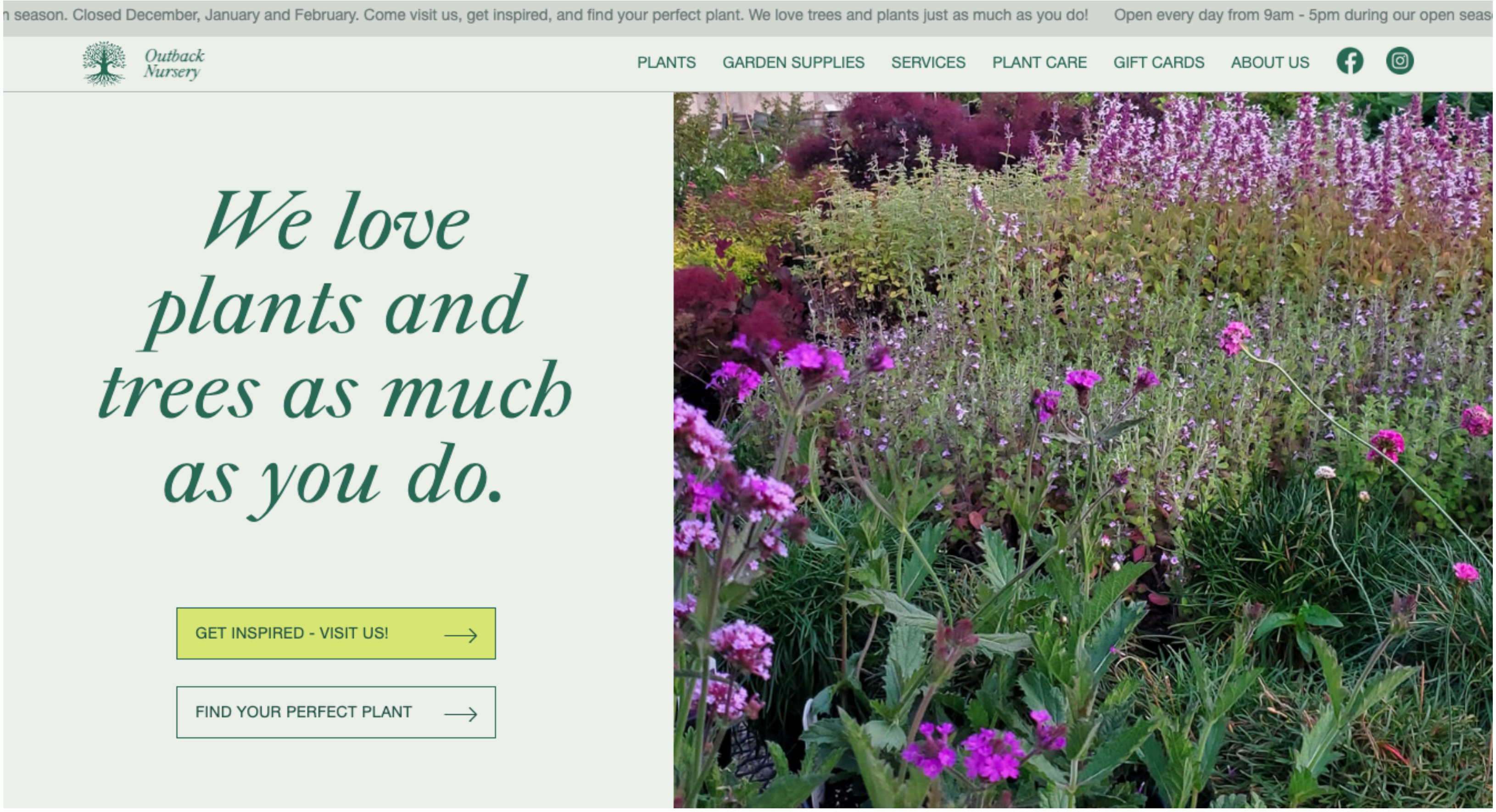
# Website Enhancement Report

## OUTBACK NURSERY DEMO SITE



n season. Closed December, January and February. Come visit us, get inspired, and find your perfect plant. We love trees and plants just as much as you do! Open every day from 9am - 5pm during our open seas

We love plants and trees as much as you do.

# Table of Contents

# 01 Website Background Information

I developed the Outback Nursery Demo Site as part of my Capstone 309 project this term. I came up with a comprehensive design system for the site, conducted an in-depth content analysis, mapped out the site's architecture, and revamped its layout and structure. Additionally, I curated the typography, colors, graphics and icons to elevate the website's visual appeal and user experience.

For the final stage of the project, I coded the website using HTML, CSS and vanilla JavaScript.

# 02 Features Added

Here are descriptions of the three JavaScript features I added to enhance the Outback Nursery demo site:

1. Banner Text Scrolling Effect:
   - When a user clicks on an element with the ID "scrollingText", it toggles the class "paused" on and off. This class controls the animation scrolling effect on the banner text.
2. Accordion Functionality:
   - This feature adds interactivity to the accordion element on the webpage. When a button within the accordion is clicked, it opens or closes the corresponding panel. It also ensures that only one panel can be open at a time by closing all other panels.
3. Carousel and Pagination:
   - This feature creates a carousel with pagination dots. It allows users to navigate between slides by clicking on the pagination dots or using previous and next buttons. Additionally, it automatically cycles through the slides at regular intervals, pausing the automatic sliding when the mouse hovers over the carousel and resuming it when the mouse moves away.

# 03 JavaScript Techniques

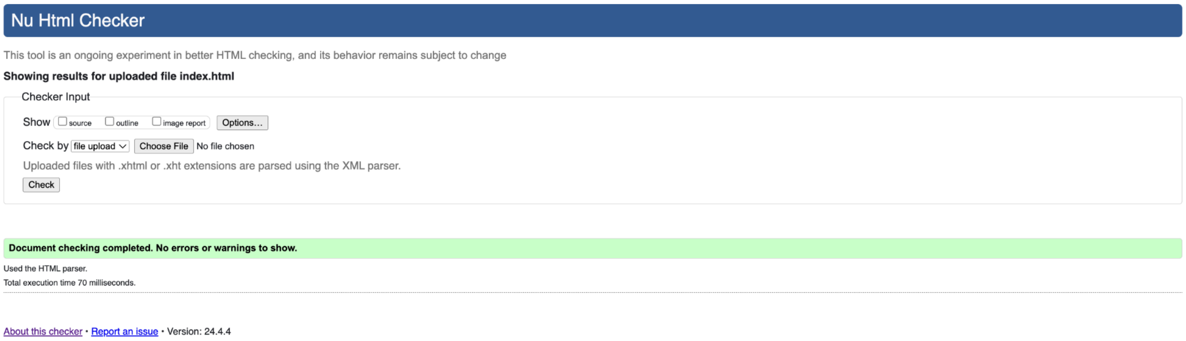
Here are the JavaScript techniques used for each effect:

1. Banner Text Scrolling Effect:
    - Event Listener: The addEventListener method is used to listen for a "click" event on the element with the ID "scrollingText".
    - Toggle Class: Within the event listener function, the classList.toggle() method is used to toggle the "paused" class on and off, which triggers the CSS animation scrolling effect for the banner text.
2. Accordion Functionality:
    - DOM Selection and Manipulation: The script selects all accordion buttons and panels using querySelectorAll() and getElementById(), respectively, to access and manipulate their properties.
    - Event Handling: Event listeners are added to each accordion button to respond to user clicks. When a button is clicked, the script identifies the corresponding panel and adjusts its height accordingly to expand or collapse it.
    - Looping: The script uses forEach() loops to iterate through collections of buttons and panels, enabling consistent handling of multiple elements.
3. Carousel and Pagination:
    - Function Declarations: Functions like showSlides, moveSlide, and currentSlide are declared to manage the carousel's behavior, including displaying slides, moving between slides, and setting the current slide.
    - Interval-based Animation: The setInterval() function is used to automatically advance the carousel's slides at regular intervals, defined in milliseconds. This creates the auto-scrolling effect.
    - Event Handling (Mouse Events): Event listeners are added to the carousel element to pause and resume the automatic sliding when the mouse hovers

over or moves away from the carousel area. This is achieved using the "mouseover" and "mouseout" events.

# 04 HTML & CSS Validation Results

## HTML Validation Results



Nu Html Checker

This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change

**Showing results for uploaded file index.html**

Checker Input

Show □ source □ outline □ image report Options...

Check by file upload Choose File No file chosen

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

Check

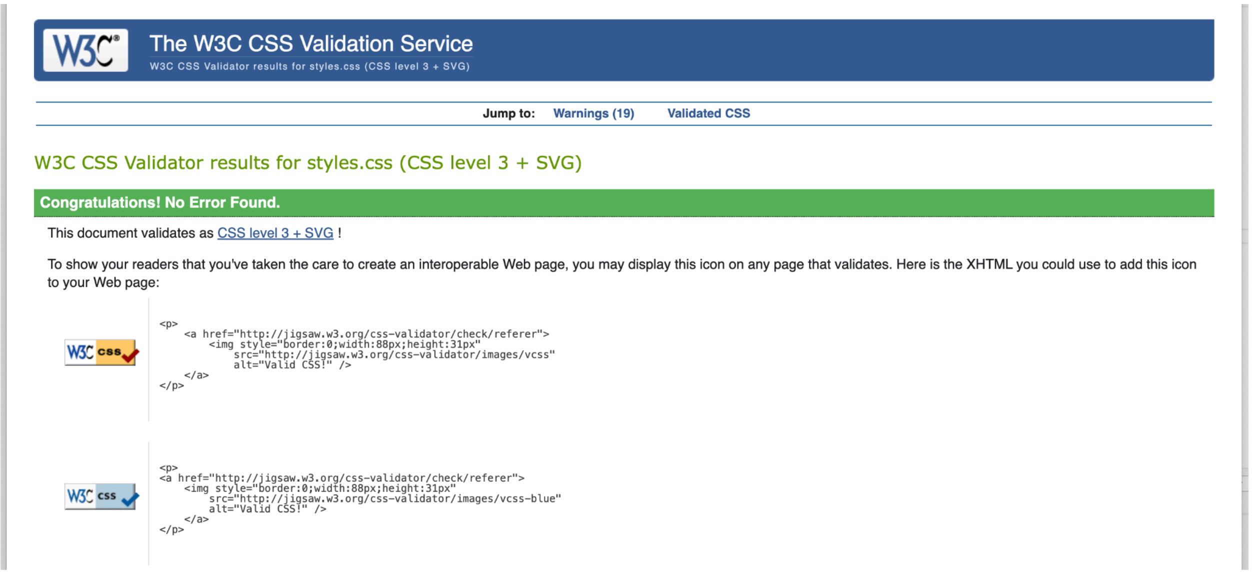
**Document checking completed. No errors or warnings to show.**

Used the HTML parser.

Total execution time 70 milliseconds.

About this checker • Report an issue • Version: 24.4.4

## CSS Validation Results



The W3C CSS Validation Service

W3C CSS Validator results for styles.css (CSS level 3 + SVG)

Jump to: Warnings (19) Validated CSS

W3C CSS Validator results for styles.css (CSS level 3 + SVG)

**Congratulations! No Error Found.**

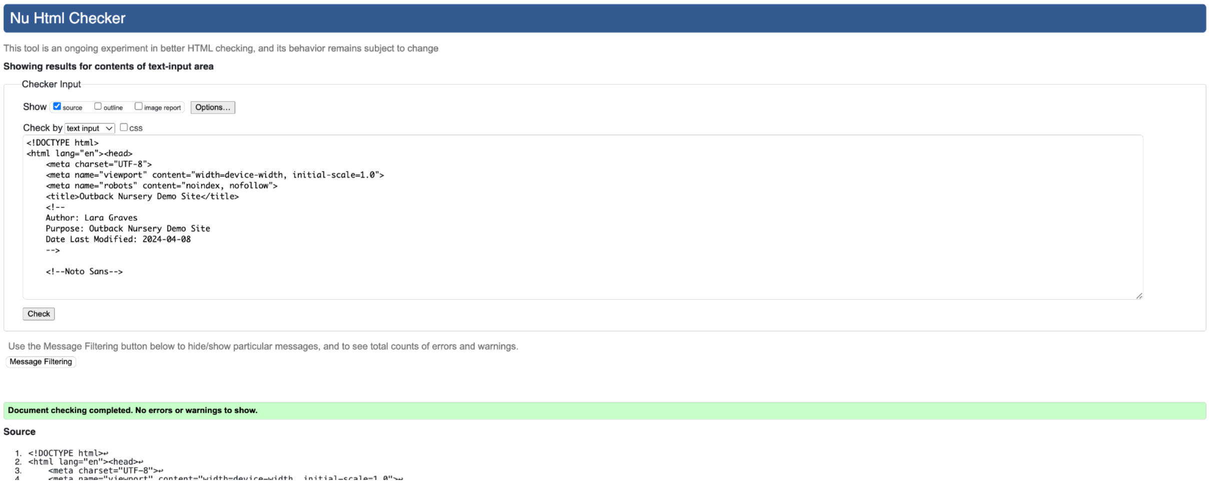
This document validates as CSS level 3 + SVG !

To show your readers that you've taken the care to create an interoperable Web page, you may display this icon on any page that validates. Here is the XHTML you could use to add this icon to your Web page:

```
<p>
    <a href="http://jigsaw.w3.org/css-validator/check/referer">
        <img style="border:0;width:88px;height:31px"
            src="http://jigsaw.w3.org/css-validator/images/vcss"
            alt="Valid CSS!" />
    </a>
</p>
```

```
<p>
<a href="http://jigsaw.w3.org/css-validator/check/referer">
    <img style="border:0;width:88px;height:31px"
        src="http://jigsaw.w3.org/css-validator/images/vcss-blue"
        alt="Valid CSS!" />
    </a>
</p>
```

**Dynamically Created HTML Code Validation Results**



This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change

**Showing results for contents of text-input area**

Checker Input

Show ☑ source □ outline □ image report

Check by text input □ css

```
<!DOCTYPE html>
<html lang="en"><head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta name="robots" content="noindex, nofollow">
    <title>Outback Nursery Demo Site</title>
    <!--
    Author: Lara Graves
    Purpose: Outback Nursery Demo Site
    Date Last Modified: 2024-04-08
    -->

    <!--Noto Sans-->
```

Use the Message Filtering button below to hide/show particular messages, and to see total counts of errors and warnings.

**Document checking completed. No errors or warnings to show.**

**Source**

```
<!DOCTYPE html>↵
<html lang="en"><head>↵
    <meta charset="UTF-8">↵
    <meta name="viewport" content="width=device-width, initial-scale=1.0">↵
```

# 05 Browser Testing

Tested in:

1. Google Chrome: Version Version 118.0.5993.117
2. Firefox: Version 16.6

# 06 Project Work Experience

It took me approximately 7-8 hours to complete the project. Initially, I spent some time planning and designing the features I wanted to implement, which included a banner text scrolling effect, an accordion menu for content organization, and a carousel with pagination for showcasing testimonials. The coding process itself took up the majority of the time, as I carefully wrote and debugged the JavaScript code to ensure smooth functionality and compatibility across different browsers.

From the project work, I learned several valuable lessons. Firstly, I deepened my understanding of JavaScript programming techniques, particularly in manipulating the Document Object Model (DOM) and handling events. This hands-on experience allowed me to grasp concepts such as closures and event propagation more effectively. Additionally, I honed my problem-solving skills by tackling various coding challenges encountered during the development process. By experimenting with different approaches and researching online resources, I was able to overcome obstacles and achieve the desired enhancements.

Throughout the project, I found several resources to be immensely helpful. Online tutorials and documentation on JavaScript provided valuable insights into language syntax and best practices. Furthermore, I frequently referred to sample code snippets and documentation related to DOM manipulation and animation effects. The following resources served as invaluable references and guided me through the implementation of the three features:

- https://blog.hubspot.com/website/scrolling-text-css
- https://dev.to/cwrcode/create-testimonial-slider-using-html-css-and-javascript-26gg
- https://www.w3schools.com/howto/howto_js_accordion.asp
- https://www.cssscript.com/responsive-tabs-accordion/

Despite significant progress, I encountered several challenges throughout the project. Mastering and implementing advanced JavaScript concepts, such as closures and event delegation, posed initial hurdles. Additionally, debugging errors and ensuring cross-browser compatibility demanded ongoing attention to detail.

Integrating JavaScript seamlessly with existing HTML and CSS was another noteworthy challenge, requiring troubleshooting and code refactoring. For instance, addressing the issue of a large gap on the left-hand side after the scrolling banner text was initially challenging. To rectify this, I created an additional paragraph of identical text to ensure a continuous scrolling effect. I also adjusted the duration of the transition and fine-tuned the scrolling keyframe animation for the translation values to achieve a seamless appearance, giving the impression of a single continuous loop across the banner.

The accordion section presented its own set of obstacles, particularly in achieving different layouts across mobile and desktop views. However, leveraging DOM operations enabled me to precisely target the correct panel to open upon button click, ultimately resolving the issue.

In conclusion, my project experience provided me with valuable insights into JavaScript programming and web development. By persistently seeking solutions, leveraging online resources, and applying problem-solving strategies, I was able to overcome obstacles and successfully implement the desired features. Moving forward, I aim to continue refining my JavaScript skills and exploring new techniques to enhance my proficiency in web development.