# Module 2 Assignment : Exploring JavaScript Topics with EJS, Node.js, and Express

Devin Fries

Department of Information Technology, Arizona State University

IFT 458: Middleware Programming & Database Security

Dr. Dinesh Sthapit

September 1, 2023

1. .**ejs Code Analysis (All Screenshots are on the last page)**: This part of the result.ejs code is written in JavaScript “<p>Sum: <%= sum %></p>

<p>Difference: <%= difference %></p>

<p>Product: <%= product %></p>

<p>Quotient: <%= quotient %></p>

“. **Control Structures**:

The provided code doesn't include explicit control structures like loops or conditionals within the EJS template. It mainly focuses on rendering dynamic content based on predefined variables (sum, difference, product, and quotient) without any conditional logic or loops.

**Variable Access and Display**:

Variables are accessed and displayed within the HTML content using the <%= variableName %> syntax. In this case, the EJS template is set up to display the values of sum, difference, product, and quotient within the HTML document. For example:

“<p>Sum: <%= sum %></p>”.

**EJS-Specific Syntax**:

The main EJS-specific syntax used in this file is the <% and %> tags to denote JavaScript code blocks and the <%= variableName %> syntax for variable interpolation.

Also, the action attribute of the <form> element (action="/calculate") indicates that when the form is submitted, it will be processed by the server-side code that corresponds to the route "/calculate." This route likely contains JavaScript code that handles the form submission and calculates the values for sum, difference, product, and quotient before rendering this EJS template.

In summary, this code demonstrates the use of EJS to embed JavaScript code within an HTML template to dynamically generate content. It primarily focuses on rendering dynamic values within the HTML document based on predefined variables, without the use of complex control structures like loops or conditionals within the template itself.

2. **HTML Code Analysis (Screenshot on last page)**: In the HTML code of result.ejs, the following parts were dynamically generated by the file:

The values of <%= sum %>, <%= difference %>, <%= product %>, and <%= quotient %> were dynamically inserted into the HTML code. In the original .ejs file, these placeholders were used to display the results of calculations, but in the rendered HTML, they have been replaced with actual values (e.g., "15," "5," "50," and "3").

Now, let's compare the rendered HTML code to the corresponding JavaScript code in the result.ejs file:

The JavaScript code in the result.ejs file doesn't appear in the rendered HTML. Instead, the JavaScript code is used to calculate values (sum, difference, product, and quotient) and assign them to variables. These variables are then inserted into the HTML using the <%= variableName %> syntax. The original .ejs code contains placeholders like <%= sum %>, but these placeholders have been replaced with actual values in the rendered HTML.

In conclusion, the result.ejs file contains JavaScript code to calculate values, but the rendered HTML displays the results of those calculations, with the dynamic content inserted where the placeholders were originally defined in the .ejs file. The rendered HTML does not contain the JavaScript code itself; it only displays the final output of that code.

3. **Documentation**:

**Introduction**:

In web development, the use of templating engines like EJS (Embedded JavaScript) can greatly enhance the flexibility and efficiency of creating dynamic web pages. This report presents an analysis of a sample .ejs file, which is a template containing JavaScript code embedded within HTML, and compares it to the HTML code generated in the browser. It also discusses the benefits and advantages of using .ejs templates over writing static HTML code, along with scenarios where .ejs templates can be beneficial.

**Comparing result.ejs Code with Generated HTML**:

The result.ejs file combines HTML and JavaScript to create dynamic content. It uses special delimiters, <% and %>, to enclose JavaScript code blocks. Variables are displayed within the HTML using <%= variableName %> syntax. When I view the rendered HTML in the browser, I noticed the following key observations:

**Dynamic Content Generation**: The most significant difference is that the .ejs code contains placeholders like <%= sum %>, whereas the generated HTML displays actual values. The .ejs file calculates these values and inserts them into the HTML dynamically. This dynamic content generation is a fundamental feature of .ejs templates.

**Control Structures**: The .ejs code does not include explicit control structures (e.g., loops or conditionals) within the template itself. Instead, it focuses on rendering dynamic content based on predefined variables. Any control structures or data processing logic likely reside in server-side code that provides data to the template.

**Benefits of Using .ejs Templates**:

Using .ejs templates offers several benefits in web development:

Dynamic Content: .ejs templates enable developers to generate dynamic content easily. They allow the integration of JavaScript code within HTML templates, making it possible to display real-time data, perform calculations, and respond to user interactions on the client side.

Code Reusability: Templates promote code reusability. Common elements like headers, footers, and navigation menus can be included in multiple pages without duplicating code. Changes made to a template propagate to all pages that use it.

Separation of Concerns: .ejs templates encourage a separation of concerns between HTML markup and JavaScript logic. This separation makes the codebase more maintainable and easier to collaborate on, with frontend and backend developers working on their respective areas.

Efficiency: Developers can work more efficiently with .ejs templates, as they don't need to manually generate HTML for each possible scenario. Instead, they can focus on the logic and data that power the dynamic content.

**Scenarios for Using .ejs Templates**:

EJS templates are beneficial in various web development scenarios:

Data-Driven Websites: Websites that display real-time data, such as social media feeds, stock market updates, or weather information, can use .ejs templates to continuously update content without refreshing the entire page.

User Dashboards: User dashboards often contain personalized content and widgets. .ejs templates make it easy to render and update these elements based on user preferences and actions.

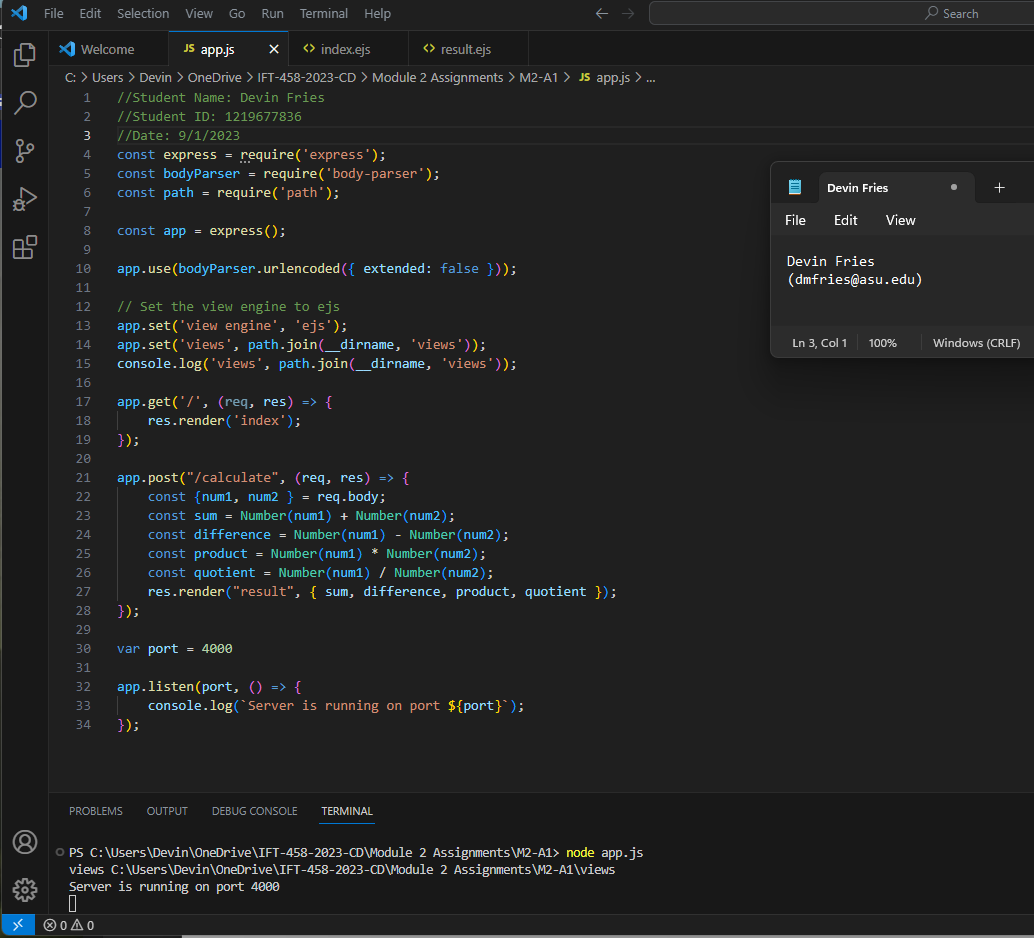
E-commerce Sites: Product listings, shopping carts, and checkout pages benefit from dynamic content rendering. .ejs templates allow for smooth interactions and immediate feedback to users.

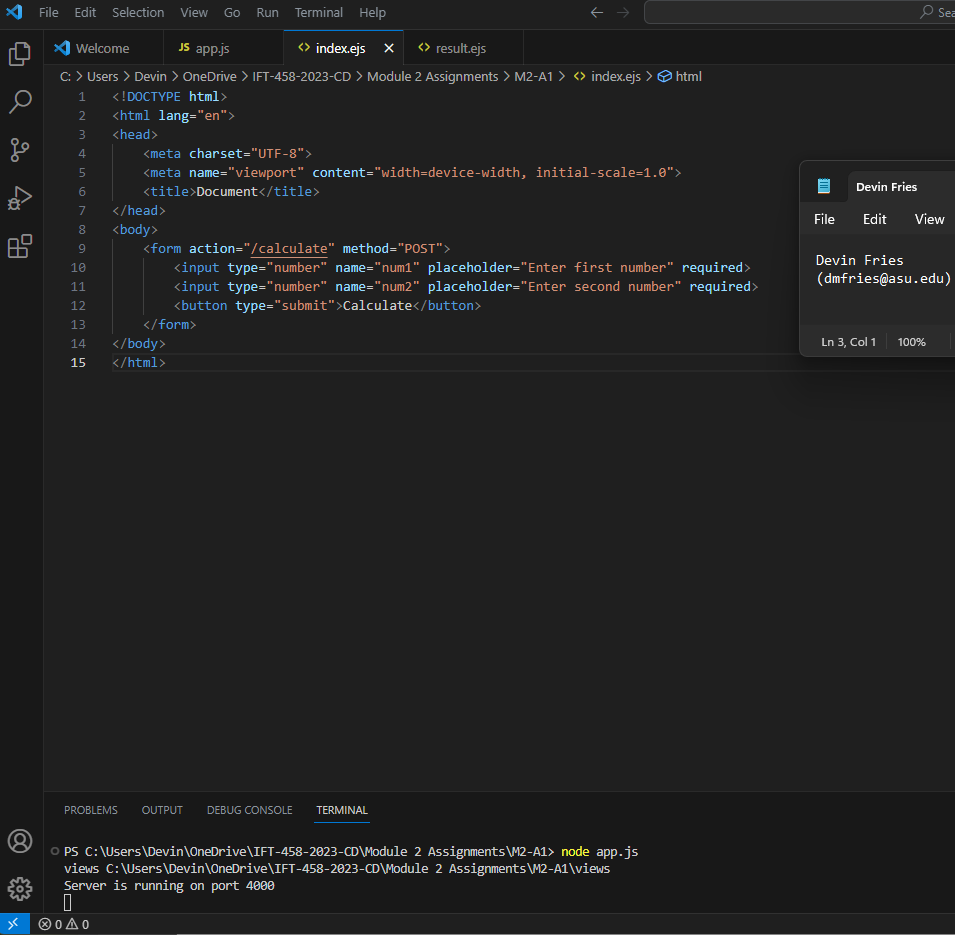
Content Management Systems (CMS): CMS platforms use templates to standardize page layouts and allow users to update content. .ejs templates can make it straightforward to incorporate new content while maintaining a consistent design.

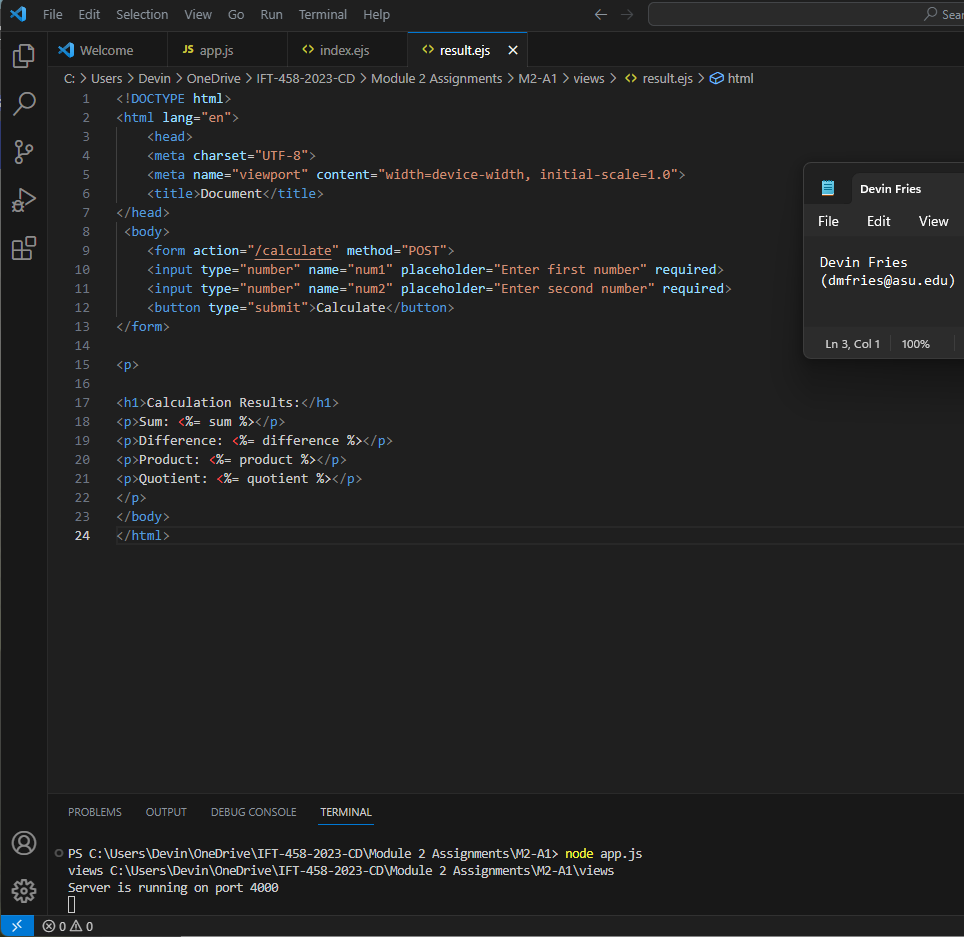
Conclusion:

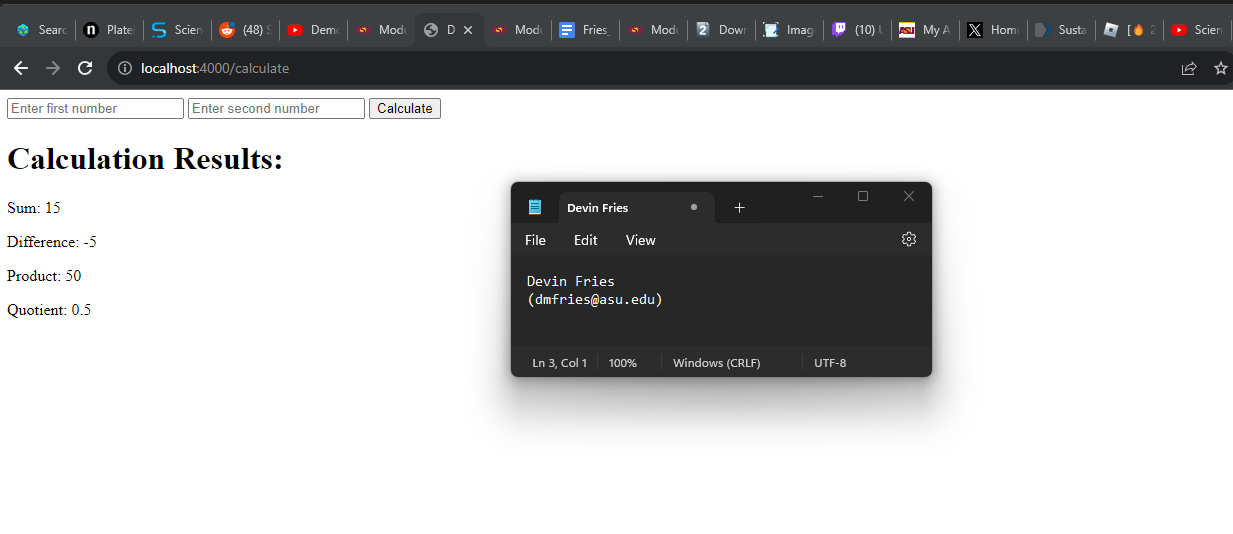
In summary, .ejs templates provide a powerful mechanism for generating dynamic web content. They combine HTML and JavaScript, allowing developers to create interactive web applications efficiently. Key benefits include code reusability, separation of concerns, and efficient handling of dynamic content. .ejs templates are particularly useful in scenarios where real-time data updates, user interactivity, and code efficiency are essential components of web development.

4. **Screenshots**:

**App.js: **

**Index.ejs**: **(next page)**

**result.ejs: (next page)**

**Localhost:4000 website calculation**: **(next page)**

**References**

EJS. (n.d.). Embedded JavaScript templating. EJS. Retrieved from <https://ejs.co/#features>