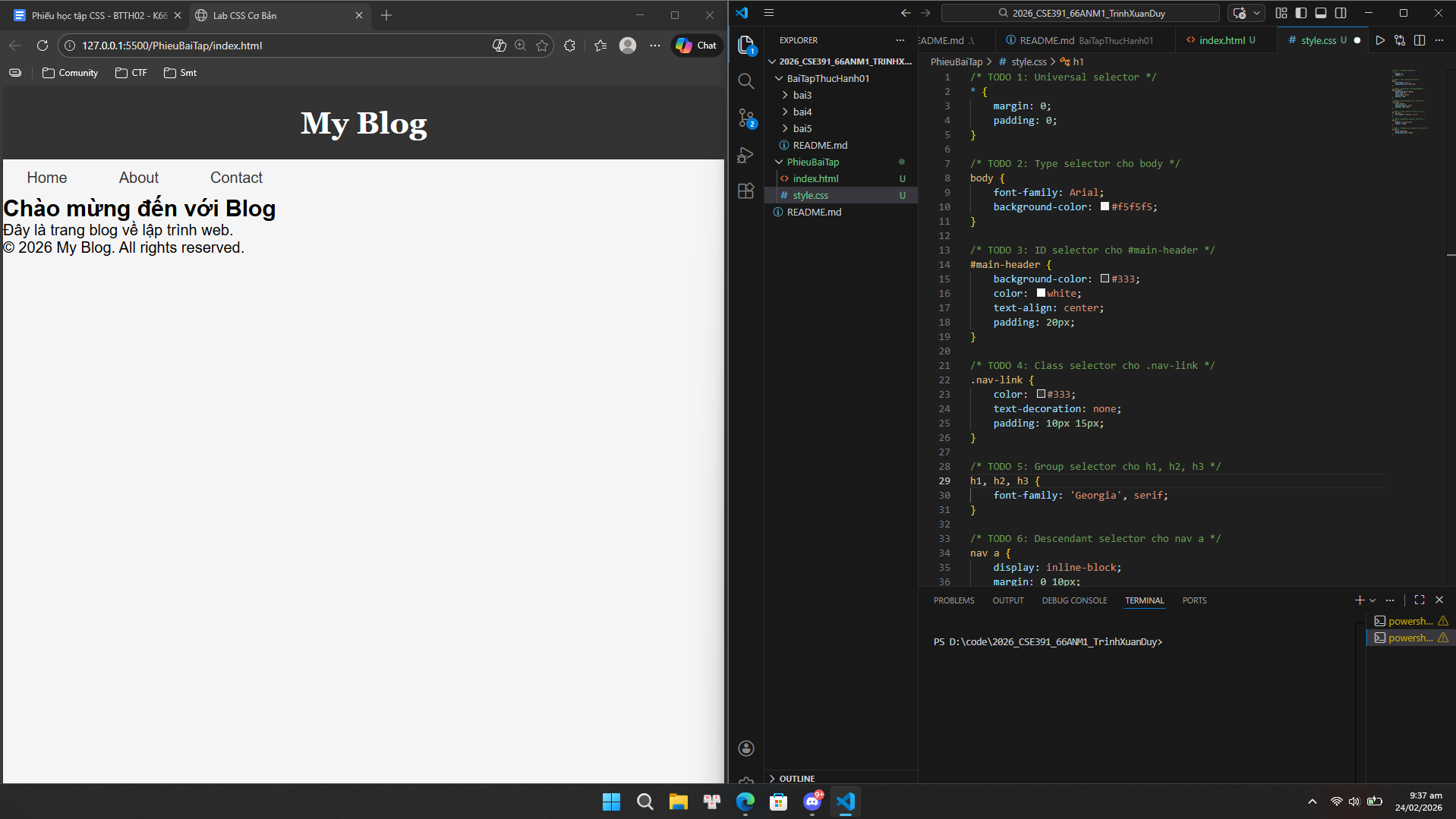
background-color: #eee;

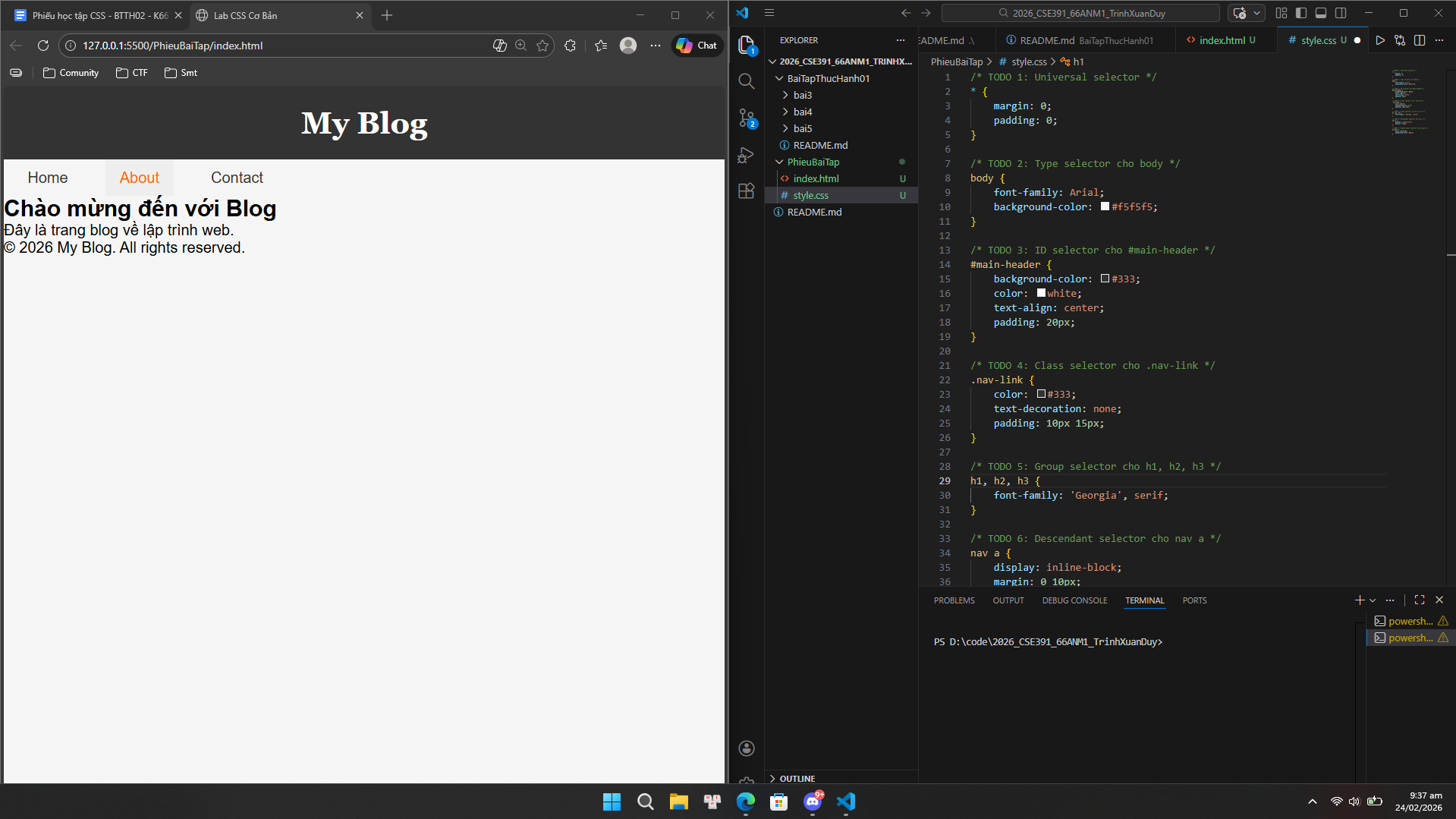
}

**Kết quả mong đợi:**

1. Menu nằm ngang, các link cách đều nhau
2. Khi rê chuột vào link, màu chữ và nền thay đổi
3. Header có nền tối, chữ trắng, căn giữa

**Ảnh kết quả (Dán ảnh trình duyệt vào đây):**

*(Dán ảnh 1: Giao diện bình thường)*

*(Dán ảnh 2: Khi hover vào link)*



**PHIẾU HỌC TẬP --- BOX MODEL & BỐ CỤC CƠ BẢN**

**Tài liệu đã xem (Link):**

**1. Kiến thức cốt lõi: CSS Box Model**

**1.1. Box Model gồm những thành phần gì?**

Vẽ sơ đồ Box Model (từ trong ra ngoài):

┌─────────────────────────────────────┐  
│ MARGIN (lề ngoài) │  
│ ┌───────────────────────────────┐ │  
│ │ BORDER (đường viền) │ │  
│ │ ┌─────────────────────────┐ │ │  
│ │ │ PADDING (lề trong) │ │ │  
│ │ │ ┌───────────────────┐ │ │ │  
│ │ │ │ CONTENT │ │ │ │  
│ │ │ │ (Nội dung) │ │ │ │  
│ │ │ └───────────────────┘ │ │ │  
│ │ └─────────────────────────┘ │ │  
│ └───────────────────────────────┘ │  
└─────────────────────────────────────┘

**Giải thích:**

* **Content**: Là **phần nội dung chính** của phần tử (văn bản, hình ảnh, video...).
* **Padding**: Là **khoảng cách giữa content và border**.
* **Border**: Là **đường viền bao quanh padding và content**.
* **Margin**: Là **khoảng cách bên ngoài border**, tạo khoảng cách giữa phần tử này với phần tử khác.

**1.2. Hai cách tính kích thước Box**

Hoàn thành bảng:

| **Thuộc tính** | **Cách tính width** | **Ưu/Nhược điểm** |
| --- | --- | --- |
| box-sizing: content-box; | width = content  Tổng chiều rộng thực tế = content + padding + border | *Mặc định của CSS*  *Dễ bị vỡ layout vì padding & border làm tăng kích thước thực tế* |
| box-sizing: border-box; | width = content + padding + border (tất cả nằm trong width đã khai báo) | Dễ kiểm soát kích thước  Không bị tràn layout khi thêm padding/border  Không phải mặc định (nhưng hiện nay dùng phổ biến) |

**Câu hỏi:** Nên dùng box-sizing nào cho toàn bộ trang? Tại sao?



**2. Bài tập thực hành 2 (35 phút): Box Model & Layout Card**

**Mục tiêu:**

* Tạo 3 card bài viết với box model rõ ràng
* Dùng margin, padding, border để tách biệt các card
* Áp dụng box-sizing: border-box
* Bố trí card theo hàng ngang (dùng display: inline-block hoặc float)

**Thêm HTML sau vào trong <main> (sau phần intro):**

<section class="posts">  
 <article class="card">  
 <h3>Bài viết 1</h3>  
 <p>Mô tả ngắn về bài viết 1.</p>  
 <a href="#" class="btn">Đọc thêm</a>  
 </article>  
   
 <article class="card">  
 <h3>Bài viết 2</h3>  
 <p>Mô tả ngắn về bài viết 2.</p>  
 <a href="#" class="btn">Đọc thêm</a>  
 </article>  
   
 <article class="card">  
 <h3>Bài viết 3</h3>  
 <p>Mô tả ngắn về bài viết 3.</p>  
 <a href="#" class="btn">Đọc thêm</a>  
 </article>  
</section>

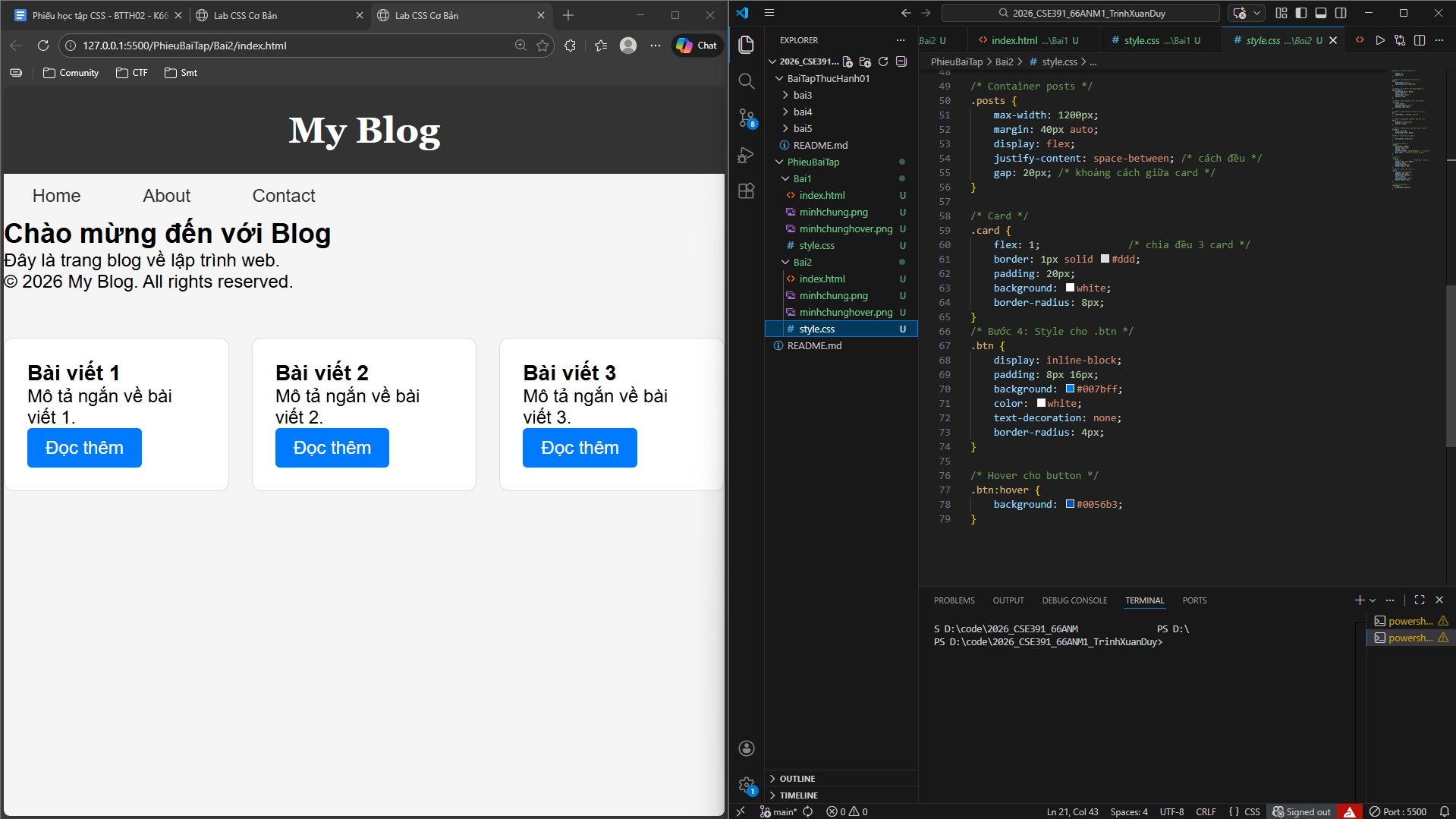
**Yêu cầu CSS:**

/\* Bước 1: Reset box-sizing cho toàn trang \*/  
\* {  
 box-sizing: border-box;  
}  
  
/\* Bước 2: Style cho .posts container \*/  
.posts {  
 /\* TODO: Căn giữa, margin trên dưới, max-width \*/  
}  
  
/\* Bước 3: Style cho .card \*/  
.card {  
 /\* TODO:   
 - width: 30% (hoặc calc(33.33% - 20px))  
 - display: inline-block  
 - border: 1px solid #ddd  
 - padding: 20px  
 - margin: 10px  
 - background: white  
 - border-radius: 8px  
 \*/  
}  
  
/\* Bước 4: Style cho .btn \*/  
.btn {  
 /\* TODO:   
 - display: inline-block  
 - padding: 8px 16px  
 - background: #007bff  
 - color: white  
 - text-decoration: none  
 - border-radius: 4px  
 \*/  
}  
  
.btn:hover {  
 /\* TODO: background: #0056b3 \*/  
}

**Kết quả mong đợi:**

1. Ba card nằm ngang trên 1 hàng
2. Mỗi card có viền, padding, margin rõ ràng
3. Nút "Đọc thêm" có màu xanh, hover đổi màu đậm hơn

**Ảnh kết quả (Dán ảnh vào đây):**

*(Dán ảnh: Giao diện 3 card)*

**PHIẾU HỌC TẬP --- FLEXBOX & CSS GRID**

**Tài liệu đã xem (Link):**

**1. Kiến thức cốt lõi: So sánh Flexbox và Grid**

Hoàn thành bảng sau:

| **Đặc điểm** | **Flexbox** | **CSS Grid** |
| --- | --- | --- |
| **Bố cục** | *1 chiều* | *2 chiều* |
| **Phù hợp với** | *Sắp xếp các thành phần nhỏ, căn chỉnh nội dung bên trong một phần tử chứa (container), hoặc phân bổ khoảng trống linh hoạt.* | *Thiết lập cấu trúc tổng thể (layout) của trang web, hoặc các giao diện phức tạp, cần chia lưới (grid) rõ ràng, đối xứng.* |
| **Thuộc tính chính** | display: flex;  justify-content  align-items | display: grid;  grid-template-columns  grid-template-rows |
| **Ví dụ sử dụng** | *Thiết kế thanh điều hướng (Navbar), căn giữa nội dung tuyệt đối, hoặc sắp xếp một dãy các nút bấm (buttons).* | *Bố cục trang cơ bản (Header, Sidebar, Main Content, Footer), tạo lưới thư viện hình ảnh (Image Gallery).* |

**1.1. Flexbox - Các thuộc tính quan trọng**

.flex-container {  
 display: flex;  
   
 /\* Căn chỉnh theo trục ngang (main axis) \*/  
 justify-content: \_\_\_\_\_\_\_\_\_\_\_; /\* flex-start | center | space-between | space-around \*/  
   
 /\* Căn chỉnh theo trục dọc (cross axis) \*/  
 align-items: \_\_\_\_\_\_\_\_\_\_\_; /\* flex-start | center | flex-end | stretch \*/  
   
 /\* Cho phép xuống hàng \*/  
 flex-wrap: \_\_\_\_\_\_\_\_\_\_\_; /\* nowrap | wrap \*/  
   
 /\* Khoảng cách giữa các item \*/  
 gap: \_\_\_\_\_\_\_\_\_\_\_;  
}

**1.2. CSS Grid - Các thuộc tính quan trọng**

.grid-container {  
 display: grid;  
   
 /\* Định nghĩa số cột và kích thước \*/  
 grid-template-columns: \_\_\_\_\_\_\_\_\_\_\_; /\* repeat(3, 1fr) | 200px 1fr 2fr \*/  
   
 /\* Định nghĩa số hàng và kích thước \*/  
 grid-template-rows: \_\_\_\_\_\_\_\_\_\_\_;  
   
 /\* Khoảng cách giữa các ô \*/  
 gap: \_\_\_\_\_\_\_\_\_\_\_;  
}



**2. Bài tập thực hành 3A (20 phút): Flexbox - Menu & Card Layout**

**Mục tiêu:**

* Dùng Flexbox cho menu ngang
* Dùng Flexbox cho layout card (thay thế inline-block)
* Responsive: card xuống 1 cột khi màn hình nhỏ

**Yêu cầu CSS (cải tiến code cũ):**

/\* --- Thiết lập cơ bản --- \*/

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

}

body {

font-family: Arial, sans-serif;

background-color: #f5f5f5;

line-height: 1.6;

}

#main-header {

background-color: #333;

color: white;

text-align: center;

padding: 20px;

}

h1 {

font-family: 'Georgia', serif;

}

/\* --- TODO: Cải tiến nav bằng Flexbox --- \*/

nav {

display: flex;

justify-content: center;

align-items: center;

gap: 20px;

background: #f8f9fa;

padding: 15px;

}

.nav-link {

color: #333;

text-decoration: none;

padding: 5px 10px;

transition: 0.3s;

}

.nav-link:hover {

color: #ff6600;

background-color: #eee;

border-radius: 4px;

}

/\* --- Section Intro --- \*/

.intro {

text-align: center;

padding: 40px 20px;

}

/\* --- TODO: Cải tiến .posts bằng Flexbox --- \*/

.posts {

display: flex;

flex-wrap: wrap;

gap: 20px;

justify-content: center;

max-width: 1200px;

margin: 40px auto;

padding: 0 20px;

}

/\* --- TODO: Cải tiến .card --- \*/

.card {

flex: 1 1 300px; /\* Responsive tự động: tối thiểu 300px, co giãn linh hoạt \*/

border: 1px solid #ddd;

padding: 20px;

background: white;

border-radius: 8px;

box-shadow: 0 2px 5px rgba(0,0,0,0.1);

}

/\* --- Button Style --- \*/

.btn {

display: inline-block;

margin-top: 15px;

padding: 8px 16px;

background: #007bff;

color: white;

text-decoration: none;

border-radius: 4px;

transition: background 0.3s;

}

.btn:hover {

background: #0056b3;

}

/\* --- Footer --- \*/

footer {

text-align: center;

padding: 20px;

background: #333;

color: white;

margin-top: 40px;

}

/\* --- TODO: Responsive (Màn hình < 768px) --- \*/

@media (max-width: 768px) {

.posts {

flex-direction: column;

align-items: center;

}

.card {

flex: 1 1 100%;

width: 100%; /\* Đảm bảo card chiếm hết chiều ngang \*/

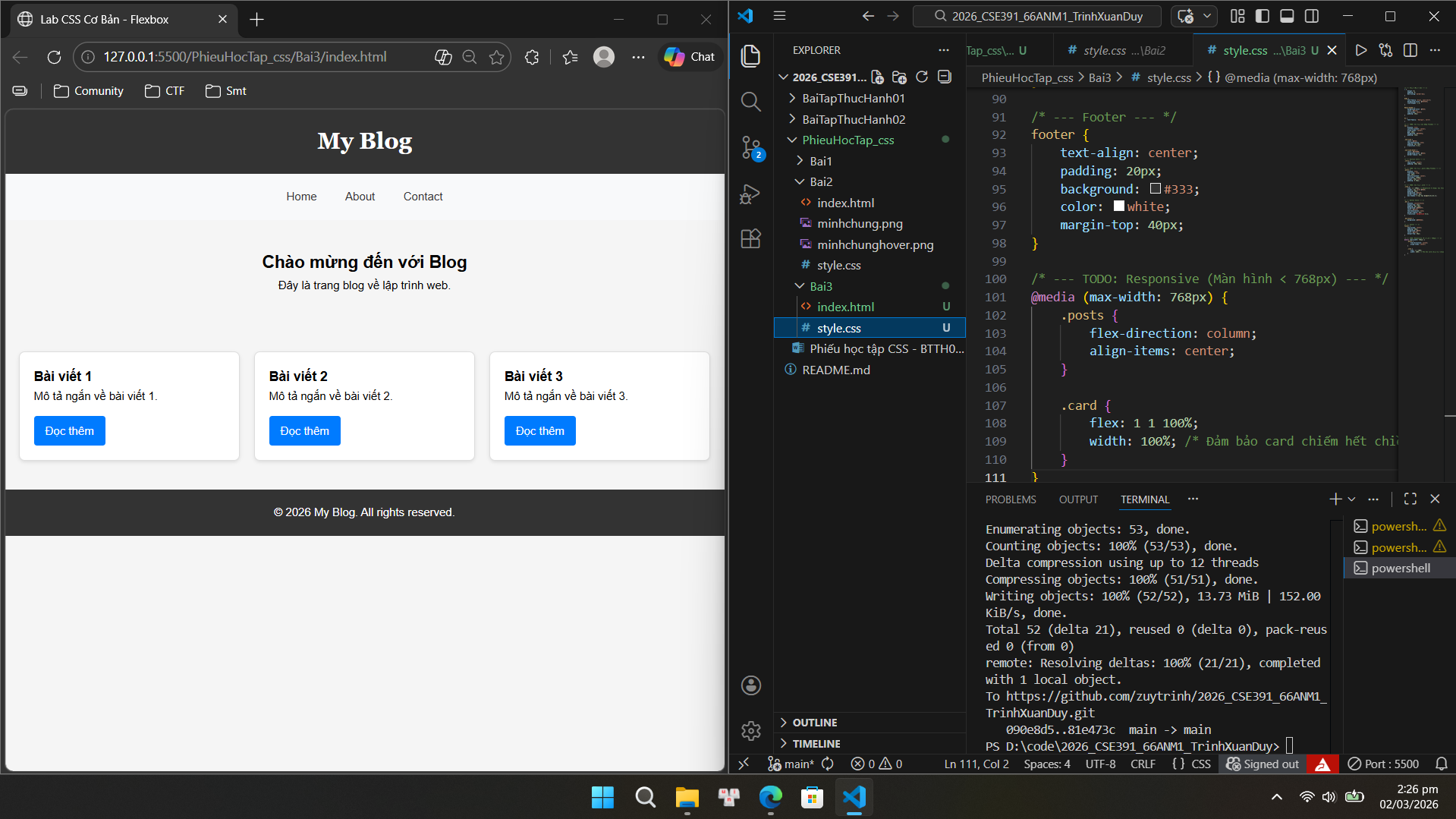
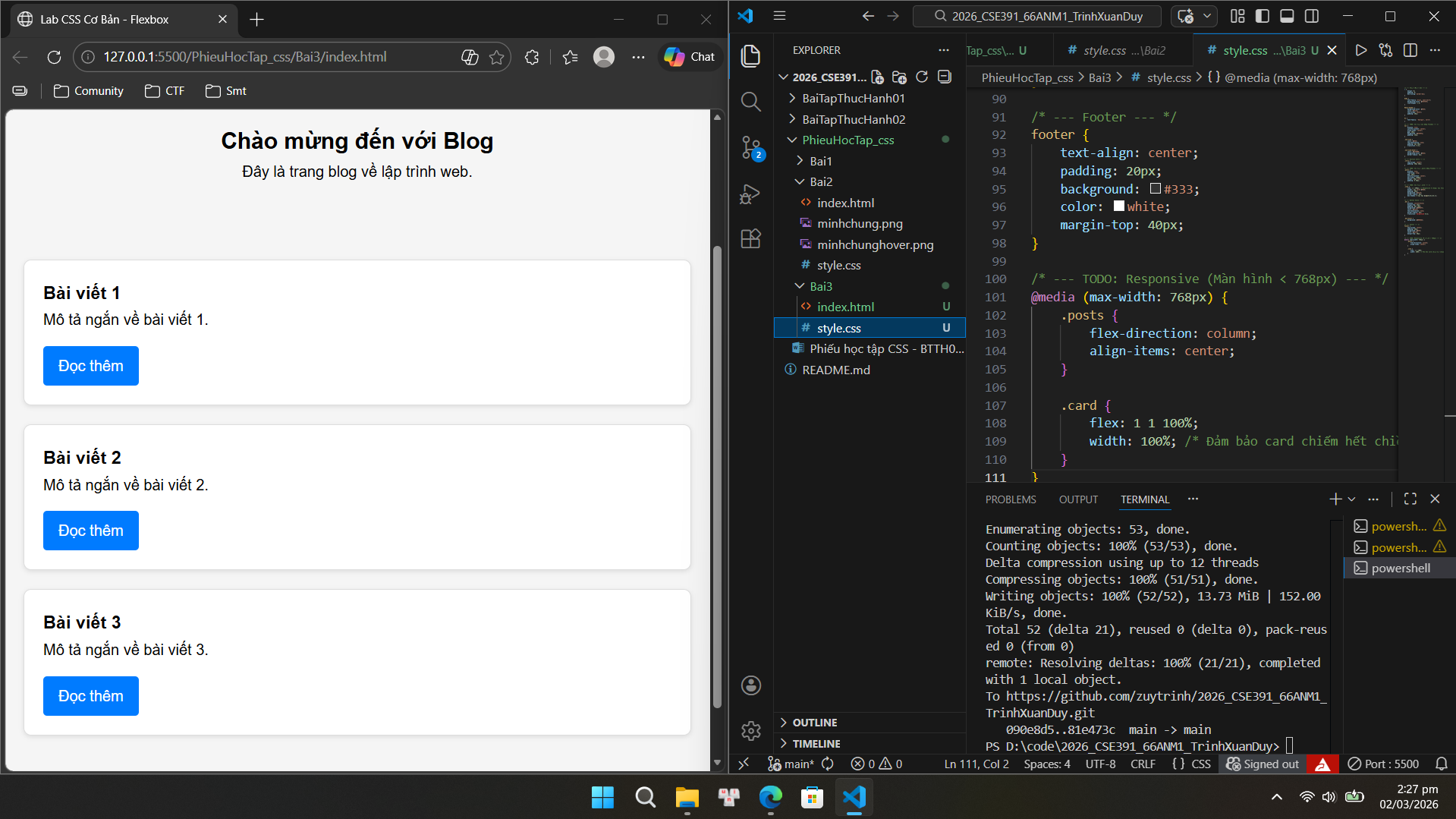
}

}

**Kết quả mong đợi:**

1. Menu các link nằm ngang, căn giữa, cách đều
2. Card layout linh hoạt, tự động xuống hàng khi hết chỗ
3. Khi thu nhỏ màn hình < 768px, card xếp thành 1 cột

**Ảnh kết quả (Dán ảnh vào đây):**

*(Dán ảnh 1: Desktop view - 3 card ngang)*  
*(Dán ảnh 2: Mobile view - card xếp dọc)*



**3. Bài tập thực hành 3B (20 phút): CSS Grid - Gallery Layout**

**Mục tiêu:**

* Tạo lưới 2 hàng x 3 cột bằng CSS Grid
* Responsive: tự động giảm số cột khi màn hình nhỏ

**Thêm HTML sau vào cuối <main>:**

<section class="gallery">  
 <div class="item">1</div>  
 <div class="item">2</div>  
 <div class="item">3</div>  
 <div class="item">4</div>  
 <div class="item">5</div>  
 <div class="item">6</div>  
</section>

**Yêu cầu CSS:**

.gallery {

display: grid;

/\* Tạo 3 cột bằng nhau mặc định \*/

grid-template-columns: repeat(3, 1fr);

gap: 16px;

max-width: 1200px;

margin: 40px auto;

padding: 0 20px;

}

.item {

background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);

color: white;

display: flex;

justify-content: center;

align-items: center;

height: 150px;

font-size: 2rem;

border-radius: 8px;

transition: transform 0.3s ease;

}

.item:hover {

transform: scale(1.05); /\* Hiệu ứng nhẹ khi di chuột \*/

}

/\* --- Responsive: Tự động giảm cột khi màn hình nhỏ --- \*/

@media (max-width: 900px) {

.gallery {

/\* Dùng auto-fit kết hợp minmax để tự nhảy dòng khi cột < 200px \*/

grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));

}

}

@media (max-width: 480px) {

.gallery {

/\* Trên màn hình rất nhỏ, đảm bảo item không bị quá bé \*/

grid-template-columns: 1fr;

}

}

**Giải thích kỹ thuật:**

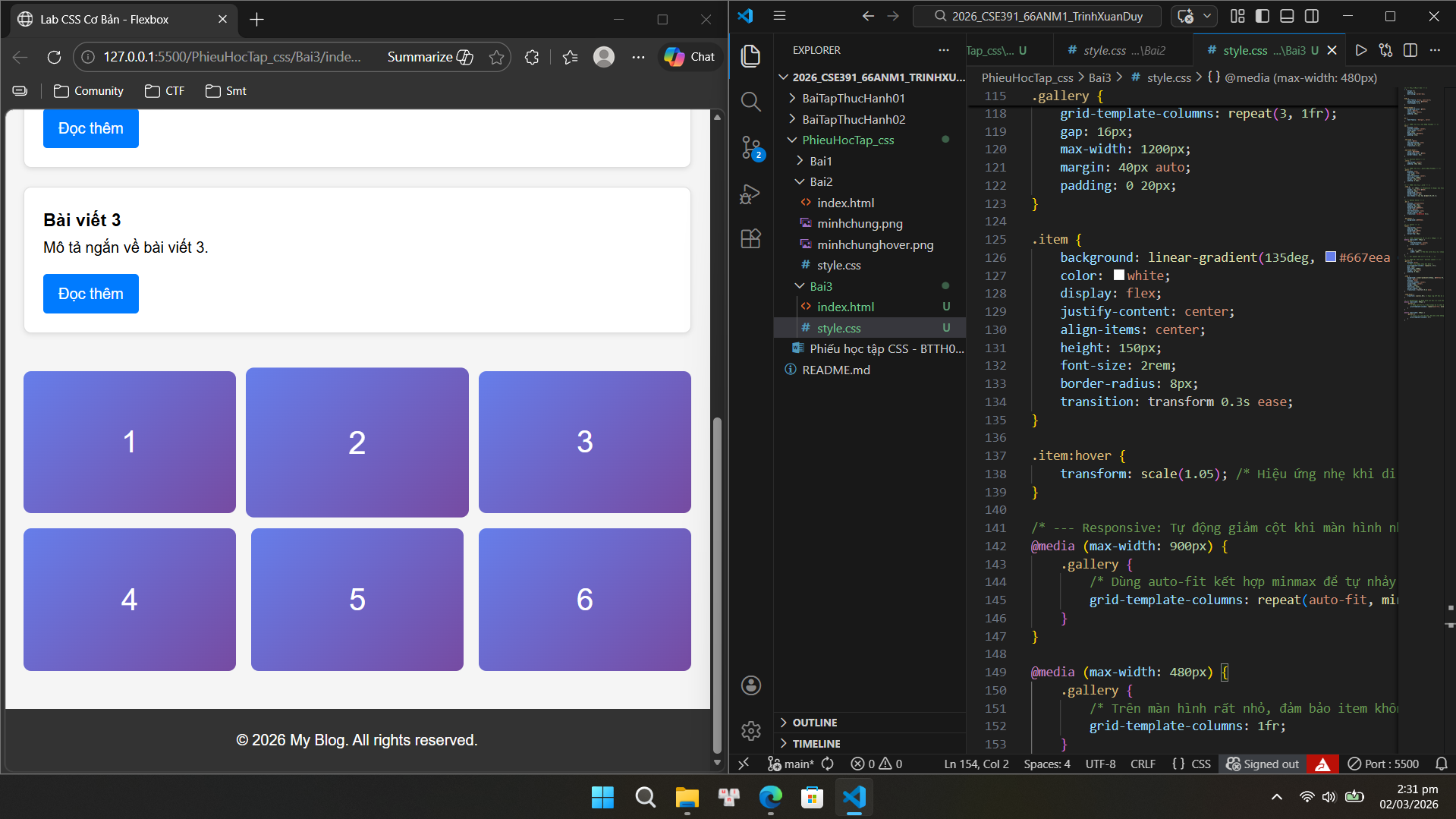
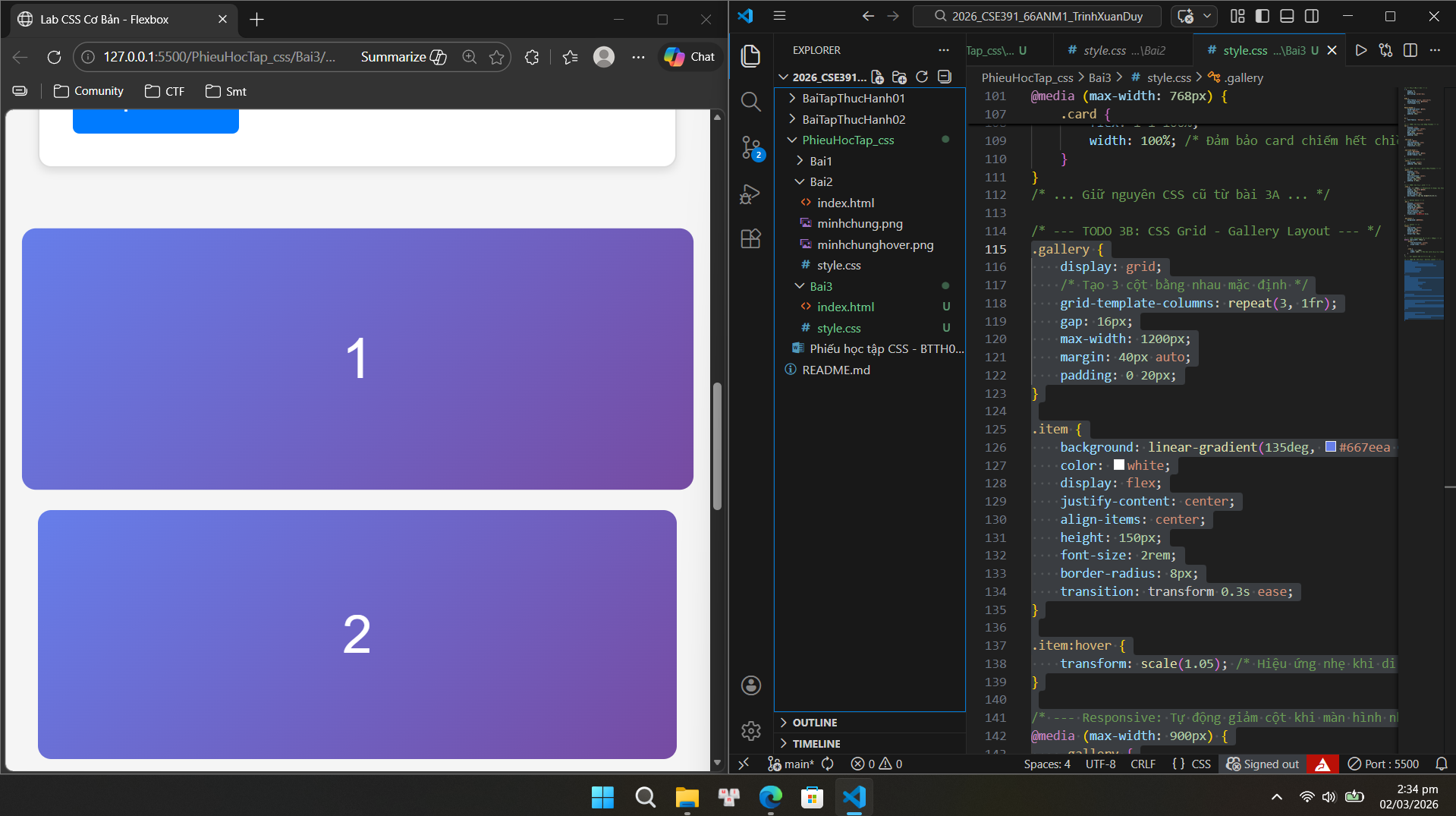
**repeat(auto-fit, minmax(200px, 1fr))** có nghĩa gì?

* repeat(): Lặp lại số cột/hàng theo số lần chỉ định.
* auto-fit: Tự động tính số cột để lấp đầy chiều ngang của hàng.
* minmax(200px, 1fr): Giới hạn kích thước: tối thiểu 200px và tối đa là chiếm hết phần còn lại (1fr).

**Kết quả mong đợi:**

1. Desktop: 3 cột, 2 hàng (6 ô)
2. Tablet: 2 cột, 3 hàng
3. Mobile: 1 cột, 6 hàng

**Ảnh kết quả (Dán ảnh vào đây):**

*(Dán ảnh 1: Desktop - 3 cột)******(Dán ảnh 2: Mobile - 1 cột)*



**PHIẾU HỌC TẬP --- ĐỊNH VỊ CSS & RESPONSIVE DESIGN**[**[2]**](#s2ls7lhhivk2)

**Tài liệu đã xem (Link):**

**1. Kiến thức cốt lõi: CSS Position**

Hoàn thành bảng:

| **Giá trị** | **Ý nghĩa** | **Ví dụ sử dụng** |
| --- | --- | --- |
| static | *Mặc định, phần tử xuất hiện theo luồng tự nhiên của trang.* | *Không bị ảnh hưởng bởi top, bottom, left, right.* |
| relative | *Di chuyển phần tử so với vị trí gốc của chính nó.* | Làm "mốc" (cha) cho phần tử con dùng absolute. |
| absolute | Định vị so với phần tử cha gần nhất có position (khác static). | *Đặt icon lên góc ảnh hoặc làm menu dropdown.* |
| fixed | *Cố định so với cửa sổ trình duyệt (viewport).* | *Thanh menu (Navbar) hoặc nút "Lên đầu trang" luôn hiện diện.* |
| sticky | *Giữ phần tử ở vị trí cụ thể khi cuộn trang (kết hợp giữa relative và fixed).* | *Tiêu đề bảng hoặc mục lục luôn dính ở trên cùng khi cuộn.* |

**2. Responsive Design - Media Queries**

**Cú pháp cơ bản:**

/\* Desktop First \*/  
@media (max-width: 768px) {  
 /\* Code CSS cho màn hình nhỏ hơn 768px \*/  
}  
  
/\* Mobile First \*/  
@media (min-width: 768px) {  
 /\* Code CSS cho màn hình lớn hơn 768px \*/  
}

**Breakpoints phổ biến:**

* **Mobile**: < 576px
* **Tablet**: 576px - 768px
* **Desktop**: > 768px
* **Large Desktop**: > 1200px



**3. Bài tập thực hành 4 (30 phút): Tạo Header Sticky + Responsive**

**Yêu cầu:**

1. Header cố định trên cùng khi scroll (sticky)
2. Thêm nút "Scroll to Top" (fixed, góc dưới bên phải)
3. Responsive: Menu chuyển icon burger khi mobile (chỉ làm style, không cần JavaScript)

**Cập nhật HTML:**

<header id="main-header" class="sticky-header">  
 <h1>My Blog</h1>  
</header>  
  
<!-- Nút Scroll to Top -->  
<a href="#" class="scroll-top">↑</a>

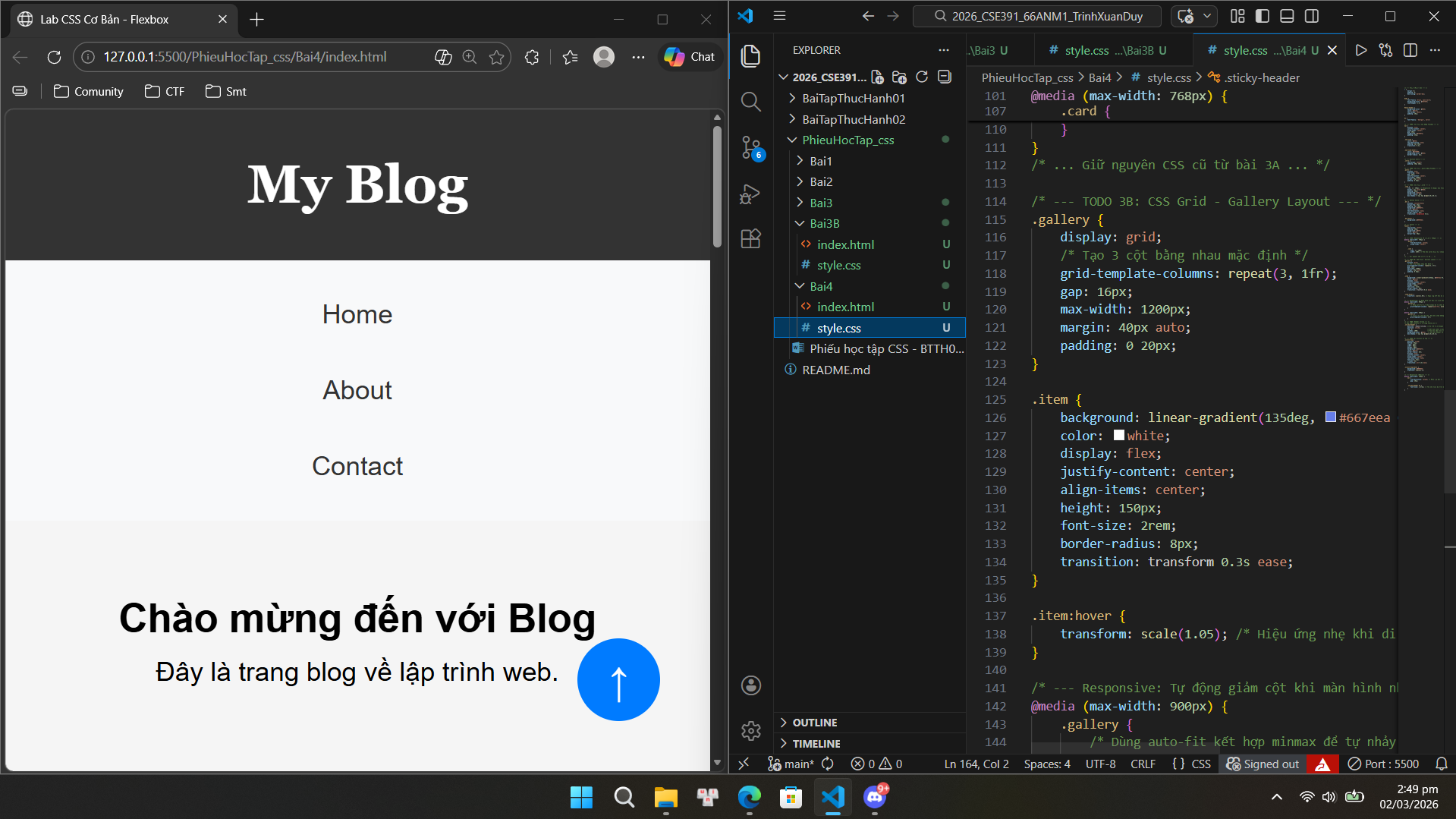
**Yêu cầu CSS:**

/\* Header sticky \*/  
.sticky-header {  
 position: sticky;  
 top: 0;  
 z-index: 100;  
 /\* TODO: Thêm box-shadow để tạo hiệu ứng nổi \*/  
}  
  
/\* Nút Scroll to Top \*/  
.scroll-top {  
 position: fixed;  
 bottom: 30px;  
 right: 30px;  
 /\* TODO:  
 - width: 50px, height: 50px  
 - background: #007bff  
 - color: white  
 - border-radius: 50%  
 - display: flex, justify/align center  
 - text-decoration: none  
 - font-size: 24px  
 \*/  
}  
  
.scroll-top:hover {  
 /\* TODO: background: #0056b3, transform: scale(1.1) \*/  
}  
  
/\* Responsive \*/  
@media (max-width: 768px) {  
 nav {  
 /\* TODO: flex-direction: column \*/  
 }  
}

**Kết quả mong đợi:**

1. Header luôn cố định trên cùng khi scroll
2. Nút "↑" ở góc dưới bên phải, hover phóng to
3. Mobile: menu xếp dọc

**Ảnh kết quả (Dán ảnh vào đây):**

*(Dán ảnh: Giao diện có header sticky và nút scroll)*

**MINI TEST - Tự Kiểm Tra**

**Phần 1: Selector & Cú pháp**

**1. Selector nào chọn tất cả thẻ <p> nằm trong <div>?**

* A. div p
* B. div > p
* C. div + p

**2. Pseudo-class :hover dùng để?**

* A. Chọn phần tử đầu tiên
* B. Chọn phần tử khi rê chuột
* C. Chọn phần tử cuối cùng

**Phần 2: Box Model**

**3. box-sizing: border-box có nghĩa?**

* A. Width bao gồm content + padding + border
* B. Width chỉ tính content
* C. Width tính cả margin

**4. Thứ tự từ trong ra ngoài của Box Model?**

* A. Content → Padding → Border → Margin
* B. Content → Border → Padding → Margin
* C. Content → Margin → Padding → Border

**Phần 3: Flexbox & Grid**

**5. Flexbox phù hợp với layout?**

* A. 1 chiều (hàng hoặc cột)
* B. 2 chiều (hàng và cột)
* C. Không phù hợp với layout

**6. justify-content: space-between có tác dụng?**

* A. Căn giữa các item
* B. Chia đều khoảng trống giữa các item
* C. Chia đều khoảng trống cả 2 bên

**7. Grid tốt hơn Flexbox khi nào?**

* A. Menu ngang đơn giản
* B. Layout phức tạp nhiều hàng cột
* C. Không có sự khác biệt

**Phần 4: Position & Responsive**

**8. position: fixed có đặc điểm?**

* A. Cố định so với viewport
* B. Cố định so với phần tử cha
* C. Không cuộn theo trang

**9. Media query @media (max-width: 768px) áp dụng cho?**

* A. Màn hình > 768px
* B. Màn hình ≤ 768px
* C. Chỉ màn hình = 768px



**Đáp án Mini Test**

**👉 Nhấn vào đây để xem đáp án**

**Phần 1:**

1. **A** - div p (descendant selector)
2. **B** - Chọn phần tử khi rê chuột

**Phần 2:**  
3. **A** - Width bao gồm content + padding + border  
4. **A** - Content → Padding → Border → Margin

**Phần 3:**  
5. **A** - 1 chiều (hàng hoặc cột)  
6. **B** - Chia đều khoảng trống giữa các item  
7. **B** - Layout phức tạp nhiều hàng cột

**Phần 4:**  
8. **A** - Cố định so với viewport  
9. **B** - Màn hình ≤ 768px



**📌 TÓM TẮT CHƯƠNG (Self-Reflection)**

**Những gì tôi đã hiểu:**

1. Cách sử dụng **Flexbox** để căn giữa menu và tạo bố cục thẻ (card) linh hoạt.
2. Cách dùng **CSS Grid** (repeat, minmax, auto-fit) để tạo lưới ảnh tự động co giãn theo màn hình.
3. Sự khác biệt giữa **Position Fixed** (cố định nút bấm) và **Position Sticky** (dính header khi cuộn).

**Những gì tôi vẫn chưa rõ:**

1. Cách xử lý xung đột khi thuộc tính overflow của thẻ cha làm mất hiệu ứng sticky của thẻ con.
2. Cách kết hợp nhuần nhuyễn giữa đơn vị fr trong Grid và % trong Flexbox để tối ưu giao diện.

**Mục tiêu tiếp theo:**

Học cách sử dụng **JavaScript** cơ bản để xử lý sự kiện click cho nút "Scroll to Top" và menu burger trên mobile.



**Ghi chú của giáo viên:**

**Ngày hoàn thành:** 02/03/2026

**Chữ ký sinh viên:** Duy

