Group Project Final

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

GitHub:
- User ID: nhannguyensf
- Link: [WebDev317-03-Fall2023/group-project-final]

Assignment description:

Our team is embarking on an exciting project to develop a comprehensive website. The site will be crafted using HTML and CSS for structuring and styling, JavaScript for dynamic interactivity, and an SQL database for robust data management. This project aims to deliver a fully navigable and user-friendly website, encompassing a homepage, various product/resource pages, and intermediate pages that seamlessly connect the entire site.

What was added for this final version using JavaScript and SQL:
- Enabled the homepage to populate dynamically by calling data from the database.
    - Using JavaScript and SQL, the homepage of the website is designed to dynamically display content based on the data stored in the database.
    - Information like the names and photos of products are updated based on the database entries.
    - This approach allows for real-time updates without the need for manual page modifications, ensuring that the homepage always presents the most current and relevant information to users.
- Introduced a dynamic population of individual product pages.
    - Each product page is populated dynamically using data retrieved from the SQL database.
    - Product details such as images, descriptions, specifications, and prices are fetched and displayed in real-time.
    - This feature ensures consistency of information across the platform and allows for easy updates and maintenance of product data.
- Added the feature to filter products by type.
    - Implemented an interactive feature allowing users to filter products based on their types or categories.
    - Users can select a product category, and the page will dynamically update to show only products that match the selected type.

    - This feature enhances the user experience by making it easier to navigate and find specific products in a potentially vast inventory.
- Implemented the ability to filter products based on price.
    - Users can specify a price range, and the JavaScript code interacts with the SQL database to retrieve and display only the products that fall within the selected price bracket.
    - This feature aids in improving the shopping experience for customers who have specific budgetary requirements.
- Started the initial steps for adding book review features and saving them to the database.
    - Began the implementation of a book review feature, allowing users to submit reviews for products.
    - Made user's recent reviews the top of the reviews section for a more responsive feel.
    - Reviews, along with user ratings, can be added through a user interface and are stored in the SQL database.
    - Plans include displaying these reviews on product pages and using them for recommendations and ratings.
- Added the account creation feature, hashed the password and saved it to the database.
    - Developed an account creation feature where users can sign up for an account on the website.
    - Passwords are securely hashed using Bcrypt algorithms before being stored in the database.
    - This feature not only enhances user engagement by personalizing the experience but also ensures high standards of security and data protection.

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.
   - Add SQL and database.

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 updates. This reduces the chances of major conflicts.

2. **Inconsistencies in the codebase**

Description:

_We had different styles 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 merging 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 the Prettier tool to automatically detect and fix style issues. Integrating these tools into the 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.

Things Learned:

- Gained familiarity with creating a MySQL database.
  - Acquired skills in setting up and managing a MySQL database, a popular relational database management system.
  - Learned about the basic structure of a MySQL database, including the creation of databases and tables.
  - Understood how to define table schemas, specifying columns, data types, and constraints to store structured data efficiently.
- Learned how to export and import databases.
  - Gained experience in exporting and importing MySQL databases, a crucial skill for data backup and transfer.
  - Mastered the use of command-line utilities to export databases as SQL files and import them into different environments, ensuring data portability and ease of replication.
- Practiced altering table data.
  - Developed competency in modifying table structure and content within a MySQL database.
  - Learned how to add, delete, and modify columns in existing tables, as well as how to update data records, catering to evolving data storage needs.
  - Understood the importance of maintaining data integrity and structure while making alterations.
- Gained experience in setting up server-side code with Express.
  - Learned to set up server-side environments using Express, a fast, unopinionated, minimalist web framework for Node.js.
  - Became familiar with creating routes, handling HTTP requests (GET, POST, PUT, DELETE), and sending responses back to the client.
- Learned to call server-side API methods from client-side JavaScript
  - Acquired skills in making AJAX calls from client-side JavaScript to server-side APIs built with Express.

- Understood the process of sending data to the server and fetching data from the server asynchronously without reloading the web page.
- Gained knowledge in handling JSON data, managing API responses, and updating the DOM dynamically based on server responses.

These learning experiences contributed to a comprehensive understanding of full-stack web development. From database management with MySQL to server-side programming with Express and client-side scripting, these skills form the foundation for building robust, dynamic, and interactive web applications. They also pave the way for further exploration and mastery in the field of web development.

This project is not just about building a website; it's about crafting an experience. From the welcoming homepage to the in-depth product pages and the reliable SQL database at its core, every element of the site is designed with the user in mind. We aim to deliver a product that is not only a repository of information and resources but also a reflection of quality and innovation in web development.