Capsitech
IT Solution for the Businesses

Web Development Training Curriculum

www.capsitech.com

Table of Contents:

S. No.	Modules
1	Static UI Development
2	JavaScript

Module 1: Static UI Development

1. HTML Basics

- Structure of an HTML document.
- Common elements: headings, paragraphs, links, images, lists.
- References:
  - W3Schools HTML Tutorial
  - freeCodeCamp: Learn HTML

2. CSS Basics

- Selectors, properties, and values.
- Styling text and elements: colors, fonts, spacing, and borders.
- Box model concepts.
- References:
  - W3Schools CSS Tutorial
  - MDN Web Docs: CSS Basics

3. Advanced HTML & CSS

- Semantic HTML: <header>, <footer>, <section>, <article>.
- Layout techniques: Flexbox and Grid.
- Positioning elements: static, relative, absolute, fixed, sticky.
- References:
  - MDN Web Docs: HTML Elements
  - CSS-Tricks: A Complete Guide to Flexbox
  - CSS-Tricks: A Complete Guide to Grid
  - MDN Web Docs: CSS Positioning

4. Responsive Design

- Principles of responsive design.
- Media queries and relative units (%, em, vh).
- References:
  - MDN Web Docs: Responsive Design
  - freeCodeCamp: Responsive Web Design Principles

5. CSS Frameworks: Tailwind CSS

- Introduction to utility-first CSS.
- Configuring Tailwind for a project.

- Common utility classes: typography, layout, spacing, and colors.
- Building components like buttons, cards, and navigation bars.
- References:
  - Tailwind CSS Documentation
  - freeCodeCamp: Tailwind CSS Crash Course

6. CSS Frameworks: Bootstrap

- Overview of Bootstrap's grid system.
- Using prebuilt components: modals, carousels, forms, and buttons.
- Customizing themes using Bootstrap variables.
- References:
  - Bootstrap Documentation
  - freeCodeCamp: Bootstrap 5 Tutorial

7. Animations and Interactivity

- CSS transitions and animations.
- Pseudo-classes (:hover, :focus) and pseudo-elements (::before, ::after).
- Adding interactivity with Bootstrap and Tailwind utilities.
- References:
  - MDN Web Docs: CSS Transitions
  - CSS-Tricks: A Complete Guide to Hover Effects
  - Tailwind CSS Animation Utilities

8. Figma for Developers

- Understanding design files in Figma.
- Extracting assets and inspecting design details.
- Translating Figma designs into HTML/CSS code.
- References:
  - Dev: Figma for Beginners

Task for Module 1

Build a Complete Responsive Website

Objective:

Create a responsive website that adheres to modern design and development standards using HTML, CSS, and a CSS framework of your choice (Tailwind CSS or Bootstrap).

Instructions:

- Design link will be provided by your mentor.
- Implement clean and well-structured code.
- Utilize the provided assets and design guidelines.
- Use Tailwind CSS or Bootstrap to style and layout the website.
- Ensure responsiveness across various screen sizes using appropriate grid and flex utilities from the chosen framework.
- Add interactivity and animations such as hover effects and transitions to enhance user experience.

Deliverables:

1. Comprehensive documentation detailing the following:

a. Design choices and considerations.

b. Frameworks and tools used.

c. Key features and implementation details.

Module 2: JavaScript

1. Introduction to JavaScript
    - What is JavaScript, and how to execute JavaScript code.
    - References:
        - MDN Web Docs: Introduction to JavaScript
2. Variables
    - let, const, var: Differences in behavior and scope.
    - References:
        - Var vs Const vs Let (ES6) - Beau teaches JavaScript
        - MDN Web Docs: var, let, and const
3. Comparisons
    - Different types of comparisons: ==, ===, Object.is.
    - References:
        - MDN Web Docs: Equality Comparisons
4. Data Types
    - Value types vs. reference types.
    - References:
        - JavaScript.info: Data Types
5. Functions
    - Writing and understanding functions in JavaScript.
    - References:
        - MDN Web Docs: Functions
6. Fetch API
    - Introduction to the fetch API for making network requests.
    - References:
        - MDN Web Docs: Fetch API
7. Local Storage
    - Storing and retrieving data using localStorage.
    - References:
        - MDN Web Docs: Window.localStorage
8. DOM Manipulation
    - Basic JavaScript DOM manipulation.

- References:
  - MDN Web Docs: DOM Manipulation

9. Form Handling
   - Form submission and error handling with JavaScript.
   - References:
     - JavaScript.info: Forms
10. Task: Contact Form
    - Create a contact form with HTML, CSS, and JavaScript.
    - Apply proper validation for each field using JavaScript.
11. Arrays
    - JavaScript arrays and array helper methods like map, forEach, filter.
    - References:
      - MDN Web Docs: Array Methods
12. Objects
    - JavaScript objects and helper methods like Object.keys, Object.values, and Object.entries.
    - References:
      - MDN Web Docs: Working with Objects
13. Spread and Rest Operators
    - Using spread operators to create shallow copies of objects and arrays.
    - Understanding rest operators.
    - References:
      - JavaScript.info: Rest and Spread
14. Destructuring
    - Array and object destructuring.
    - References:
      - Destructuring in ES6 - Beau teaches JavaScript
      - MDN Web Docs: Destructuring Assignment
15. Functions as Variables
    - Treating functions like normal variables and using callbacks.
    - References:
      - JavaScript.info: Functions

16. Timers

- Using setTimeout and setInterval.
- References:
  - MDN Web Docs: Timers

17. Promises and Async/Await

- Understanding promises, then, catch.
- Using async/await and handling errors with try-catch blocks.
- References:
  - MDN Web Docs: Promises
  - JavaScript.info: Async/Await

Capsitech
IT Solution for the Businesses

Task for Module 2

Build a Complete Contact Form

Objective: Create a fully functional contact form that adheres to modern JavaScript standards, including validation, interactivity, and proper error handling.

Instructions:

1. Use HTML, CSS, and JavaScript to create a contact form.
2. Include validation for fields such as name, email, and message.
3. Leverage JavaScript DOM manipulation to dynamically show error messages.
4. Implement a feature to simulate form submission and success feedback using promises.
5. Style the form using modern CSS techniques or frameworks like Tailwind CSS or Bootstrap.
6. Ensure responsiveness for various screen sizes.

Deliverables:

- Comprehensive documentation including:
  - Validation and interactivity features.
  - Frameworks and tools used.
  - Code structure and comments for better readability.