**PHIẾU HỌC TẬP CSS CƠ BẢN**

**Họ và tên:** ĐẶNG QUỐC CƯỜNG

**Mã sinh viên:** 2451271083

**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: Cấu trúc và Nội dung (Khung sườn của ngôi nhà)
* CSS chịu trách nhiệm: Giao diện & Trình bày (Sơn màu & Nội thất)
* JavaScript chịu trách nhiệm: Hành vi & Tương tác (Hệ thống điện & nước)

**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): áp dụng style cho tất cả các phần tử HTML có tên thẻ tương ứng trong trang.
* **Property** (thuộc tính): thuộc tính áp dụng cho thẻ
* **Value** (giá trị): giá trị của property áp dụng

**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** | *Nằm trực tiếp bên trong thẻ mở của một phần tử HTML (sử dụng thuộc tính style="...").* | *Áp dụng nhanh chóng; Có độ ưu tiên cao nhất (dễ dàng ghi đè các luật CSS khác).* | *Làm mã HTML trở nên lộn xộn, khó đọc; Cực kỳ khó bảo trì và không thể tái sử dụng cho các phần tử khác.* |
| **Internal CSS** | *Nằm bên trong cặp thẻ <style>, được đặt ở phần <head> của trang HTML.* | *Gom mã CSS về một chỗ nên dễ quản lý hơn Inline; Không cần tạo thêm file vật lý mới.* | *Chỉ có tác dụng trên một trang HTML duy nhất; Không thể dùng chung cho các trang khác của website.* |
| **External CSS** | *Nằm ở một file có đuôi .css riêng biệt, được nhúng vào file HTML bằng thẻ <link> trong phần <head>.* | *Tách biệt hoàn toàn nội dung (HTML) và giao diện (CSS); Có thể tái sử dụng cho hàng trăm trang web khác nhau; Trình duyệt tự lưu bộ nhớ đệm (cache) giúp web tải nhanh hơn.* | *Tốn thêm một yêu cầu tải (HTTP request) từ máy chủ để lấy file CSS về trong lần truy cập đầu tiên.* |

**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 HTML có trên trang web. Thường dùng để reset CSS mặc định của trình duyệt.* |
| **Type Selector** | element | p { color: blue; } | *Chọn tất cả các phần tử dựa trên tên thẻ HTML (ví dụ: chọn tất cả các đoạn văn bản <p>).* |
| **Class Selector** | .className | .btn { padding: 10px; } | *Chọn tất cả các phần tử có thuộc tính class khớp với tên được khai báo. Một class có thể được dùng chung cho nhiều phần tử khác nhau.* |
| **ID Selector** | #idName | #header { background: gray; } | *Chọn một phần tử duy nhất mang thuộc tính id khớp với tên được khai báo. Lưu ý: ID là định danh duy nhất, không được trùng lặp trên cùng một trang HTML.* |
| **Descendant Selector** | parent child | nav a { text-decoration: none; } | *Chọn tất cả các phần tử con (child) nằm lồng bên trong một phần tử cha (parent) được chỉ định.* |
| **Group Selector** | selector1, selector2 | h1, h2, h3 { font-family: Arial; } | *Chọn nhiều phần tử khác nhau cùng một lúc để áp dụng chung một tập hợp các quy tắc CSS. Các selector được gộp lại và phân tách bằng dấu phẩy. Việc này giúp rút ngắn mã CSS và tránh phải viết lặp lại các thuộc tính giống nhau.* |
| **Pseudo-class Selector** | selector:state | a:hover { color: red; } | *Chọn và thay đổi giao diện của phần tử dựa trên một trạng thái đặc biệt hoặc tương tác của người dùng thay vì cấu trúc HTML tĩnh. Ví dụ: đổi màu khi người dùng di chuột qua (:hover), khi người dùng bấm vào (:active), hoặc nhắm vào phần tử đầu tiên của một danh sách (:first-child).* |

**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>  
 <!-- TODO: Liên kết file style.css vào đây -->  
</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>.

<!-- TODO: <link rel="stylesheet" href="style.css"> -->

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

/\* TODO 1: Dùng universal selector để reset margin và padding về 0 cho toàn trang \*/  
\* {

margin: 0;

padding: 0;

}  
/\* TODO 2: Dùng type selector cho body \*/  
/\* Yêu cầu: font-family: Arial; background-color: #f5f5f5; \*/  
body {

font-family: Arial;

background-color: #f5f5f5;

}  
/\* TODO 3: Dùng ID selector cho #main-header \*/  
/\* Yêu cầu: background-color: #333; color: white; text-align: center; padding: 20px; \*/  
#main-header {

background-color: #333;

color: white;

text-align: center;

padding: 20px;

}  
/\* TODO 4: Dùng class selector cho .nav-link \*/  
/\* Yêu cầu: color: #333; text-decoration: none; padding: 10px 15px; \*/  
.nav-link {

color: #333;

text-decoration: none;

padding: 10px 15px;

}  
/\* TODO 5: Dùng group selector cho h1, h2, h3 \*/  
/\* Yêu cầu: font-family: 'Georgia', serif; \*/  
h1, h2, h3 {

font-family: 'Georgia', serif;

}  
/\* TODO 6: Dùng descendant selector cho nav a \*/  
/\* Yêu cầu: display: inline-block; margin: 0 10px; \*/  
nav a {

display: inline-block;

margin: 0 10px;

}  
/\* TODO 7: Dùng pseudo-class selector cho a:hover \*/  
/\* Yêu cầu: color: #ff6600; background-color: #eee; \*/  
a:hover {

color: #ff6600;

background-color: #eee;

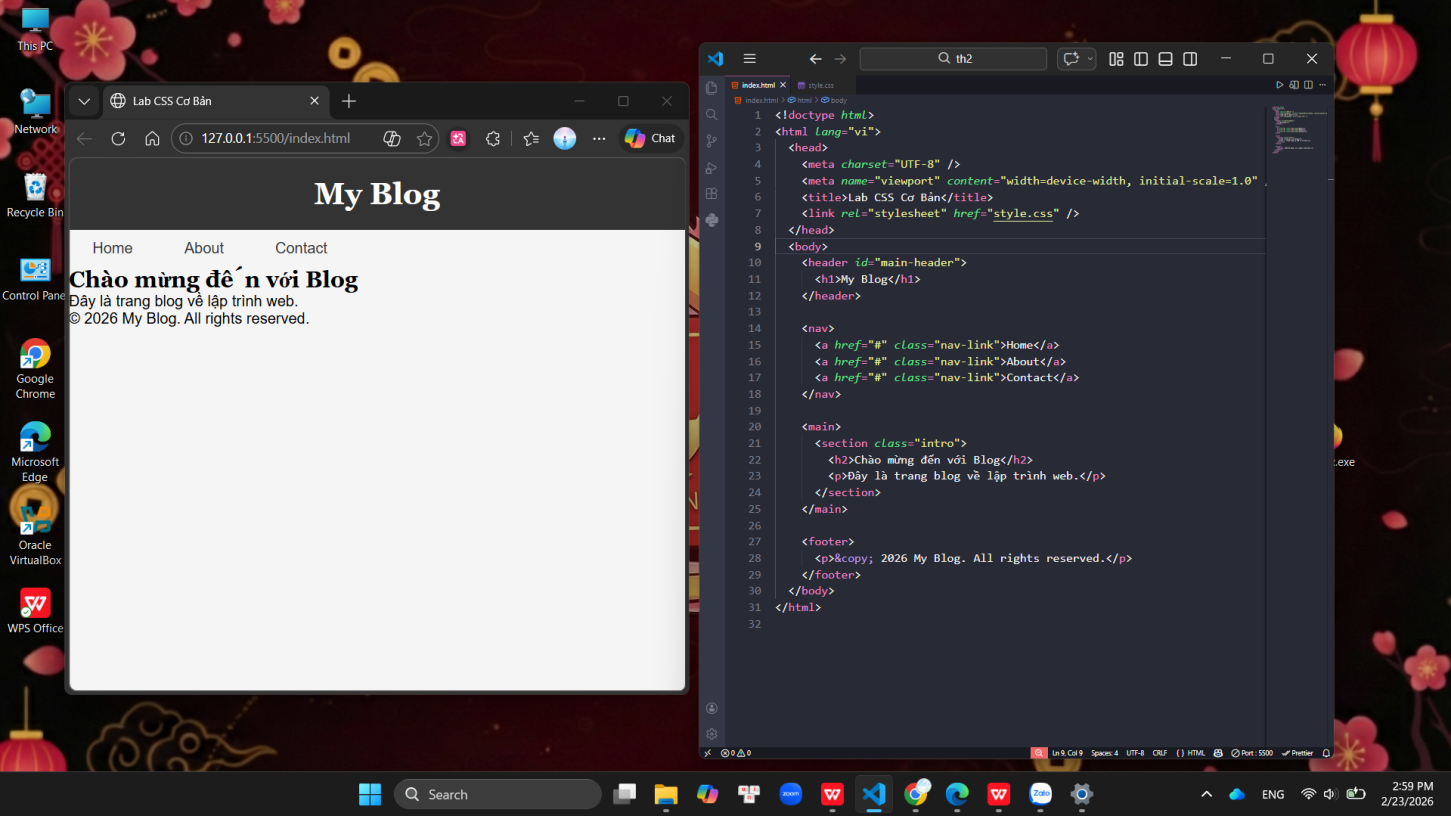
}

**Kết quả mong đợi:**

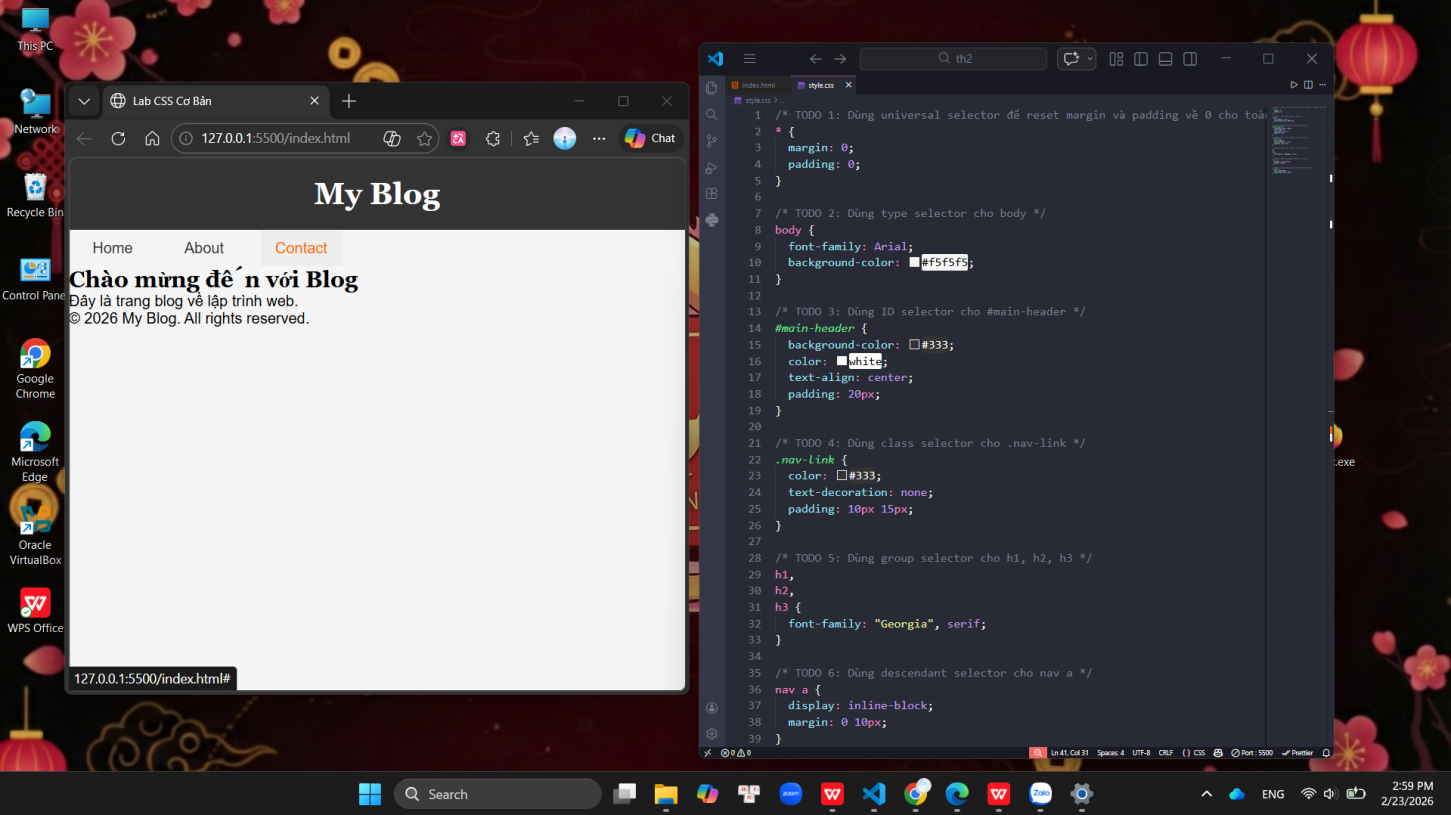
1. Menu nằm ngang, các link cách đều nhau
2. Khi rê chuột vào link, màu chữ và nền thay đổi
3. Header có nền tối, chữ trắng, căn giữa

**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) │ <- Tạo khoảng cách với hàng xóm  
│ ┌───────────────────────────────┐ │  
│ │ BORDER (đường viền) │ │ <- Đường bao quanh  
│ │ ┌─────────────────────────┐ │ │  
│ │ │ PADDING (lề trong) │ │ │ <- Khoảng đệm bảo vệ nội dung  
│ │ │ ┌───────────────────┐ │ │ │  
│ │ │ │ CONTENT │ │ │ │ <- Text, Ảnh  
│ │ │ │ (Nội dung) │ │ │ │  
│ │ │ └───────────────────┘ │ │ │  
│ │ └─────────────────────────┘ │ │  
│ └───────────────────────────────┘ │  
└─────────────────────────────────────┘

**Giải thích:**

* **Content**: Nơi chứa text, ảnh. Kích thước xác định bởi width và height.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
* **Padding**: Vùng đệm trong suốt bao quanh content. (Ví dụ: Chữ cách viền một chút cho dễ đọc).\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
* **Border**: Đường viền bao quanh padding.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
* **Margin**: Khoảng trống trong suốt bên ngoài border. Giúp đẩy các phần tử khác ra xa.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**1.2. Hai cách tính kích thước Box**

Hoàn thành bảng:

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Cách tính width** | **Ưu/Nhược điểm** |
| box-sizing: content-box; | ***Chiều rộng thực tế*** *= width + padding-left + padding-right + border-left + border-right)* | *Nhược: Phình to hơn kích thước mong muốn* |
| box-sizing: border-box; | ***Chiều rộng thực tế*** *= width* | *Ưu: Giữ nguyên box width mong muốn*  *Nhược: Làm nhỏ nội dung nếu thêm padding hoặc border dày* |

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

Nên dùng thuộc tính box-sizing: **border-box**; cho toàn bộ trang vì nó giúp cố định chính xác kích thước phần tử theo đúng thông số bạn đã khai báo. Bất kỳ lớp đệm (padding) hay đường viền (border) nào được thêm vào sẽ tự động lấn không gian vào bên trong hộp thay vì cộng dồn làm phình to kích thước ra ngoài, giúp kiểm soát bố cục dễ dàng và không bị vỡ giao diện.

**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 \*/

max-width: 1000px;

margin: 40px auto;

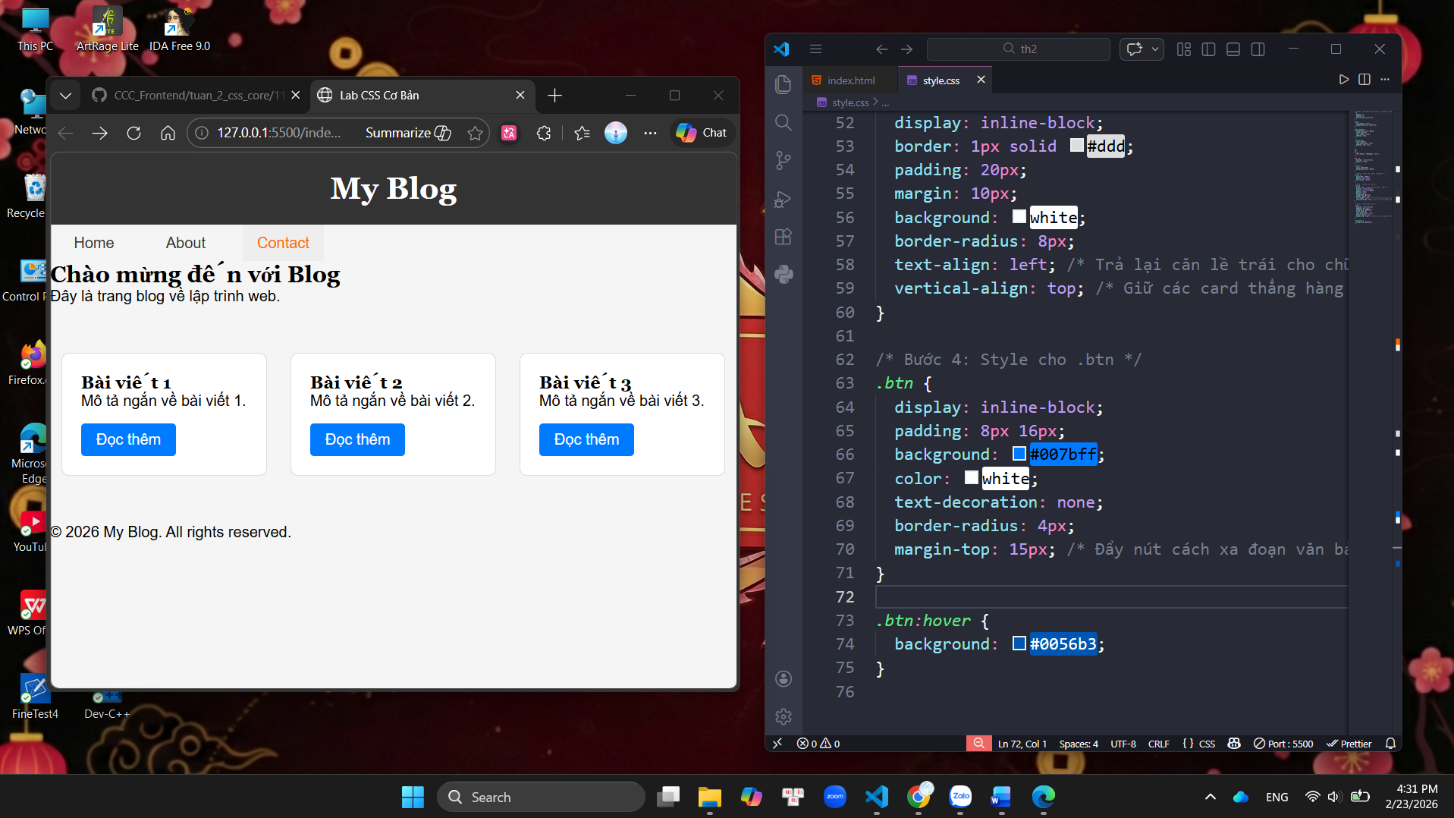
text-align: center;  
}  
  
/\* Bước 3: Style cho .card \*/  
.card {  
 /\* TODO:   
 - width: 30% (hoặc calc(33.33% - 20px))  
 - display: inline-block  
 - border: 1px solid #ddd  
 - padding: 20px  
 - margin: 10px  
 - background: white  
 - border-radius: 8px  
 \*/  
}  
  
/\* Bước 4: Style 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 (chỉ theo hàng* ***hoặc*** *cột)* | *2 chiều (kiểm soát cả hàng* ***và*** *cột cùng lúc)* |
| **Phù hợp với** | *Căn chỉnh chi tiết các thành phần nhỏ, các cụm component linh hoạt.* | *Thiết kế cấu trúc, chia khung bố cục tổng thể cho khối lớn hoặc toàn bộ trang.* |
| **Thuộc tính chính** | display: flex;  justify-content  align-items | display: grid;  grid-template-columns  grid-template-rows |
| **Ví dụ sử dụng** | *Thanh điều hướng (Menu/Navbar), danh sách các nút bấm, thanh công cụ.* | *Bố cục trang web cơ bản (Header, Sidebar, Main, Footer), thư viện ảnh dạng lưới (Gallery)* |

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

.flex-container {  
 display: flex;  
   
 /\* Căn chỉnh theo trục ngang (main axis) \*/  
 justify-content: space-between; /\* flex-start | center | space-between | space-around \*/  
   
 /\* Căn chỉnh theo trục dọc (cross axis) \*/  
 align-items: center; /\* flex-start | center | flex-end | stretch \*/  
   
 /\* Cho phép xuống hàng \*/  
 flex-wrap: wrap; /\* nowrap | wrap \*/  
   
 /\* Khoảng cách giữa các item \*/  
 gap: 20px;  
}

**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); /\* repeat(3, 1fr) | 200px 1fr 2fr \*/  
   
 /\* Định nghĩa số hàng và kích thước \*/  
 grid-template-rows: auto;  
   
 /\* Khoảng cách giữa các ô \*/  
 gap: 20px;  
}

**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):**

/\* Cải tiến nav bằng Flexbox \*/  
nav {  
 /\* TODO:  
 - display: flex  
 - justify-content: center  
 - align-items: center  
 - gap: 20px  
 - background: #f8f9fa  
 - padding: 15px  
 \*/  
}  
  
/\* Cải tiến .posts bằng Flexbox \*/  
.posts {  
 /\* TODO:  
 - display: flex  
 - flex-wrap: wrap  
 - gap: 20px  
 - justify-content: center (hoặc space-between)  
 - max-width: 1200px  
 - margin: 40px auto  
 \*/  
}  
  
.card {  
 /\* TODO:  
 - flex: 0 0 calc(33.33% - 20px)  
 - (hoặc flex: 1 1 300px cho responsive tự động)  
 \*/  
}  
  
/\* Responsive: Khi màn hình < 768px, card chiếm full width \*/  
@media (max-width: 768px) {  
 .posts {  
 /\* TODO: flex-direction: column \*/  
 }  
   
 .card {  
 /\* TODO: flex: 1 1 100% \*/  
 }  
}

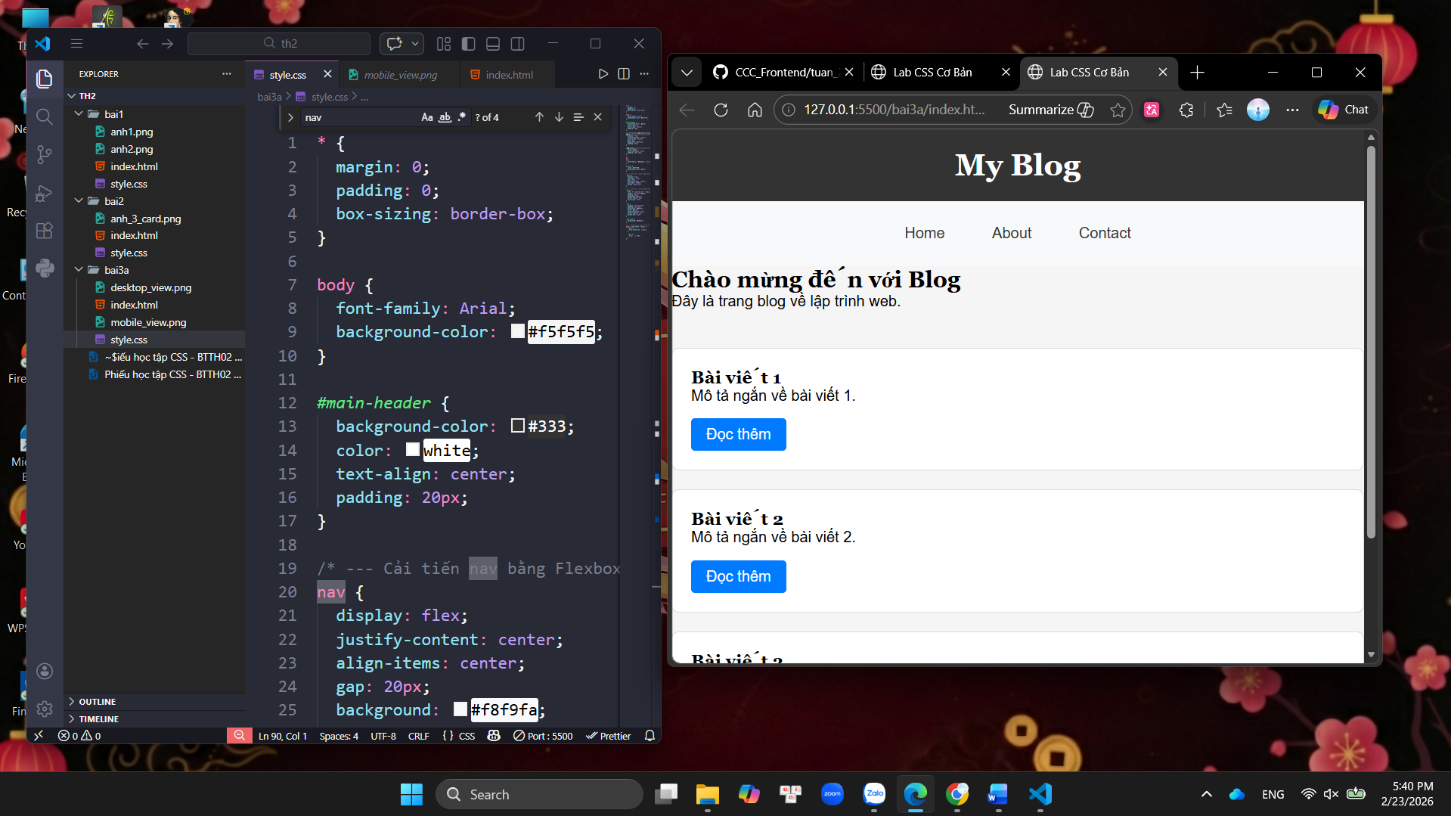
**Kết quả mong đợi:**

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 {  
 /\* TODO:  
 - display: grid  
 - grid-template-columns: repeat(3, 1fr) ← 3 cột bằng nhau  
 - gap: 16px  
 - max-width: 1200px  
 - margin: 40px auto  
 \*/  
}  
  
.item {  
 /\* TODO:  
 - 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  
 \*/  
}  
  
/\* Responsive: Tự động giảm cột khi màn hình nhỏ \*/  
.gallery {  
 /\* TODO: Dùng auto-fit/auto-fill \*/  
 grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));  
}

**Giải thích kỹ thuật:**

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

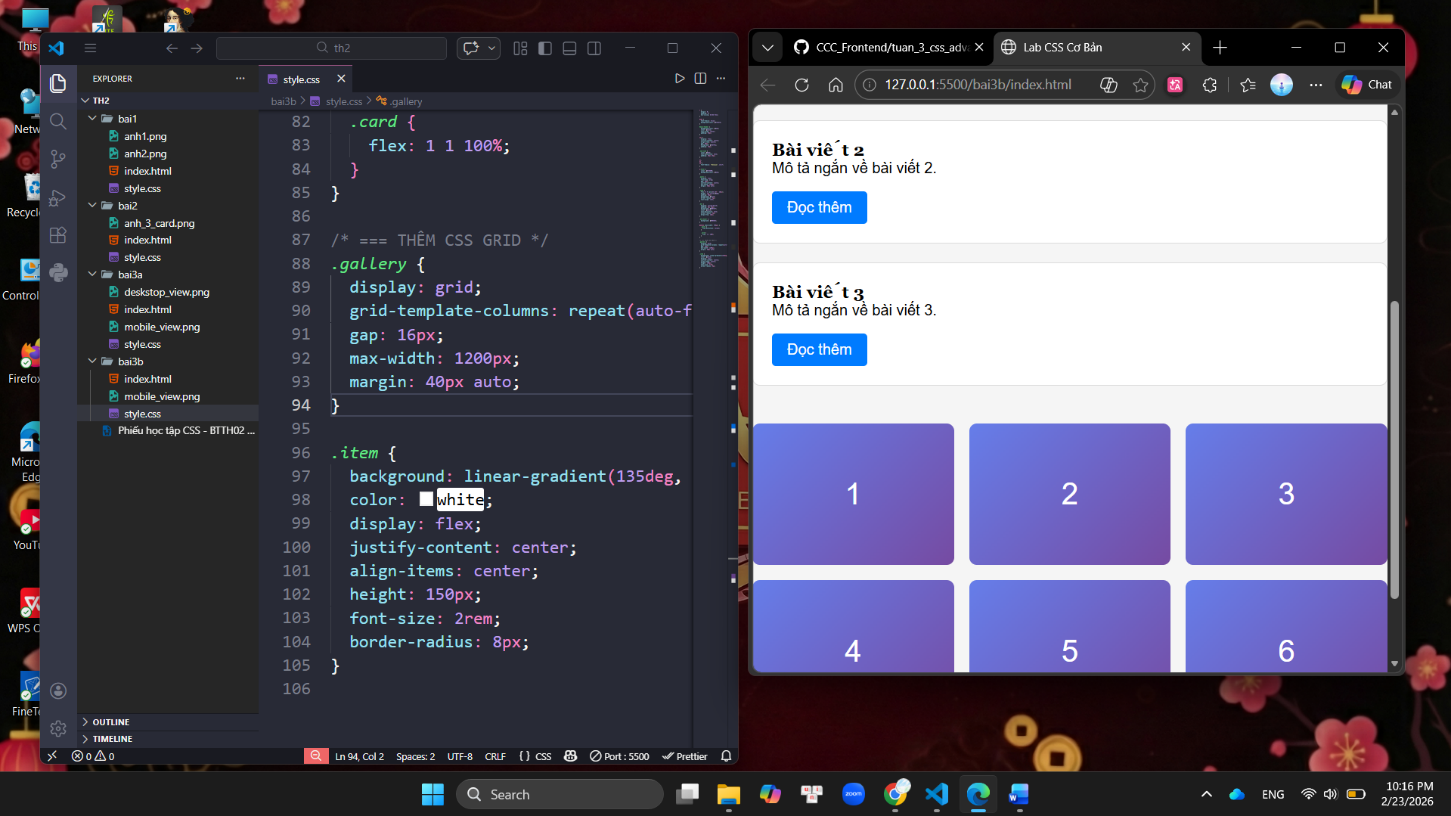
* repeat(): Hàm lặp lại việc tạo cột tự động để rút gọn mã CSS, thay vì phải khai báo kích thước cho từng cột một cách thủ công.
* auto-fit: Tự động tính toán và nhét số lượng cột tối đa có thể vào một hàng. Nếu hàng còn dư chỗ trống, nó sẽ kéo giãn các cột ra để lấp đầy toàn bộ hàng đó.
* minmax(200px, 1fr): Thiết lập kích thước giới hạn linh hoạt cho cột. Mỗi cột có thể co giãn để chiếm không gian trống (1fr), nhưng chiều rộng tối thiểu không bao giờ được phép nhỏ hơn 200px (nếu màn hình quá nhỏ khiến cột bị ép dưới 200px, cột đó sẽ tự động rớt xuống hàng dưới).

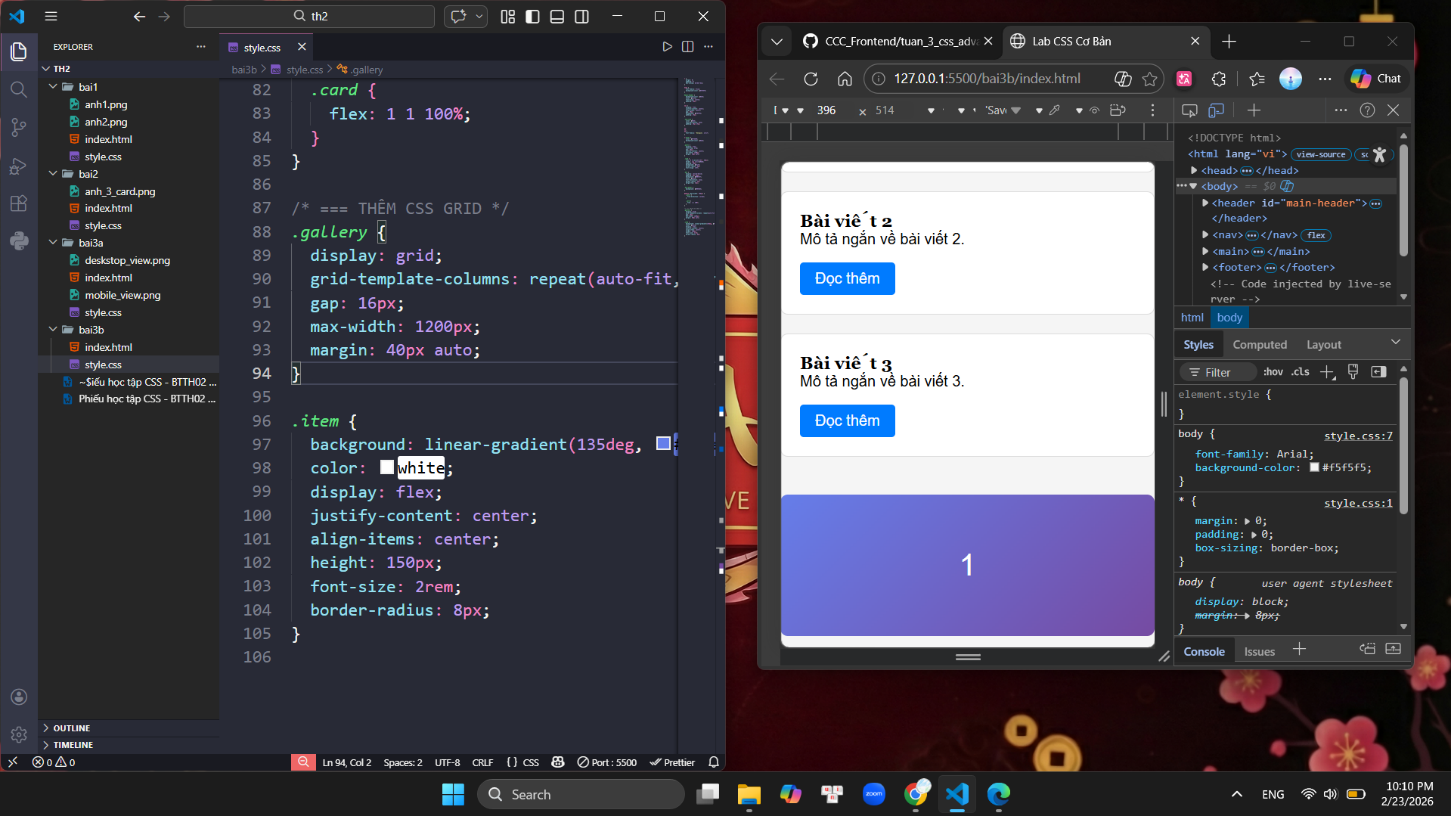
**Kết quả mong đợi:**

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]**](#fn2)

**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 nằm theo thứ tự luồng bình thường của trang web. Các thuộc tính top/bottom/left/right không có tác dụng.* | *Các khối văn bản, đoạn văn (thẻ <p>, <div>) hiển thị bình thường từ trên xuống dưới.* |
| relative | *Định vị tương đối so với* ***vị trí gốc ban đầu*** *của chính nó. Không làm ảnh hưởng/mất không gian chiếm chỗ ban đầu của nó.* | *Dịch chuyển nhẹ phần tử lên/xuống/trái/phải, hoặc dùng làm khung chứa (điểm tựa) cho phần tử con định vị bằng absolute.* |
| absolute | *Định vị tuyệt đối dựa theo* ***phần tử cha bao ngoài gần nhất*** *có cài đặt position (khác static). Bị rút hoàn toàn khỏi luồng trang web (không còn chiếm chỗ).* | *Nút tắt "X" ở góc hộp thoại (modal), dấu chấm đỏ thông báo (badge) nằm đè trên icon giỏ hàng.* |
| fixed | *Định vị cố định theo* ***cửa sổ trình duyệt (viewport)****. Luôn đứng yên một chỗ kể cả khi người dùng cuộn trang.* | *Thanh điều hướng (menu/header) luôn dính trên cùng, nút chat ở góc dưới màn hình, nút "Back to top".* |
| sticky | *Lai giữa relative và fixed. Hoạt động như relative bình thường, nhưng khi người dùng cuộn trang tới một ngưỡng chỉ định (vd: top: 0) thì nó dính chặt lại màn hình như fixed.* | *Dòng tiêu đề của một bảng (table header) luôn dính trên cùng khi cuộn xem danh sách dài, thanh mục lục trượt theo bài viết.* |

**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:**

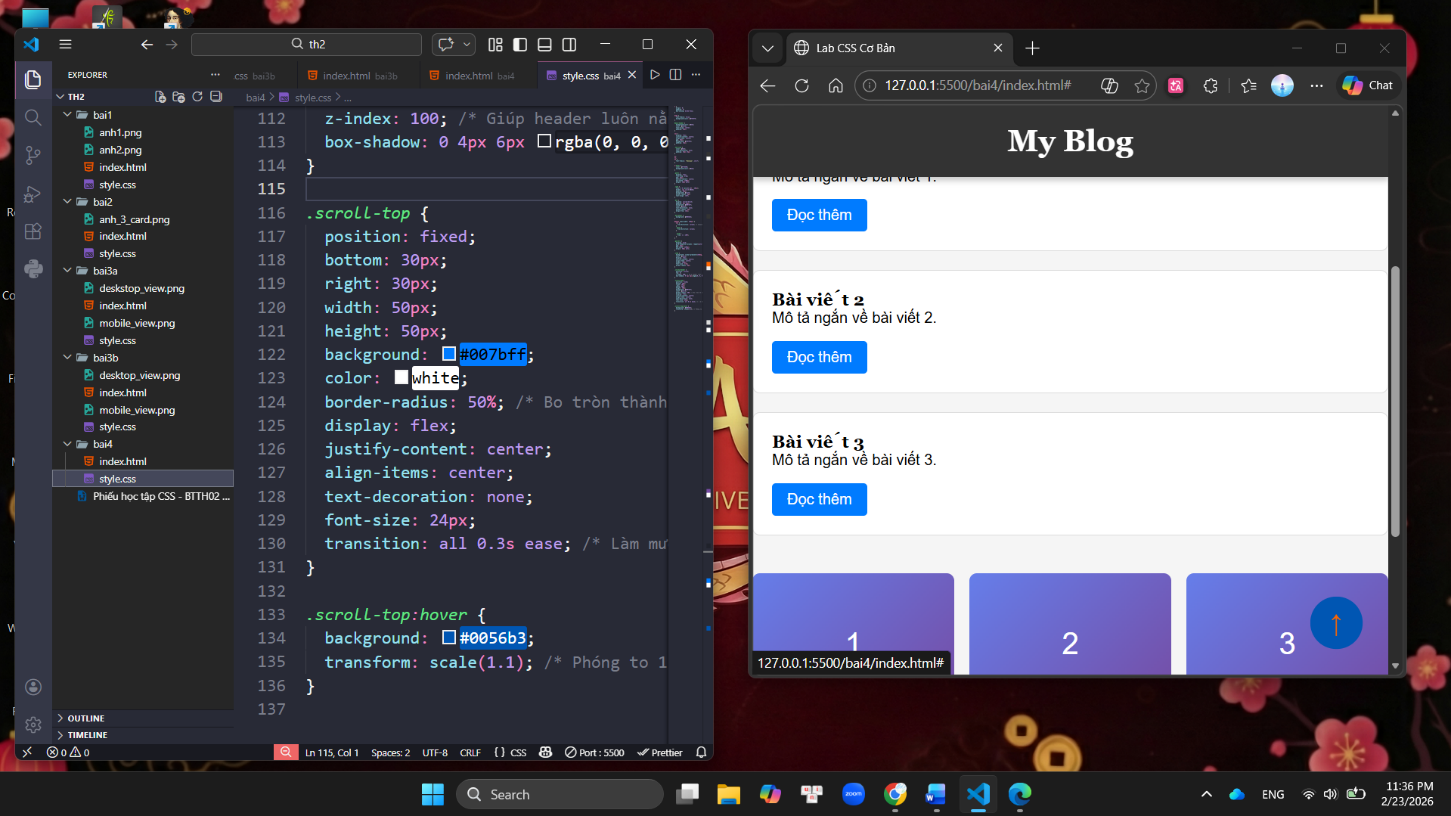
/\* Header sticky \*/  
.sticky-header {  
 position: sticky;  
 top: 0;  
 z-index: 100;  
 /\* TODO: Thêm box-shadow để tạo hiệu ứng nổi \*/  
}  
  
/\* Nút Scroll to Top \*/  
.scroll-top {  
 position: fixed;  
 bottom: 30px;  
 right: 30px;  
 /\* TODO:  
 - width: 50px, height: 50px  
 - background: #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. Các nội dung về html

**Những gì tôi vẫn chưa rõ:**

1. Các nội dung về CSS

**Mục tiêu tiếp theo:**

Nắm rõ hơn HTML và tìm hiểu lại CSS

**Ghi chú của giáo viên:**

**Ngày hoàn thành:** 23/02/2026

**Chữ ký sinh viên:**

