DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

ECE 9065 – Web Application Development

Lab Assignment #3 - Stock and User Information Dashboard

Deadlines:

- Submission deadline: Friday, Nov. 22, 2024, at 11:55 pm.
- Late submissions are accepted until Nov. 25, 2024, at 11:55 pm with a 1% penalty each day.
- Demonstration deadline: End of Nov 29, 2024. (if requested)

Overview:

This lab assignment is designed to help students develop a web-based **Computer Stock and User Information Dashboard**. The dashboard will allow users to view, manage, and interact with computer model data and associated user portfolios. The lab is split into two parts:

1. **Part 1**: Building the core interface and user interaction for displaying and managing user and computer stock data.
2. **Part 2**: Implementing a detailed computer information viewer using advanced JavaScript techniques.

Prerequisites:

- Knowledge of HTML, CSS, JavaScript (ES6), and basic web development concepts.
- Familiarity with JSON data, array functions, classes, and modules.
- Understanding of Git for version control.

Setup:

- **GitHub Repository**: Clone the provided GitHub repository.
  - **Clone Link**: GitHub Classroom
- **Files**: Starter files include HTML, CSS, JS, and JSON files. Each section of the lab will reference specific files and tasks.

Objectives:

- Implement a dynamic, interactive user interface for managing computer data.
- Use JavaScript to fetch and manipulate JSON data.
- Apply array methods, classes, and event handling to build a responsive and user-friendly dashboard.

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

ECE 9065 – Web Application Development

Lab Assignment #3 - Stock and User Information Dashboard

Part 1: Computer Stock Dashboard

Overview:

The goal of Part 1 is to create a dashboard that displays users and their associated computer models. Users can interact with the interface to view, update, and manage data.

Objectives:

- Create a dynamic interface to list users and display their portfolios.
- Implement functionality to view, edit, and delete user and stock data.
- Style the application using CSS to ensure a responsive, clean layout.

Task 1: Display Users and Portfolios

1. **User List**:
    - Fetch data from `users.json` to display a list of users on the left side of the dashboard.
    - Each user should be clickable, showing their details in a form when selected.
2. **User Details**:
    - Create a form that displays detailed information about the selected user (first name, last name, email, etc.).
    - The form should allow editing, saving changes, and deleting the user.
3. **Computer Stock Portfolio**:
    - Fetch and display the user's computer stock data from their portfolio.
    - Use the `stocks-complete.json` file to map computer models to detailed information.
    - Each computer entry should show:
        - Model name
        - Quantity owned
        - An action to view more details or delete from the portfolio.
4. **Styling and Grid Layout**:
    - Use CSS to create a grid-based layout, separating users, details, and stock information.
    - Ensure that the interface is responsive, and working well on both desktop and mobile.

Example Implementation (Images from ch09):

Bonus (Optional):

- **Local Storage**: Implement local storage to persist user edits across sessions.
- **Search Functionality**: Add a search bar to quickly find users or computer models.

Files Needed:

- users.json: Contains user information and computer portfolios.
- stocks-complete.json: Details about each computer model.
- Lab3-p1.css: CSS for grid and layout styling.
- Lab3-p1.js: JavaScript for data fetching and DOM manipulation.

Part 2: Computer Information Viewer (Adapted from Chapter 10)

Overview:

In Part 2, you will build on the existing dashboard to add a detailed computer information viewer. This viewer will allow users to explore different computer models, see detailed specifications, and interact with advanced UI elements.

Objectives:

- Fetch and display data dynamically using the Fetch API.
- Use JavaScript array functions, classes, and prototypes.
- Build a clean, responsive interface to browse and view computer models.

Task 1: Dynamic Computer Information Display

1. **Data Fetching**:
   - Use the Fetch API to load computer data from `stocks-formatted.json`.
   - Display a list of computer models, categorized (e.g., Laptops, Desktops, Mini PCs).
2. **Filtering and Sorting**:
   - Implement filters (by category, manufacturer) to help users find specific models.
   - Allow sorting (by popularity, release date) for easy browsing.
3. **Computer Details Viewer**:
   - When a user clicks on a computer model, display detailed information on the right:
     - Full specifications
     - Manufacturer details
     - Popularity score or rating
   - Use `stocks-formatted.json` to map model identifiers to full descriptions.
4. **Advanced JavaScript Usage**:
   - Create a Computer class that encapsulates all relevant data.
   - Use array methods like .filter(), .map(), and .forEach() for data management.

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

ECE 9065 – Web Application Development

Lab Assignment #3 - Stock and User Information Dashboard

Example Implementation:

Refer to images ch10-1, ch10-2, and ch10-3 for a sample layout and features of the computer information viewer. Use these references to structure your JavaScript and CSS.

Files Needed:

- **stocks-formatted.json**: Contains detailed computer information.
- **Lab3-p2.css**: CSS for detailed viewer layout and styling.
- **Lab3-p2.js**: JavaScript for dynamic data fetching and display.

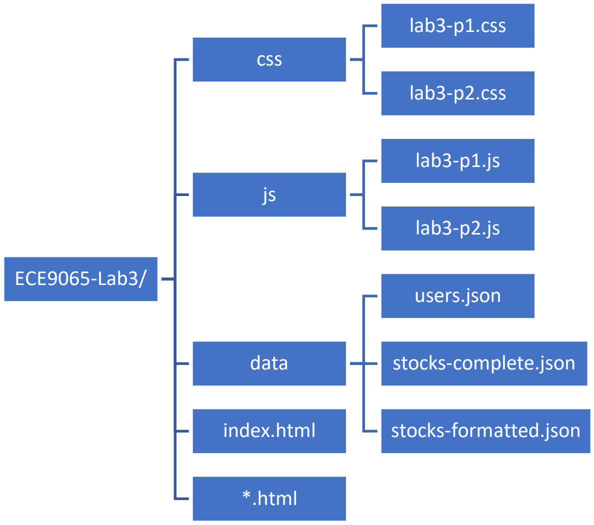
Submission Guidelines:

1. **Push Code to GitHub**:
   - Commit frequently with clear, descriptive messages.
   - Ensure your code is structured and commented.
2. **Evaluation Criteria**:
   - **UI Design and Responsiveness** (30 points): Clean, user-friendly, and responsive interface.
   - **Functionality** (40 points): Properly implemented features, smooth data handling.
   - **Code Quality** (20 points): Clean code, well-organized, and modular.
   - **Documentation** (10 points): Appropriate comments, clear code documentation.

Additional Resources:

- MDN Web Docs
- JavaScript Array Functions
- CSS Grid Layout

Example Submission Structure:



Good Luck!

This lab assignment encourages creativity and technical skills. Be sure to customize the interface for a polished, user-friendly dashboard. Happy coding!