

25 HTML & CSS Projects for Beginners



HTML



How to Use This Template

Thanks for downloading HubSpot's
25 HTML & CSS Projects for Beginners!

Inside, you'll find step-by-step HTML & CSS projects, including building a website for your resume, building a basic contact form, creating a product landing page, and more. In the end, you'll have 25 completed projects to add to your portfolio. Plus, we've included a quick reference guide to common HTML & CSS elements, such as color & font codes.



Create a compelling website with HubSpot's free drag-and-drop website builder—no coding required

- ✓ Build and edit your website without any coding or technical skills
- ✓ Publish your business website faster with a pre-built theme
- ✓ Leverage custom modules to deliver a great user experience

Build Your Website for Free

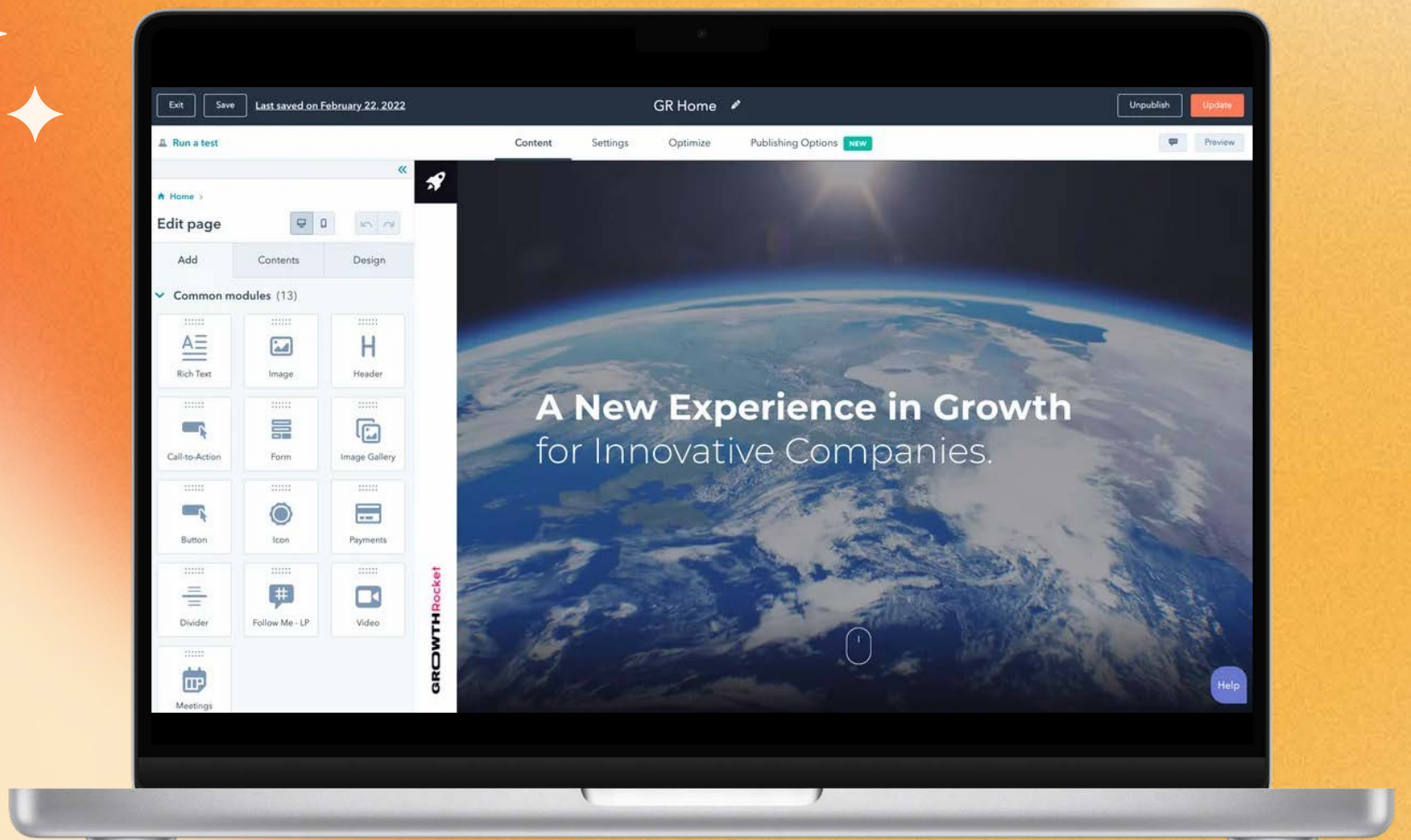


Table of Contents

5 Quick Reference: Common HTML & CSS Elements

7 How To Make HTML & CSS Projects

8 25 HTML & CSS Projects for Beginners

Projects

- | | |
|--|---|
| 1. Build a Personal Profile Card | 14. Build a Basic Portfolio Page |
| 2. Create a Simple Product Landing Page | 15. Create a Basic Product Page |
| 3. Build a Basic Contact Form | 16. Create a Simple “Coming Soon” Page |
| 4. Create a Simple Blog Post Layout | 17. Build a Simple “404 Error” Page |
| 5. Build a Simple Image Gallery | 18. Create a Basic Weather Info Page |
| 6. Create a Simple Resume Page | 19. Build a Simple Countdown Timer Page |
| 7. Create a Simple FAQ Page | 20. Create a Basic Testimonials Page |
| 8. Build a Simple To-Do List | 21. Build a Simple Travel Destination Page |
| 9. Create a Basic Newsletter Signup Form | 22. Create a Basic Movie Review Page |
| 10. Build a Basic Pricing Table | 23. Build a Simple Online Poll Page |
| 11. Create a Basic Event Invitation Page | 24. Create a Basic Music Playlist Page |
| 12. Build a Basic Recipe Page | 25. Create a Basic “Meet the Team” Page |
| 13. Create a Simple Business Card Webpage | |



Quick Reference:

Common HTML & CSS Elements

This section provides a quick reference to common elements marketers might use frequently when working with HTML and CSS templates, such as color, fonts, and dimensions.

Common Colors

Color Name	HEX Code	RGB Value	Visual
White	#FFFFFF	rgb (255, 255, 255)	
Black	#000000	rgb (0, 0, 0)	
Red	#FF0000	rgb (255, 0, 0)	
Blue	#007BFF	rgb (0, 123, 255)	
Green	#28A745	rgb (40, 167, 69)	
Yellow	#FFD700	rgb (255, 215, 0)	
Gray	#6C757D	rgb (108, 117, 125)	

Common Fonts

Font Name	CSS Code	Visual
Arial	font-family: Arial, sans-serif;	This is an example.
Times New Roman	font-family: "Times New Roman", serif;	This is an example.
Helvetica	font-family: Helvetica, sans-serif;	This is an example.
Georgia	font-family: Georgia, serif;	This is an example.
Verdana	font-family: Verdana, sans-serif;	This is an example.
Courier New	font-family: "Courier New", monospace;	This is an example.
Open Sans (Google)	font-family: "Open Sans", sans-serif;	This is an example.

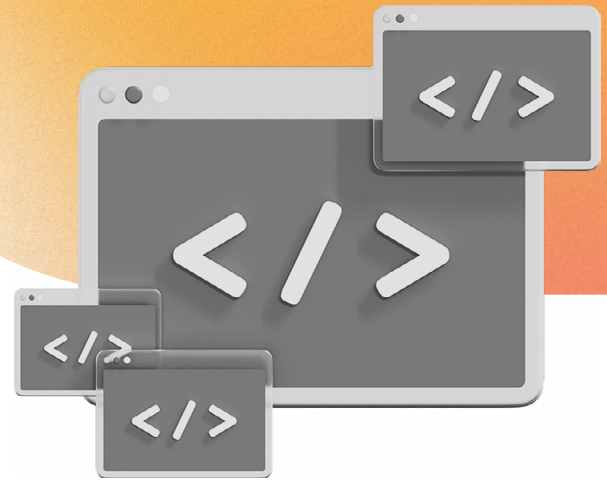
Font Sizes

Size	CSS Code	Use Case
Small Text	font-size: 0.875rem;	Fine print, disclaimers
Body Text	font-size: 1rem;	Standard paragraphs
Subheadings	font-size: 1.25rem;	H2 or H3-sized headings
Large Headings	font-size: 2rem;	Main page titles, hero sections

Image Dimensions

Element	Recommended Size	Use Case
Hero Image	1920px x 1080px	Website header or hero section
Profile Picture	150px x 150px	Team or avatar images
Thumbnail Image	300px x 200px	Blog post or product previews
Button Height/Width	40px x 120px	Call-to-action buttons

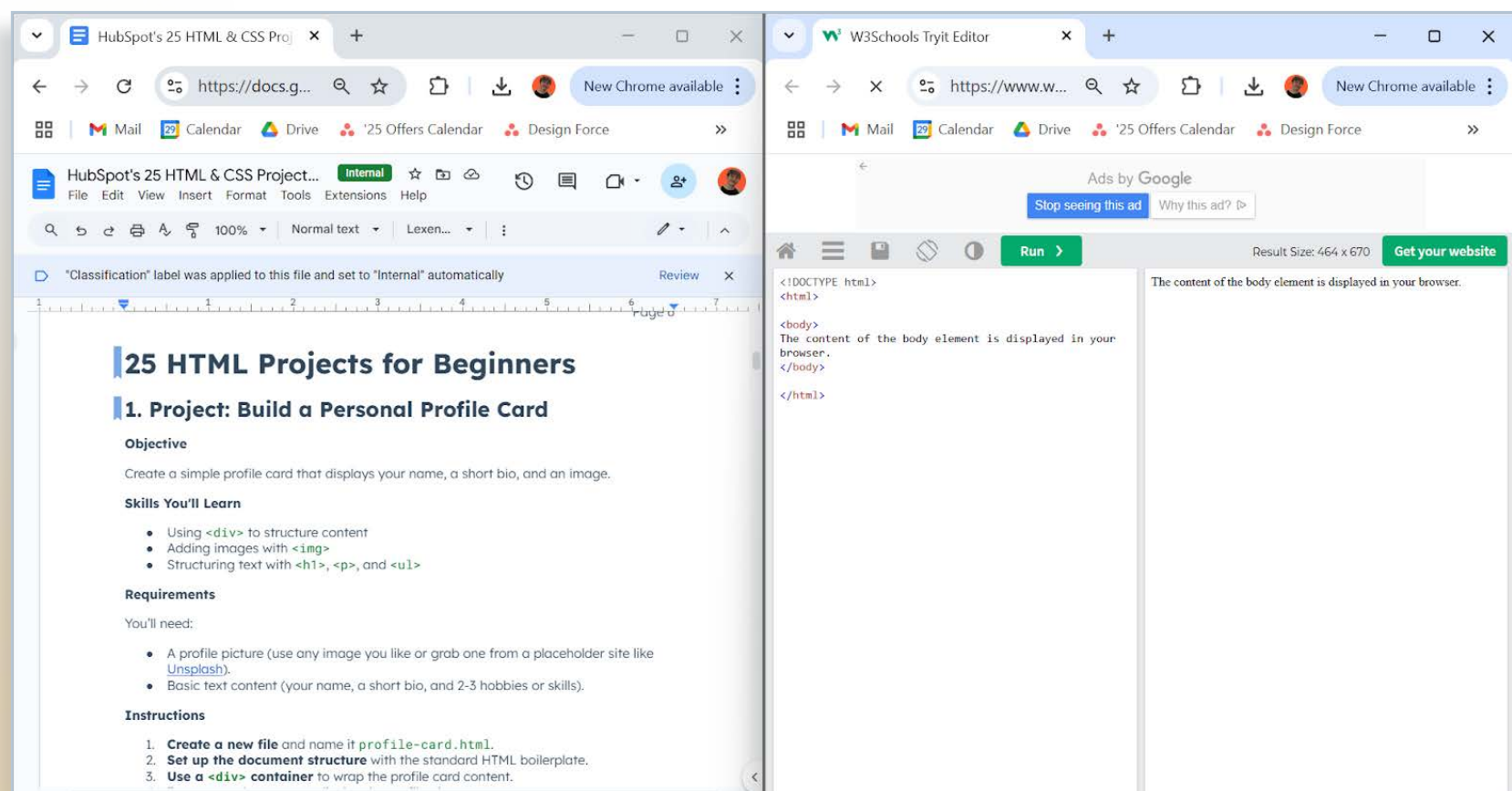
How To Make HTML & CSS Projects



To test and experiment with your HTML projects, you can use online code editors. Two great options are [CodePen](#) and [W3Schools Tryit Editor](#).

CodePen is a powerful tool with live previews and collaboration features, but it requires creating a free account to save your work. W3Schools Tryit Editor is a more budget-friendly, no-signup option that lets you quickly edit and run HTML code without saving. Both are excellent for testing and learning as you build your projects!

Once you've selected your code editor, consider putting this guide and the editor side by the side, as shown in the image below.



25 HTML & CSS Projects for Beginners



1. Project:

Build a Personal Profile Card

Objective

Create a simple profile card that displays your name, a short bio, and an image.

Skills You'll Learn

- Using `<div>` to structure content
- Adding images with ``
- Structuring text with `<h1>`, `<p>`, and ``

Requirements

You'll need:

- A profile picture (use any image you like or grab one from a placeholder site like [Unsplash](#)).
- Basic text content (your name, a short bio, and 2-3 hobbies or skills).

Instructions

1. **Create a new file** and name it `profile-card.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use a `<div>` container** to wrap the profile card content.
4. **Insert an `` tag** to display the profile picture.
5. **Add a heading (`<h1>`)** for your name.
6. **Write a short bio** inside a `<p>` tag.
7. **Create a list of hobbies or skills** using `` and ``.
8. **Test the layout** by opening the file in a browser.

Challenges/Extensions

- Add inline CSS to center the card on the page.
- Use additional tags like `<hr>` to add a line separating sections.
- Replace the unordered list with a table to organize skills or hobbies in columns.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Profile Card</title>
</head>
<body>
  <div class="profile-card">
    
    <h1>Your Name</h1>
    <p>Hello! I'm [Your Name], and I'm passionate about web
design. I love coding and creating visually stunning designs.</p>
    <ul>
      <li>Coding</li>
      <li>Photography</li>
      <li>Reading</li>
    </ul>
  </div>
</body>
</html>
```

2. Project:

Create a Simple Product Landing Page

Objective

Build a single-page product landing page that highlights a product, includes a brief description, and has a call-to-action (CTA) button.

Skills You'll Learn

- Structuring content with `<section>` and `<header>`
- Using `<a>` tags for links
- Adding buttons and placeholder content

Requirements

You'll need:

- A product idea (real or fictional)
- Placeholder text for the product name, description, and CTA
- An image (you can use a placeholder image from [Placeholder.com](https://placeholder.com))

Instructions

1. **Create a new file** and name it `landing-page.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use a `<header>` container** to display the product name and slogan.
4. **Create a `<section>`** for the product description and benefits.
5. **Insert an `` tag** to display a product image.
6. **Add a call-to-action button** using `<a>` or `<button>` with a purchase or learn more link.
7. **Test the layout** by opening the file in a browser.

Challenges/Extensions

- Add more sections, like testimonials or product features.
- Use inline CSS to style the button and make it visually appealing.
- Make the CTA link open in a new tab using `target="_blank"`.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Product Landing Page</title>
</head>
<body>
  <header>
    <h1>Amazing Product</h1>
    <p>The only product you'll ever need!</p>
  </header>
  <section>
    <h2>Why Choose Amazing Product?</h2>
    <p>It's fast, reliable, and built with love. This product
will simplify your life in ways you never imagined.</p>
    
    <a href="https://example.com" class="cta-button">Buy Now</a>
  </section>
</body>
</html>
```

3. Project:

Build a Basic Contact Form

Objective

Create a simple contact form that allows users to enter their name, email, and a message.

Skills You'll Learn

- Structuring forms with `<form>`, `<input>`, `<textarea>`, and `<button>`
- Using form attributes like `required` and `placeholder`
- Understanding the basics of user input handling

Requirements

You'll need:

- Basic knowledge of form elements in HTML
- Placeholder text for form fields
- A submit button

Instructions

1. **Create a new file** and name it `contact-form.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use a `<form>` element** to wrap the contact fields.
4. **Add a `<label>` and `<input>` field** for the user's name.
5. **Add a `<label>` and `<input>` field** for the user's email.
6. **Include a `<label>` and `<textarea>`** for the message.
7. **Add a submit button** using `<button type="submit">`.
8. **Test the form** by entering sample data and clicking submit.

Challenges/Extensions

- Add a dropdown (`<select>`) for users to choose an inquiry type (e.g., General, Support, Sales).
- Use a radio button group to let users select their preferred contact method (Email, Phone).
- Wrap everything in a `<fieldset>` with a `<legend>` for better structure.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Contact Form</title>
</head>
<body>
  <h2>Contact Us</h2>
  <form>
    <label for="name">Name:</label>
    <input type="text" id="name" name="name" placeholder="Enter your name"
required>

    <label for="email">Email:</label>
    <input type="email" id="email" name="email" placeholder="Enter your
email" required>

    <label for="message">Message:</label>
    <textarea id="message" name="message" rows="4" placeholder="Type your
message here..." required></textarea>

    <button type="submit">Send</button>
  </form>
</body>
</html>
```

4. Project:

Create a Simple Blog Post Layout

Objective

Design a basic blog post layout with a title, author name, date, and main content.

Skills You'll Learn

- Structuring a blog post with `<article>` and `<section>`
- Formatting text with headings and paragraphs
- Adding an image to a blog post

Requirements

You'll need:

- A blog post title and sample content
- An image (you can use a placeholder from [Unsplash](https://unsplash.com) or [Placeholder.com](https://placeholder.com))
- A fake author name and date

Instructions

1. **Create a new file** and name it `blog-post.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use `<article>` to wrap the blog post** and include a `<header>` for the title, author, and date.
4. **Insert an image** below the title using the `` tag.
5. **Add multiple paragraphs** of sample text to simulate a real blog post.
6. **Test the layout** by opening the file in a browser.

Challenges/Extensions

- Add a `<footer>` with a “Read More” button or author bio.
- Use `<blockquote>` to highlight a key quote in the post.
- Add a comments section with an input box.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Simple Blog Post</title>
</head>
<body>
  <article>
    <header>
      <h1>My First Blog Post</h1>
      <p>By <strong>Jane Doe</strong> | Published on <time datetime="2025-01-29">January 29, 2025</time></p>
    </header>
    
    <section>
      <p>Welcome to my first blog post! This is where I share my thoughts on web development, design, and technology.</p>
      <p>Learning HTML is the first step in becoming a great web developer. By understanding how to structure content, you can create readable, accessible, and well-organized pages.</p>
      <blockquote>"Web development is both an art and a science. Structure matters!"</blockquote>
      <p>Stay tuned for more posts, where I'll dive into CSS and JavaScript to make these pages more dynamic.</p>
    </section>
  </article>
</body>
</html>
```

5. Project:

Build a Simple Image Gallery

Objective

Create a basic image gallery displaying multiple images in a structured layout.

Skills You'll Learn

- Using the `` tag to display images
- Structuring an image grid with `<div>`
- Linking images to larger versions

Requirements

You'll need:

- At least six images (you can use placeholder images from [Placeholder.com](https://placeholder.com) or [Unsplash](https://unsplash.com))
- A basic understanding of `<div>` for grouping content

Instructions

1. **Create a new file** and name it `image-gallery.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Create a `<div>` container** to hold all images.
4. **Add multiple `` elements** inside the container.
5. **Wrap each `` in an `<a>` tag** to allow users to click for a larger view.
6. **Test the gallery** by opening the file in a browser.

Challenges/Extensions

- Add captions below each image using `<figcaption>`.
- Organize images into categories using multiple `<section>` elements.
- Add inline CSS to make the images display in a grid.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Simple Image Gallery</title>
</head>
<body>
  <h2>Image Gallery</h2>
  <div class="gallery">
    <a href="https://via.placeholder.com/800">
      
    </a>
    <a href="https://via.placeholder.com/800">
      
    </a>
    <a href="https://via.placeholder.com/800">
      
    </a>
    <a href="https://via.placeholder.com/800">
      
    </a>
    <a href="https://via.placeholder.com/800">
      
    </a>
    <a href="https://via.placeholder.com/800">
      
    </a>
  </div>
</body>
</html>
```

6. Project:

Create a Simple Resume Page

Objective

Build a basic online resume that includes a name, contact information, summary, skills, and experience.

Skills You'll Learn

- Structuring a resume with `<header>`, `<section>`, and ``
- Formatting text with `<h1>`, `<p>`, and ``
- Organizing information using lists

Requirements

You'll need:

- Placeholder text for your name, summary, and work experience
- A list of skills and previous job titles
- Optional: A profile image

Instructions

1. **Create a new file** and name it `resume.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use a `<header>` element** to display the name and contact information.
4. **Create a `<section>` for a short summary** about yourself.
5. **Add a skills section** using an unordered list (``).
6. **Include a work experience section** listing past job titles and companies.
7. **Test the layout** by opening the file in a browser.

Challenges/Extensions

- Add a profile picture using the `` tag.
- Use `<hr>` to visually separate sections.
- Create a separate section for education history.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>My Resume</title>
</head>
<body>
  <header>
    <h1>John Doe</h1>
    <p>Email: johndoe@example.com | Phone: (123) 456-7890</p>
  </header>

  <section>
    <h2>Summary</h2>
    <p>Motivated web developer with a passion for creating user-friendly websites. Skilled in HTML, CSS,
and JavaScript.</p>
  </section>

  <section>
    <h2>Skills</h2>
    <ul>
      <li>HTML & CSS</li>
      <li>JavaScript</li>
      <li>Responsive Web Design</li>
      <li>SEO Optimization</li>
    </ul>
  </section>

  <section>
    <h2>Work Experience</h2>
    <h3>Web Developer - ABC Company</h3>
    <p>2022 - Present</p>
    <ul>
      <li>Developed responsive websites for clients.</li>
      <li>Optimized web pages for faster loading times.</li>
    </ul>

    <h3>Junior Developer - XYZ Agency</h3>
    <p>2020 - 2022</p>
    <ul>
      <li>Assisted in building landing pages.</li>
      <li>Worked with a team to design user interfaces.</li>
    </ul>
  </section>
</body>
</html>
```

7. Project:

Create a Simple FAQ Page

Objective

Build a Frequently Asked Questions (FAQ) page with collapsible sections for each question.

Skills You'll Learn

- Structuring FAQs with `<details>` and `<summary>`
- Using semantic HTML for better readability
- Organizing content efficiently

Requirements

You'll need:

- At least three sample questions and answers
- Basic understanding of `<details>` and `<summary>` tags

Instructions

1. **Create a new file** and name it `faq.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. Use `<h1>` for the **page title** at the top.
4. **Create a `<section>`** to contain all FAQ items.
5. Use `<details>` and `<summary>` for each question, with the answer inside `<details>`.
6. **Test the functionality** by clicking on each question to expand/collapse it.

Challenges/Extensions

- Add more questions to make it a comprehensive FAQ.
- Style the FAQ page using inline CSS to improve readability.
- Include a contact link for users to submit additional questions.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>FAQ Page</title>
</head>
<body>
  <h1>Frequently Asked Questions</h1>
  <section>
    <details>
      <summary>What is this page about?</summary>
      <p>This is a simple FAQ page that answers common questions.</p>
    </details>
    <details>
      <summary>How does the FAQ section work?</summary>
      <p>Click on any question to expand and view the answer.</p>
    </details>
    <details>
      <summary>Can I add more questions?</summary>
      <p>Yes! Just add more `<details>` elements with questions and answers.</p>
    </details>
  </section>
</body>
</html>
```

8. Project:

Build a Simple To-Do List

Objective

Create a basic to-do list where users can add tasks.

Skills You'll Learn

- Structuring lists with `` and ``
- Using `<input>` fields for user input
- Adding a submit button for task entry

Requirements

You'll need:

- An input field for adding tasks
- A button to submit tasks
- An unordered list (``) to display tasks

Instructions

1. **Create a new file** and name it `todo-list.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Add an `<h1>` title** at the top.
4. **Create a `<form>`** with an `<input>` field and a submit button.
5. **Use a ``** below the form to display tasks.
6. **Test by manually adding tasks** inside the list.

Challenges/Extensions

- Add placeholder text in the input field.
- Use inline CSS to style the list.
- Create a version using JavaScript to allow dynamic task addition.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Simple To-Do List</title>
</head>
<body>
  <h1>To-Do List</h1>
  <form>
    <input type="text" placeholder="Enter a new task">
    <button type="submit">Add Task</button>
  </form>
  <ul>
    <li>Example Task 1</li>
    <li>Example Task 2</li>
  </ul>
</body>
</html>
```

9. Project:

Create a Basic Newsletter Signup Form

Objective

Build a simple email signup form that allows users to subscribe to a newsletter.

Skills You'll Learn

- Structuring forms with `<form>` and `<input>`
- Using the `email` input type
- Creating a submit button for form submission

Requirements

You'll need:

- An input field for an email address
- A submit button
- A short description encouraging users to subscribe

Instructions

1. **Create a new file** and name it `newsletter-signup.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Add an `<h1>` title** for the signup section.
4. **Create a `<form>`** containing:
 - A text description encouraging users to sign up.
 - An `<input>` field with `type="email"`.
 - A submit `<button>`.
5. **Test by entering an email address** in the field and clicking submit.

Challenges/Extensions

- Add a `placeholder` text inside the email input field.
- Use the `required` attribute to prevent empty submissions.
- Style the form using inline CSS to make it more visually appealing.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Newsletter Signup</title>
</head>
<body>
  <h1>Subscribe to Our Newsletter</h1>
  <p>Stay updated with the latest news and exclusive offers!</p>
  <form>
    <input type="email" placeholder="Enter your email" required>
    <button type="submit">Subscribe</button>
  </form>
</body>
</html>
```

10. Project:

Build a Basic Pricing Table

Objective

Create a simple pricing table that displays different plans with features and prices.

Skills You'll Learn

- Structuring tables with `<table>`, `<tr>`, and `<td>`
- Using `<th>` for table headers
- Displaying pricing and feature comparisons

Requirements

You'll need:

- At least three pricing plans (e.g., Basic, Pro, Premium)
- A list of features for each plan
- Price values for each plan

Instructions

1. **Create a new file** and name it `pricing-table.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use a `<table>`** to display pricing plans.
4. **Create a `<thead>` section** with column headers for plan names.
5. **Use a `<tbody>`** to list features and prices.
6. **Test the layout** by opening the file in a browser.

Challenges/Extensions

- Highlight the most popular plan using the `` tag.
- Add a “Sign Up” button in each column.
- Style the table with inline CSS to improve readability.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Pricing Table</title>
</head>
<body>
  <h1>Our Pricing Plans</h1>
  <table border="1">
    <thead>
      <tr>
        <th>Plan</th>
        <th>Basic</th>
        <th>Pro</th>
        <th>Premium</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <td>Price</td>
        <td>$9.99/month</td>
        <td>$19.99/month</td>
        <td>$29.99/month</td>
      </tr>
      <tr>
        <td>Storage</td>
        <td>10GB</td>
        <td>50GB</td>
        <td>100GB</td>
      </tr>
      <tr>
        <td>Support</td>
        <td>Email Support</td>
        <td>Priority Email Support</td>
        <td>24/7 Phone Support</td>
      </tr>
    </tbody>
  </table>
</body>
</html>
```

11. Project:

Create a Basic Event Invitation Page

Objective

Build a simple event invitation webpage that includes event details, an image, and an RSVP form.

Skills You'll Learn

- Structuring event details with `<section>` and `<h2>`
- Using `` to display an event banner
- Creating an RSVP form with `<input>` fields

Requirements

You'll need:

- A title for the event
- Date, time, and location details
- A banner image (use a placeholder if needed)
- A form for users to RSVP

Instructions

1. **Create a new file** and name it `event-invitation.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Add an `<h1>`** for the event title.
4. **Include an `` tag** to display an event banner.
5. **Use a `<section>`** to list event details such as date, time, and location.
6. **Create a simple RSVP form** with input fields for name and email.
7. **Test the page** by opening it in a browser.

Challenges/Extensions

- Add a dropdown (<select>) for meal preferences.
- Include a confirmation message after submitting the form.
- Use inline CSS to make the invitation visually appealing.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Event Invitation</title>
</head>
<body>
  <h1>You're Invited to Our Annual Gala!</h1>
  

  <section>
    <h2>Event Details</h2>
    <p><strong>Date:</strong> March 15, 2025</p>
    <p><strong>Time:</strong> 7:00 PM - 10:00 PM</p>
    <p><strong>Location:</strong> Grand Ballroom, City Center</p>
  </section>

  <section>
    <h2>RSVP</h2>
    <form>
      <label for="name">Name:</label>
      <input type="text" id="name" name="name" required>

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

      <button type="submit">Confirm Attendance</button>
    </form>
  </section>
</body>
</html>
```

12. Project:

Build a Basic Recipe Page

Objective

Create a simple recipe webpage that includes a recipe title, ingredients list, steps, and an image.

Skills You'll Learn

- Structuring content with `<section>` and `<h2>`
- Using lists (`` and ``) for ingredients and steps
- Displaying an image with ``

Requirements

You'll need:

- A recipe title and description
- A list of ingredients
- Step-by-step cooking instructions
- A food image (use a placeholder if needed)

Instructions

1. **Create a new file** and name it `recipe.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Add an `<h1>`** for the recipe title.
4. **Include an `` tag** to display the dish.
5. **Create an ingredients list** using ``.
6. **Write step-by-step instructions** using an `` ordered list.
7. **Test the layout** by opening the file in a browser.

Challenges/Extensions

- Add a “Prep Time” and “Servings” section.
- Include a notes section for additional tips.
- Use inline CSS to improve readability.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Recipe Page</title>
</head>
<body>
  <h1>Delicious Chocolate Chip Cookies</h1>
  

  <section>
    <h2>Ingredients</h2>
    <ul>
      <li>2 cups all-purpose flour</li>
      <li>1/2 tsp baking soda</li>
      <li>1/2 tsp salt</li>
      <li>3/4 cup unsalted butter, melted</li>
      <li>1 cup brown sugar</li>
      <li>1/2 cup white sugar</li>
      <li>1 tbsp vanilla extract</li>
      <li>1 egg + 1 egg yolk</li>
      <li>2 cups chocolate chips</li>
    </ul>
  </section>

  <section>
    <h2>Instructions</h2>
    <ol>
      <li>Preheat oven to 325°F (165°C) and line a baking sheet with parchment paper.</li>
      <li>In a bowl, mix flour, baking soda, and salt.</li>
      <li>In another bowl, beat melted butter, brown sugar, and white sugar.</li>
      <li>Stir in vanilla, egg, and egg yolk.</li>
      <li>Gradually mix dry ingredients into wet ingredients.</li>
      <li>Fold in chocolate chips and scoop dough onto baking sheet.</li>
      <li>Bake for 12-15 minutes until edges are golden brown.</li>
      <li>Let cool before serving.</li>
    </ol>
  </section>
</body>
</html>
```

13. Project:

Create a Simple Business Card Webpage

Objective

Build a digital business card webpage that displays a name, job title, contact information, and a profile image.

Skills You'll Learn

- Structuring content with `<section>` and `<h2>`
- Using `` for a profile picture
- Formatting contact details with `<p>` and `<a>`

Requirements

You'll need:

- A name and job title
- Contact details (email, phone, website)
- A profile image (use a placeholder if needed)

Instructions

1. **Create a new file** and name it `business-card.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Add an `<h1>`** for the name and a `<p>` for the job title.
4. **Insert an ``** tag for the profile picture.
5. **Use `<p>` and `<a>` tags** to list contact details.
6. **Test the layout** by opening the file in a browser.

Challenges/Extensions

- Add social media links using `<a>` tags.
- Use inline CSS to style the card and center it.
- Include a “Download vCard” button for easy contact saving.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Digital Business Card</title>
</head>
<body>
  <section>
    
    <h1>John Doe</h1>
    <p>Web Developer</p>
    <p>Email: <a href="mailto:johndoe@example.com">johndoe@
example.com</a></p>
    <p>Phone: <a href="tel:+1234567890">(123) 456-7890</a></p>
    <p>Website: <a href="https://example.com" target="_
blank">example.com</a></p>
  </section>
</body>
</html>
```

14. Project:

Build a Basic Portfolio Page

Objective

Create a personal portfolio webpage showcasing projects, skills, and a short bio.

Skills You'll Learn

- Structuring a webpage with `<header>`, `<section>`, and `<footer>`
- Using `` to list skills or projects
- Adding navigation links with `<a>`

Requirements

You'll need:

- A profile image (use a placeholder if needed)
- A short bio
- A list of skills and projects

Instructions

1. **Create a new file** and name it `portfolio.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use a `<header>`** to display a name and navigation links.
4. **Create an “About Me” section** with a short bio.
5. **Add a “Skills” section** using an unordered list (``).
6. **Include a “Projects” section** with project names and links.
7. **Create a `<footer>`** for contact information.
8. **Test the layout** by opening the file in a browser.

Challenges/Extensions

- Add a “Download Resume” button using `<a>` with a file link.
- Use inline CSS to make the layout more visually appealing.
- Include social media icons linking to profiles.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Portfolio</title>
</head>
<body>
  <header>
    <h1>Jane Doe</h1>
    <nav>
      <a href="#about">About</a> |
      <a href="#skills">Skills</a> |
      <a href="#projects">Projects</a> |
      <a href="#contact">Contact</a>
    </nav>
  </header>

  <section id="about">
    <h2>About Me</h2>
    <p>I'm a passionate web developer with experience in HTML, CSS, and JavaScript. I love building creative and user-friendly websites.</p>
  </section>

  <section id="skills">
    <h2>Skills</h2>
    <ul>
      <li>HTML & CSS</li>
      <li>JavaScript</li>
      <li>Responsive Design</li>
      <li>SEO Optimization</li>
    </ul>
  </section>

  <section id="projects">
    <h2>Projects</h2>
    <ul>
      <li><a href="#">Personal Blog</a></li>
      <li><a href="#">Portfolio Website</a></li>
      <li><a href="#">E-commerce Landing Page</a></li>
    </ul>
  </section>

  <footer id="contact">
    <h2>Contact</h2>
    <p>Email: <a href="mailto:jane.doe@example.com">jane.doe@example.com</a></p>
    <p>LinkedIn: <a href="https://linkedin.com/in/janedoe" target="_blank">linkedin.com/in/janedoe</a></p>
  </footer>
</body>
</html>
```

15. Project:

Create a Basic Product Page

Objective

Build a simple product page showcasing an item with a description, price, and purchase button.

Skills You'll Learn

- Structuring a product display with `<section>` and `<h2>`
- Using `` for product images
- Creating a call-to-action (CTA) button with `<button>`

Requirements

You'll need:

- A product name and description
- A product image (use a placeholder if needed)
- A price display and purchase button

Instructions

1. **Create a new file** and name it `product-page.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use a `<header>`** for the product name.
4. **Insert an ``** tag for the product image.
5. **Write a product description** inside a `<p>` tag.
6. **Display the price** inside a `<h3>` tag.
7. **Add a purchase button** inside a `<button>` tag.
8. **Test the layout** by opening the file in a browser.

Challenges/Extensions

- Add a “Reviews” section with sample customer feedback.
- Include a “Related Products” section linking to similar items.
- Use inline CSS to make the product page visually appealing.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Product Page</title>
</head>
<body>
  <header>
    <h1>Wireless Headphones</h1>
  </header>

  <section>
    
    <p>Experience crystal-clear sound with our latest wireless
headphones. Designed for comfort and long-lasting battery life.</p>
    <h3>Price: $99.99</h3>
    <button>Buy Now</button>
  </section>
</body>
</html>
```

16. Project:

Create a Simple “Coming Soon” Page

Objective

Build a minimal “Coming Soon” page that informs visitors about an upcoming website or product launch.

Skills You’ll Learn

- Using `<h1>`, `<p>`, and `<section>` for structure
- Adding a countdown or message with text
- Creating an email signup field for early access

Requirements

You’ll need:

- A title announcing something is “Coming Soon”
- A short description or teaser text
- An email input field for users to get updates (optional)

Instructions

1. **Create a new file** and name it `coming-soon.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use an `<h1>`** for the main announcement.
4. **Write a short description** using a `<p>` tag.
5. **(Optional) Add an email signup form** with an input and button.
6. **Test the page** by opening it in a browser.

Challenges/Extensions

- Add a countdown timer using JavaScript.
- Include social media icons for users to follow updates.
- Use inline CSS to center the content on the page.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Coming Soon</title>
</head>
<body>
  <section>
    <h1>Something Exciting is Coming Soon!</h1>
    <p>We're working on something amazing. Stay tuned for
updates.</p>
    <form>
      <input type="email" placeholder="Enter your email"
required>
      <button type="submit">Notify Me</button>
    </form>
  </section>
</body>
</html>
```

17. Project:

Build a Simple “404 Error” Page

Objective

Create a custom 404 error page that informs users when they reach a broken or missing link

Skills You’ll Learn

- Structuring a message with `<h1>`, `<p>`, and `<a>`
- Using semantic HTML for accessibility
- Creating a “Go Back” or “Return Home” button

Requirements

You’ll need:

- A message telling users the page isn’t found
- A link back to the homepage
- An optional image or icon to enhance the design

Instructions

1. **Create a new file** and name it `404.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use an `<h1>`** to display “Page Not Found.”
4. **Write a short message** explaining the issue inside a `<p>` tag.
5. **Add a link or button** that takes users back to the homepage.
6. **(Optional) Include an image** for visual appeal.
7. **Test the page** by opening it in a browser.

Challenges/Extensions

- Add an animated effect using CSS.
- Include a search bar to help users find what they need.
- Display a random fun fact or joke to lighten the mood.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>404 Not Found</title>
</head>
<body>
  <section>
    <h1>404 - Page Not Found</h1>
    <p>Oops! The page you're looking for doesn't exist.</p>
    <a href="index.html">Return to Home</a>
  </section>
</body>
</html>
```

18. Project:

Create a Basic Weather Info Page

Objective

Build a simple webpage that displays weather information for a city, including temperature and conditions.

Skills You'll Learn

- Structuring content with `<section>` and `<h2>`
- Using `` for weather icons
- Formatting temperature and condition details

Requirements

You'll need:

- A city name
- Sample weather details (temperature, condition)
- A weather icon (use a placeholder if needed)

Instructions

1. **Create a new file** and name it `weather.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use an `<h1>`** for the city name.
4. **Add an `` tag** for a weather icon.
5. **Display the temperature** using a `<h2>` tag.
6. **Write a short description** of the weather condition inside a `<p>` tag.
7. **Test the page** by opening it in a browser.

Challenges/Extensions

- Add multiple cities with their weather details.
- Use a `<table>` to compare temperatures across different locations.
- Link to a real weather API to pull live data (advanced).

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Weather Info</title>
</head>
<body>
  <h1>Weather in New York</h1>
  
  <h2>72°F</h2>
  <p>Sunny with a light breeze.</p>
</body>
</html>
```

19. Project:

Build a Simple Countdown Timer Page

Objective

Create a webpage that displays a countdown timer for an upcoming event.

Skills You'll Learn

- Structuring content with `<h1>`, `<p>`, and ``
- Displaying dynamic numbers for a countdown
- Understanding basic time formatting

Requirements

You'll need:

- An event title
- A target countdown date and time
- A placeholder for the countdown display

Instructions

1. **Create a new file** and name it `countdown.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use an `<h1>`** for the event title.
4. **Write a short message** explaining the countdown inside a `<p>` tag.
5. **Add a ``** where the countdown timer will be displayed.
6. **Test the page** by opening it in a browser.

Challenges/Extensions

- Add JavaScript to make the countdown timer dynamic.
- Include a background image for a festive look.
- Style the countdown numbers with inline CSS.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Countdown Timer</title>
</head>
<body>
  <h1>Countdown to New Year's Eve!</h1>
  <p>Time left until the new year:</p>
  <h2><span>00 Days 00:00:00</span></h2>
</body>
</html>
```

20. Project:

Create a Basic Testimonials Page

Objective

Build a simple testimonials page that displays customer reviews and feedback.

Skills You'll Learn

- Structuring testimonials with `<section>` and `<blockquote>`
- Using `` for customer profile pictures
- Formatting names and reviews properly

Requirements

You'll need:

- At least three sample testimonials
- Customer names and feedback text
- Optional: Profile pictures for each customer

Instructions

1. **Create a new file** and name it `testimonials.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use an `<h1>`** for the page title.
4. **Create a `<section>`** for each testimonial.
5. **Wrap the review text inside a `<blockquote>` tag.**
6. **Display the customer's name below the testimonial.**
7. **(Optional) Add a profile picture** using an `` tag.
8. **Test the page** by opening it in a browser.

Challenges/Extensions

- Add more testimonials for variety.
- Use inline CSS to style the layout.
- Include star ratings using Unicode symbols (★).

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Customer Testimonials</title>
</head>
<body>
  <h1>What Our Customers Say</h1>

  <section>
    
    <blockquote>"This product changed my life! Highly recommended!"</
blockquote>
    <p>- Sarah Johnson</p>
  </section>

  <section>
    
    <blockquote>"Excellent service and fast delivery. Will buy again!"</
blockquote>
    <p>- Mark Williams</p>
  </section>

  <section>
    
    <blockquote>"Great quality and value for money. Five stars!" ★★★★★
</blockquote>
    <p>- Emily Davis</p>
  </section>
</body>
</html>
```

21. Project:

Build a Simple Travel Destination Page

Objective

Create a webpage showcasing a travel destination with images, a description, and key attractions.

Skills You'll Learn

- Structuring content with `<section>` and `<h2>`
- Using `` for destination photos
- Formatting travel details using lists

Requirements

You'll need:

- A destination name and description
- At least one image of the location
- A list of top attractions or activities

Instructions

1. **Create a new file** and name it `travel-destination.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use an `<h1>`** for the destination name.
4. **Write a short description** inside a `<p>` tag.
5. **Insert an ``** tag to display an image of the destination.
6. **Create a list of attractions** using ``.
7. **Test the page** by opening it in a browser.

Challenges/Extensions

- Add a “Best Time to Visit” section.
- Include a Google Maps embed for location reference.
- Use inline CSS to make the layout more visually appealing.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Visit Paris</title>
</head>
<body>
  <h1>Explore Paris, France</h1>
  
  <p>Paris, the City of Light, is known for its rich history,
stunning architecture, and world-class cuisine.</p>

  <h2>Top Attractions</h2>
  <ul>
    <li>The Eiffel Tower</li>
    <li>The Louvre Museum</li>
    <li>Notre-Dame Cathedral</li>
    <li>Champs-Élysées & Arc de Triomphe</li>
    <li>Seine River Cruises</li>
  </ul>
</body>
</html>
```

22. Project:

Create a Basic Movie Review Page

Objective

Build a simple movie review page that includes a movie title, poster, rating, and a short review.

Skills You'll Learn

- Structuring content with `<section>` and `<h2>`
- Using `` for a movie poster
- Formatting review details with `<p>` and ``

Requirements

You'll need:

- A movie title
- A poster image (use a placeholder if needed)
- A short review and rating

Instructions

1. **Create a new file** and name it `movie-review.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use an `<h1>`** for the movie title.
4. **Insert an `` tag** for the movie poster.
5. **Write a short review** inside a `<p>` tag.
6. **Display a rating** using stars (★) or a numeric score.
7. **Test the page** by opening it in a browser.

Challenges/Extensions

- Add a “Cast & Crew” section with actor names.
- Include a “Watch Trailer” button linking to a YouTube video.
- Create multiple reviews and format them in a `<table>`.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Movie Review: Inception</title>
</head>
<body>
  <h1>Inception (2010)</h1>
  
  <p><strong>Director:</strong> Christopher Nolan</p>
  <p><strong>Genre:</strong> Science Fiction, Thriller</p>

  <h2>Review</h2>
  <p>"Inception is a mind-bending thriller that masterfully blends
dream and reality. With stunning visuals and a gripping storyline, it
keeps audiences on the edge of their seats."</p>

  <h2>Rating: ★★★★★ (5/5)</h2>
</body>
</html>
```

23. Project:

Build a Simple Online Poll Page

Objective

Create a webpage where users can vote on a question by selecting an option.

Skills You'll Learn

- Structuring a poll with `<form>` and `<input type="radio">`
- Grouping related inputs using `<fieldset>` and `<legend>`
- Creating a submit button for voting

Requirements

You'll need:

- A poll question
- At least three answer choices
- A submit button

Instructions

1. **Create a new file** and name it `poll.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use an `<h1>`** for the poll question.
4. **Create a `<form>` element** for user input.
5. **Use `<fieldset>` and `<legend>`** to group the poll options.
6. **Add multiple radio buttons (`<input type="radio">`)** for the answer choices.
7. **Include a submit button** to collect votes.
8. **Test the page** by selecting an option and clicking submit.

Challenges/Extensions

- Add a “View Results” section showing poll percentages.
- Use JavaScript to store and display poll results dynamically.
- Style the form with inline CSS for better presentation.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Online Poll</title>
</head>
<body>
  <h1>What's your favorite programming language?</h1>
  <form>
    <fieldset>
      <legend>Choose one:</legend>
      <input type="radio" id="html" name="language" value="HTML">
      <label for="html">HTML</label><br>

      <input type="radio" id="css" name="language" value="CSS">
      <label for="css">CSS</label><br>

      <input type="radio" id="javascript" name="language"
value="JavaScript">
      <label for="javascript">JavaScript</label><br>

      <input type="radio" id="python" name="language" value="Python">
      <label for="python">Python</label><br>
    </fieldset>
    <button type="submit">Vote</button>
  </form>
</body>
</html>
```

24. Project:

Create a Basic Music Playlist Page

Objective

Build a simple webpage that displays a list of songs in a music playlist with play buttons.

Skills You'll Learn

- Structuring content with `<section>` and `<h2>`
- Using `` for a song list
- Adding audio controls with the `<audio>` tag

Requirements

You'll need:

- A playlist title
- A list of at least three songs
- Play buttons using the `<audio>` tag (with sample MP3 links or placeholders)

Instructions

1. **Create a new file** and name it `playlist.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use an `<h1>`** for playlist title.
4. **Create an unordered list (``)** to display songs.
5. **For each song, add an `<audio>` tag** with controls for playback.
6. **Test the page** by playing the sample audio files.

Challenges/Extensions

- Add album artwork for each song using ``.
- Include song durations and artist names.
- Use inline CSS to style the playlist layout.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>My Music Playlist</title>
</head>
<body>
  <h1>My Favorite Songs</h1>
  <ul>
    <li>
      <strong>Song 1 - Artist A</strong><br>
      <audio controls>
        <source src="sample1.mp3" type="audio/mpeg">
        Your browser does not support the audio element.
      </audio>
    </li>
    <li>
      <strong>Song 2 - Artist B</strong><br>
      <audio controls>
        <source src="sample2.mp3" type="audio/mpeg">
        Your browser does not support the audio element.
      </audio>
    </li>
    <li>
      <strong>Song 3 - Artist C</strong><br>
      <audio controls>
        <source src="sample3.mp3" type="audio/mpeg">
        Your browser does not support the audio element.
      </audio>
    </li>
  </ul>
</body>
</html>
```

25. Project:

Create a Basic “Meet the Team” Page

Objective

Build a simple webpage that showcases team members with their names, roles, and profile pictures.

Skills You’ll Learn

- Structuring team profiles with `<section>` and `<h2>`
- Using `` for profile pictures
- Formatting names and roles properly

Requirements

You’ll need:

- A page title introducing the team
- At least three team members with names, roles, and photos
- Optional: Short bios for each member

Instructions

1. **Create a new file** and name it `team.html`.
2. **Set up the document structure** with the standard HTML boilerplate.
3. **Use a `<h1>`** for the page title.
4. **Create a `<section>`** for each team member.
5. **Add an `` tag** for the profile picture.
6. **Display the name inside an `<h3>`** and the role inside a `<p>`.
7. **Test the page** by opening it in a browser.

Challenges/Extensions

- Add a “Contact” button under each team member.
- Use inline CSS to format the layout neatly.
- Include social media links for each team member.

Solution Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Meet the Team</title>
</head>
<body>
  <h1>Meet Our Team</h1>

  <section>
    
    <h3>Jane Doe</h3>
    <p>CEO & Founder</p>
    <p>Jane leads our company with a passion for innovation and excellence.</p>
  </section>

  <section>
    
    <h3>John Smith</h3>
    <p>Lead Developer</p>
    <p>John is responsible for building cutting-edge web applications.</p>
  </section>

  <section>
    
    <h3>Emily Johnson</h3>
    <p>Marketing Manager</p>
    <p>Emily crafts creative strategies to reach and engage our audience.</p>
  </section>
</body>
</html>
```

Create a compelling website with HubSpot's free drag-and-drop website builder—no coding required

- ✓ Build and edit your website without any coding or technical skills
- ✓ Publish your business website faster with a pre-built theme
- ✓ Leverage custom modules to deliver a great user experience

Build Your Website for Free

