Group Project 3 - JavaScript

Team Members:
- Austin Ng
- Cesar Herrera
- Ken Gu
- Nhan Nguyen

GitHub:
- User ID: nhannguyensf
- Link: WebDev317-03-Fall2023/group-project-3-JavaScript

Assignment description:

My team is tasked with coding the website using HTML, CSS, and integrating JavaScript based on the design we have from the last assignment. Teams need to develop a fully navigable site, including a homepage, product/resource pages, and intermediate pages.

What was added using JavaScript:

- Consistent Headers and Footers: Initially, headers and footers were embedded using iframes, which, while functional, had limitations in terms of performance and SEO. JavaScript provided a more elegant solution. By using JavaScript, we dynamically loaded the header and footer content into each page. This was typically done through AJAX requests fetching HTML content or by inserting common JavaScript functions that generate the header and footer on each page load.

- Toggle Dark/Light Mode button: We introduced a toggle button, easily accessible on the website, allowing users to switch between light and dark modes. This was achieved using JavaScript to dynamically change the CSS classes or styles of the website elements, thereby altering the color schemes.

- Homepage Slideshow: The homepage features a JavaScript-powered slideshow, which automatically cycles through promotional images or featured content. This could include new book releases, special offers, or any other key information that needs highlighting.

- Form Validation: To enhance user interaction, we implemented form validation using JavaScript. By validating user inputs on the client side, we improve the overall efficiency of the form submission process, ensure data integrity, and enhance user experience by providing real-time feedback.

- Dynamic Content on Product Pages: JavaScript was used to dynamically display product information, including prices and formats. This was achieved by reading data attributes (like data-price) and updating the corresponding HTML elements, preparing the groundwork for future database integration.

- Shopping Cart Functionality: The shopping cart functionality was significantly enhanced with JavaScript. It included dynamically displaying items from local storage, calculating and updating the total price, and providing options to remove items from the cart. When users added items to their cart, JavaScript was used to prompt them with a confirmation dialog. This dialog asked whether they wanted to continue shopping or proceed to the cart page, improving user experience and providing a smooth navigation flow. JavaScript's local storage feature was utilized to store and retrieve cart data. This allowed for persistent storage of cart items, ensuring that user selections were remembered across different browsing sessions.

Approach / What we did:

1. Setup and Collaboration:
   - Clone the provided starting repository.
   - Set up a GitHub repository for the team.
   - Ensure all team members have access to push and pull from the repository.

2. Planning:
   - Review the website's design, layout, and content with our team.
   - Decide on the navigation structure and the content for each page.
   - Assign tasks to each team member based on their strengths and preferences.

3. File Organization:
   - Use the Public directory as the root for all our website files.
   - Inside public, create separate directories for images (images), styles (CSS), and product pages (products), and add a Scripts folder to contain the .js files.
   - Ensure all files and directories have meaningful names.

4. Coding:
   - Start with the available structure of the HTML pages.
   - Add content to the home page, product/resource pages, intermediate pages, etc.
   - Improve the style using more CSS. Ensure consistency across all pages.
   - Test the website in different browsers to ensure compatibility.
   - Add JavaScript to make the website more interactive, user-friendly, and dynamic.

5. Review and Iteration:
   - Regularly commit and push changes to the shared repository.
   - Review each other's work and provide feedback.

- Make necessary revisions based on feedback and testing.

6. Documentation:
   - Write a PDF document following the requirements.

7. Submission:
   - Ensure all files and directories are within the public directory in our repository.
   - Use the command provided by the instructor to tag the version of our repository we want to submit.
   - Submit the PDF to Canvas.

Issues:

1. **Need to control the version when collaborating**

Description:

_Changes made by one developer can conflict with changes made by another.

Solution:

_Adopt a VCS (Git and GitHub platform). It allows multiple developers to work on the same project without interfering with each other's changes. Each developer can create branches, make changes, and then merge them back into the main codebase.

_Ensure that all team members understand the basics of using GitHub. Conduct workshops or training sessions about doing branches in Git.

_Implement a branching strategy like feature branching, Gitflow, or trunk-based development. This ensures that the main codebase remains stable, while development continues in separate branches.

_Encourage team members to regularly pull the latest changes from the main branch and push their own updates. This reduces the chances of major conflicts.

2. **Inconsistencies in the codebase**

Description:

_We had different style of writing code, such as indentation levels, varied naming conventions, or diverse ways of structuring code. This made the code hard to read, understand, and maintain. It led to merge conflicts and other collaboration issues.

Solution:

_We had team meetings to agree on a coding standard, which was based on widely-accepted industry standards or tailored to the team's preferences.

_We used Prettier tool to automatically detect and fix style issues. Integrating these tools into development process ensures that code adheres to a consistent style.

_ Implement a code review process where team members review each other's code before it's merged. This not only catches style inconsistencies but also improves code quality by leveraging multiple sets of eyes.

_ Hold regular meetings or check-ins to discuss any challenges or discrepancies in coding styles. This helped in addressing any deviations early on.