This report outlines enhancements made to the Comox Valley Farmers Market website originally developed in DGL 103 and improved in DGL 113. It covers the addition of new features, implementation of JavaScript techniques, validation results, browser testing outcomes, and insights gained from the project implementation process.

Website Enhancement Report

Comox Valley Farmers Market

Emily Boucher DGL113 Winter 2024

Emily Boucher
April 2, 2024

Table of Contents

Table of Figures

Emily Boucher
April 2, 2024

Website Background Information

This website was originally created in the DGL103 CSS & HTML class where the objective was to take a current website and recreate it using CSS and HTML to make it more accessible and user-friendly. The original website had slow loading times and felt very unorganized and inaccessible, making it difficult for users to find specific information. In the updated version I condensed the website to 3 main pages, the home page, community page, and market page, utilizing JavaScript to play a video in the background on a loop, mobile-first design, navigation bars and smooth scroll.

Enhancement Strategy & JavaScript Techniques and Implementation

In this project, three additional JavaScript features were integrated into the Comox Valley Farmers Market website to enhance its functionality and user experience.

Feature 1: Market Carousel on the "Our Markets" Page

The addition of a carousel on the "Our Markets" page allows users to browse through multiple market photos horizontally. This carousel feature enhances user interaction and engagement by providing a visually appealing way to explore an image gallery. JavaScript was utilized to enable sliding motion upon user interaction, with logic to handle boundary conditions for a seamless carousel experience.

JavaScript Techniques and Implementation

DOM Selection and Initialization:

- Elements such as **track**, **slides**, **nextButton**, **prevButton**, **dotsNav**, and **dots** are selected from the DOM using **document.querySelector** and **Array.from** to convert **NodeList** into an array.
- **slideWidth** is calculated by retrieving the width of the first slide using **getBoundingClientRect().width**.

Setting Initial Slide Positions:

- The **setSlidePosition** function is defined to arrange each slide next to one another horizontally by setting the **left** style property based on the slide index multiplied by **slideWidth**.
- This function is applied to each slide using **slides.forEach(setSlidePosition)**.

Moving to the Selected Slide:

- The **moveToSlide** function is used to transition the carousel to the target slide by updating the **transform** property of the **track** element to shift it horizontally.
- The current and target slides are managed by adding and removing the **current-slide** class accordingly.

Updating Navigation Dots:

- The **updateDots** function updates the navigation dots by adding or removing the **current-slide** class to specify the currently active dot.

Show/Hide Navigation Arrows:

- The **hideShowArrows** function controls the visibility of navigation arrows (**prevButton** and **nextButton**) based on the current slide's position in the carousel.
- Arrows are shown or hidden by adding or removing the **is-hidden** class.

Event Listeners:

- Event listeners are added to the **prevButton**, **nextButton**, and **dotsNav** elements to handle user interactions (clicks on navigation arrows or dots).
- When the left arrow (**prevButton**) is clicked, it triggers a transition to the previous slide, updates the navigation dots, and adjusts arrow visibility.
- Similarly, when the right arrow (**nextButton**) is clicked, it transitions to the next slide, updates dots, and manages arrow visibility.
- Clicking on a dot (**dotsNav**) directly transitions to the corresponding slide, updates dots, and adjusts arrow visibility accordingly.

The JavaScript techniques used in this carousel involve DOM manipulation, event handling, and CSS style updates to create a dynamic and interactive carousel experience on the website. The code manages slide transitions, updates navigation indicators, and controls arrow visibility to enhance user navigation and engagement with the carousel component.

Feature 2: New Tab Confirmation for Document Access

Implemented on the community page, this feature offers users the choice to open a document in a new tab for easy access and to improve the accessibility of external resources. By adding a JavaScript function directly to the page, users benefit from enhanced flexibility in accessing resources without disrupting their browsing experience. A confirmation dialogue box prompts users to choose whether to open the document in a new tab, streamlining document retrieval and navigation within the website. This will significantly enhance the websites interface as the “Rules Document” now lives in the main of the repository and will be accessible independently of the original CVFM site.

JavaScript Techniques and Implementation

Enhanced User Experience

- Users can conveniently choose to open a document link in a new tab, providing flexibility and control over their browsing experience.

JavaScript Techniques Used

- **Event Handling**: Utilizes JavaScript event listeners (**addEventListener**) to detect user interaction with the document access button.
- **User Confirmation**: Implements a confirmation dialog box (**confirm**) to prompt users before opening the document in a new tab.
- **Conditional Navigation**: Redirects users to the document link in a new tab based on their confirmation choice.

Implementation Details:

- **Confirmation Dialog**: Upon clicking the document access button, a confirmation dialog prompts the user with the option to open the document in a new tab.

- **Conditional Navigation**: If the user confirms (chooses "Yes" in the dialog), the document opens in a new tab; otherwise (chooses "No"), the current page remains unchanged.
- **JavaScript Function**: The **checker()** function controls the behavior of the confirmation dialog and navigates to the document link accordingly.
- This JavaScript feature enriches the accessibility of document resources on the website, empowering users with a seamless and intuitive method to interact with external content.

Feature 3: Scroll to Top Button Across All Pages

The incorporation of a "Scroll to Top" button on every page of the farmers market website significantly enhances user navigation. This button allows users to swiftly return to the top of any page with a single click, reducing manual scrolling and improving overall accessibility. JavaScript was leveraged to monitor scroll position dynamically, toggling the visibility of the button based on predefined thresholds. Smooth scrolling functionality was implemented to ensure a seamless transition to the top of the page upon user interaction. These added features not only elevate the website's usability and accessibility but also contribute to an enriched browsing experience for visitors of the Comox Valley Farmers Market website.

JavaScript Techniques Used:

- **Event Handling**: The **window.addEventListener** method is employed to detect scroll events on the webpage.
- **DOM Manipulation**: The JavaScript code interacts with the Document Object Model (DOM) to dynamically add or remove CSS classes for the scroll-to-top button based on the user's scroll position.
- **Smooth Scrolling Animation**: Custom smooth scrolling animation is implemented using JavaScript to create a visually appealing transition to the top of the page.

Implementation Details:

- **Scroll Position Monitoring**: The **window.addEventListener('scroll')** function continuously monitors the scroll position. When the user scrolls beyond a specified threshold (e.g., 20 pixels from the top), the scroll-to-top button becomes visible by adding the **show** class to its HTML element.
- **Smooth Scrolling Function**: The **scrollToTop()** function is triggered when the scroll-to-top button is clicked. This function calculates the scrolling distance and uses a custom easing function (**easeInOutCubic**) to animate the scroll behavior smoothly back to the top of the page.
- **Performance Optimization**: The implementation considers performance optimization by using **requestAnimationFrame** for the scrolling animation, ensuring a more efficient and visually pleasing user experience.

Advantages:

- **Enhanced Navigation**: Users can easily navigate back to the top of the page with a single click, improving overall accessibility and usability.
- **Interactive User Experience**: The smooth scrolling animation adds a touch of interactivity and sophistication to the website, enhancing user engagement.

- **Cross-Page Consistency**: The scroll-to-top button is available across all pages, ensuring consistency in user interface elements and providing a seamless browsing experience throughout the website.

Validation Results

3 Webpages were checked for both HTML and dynamic HTML, while only 1 CSS styles sheet required checking. Web pages tested on:

Safari Version 16.6 (18615.3.12.11.2)

- Auto-play JavaScript function on Safari does not work in this browser.

Google Chrome Version 123.0.6312.107 (Official Build) (x86_64)

- All functions work as expected

HTML Results

All three HTML pages that had changes made to them for this project were put through HTML Validation checks on the W3 website. No major issues came about, however, a similar issue was across all 3 pages for the naming of the body class, I believe this to be irrelevant.

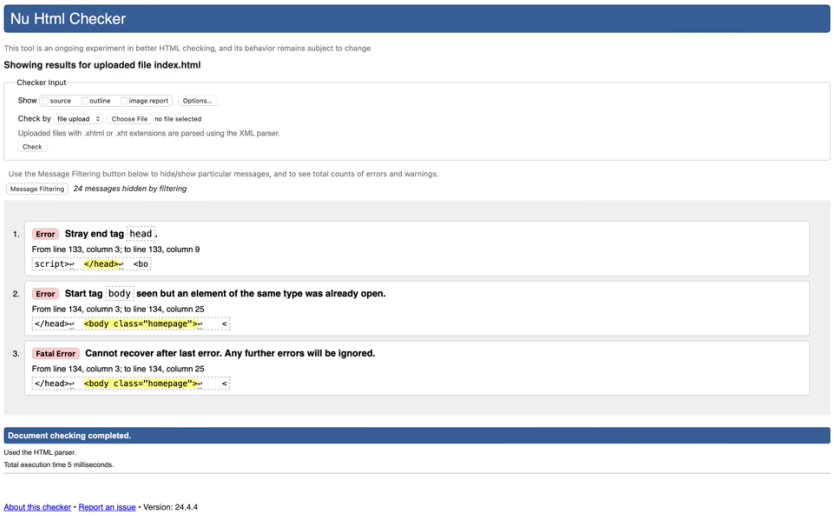

Figure 1 Homepage HTML check

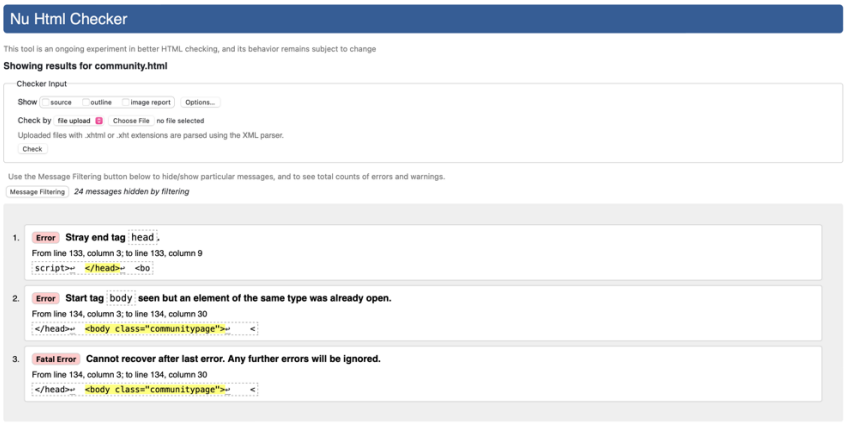

Figure 2 Community Page HTML Check

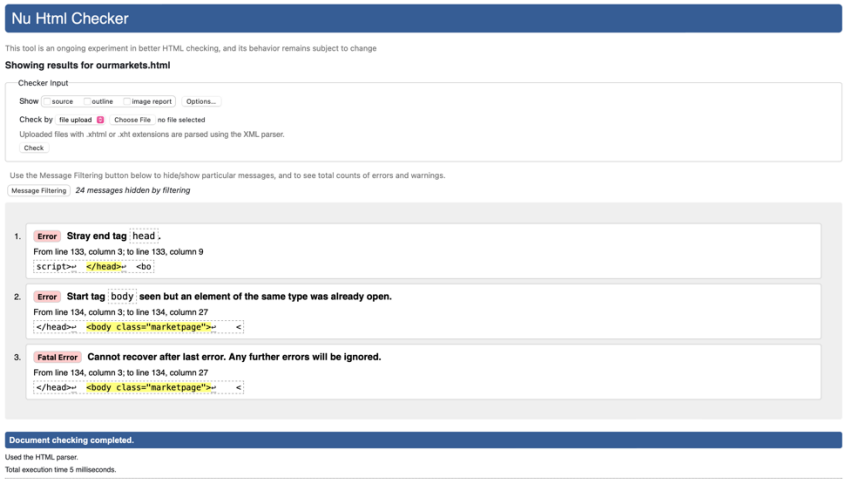

Figure 3 Our Markets Page HTML Check

CSS Validation Results

The main stylesheet which was used for the entire website was run through the CSS validator and no errors were found.

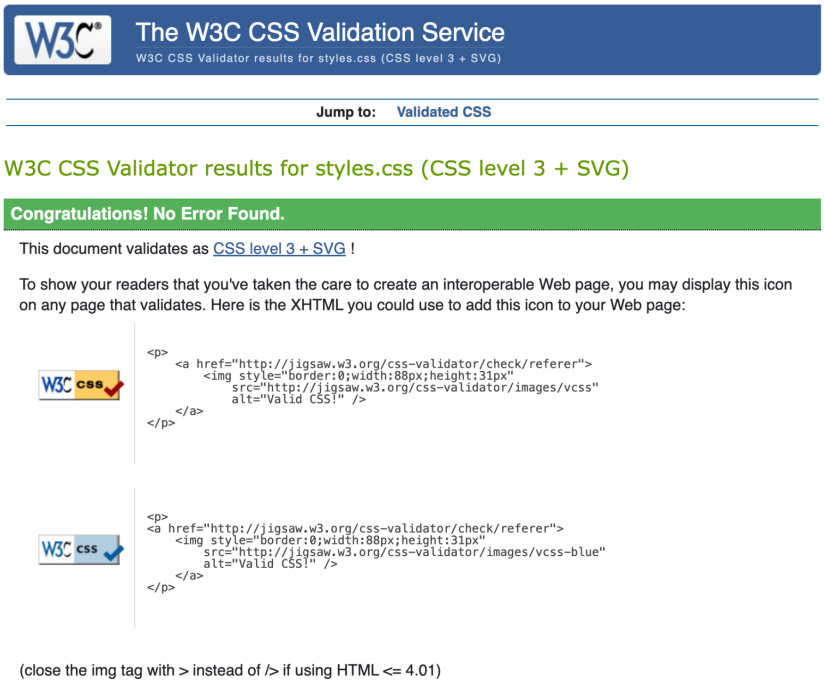

Figure 4 CSS Validation Results

Dynamic HTML Code Validation

Both the Homepage and the Market page had no errors or warnings, however, the community page ran an error due to a ul being a child of an ol. This error did not impact the function of the page so I determined it did not require any changes.

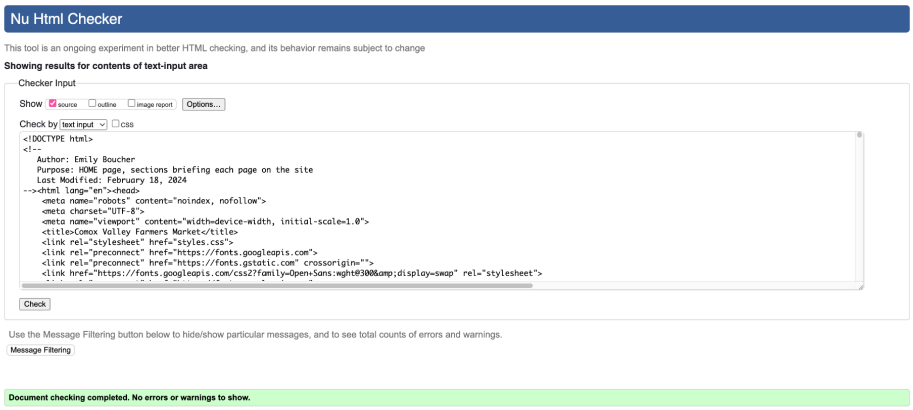

Figure 5 Homepage Dynamic HTML Validation Results

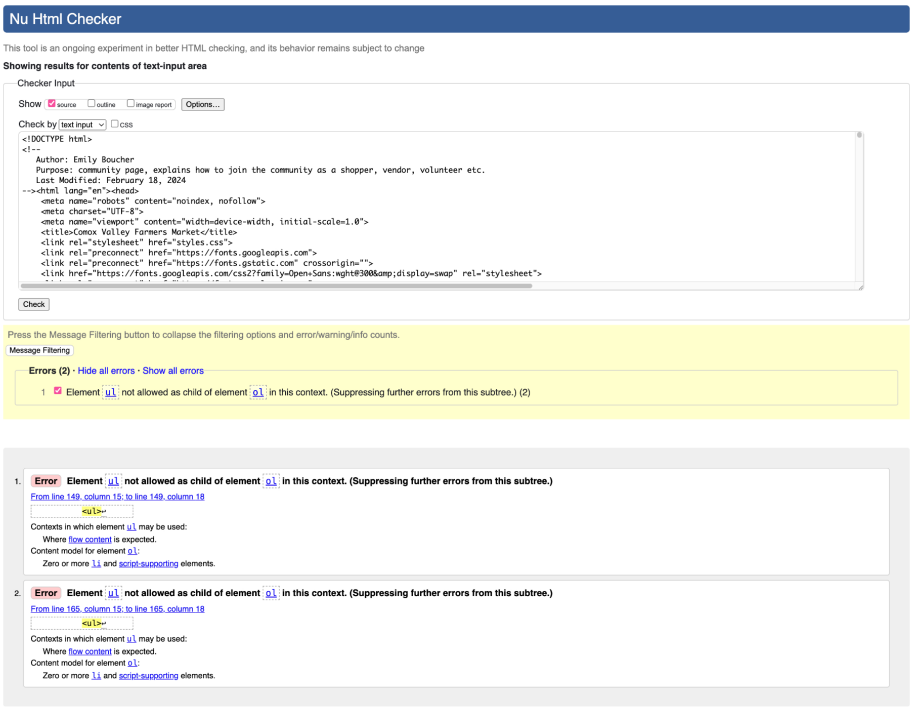

Figure 6 Community page Dynamic HTML Validation Results

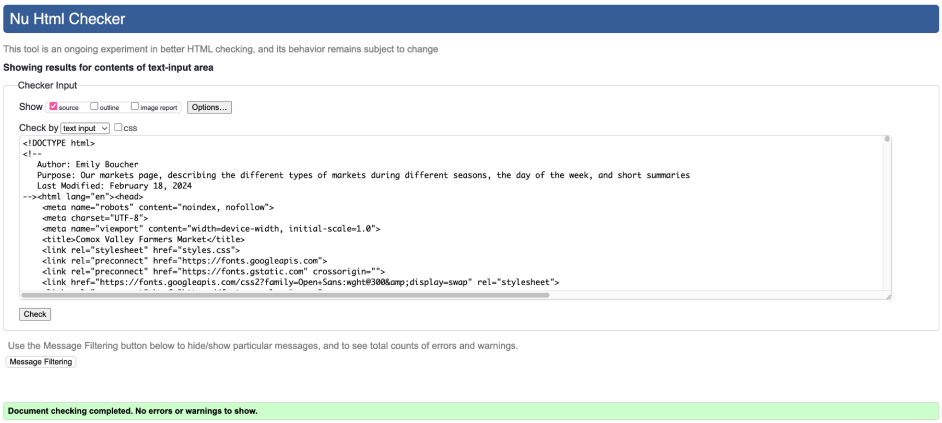

Figure 7 Our Markets Page Dynamic HTML Validation Results

Project Work Experience

It took me ____ hour(s) to complete the project.

21 hours

From the project work I learned the following:

I learned the vocabulary required to search for YouTube tutorials to learn how to do more specific JavaScript code. I was able to troubleshoot errors I was having and search online for help. I also became more familiar with organizing my functions and being able to call on them throughout the code. I appreciate using the app.js file but also learned more about adding code directly to certain pages

I found the following helpful for completing the project (textbook, lecture notes, video, online resources, etc.):

YouTube videos were very helpful for completing this project. Along with reading material and resources on the course outline and FAQ page on Brightspace.

The difficulties I encountered with the project:

Initially trying to establish JavaScript code that would enhance my website was difficult as I am very unfamiliar with dynamic code and what the possibilities actually are.

Other comments and suggestions:

I do not have any additional comments or suggestions for this project.

Conclusion

In conclusion, the enhancements made to the Comox Valley Farmers Market website have successfully addressed several key usability and accessibility issues identified in the original version. Through the strategic use of JavaScript, we've introduced interactive features like carousels, document access confirmations, and a convenient "Scroll to Top" button across all pages. These additions not only improve user navigation and engagement but also contribute to a more seamless browsing experience. The validation results confirm the robustness of the HTML and CSS implementation, underscoring the project's attention to quality and standards compliance. Overall, this project has been a valuable learning experience, providing exposure to JavaScript techniques and reinforcing the importance of user-centered design principles in web development.

References

Powell, K. (2019a, January 3). *How to code a carousel with HTML, CSS and JavaScript - From Scratch (Part 1)*. YouTube. https://www.youtube.com/watch?v=VYsVOamdB0g

Powell, K. (2019b, January 3). *How to code a carousel with HTML, CSS and JavaScript - From Scratch (Part 2)*. YouTube. https://www.youtube.com/watch?v=gBzsE0oieio

Powell, K. (2023, December 21). *Incredible scroll-based animations with CSS-only*. YouTube. https://www.youtube.com/watch?v=UmzFk68Bwdk

Author, G. (2015, March 22). *3 ways JavaScript Can Make your website more engaging*. Revenue River. https://www.revenueriver.co/thecuttingedge/javascript