Certificate Program in Full-Stack Development Project

Course title	**User Interfaces**
Course number	420-WC4-AB
Hours	60
Ponderation *Ratio of lecture, practical and homework hours*	2-2-3
Credits	2.33
Competency statement(s) and code(s)	00ST Develop non-transactional Web applications. Element 6 only: 00ST.6 Program the client-side application logic.
Prerequisite (s)	420-WB4-AB Web Design 420-SA5-AB Database
Cohort	FSD-13
Start date	January 22, 2025
End date	March 6, 2025
Day(s) and times	Monday-Friday, 9:00 AM. - 2:30 PM.
Classroom/lab number	12
Semester	Winter 2026
Teacher	Khattar Daou, Ph.D.
Teachers' contact info	Khattar.Daou@JohnAbbott.qc.ca
Course format (F2F, online, hybrid)	Online

Team Project

Enhanced Team Case Projects for Web UI with JavaScript

Enhanced Team Case Projects for Web UI with JavaScript

Your team will enhance your previously designed website by integrating JavaScript-based functionalities. Work in a group of 3–4 members to refine the structure, ensuring a scalable navigation system that supports additional pages. Each member is responsible for designing and coding at least two pages, implementing a common layout, and incorporating responsive design principles. Ensure your project adheres to best practices for maintainability and scalability.

Case 1: Team Project Setup & Navigation System

Objective:

- Establish a structured web project in teams.
- Implement a simple JavaScript-powered navigation system.

Tasks:

1. Each team creates a repository and defines folder structures.
2. Develop a navigation bar using HTML & CSS.
3. Implement JavaScript to enable active page highlighting and smooth scrolling.
4. Ensure accessibility compliance.
5. Document project setup and team responsibilities.

Case 2: Interactive Features with JavaScript

Objective:

- Implement JavaScript-based interactivity to enhance user experience.

Tasks:

1. Each student selects and implements one interactive feature (e.g., dropdown menu, modal popup, animated button effects, or a simple calculator).

2. Use event listeners to trigger interactions.
3. Ensure smooth transitions and accessibility compliance.
4. Document the feature's implementation and how it improves user experience.

Case 3: Using Arrays and Loops for Dynamic Content

Objective:

- Manipulate and display dynamic data using JavaScript arrays and loops.

Tasks:

1. Create an array of at least five items (e.g., product list, testimonials, services).
2. Use loops to dynamically generate content.
3. Assign different responsibilities (one student for the array, another for loop logic, another for UI rendering).
4. Display the data within the project's UI.

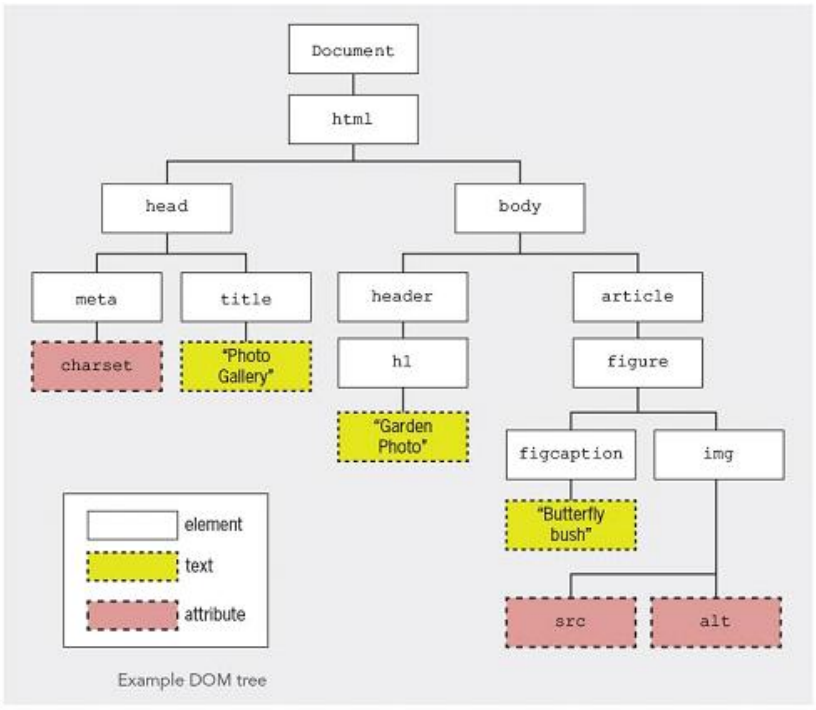
Case 4: DOM Manipulation in Action

Objective:

- Understand and manipulate the DOM dynamically.

Tasks:

1. Each student must identify and manipulate at least one DOM element (e.g., adding, removing, or modifying elements dynamically).
2. Demonstrate event-driven DOM changes.
3. Ensure visual feedback (e.g., color change, text updates, animations).
4. Document the before-and-after state of the DOM tree.



Case 5: Form Validation & Error Handling

Objective:

- Implement real-time form validation and error handling.

Tasks:

1. Implement validation for different input types (e.g., email, number, text fields).
2. Display custom error messages and prevent form submission for invalid inputs.
3. Ensure real-time feedback (e.g., valid/invalid states with CSS).
4. Test for various edge cases and document findings.

Case 6: Dynamically Updating a Table

Objective:

- Use JavaScript to update an HTML table dynamically.

Tasks:

1. Populate a table using an array of objects.
2. Implement a form that allows adding new rows.
3. Enable editing and deleting rows.
4. Use local storage to persist data.

Case 7: Feedback Form with Interactive Selection

Objective:

- Implement an interactive form with data storage.

Tasks:

1. Create a feedback form with rating options.
2. Store user selections in an array or object.
3. Use local storage to retain feedback.
4. Display summarized feedback results dynamically.

Case 8: Team-Based UI/UX Evaluation

1. **Objective:**
2. Evaluate project usability and accessibility.
3. **Tasks:**
4. Teams review each other’s projects and provide feedback on UI/UX.
5. Use Lighthouse or similar tools for accessibility testing.
6. Document findings and suggest improvements.

Case 9: Fetching & Displaying External Data (AJAX)

Objective:

- Use JavaScript to fetch and display external data.

Tasks:

1. Fetch data from an API (e.g., weather, quotes, news).
2. Display fetched data dynamically.
3. Implement error handling for failed API calls.
4. Document API usage and integration process.

Case 10: Enhancing UI with JavaScript Libraries

Objective:

- Introduce students to JavaScript libraries for better UI enhancements.

Tasks:

1. Implement a feature using a JavaScript library (e.g., jQuery, Chart.js, AOS for animations).

2. Compare native JavaScript vs. library-based implementation.
3. Ensure integration with the existing project UI.
4. Document challenges and solutions encountered.