# Teaching Guide: Building the Brew & Bean Website

This guide will walk you through teaching students how to build a simple website for a fictional coffee shop called "Brew & Bean" using HTML, CSS, and JavaScript.

## Lesson 1: HTML Structure

### Objectives:

- Understand the basic structure of an HTML document

- Learn about common HTML tags and their purposes

### Steps:

1. Start with the HTML boilerplate:

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Brew & Bean Coffee Shop</title>

</head>

<body>

</body>

</html>

```

Explain each part of this structure.

2. Add the header and navigation:

```html

<header>

<h1>Brew & Bean Coffee Shop</h1>

</header>

<nav>

<ul>

<li><a href="#home">Home</a></li>

<li><a href="#menu">Menu</a></li>

<li><a href="#about">About</a></li>

<li><a href="#contact">Contact</a></li>

</ul>

</nav>

```

Discuss the semantic meaning of `<header>` and `<nav>` tags.

3. Create the main content sections:

```html

<div class="container">

<section id="home">

<h2>Welcome to Brew & Bean</h2>

<p>Enjoy our freshly roasted coffee and homemade pastries in a cozy atmosphere.</p>

</section>

<!-- Repeat for menu, about, and contact sections -->

</div>

```

Explain the use of `<section>` tags and `id` attributes.

4. Add the order button:

```html

<button id="order-button">Order Now</button>

```

### Activity:

Have students create the HTML structure, encouraging them to add their own content or additional sections.

## Lesson 2: Styling with CSS

### Objectives:

- Understand how to apply CSS to HTML elements

- Learn about selectors, properties, and values

- Explore layout and styling techniques

### Steps:

1. Add a `<style>` tag in the `<head>` section:

```html

<style>

/\* CSS will go here \*/

</style>

```

Explain the difference between internal, external, and inline styles.

2. Style the body and basic layout:

```css

body {

font-family: Arial, sans-serif;

line-height: 1.6;

margin: 0;

padding: 0;

background-color: #f4f4f4;

}

.container {

width: 80%;

margin: auto;

overflow: hidden;

padding: 20px;

}

```

Discuss the concept of the CSS box model.

3. Style the header and navigation:

```css

header {

background-color: #3c2f2f;

color: #fff;

text-align: center;

padding: 1rem;

}

nav {

background-color: #5c4033;

color: #fff;

padding: 0.5rem;

}

nav ul {

list-style-type: none;

padding: 0;

}

nav ul li {

display: inline;

margin-right: 10px;

}

nav ul li a {

color: #fff;

text-decoration: none;

}

```

Explain CSS selectors and how specificity works.

4. Style the menu items and order button:

```css

.menu-item {

background-color: #fff;

border-radius: 5px;

padding: 10px;

margin-bottom: 10px;

}

#order-button {

display: block;

width: 200px;

margin: 20px auto;

padding: 10px;

background-color: #5c4033;

color: #fff;

text-align: center;

text-decoration: none;

border-radius: 5px;

cursor: pointer;

}

```

Discuss the difference between classes and IDs in CSS.

### Activity:

Challenge students to add their own styles, such as hover effects on links or additional colors to the design.

## Lesson 3: Adding Interactivity with JavaScript

### Objectives:

- Understand how to include JavaScript in an HTML document

- Learn about event listeners and DOM manipulation

- Implement simple interactive features

### Steps:

1. Add a `<script>` tag just before the closing `</body>` tag:

```html

<script>

// JavaScript will go here

</script>

```

Explain why it's often better to place scripts at the end of the body.

2. Add an event listener to the order button:

```javascript

document.getElementById('order-button').addEventListener('click', function() {

alert('Thank you for your order! We\'ll prepare it right away.');

});

```

Discuss how event listeners work and the concept of callback functions.

3. Implement smooth scrolling for navigation:

```javascript

document.querySelectorAll('nav a').forEach(anchor => {

anchor.addEventListener('click', function (e) {

e.preventDefault();

document.querySelector(this.getAttribute('href')).scrollIntoView({

behavior: 'smooth'

});

});

});

```

Explain how this code selects all navigation links and adds click events to them.

### Activity:

Encourage students to add their own JavaScript features, such as a simple form validation for a contact form or a toggle for showing/hiding menu descriptions.

## Extensions and Further Learning:

1. Responsive Design: Introduce media queries to make the site mobile-friendly.

2. External Files: Move CSS and JavaScript to external files and link them to the HTML.

3. Additional Pages: Create separate pages for the menu, about, and contact sections.

4. Forms: Add a contact form or newsletter signup form with JavaScript validation.

5. API Integration: Use a simple API (e.g., for displaying a random coffee fact) to introduce AJAX concepts.

Remember to emphasize best practices throughout, such as writing semantic HTML, using meaningful class and ID names, and keeping JavaScript unobtrusive. Encourage students to experiment and customize their websites to reinforce learning and creativity.