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

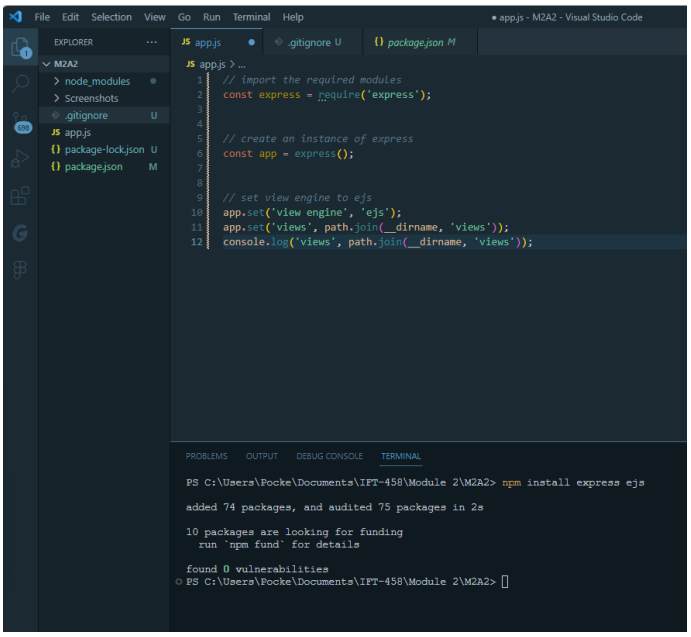
Justin Dyer

Ira A. Fulton Schools of Engineering, Arizona State University

IFT 458 Middleware Prog & Database Sec (2023 Fall)

Professor Dinesh Sthapit

September 3rd, 2023



The first image shows the new app.js JavaScript file in the M2A2 folder along with the node_modules folder produced from installing express, ejs, and body-parser. I also wrote the code that imports express and creates an instance of express named app. I then use the instance to set the view engine to ejs.

Next the code from the app.js file of module 1 assignment 2 was pasted into the app.js file. The GET route was updated to render the index page and the POST route was modified to the calculate page and the result is rendered to the result page. We changed the port to a variable named port with an assigned value of 4000 and told the app instance to listen to the port number assigned to var port.

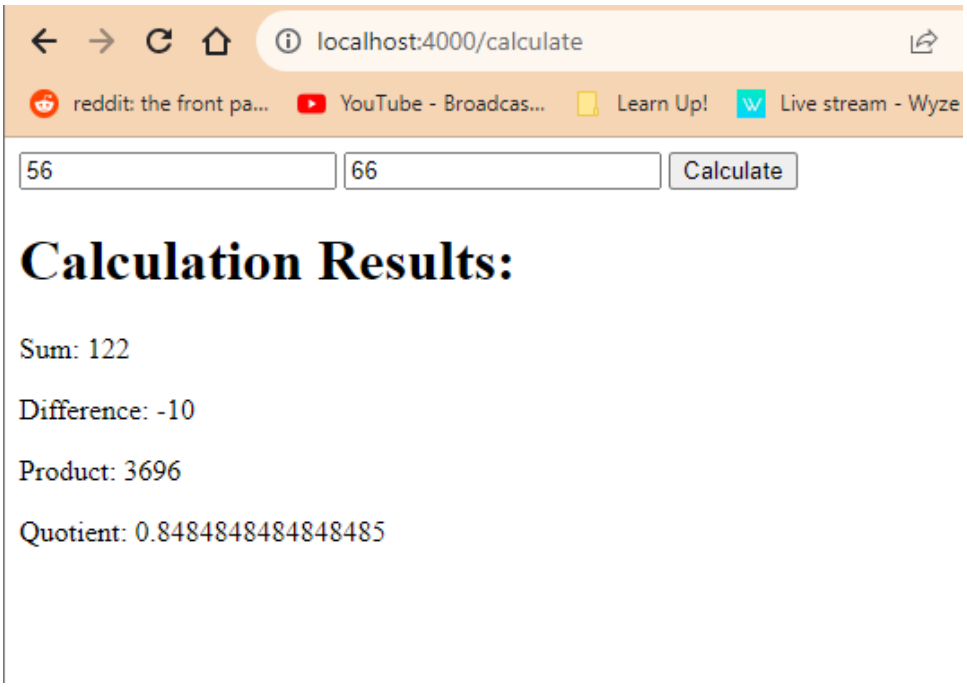
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <form action="/calculate" method="POST">
        <input type="number" name="num1" placeholder="Enter first number" required>
        <input type="number" name="num2" placeholder="Enter second number" required>
        <button type="submit">Calculate</button>
    </form>
</body>
</html>
```

Here we have created the views folder and added an embedded JavaScript (EJS) file named ‘index.ejs’. This is the index page that will display to the user allowing them to input numbers.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <form action="/calculate" method="POST">
        <input type="number" name="num1" placeholder="Enter first number" required>
        <input type="number" name="num2" placeholder="Enter second number" required>
        <button type="submit">Calculate</button>
    </form>

    <p>
        <h1>Calculation Results:</h1>
        <p>Sum: <%= sum %></p>
        <p>Difference: <%= difference %></p>
        <p>Product: <%= product %></p>
        <p>Quotient: <%= quotient %></p>
    </p>
</body>
</html>
```

Then we create another EJS file named ‘results.esj’ that will display the results to the user after they submit calculations. This uses <$= $> tags to use the Javascript variables from our ‘app.js’ file to update the webpage dynamically after the user submits.



The resulting page after entering 56 and 66 into the form and submitting. The numbers were passed into the arrow function when we POST, the req.body contains values num1 and num2 which are calculated into const sum, difference, product, and quotient. We then render the page with the EJS template from ‘result.ejs’ and this uses the const values we passed in.

```
    <p>
        <h1>Calculation Results:</h1>
        <p>Sum: <%= sum %></p>
        <p>Difference: <%= difference %></p>
        <p>Product: <%= product %></p>
        <p>Quotient: <%= quotient %></p>
    </p>
</body>
```

This shows the EJS code within the ’results.esj’ file in visual studio code. Note the syntax

<$= ‘variable name here’$>

```
<p>
    <h1>Calculation Results:</h1>
    <p>Sum: 122</p>
    <p>Difference: -10</p>
    <p>Product: 3696</p>
    <p>Quotient: 0.8484848484848485</p>
</p>
```

This is the same section in the chrome inspector view. We see that the formatting characters are gone and instead, we have the values of the const variables displayed as HTML. This shows the advantages of using EJS instead of static HTML. With EJS we can create templates for certain cases and display the necessary information dynamically using user input.

app.js

```
// Student Name: Justin Dyer

// Student ID: 1216117409

// Date: 09/02/23



// Import the required modules

const express = require('express');

const bodyParser = require('body-parser');

const path = require('path');




// Create an instance of express
```

```
const app = express();


// We use the 'body-parser' middleware to parse the incoming request bodies
app.use(bodyParser.urlencoded({ extended: false }));


// Set the view engine to ejs
app.set('view engine', 'ejs');
app.set('views', path.join(__dirname, 'views'));
console.log('views', path.join(__dirname, 'views'));


// create a route for the home page
// The GET route for the form
app.get('/', (req, res) => {
    // Render the form and pass in the current student data
    res.render('index.ejs');
});


// create a route for user to enter the numbers
app.post("/calculate", (req, res) => {
    const { num1, num2 } = req.body;
    const sum = Number(num1) + Number(num2);
    const difference = Number(num1) - Number(num2);
```

```
    const product = Number(num1) * Number(num2);

    const quotient = Number(num1) / Number(num2);

    res.render("result", { sum, difference, product, quotient });

})

var port = 4000



// Start the server on port 3000

app.listen(port, () => {

    console.log(`Server is running on port ${port}`);

})
```

index.ejs

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

    <form action="/calculate" method="POST">
```

```
        <input type="number" name="num1" placeholder="Enter first number"
required>
        <input type="number" name="num2" placeholder="Enter second number"
required>
        <button type="submit">Calculate</button>
    </form>
</body>
</html>
```

result.ejs

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <form action="/calculate" method="POST">
        <input type="number" name="num1" placeholder="Enter first number"
required>
```

```
        <input type="number" name="num2" placeholder="Enter second number"
required>
        <button type="submit">Calculate</button>
    </form>



    <p>
        <h1>Calculation Results:</h1>
        <p>Sum: <%= sum %></p>
        <p>Difference: <%= difference %></p>
        <p>Product: <%= product %></p>
        <p>Quotient: <%= quotient %></p>
    </p>
</body>
</html>
```