**Module 6 – HTML, CSS and JS in PHP**

## HTML Basics

### □ What is HTML? Explain its structure.

HTML HyperText Markup Language is the standard markup language used to create and design the structure of web pages. It tells a web browser how to display text, images, links, and other types of content.

- "HyperText" refers to the ability to link to other documents or web pages.
- "Markup Language" means it uses tags to "mark up" content for formatting and layout.

### Structure of an HTML Document

```
<!DOCTYPE html>
<html>
  <head>
    <title>My First Web Page</title>
  </head>
  <body>
    <h1>Welcome to My Website</h1>
    <p>This is a paragraph of text.</p>
  </body>
</html>
```

### □ Describe the purpose of HTML tags and provide examples of commonly used tags.

Purpose of HTML Tags

HTML tags are the building blocks of HTML. They are used to mark up content on a web page so that the browser can understand how to display or interact with that content.

Main purposes of HTML tags:

- Define structure (e.g., headings, paragraphs, sections)
- Format and display text, images, and links
- Organize media and multimedia (videos, audio)
- Enable user interaction (forms, buttons)
- Create links between pages

Commonly Used HTML Tags (with examples)

| Tag | Description | Example |
|---|---|---|
| <html> | Root of the HTML document | <html> ... </html> |
| <head> | Contains metadata and links to resources | <head> ... </head> |
| <title> | Sets the title of the web page | <title>My Page</title> |
| <body> | Holds all the visible content | <body> ... </body> |

| Tag | Description | Example |
|---|---|---|
| <h1>–<h6> | Headings, from largest (h1) to smallest (h6) | <h1>Main Heading</h1> |
|  | Paragraph of text | This is a paragraph. |
| <a> | Creates a hyperlink | <a href="https://example.com">Visit</a> |
| <img> | Embeds an image | <img src="image.jpg" alt="Image" /> |
| <ul>, <ol>, <li> | Unordered and ordered lists | <ul><li>Item</li></ul> |
|  | Generic container for block-level content | Some content |
|  | Generic container for inline content | Text |
| <br> | Line break | Line 1<br>Line 2 |
| <form> | Defines a form for user input | <form> ... </form> |
| <input> | Input field in a form | <input type="text" /> |
| <button> | Clickable button | <button>Click Me</button> |

```
<!DOCTYPE html>
<html>
  <head>
    <title>Example Page</title>
  </head>
  <body>
    <h1>Welcome to My Site</h1>
    <p>This is a paragraph of text.</p>
    <a href="https://example.com">Go to Example</a>
    <img src="photo.jpg" alt="A sample photo" />
  </body>
</html>
```

**What are the differences between block-level and inline elements? Give examples of each.**

Block-level elements start on a new line and take up the full width available (by default), stretching out to the left and right as far as it can.

Examples:

-  – generic container
-  – paragraph
- <h1> to <h6> – headings
- <ul>, <ol>, <li> – lists
- <section>, <header>, <footer>

Example Code:

```
<div>
  <h1>This is a heading</h1>
  <p>This is a paragraph inside a div.</p>
</div>
```

2. Inline Elements

Inline elements do not start on a new line and only take up as much width as necessary. They flow within a block of text or content.

Examples:

-  – generic inline container
- <a> – anchor (link)
-  – bold
- <img> – image
- <br> – line break

Example Code:

```
<p>This is a <strong>bold</strong> word in a sentence.</p>
<a href="#">Click here</a> to visit the homepage.
```

**Explain the concept of semantic HTML and why it is important.**
Semantic HTML refers to the use of HTML tags that convey meaning about the content they contain. Instead of just using generic elements like  or , semantic HTML uses meaningful tags like <header>, <article>, <footer>, etc.

Examples of Semantic Tags:

- <header> – Page or section header
- <nav> – Navigation links
- <main> – Main content
- <article> – Independent content
- <footer> – Footer content

**CSS Fundamentals**
**What is CSS? How does it differ from HTML?**
CSS stands for Cascading Style Sheets.
It is a language used to style and visually format HTML content.

Purpose of CSS:

CSS controls how HTML elements look — including:

- Colors
- Fonts
- Layout
- Spacing
- Animations
- Responsiveness

**Difference Between CSS and HTML:**

| Feature | HTML | CSS |
|---|---|---|
| Purpose | Structures content | Styles the content |
| Role | Defines elements (e.g., headings, images, links) | Defines appearance (e.g., color, size, position) |
| Language Type | Markup language | Style sheet language |
| Example Tag | This is a paragraph. | p { color: blue; font-size: 16px; } |
| File Extension | .html | .css |

 **Explain the three ways to apply CSS to a web page.**

1.Inline CSS

- Written inside an HTML tag using the style attribute.
- Example:

```
<p style="color: red;">Red text</p>
```

2. Internal CSS

- Written inside a <style> tag within the HTML <head>.
- Example:

```
<style>
  p { color: blue; }
</style>
```

3. External CSS

- Written in a separate .css file and linked with <link> in the HTML <head>.
- Example:

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

 **What are CSS selectors? List and describe the different types of selectors.**
Universal Selector (*)

- Targets all elements

Example:

```
* { margin: 0; }
```

- Element Selector

Targets elements by tag name

- Example:

```
p { color: blue; }
```

- Class Selector (.)

Targets elements with a specific class

- Example:

.highlight { background: yellow; }

- ID Selector (#)

Targets a single element with a specific ID

- Example:

#header {

font-size: 24px;

}

- Group Selector

Targets multiple selectors at once

Example:

h1, h2, p { color: green; }

- Descendant Selector

Targets elements inside another element

Example:

div p { font-style: italic; }

- Child Selector (>)

Targets direct children

Example:

ul > li { list-style: none; }

- Pseudo-class Selector (:)

Targets elements in a specific state

Example:

a:hover { color: red; }

**What is the box model in CSS? Explain its components.**

The CSS Box Model is a fundamental concept that describes how elements on a web page are structured and spaced. Every HTML element is considered as a rectangular box composed of several layers.

Components of the Box Model:

1. Content
    - The actual content inside the element (text, images, etc.).
    - Size controlled by width and height.

2. Padding
    - Space inside the element, between the content and the border.
    - Adds breathing room inside the box.
3. Border
    - The line surrounding the padding and content.
    - Can have thickness, style, and color.
4. Margin
    - Space outside the border, separating the element from others.
    - Controls the distance between elements.

### 3. Responsive Web Design

**What is responsive web design? Why is it important?**
Responsive Web Design (RWD) is an approach to building websites so that they adapt smoothly to different screen sizes and devices (desktops, tablets, smartphones).

It ensures the site looks good and works well on any device, by using flexible layouts, images, and CSS media queries.

Why is Responsive Web Design Important?

1. Better User Experience:
   Users get an optimized, easy-to-use website regardless of their device.
2. Increased Mobile Traffic:
   More people browse on mobile devices, so responsive design captures a larger audience.
3. Improved SEO:
   Search engines like Google prioritize mobile-friendly, responsive sites in rankings.
4. Cost-Effective:
   One website works across all devices — no need for separate mobile sites.

**Explain the use of media queries in CSS. Provide an example.**

Media queries allow you to apply different CSS styles based on the device's characteristics such as screen size, resolution, orientation, etc.

They are key for creating responsive designs that adapt to various devices.

```
body {
  background-color: white;
  font-size: 16px;
}

@media (max-width: 600px) {
  body {
```

```
    background-color: lightgray;
    font-size: 14px;
  }
}
```

 **What are the benefits of using a mobile-first approach in web design?**

Benefits of Using a Mobile-First Approach in Web Design

1. Better Performance
   Designing for mobile first encourages simpler, faster-loading pages, improving load times on all devices.
2. Improved User Experience
   Focuses on essential content and features for smaller screens, ensuring clarity and usability.
3. Easier Responsive Design
   Starting with mobile styles and then scaling up (progressive enhancement) helps manage CSS more efficiently.
4. Wider Audience Reach
   Mobile users are a huge portion of internet traffic; mobile-first ensures your site works well for them.
5. SEO Advantages
   Search engines like Google prioritize mobile-friendly websites, boosting your ranking.
6. Cost-Effective Development
   Avoids building separate desktop and mobile versions, saving time and resources.

4. PHP Integration

 How can PHP be used to dynamically generate HTML content? Provide examples.
-->PHP is a server-side scripting language that can generate HTML content dynamically before the page is sent to the browser.
This means PHP runs on the server, processes logic (like database queries, calculations, conditions), and outputs HTML for the client.
EXAMPLE:-

```
    <!DOCTYPE html>
<html>
<head>
    <title>Dynamic HTML with PHP</title>
</head>
<body>
    <h1>Welcome to My Website</h1>
    <p>
        <?php
            $name = "Tanisha";
            echo "Hello, $name!";
        ?>
    </p>
</body>
</html>
```

 Explain how to include CSS files in a PHP-generated HTML page.
When you generate HTML with PHP, you can include CSS files the same way you would in a normal HTML page — using the <link> tag inside the <head> section.
Since PHP just outputs HTML to the browser, the CSS linking works exactly the same.

```
<!DOCTYPE html>
<html>
<head>
    <title>PHP with CSS</title>
    <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
    <?php
        echo "<h1 class='heading'>Hello, World!</h1>";
        echo "<p class='paragraph'>This is styled using CSS.</p>";
    ?>
</body>
</html>
```

 What are the advantages of using PHP to manage HTML forms?

1. Easy Form Data Collection

- PHP can access form data sent via GET or POST methods using $_GET and $_POST.
- Example:

```
$name = $_POST['name'];
$email = $_POST['email'];
```

- This makes it simple to capture user input.

2. Validation and Error Handling

- PHP allows you to validate form data (e.g., check if fields are empty, if email is valid, etc.) before saving or using it.
- Example:

```
if (empty($_POST['email'])) {
    echo "Email is required!";
} elseif (!filter_var($_POST['email'], FILTER_VALIDATE_EMAIL)) {
    echo "Invalid email format!";
}
```

3. Security Features

- PHP can sanitize and escape form inputs to prevent SQL Injection and XSS attacks.
- Example:

```
$safe_input = htmlspecialchars($_POST['message']);
```

4. Database Integration

- PHP works seamlessly with databases (like MySQL, PostgreSQL).
- You can save form submissions directly into a database.
- Example:

```
$conn = mysqli_connect("localhost", "root", "", "mydb");
$sql = "INSERT INTO users (name, email) VALUES ('$name', '$email')";
mysqli_query($conn, $sql);
```

5. Dynamic Responses

- After processing a form, PHP can generate custom confirmation messages or redirect users.
- Example:

```
echo "Thank you, $name! Your message has been received.";
```

6. File Upload Handling

- PHP has built-in functions ($_FILES) to handle file uploads from forms.
- Example:

```
move_uploaded_file($_FILES['file']['tmp_name'], "uploads/" . $_FILES['file']['name']);
```

7. Session & Authentication Support

- PHP can manage login forms with sessions and cookies.
- Example:

```
session_start();
$_SESSION['username'] = $_POST['username'];
```

8. Cross-Platform and Widely Supported

- PHP runs on almost any web server (Apache, Nginx, IIS) and is supported by all major operating systems, making form handling very portable.

LAB EXERCISES

1. Creating a Simple Web Page

Instructions:
 Create an HTML file (e.g., index.html) that includes a header, a navigation bar, a main content section, and a footer.
 Style the page using an external CSS file (e.g., styles.css).
  Use CSS properties such as color, background-color, font-size, and padding to enhance the design.

--index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
```

```
  <title>Simple Webpage</title>
  <link rel="stylesheet" href="styles.css" />
</head>
<body>

  <header>
    <h1>My Website Header</h1>
  </header>

  <nav>
    <ul>
      <li><a href="#">Home</a></li>
      <li><a href="#">About</a></li>
      <li><a href="#">Services</a></li>
      <li><a href="#">Contact</a></li>
    </ul>
  </nav>

  <main>
    <section>
      <h2>Welcome to My Website</h2>
      <p>This is the main content section of the webpage.</p>
    </section>
  </main>

  <footer>
    <p>&copy; 2025 My Website. All rights reserved.</p>
  </footer>

</body>
</html>
```

style.css

```
body {
  font-family: Arial, sans-serif;
  margin: 0;
  padding: 0;
  background-color: #f5f5f5;
  color: #333;
}


header {
  background-color: #4CAF50;
  color: white;
  padding: 20px 0;
  text-align: center;
  font-size: 2rem;
}

nav {
```

```
  background-color: #333;
}

nav ul {
  margin: 0;
  padding: 0;
  list-style-type: none;
  display: flex;
  justify-content: center;
}

nav ul li {
  margin: 0 15px;
}

nav ul li a {
  color: white;
  text-decoration: none;
  font-size: 1.1rem;
  padding: 14px 10px;
  display: block;
}

nav ul li a:hover {
  background-color: #575757;
  border-radius: 4px;
}


main {
  padding: 30px;
  background-color: white;
  max-width: 900px;
  margin: 20px auto;
  border-radius: 8px;
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

main h2 {
  color: #4CAF50;
  font-size: 1.8rem;
}

main p {
  font-size: 1.1rem;
  line-height: 1.6;
  padding-top: 10px;
}


footer {
  background-color: #222;
```

```
  color: #ddd;
  text-align: center;
  padding: 15px 0;
  font-size: 0.9rem;
}
```

2. Form Handling with PHP

o Instructions:

 Create an HTML form that collects user information (e.g., name, email, and message).

 Use PHP to process the form data and display a confirmation message with the submitted information.

 Validate user inputs and provide appropriate feedback.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Contact Form</title>
</head>
<body>
    <h2>Contact Us</h2>
    <form action="" method="post">
        <label for="name">Name:</label><br>
        <input type="text" name="name" id="name" required><br><br>

        <label for="email">Email:</label><br>
        <input type="email" name="email" id="email" required><br><br>

        <label for="message">Message:</label><br>
        <textarea name="message" id="message" rows="4" required></textarea><br><br>

        <input type="submit" value="Submit">
    </form>
</body>
</html>

<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    echo "<h2>Submitted Data:</h2>";
    echo "Name: " . $_POST['name'] . "<br>";
    echo "Email: " . $_POST['email'] . "<br>";
    echo "Message: " . $_POST['message'] . "<br>";
}
?>
```

3. Dynamic Content Generation

o Instructions:

 Create a PHP script (e.g., dynamic-content.php) that generates a list of items (e.g., products or blog posts) from an array.

 Use a loop to display the items in a styled HTML list.
 Style the list using CSS.

4. CSS Grid and Flexbox
o Instructions:  Build a grid layout for a gallery of images or a product showcase using either CSS Grid or Flexbox.
 Ensure that the layout is responsive and adjusts based on the screen size.
 Use media queries to change the layout for mobile devices.

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="utf-8" />
 <meta name="viewport" content="width=device-width,initial-scale=1" />
 <title>Responsive Image Gallery (CSS Grid)</title>
 <style>

  * { box-sizing: border-box; }
  body {
   font-family: system-ui, -apple-system, "Segoe UI", Roboto, "Helvetica Neue", Arial;
   margin: 0;
   padding: 2rem;
   background: #f7f7f8;
   color: #111;
  }

  h1 {
   margin: 0 0 1rem 0;
   font-size: 1.5rem;
  }


  .gallery {
   display: grid;
   gap: 16px;
   grid-template-columns: repeat(4, 1fr);
  }

  .card {
   background: white;
   border-radius: 10px;
   overflow: hidden;
   box-shadow: 0 6px 18px rgba(22, 28, 37, 0.06);
   display: flex;
   flex-direction: column;
   transition: transform 180ms ease, box-shadow 180ms ease;
  }



  .card img {
   width: 100%;
```

```
  height: 170px;
  object-fit: cover;
  display: block;
 }

 .card .info {
  padding: 0.75rem;
  font-size: 0.95rem;
 }

 .card .title {
  margin: 0 0 0.35rem 0;
  font-weight: 600;
 }

 .card .subtitle {
  margin: 0;
  color: #666;
  font-size: 0.85rem;
 }


 @media (max-width: 900px) {
  .gallery { grid-template-columns: repeat(2, 1fr); }
  .card img { height: 160px; }
 }

 @media (max-width: 520px) {
  body { padding: 1rem; }
  .gallery { grid-template-columns: 1fr; gap: 12px; }
  .card img { height: 210px; }
  .card .info { padding: 0.75rem; }
 }
 </style>
</head>
<body>
 <h1>Simple Responsive Gallery</h1>

 <div class="gallery">
  <div class="card">
   <img src="cotten.jpg" alt="Mountain lake">
   <div class="info">
    <p class="subtitle">Plants</p>
   </div>
  </div>

  <div class="card">
   <img src="cotten2.jpg" alt="Husky dog">
   <div class="info">
       <p class="subtitle">Plants</p>
```

```
 </div>
</div>

<div class="card">
 <img src="cotten-3.jpg" alt="Forest path">
 <div class="info">
      <p class="subtitle">Plants</p>

 </div>
</div>

<div class="card">
 <img src="jade.jpg" alt="City skyline">
 <div class="info">
     <p class="subtitle">Plants</p>

 </div>
</div>

<div class="card">
 <img src="Indian Rose.jpg" alt="Beach sunset">
 <div class="info">
      <p class="subtitle">Plants</p>

 </div>
</div>

<div class="card">
 <img src="cactus.jpg" alt="Close-up flower">
 <div class="info">
     <p class="subtitle">Plants</p>

 </div>
</div>

<div class="card">
 <img src="Indian Rose.jpg" alt="Road in countryside">
 <div class="info">
     <p class="subtitle">Plants</p>

 </div>
</div>

<div class="card">
 <img src="cotten-3.jpg" alt="Minimal interior">
 <div class="info">
     <p class="subtitle">Plants</p>

 </div>
</div>
</div>
```

```
</body>
</html>
```

5. Styling a PHP Application
 o Instructions:
 Create a simple PHP application (e.g., a user registration page).
  Use an external CSS file to style the form elements (e.g., inputs, buttons, labels).
 Ensure that the application is visually appealing and user-friendly.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Contact Form</title>
 <link rel="stylesheet" href="A-Q-5.css">
</head>
<body>
<div class="">
        <h2>User Registration</h2>
    <form action="" method="post">
        <label for="name">Name:</label><br>
        <input type="text" name="name" id="name" required><br><br>
        <label for="email">Email:</label><br>
        <input type="email" name="email" id="email" required><br><br>
        <label for="password"> Password:</label><br>
        <input type="password" name="password" id="password" required><br><br>
  <button type="submit" value="submit"> Submit </button>
    </form>

<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    echo "<h2>Submitted Data:</h2>";
    echo "Name: " . $_POST['name'] . "<br>";
    echo "Email: " . $_POST['email'] . "<br>";
    echo "Password: " . $_POST['password'] . "<br>";
}
?>
</div>
</body>
</html>
```

CSS file:-

```
body {
    font-family: Arial, sans-serif;
    background: #f3f4f6;
    display: flex;
    justify-content: center;
    align-items: center;
```

```
    height: 100vh;
    margin: 0;
}

.header {
    background: #fff;
    padding: 25px;
    border-radius: 12px;
    box-shadow: 0 4px 10px rgba(0,0,0,0.15);
    width: 350px;
}

h2 {
    text-align: center;
    color: #333;
    margin-bottom: 20px;
}

label {
    display: block;
    margin: 10px 0 5px;
    color: #444;
    font-weight: bold;
}

input {
    width: 100%;
    padding: 10px;
    border: 1px solid #ccc;
    border-radius: 6px;
    margin-bottom: 15px;
    font-size: 14px;
}

input:focus {
    border-color: #007bff;
    outline: none;
}

button {
    width: 100%;
    padding: 12px;
    background: #007bff;
    border: none;
    border-radius: 6px;
    color: white;
    font-size: 16px;
    cursor: pointer;
    transition: 0.3s;
}
button:hover {
    background: #0056b3;
```

}

6. Implementing a Responsive Navigation Bar 
o Instructions:  Build a navigation bar using HTML
and  elements.
 Use CSS to style the navigation bar and make it responsive (e.g., using media queries).
 Implement a dropdown menu for sub-navigation items.

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

<head>
  <title>photography by monali</title>
  <link href="image2_files\photography_by_monali.png" rel="icon">

  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <!-- Latest compiled and minified CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet">

  <!-- Latest compiled JavaScript -->
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"></script>


  <style>
    h1 {
      margin-top: 30px;
      font-family: 'Gill Sans', 'Gill Sans MT', Calibri, 'Trebuchet MS', sans-serif;
    }

    .row a {
      text-decoration: none;
      color: white;
      margin-left: 100px;
      font-size: 20px;
    }

    .row a:hover {
      color: rgb(66, 75, 211);

    }

    .btn {
      width: 50px;
      float: right;
      margin-left: 10px;
    }

    .btn:hover {
```

```
      background-color: greenyellow;
      color: black;
    }

    .button-img {
      width: 100%;
    }

    .btn1 {
      width: 170px;
    }

    .img-rounded {
      height: 480px;
      width: auto;

    }

    .mainbar {
      margin-left: 337px;

    }

    .stick {
      position: sticky;
      top: 0;

    }
  </style>
</head>

<body>
  <div class="stick">
    <div class="container-fluid">
      <div class="row">
        <div class="col-md-2 bg-secondary p-2">

          <img src="image2_files/photography_by_monali.png" alt="" height="100px "
width="auto" align="center">

        </div>
        <div class="col-md-10 bg-secondary p-1">
          <h1 align="center">photography_by_monali

          </h1>
        </div>
      </div>
      <div class="row">
        <div class="col-md-9 bg-dark p-1">
          <a href="index.html">Home</a>
```

```
        <a href="gallery.html">Gallery</a>
        <a href="about us.html">About Us</a>
        <a href="Booking.html">Booking</a>

        <a href="contact.html">Contact Us</a>
        <div class="col-md-9 bg-dark p-1"></div>




      </div>
      <div class="col-md-3 bg-dark p-1">
        <a href="signin.html"><button type="button" style="width: 95px;"
            class="btn btn-success">Registation</button></a>

        <a href="login copy.html"> <button type="button " style="width: 55px;"
            class="btn btn-success ">Login</button></a>
      </div>

    </div>
  </div>
</div>
</body>
```

7. Image Gallery with Lightbox Effect
Objective: Create an image gallery that opens images in a lightbox effect.
Instructions:
Use HTML to create a gallery of images.
Implement CSS for styling and layout.
Use JavaScript or a CSS library to create a lightbox effect when images are clicked.
ANS:-

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<title>Lightbox Gallery</title>
<link rel="stylesheet" href="A-Q-7.css">
</head>
<body>
<h2>Image Gallery</h2>
<div class="gallery">
<a href="#img1"><img src="img1.jpg" alt="Image 1"></a>
<a href="#img2"><img src="img2.webp" alt="Image 2"></a>
<a href="#img3"><img src="img7.webp" alt="Image 3"></a>
<a href="#img4"><img src="img8.webp" alt="Image 4"></a>
  </div>

  <!-- Lightbox -->
  <div class="lightbox" id="img1">
    <a href="#" class="close">&times;</a>
    <img src="img1.jpg" alt="Image 1">
```

```
  </div>

  <div class="lightbox" id="img2">
    <a href="#" class="close">&times;</a>
    <img src="img2.webp" alt="Image 2">
  </div>

  <div class="lightbox" id="img3">
    <a href="#" class="close">&times;</a>
    <img src="img7.webp" alt="Image 3">
  </div>

  <div class="lightbox" id="img4">
    <a href="#" class="close">&times;</a>
    <img src="img8.webp" alt="Image 4">
  </div>
</body>
</html>
```

CSS File:-

```
body {
  font-family: Arial, sans-serif;
background: #f9f9f9;
text-align: center;
padding: 20px;
}
h2 {
margin-bottom: 20px;
}
.gallery {
display: flex;
justify-content: center;
gap: 15px;
f
 lex-wrap: wrap;
}
.gallery img {
width: 500px;
border-radius: 6px;
cursor: pointer;
transition: transform 0.3s;
}
.gallery img:hover {
transform: scale(1.05);
}
/* Lightbox styling */
.lightbox {
display: none;
position: fixed;
top: 0; left: 0;
width: 100%;
height: 100%;
```

```
background: rgba(0,0,0,0.85);
justify-content: center;
align-items: center;
z-index: 999;
}
.lightbox img {
max-width: 90%;
max-height: 80%;
border-radius: 8px;
box-shadow: 0 0 15px #000;
}
.lightbox:target {
display: flex;
}
.close {
position: absolute;
top: 20px;
right: 30px;
font-size: 32px;
color: white;
text-decoration: none;
}
.close:hover {
color: red;
}
```