## IFT458&544: Middleware Prog & Database Sec (2023 Fall)

Sai Kishore Reddy Konatham

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

Prof. Dinesh Sthapit

September 3, 2023

#### 

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

**Setup:**

A screenshot of a computer

Description automatically generated

Exercise 1: Variables:

Code:

app.js

const express = require('express');

const app = express();

const port = 4000;// Port number made as 4000 in place of the default 3000

app.set('view engine', 'ejs');

app.use(express.urlencoded({ extended: true }));

app.use(express.static('public'));

app.get('/', (req, res) => {

  res.render('index');

});

app.post('/calculate', (req, res) => {

  try {

    const num1 = parseFloat(req.body.num1);

    const num2 = parseFloat(req.body.num2);

    const addition = num1 + num2;

    const subtraction = num1 - num2;

    const multiplication = num1 \* num2;

    const division = num2 !== 0 ? num1 / num2 : 'Cannot divide by zero';

    res.render('result', {

      num1,

      num2,

      addition,

      subtraction,

      multiplication,

      division,

    });

  } catch (error) {

    res.status(500).send('An error occurred.');

  }

});

app.listen(port, () => {

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

});

**index.ejs**

<!DOCTYPE html>

<html>

<head>

  <title>Calculation Results</title>

</head>

<body>

  <h1>Calculation Results</h1>

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

    <label for="num1">Number 1:</label>

    <input type="number" id="num1" name="num1" required><br>

    <label for="num2">Number 2:</label>

    <input type="number" id="num2" name="num2" required><br>

    <button type="submit">Calculate</button>

  </form>

</body>

</html>

**result.ejs :**

<!DOCTYPE html>

<html>

<head>

  <title>Calculation Result</title>

</head>

<body>

  <h1>Calculation Result</h1>

  <p>Number 1: <%= num1 %></p>

  <p>Number 2: <%= num2 %></p>

  <p>Addition: <%= addition %></p>

  <p>Subtraction: <%= subtraction %></p>

  <p>Multiplication: <%= multiplication %></p>

  <p>Division: <%= division %></p>

  <a href="/">Back</a>

</body>

</html>

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Embedded JavaScript Code:**

In the .ejs file, JavaScript code is seamlessly integrated using the <% %> tags. These tags facilitate the execution of JavaScript logic within the template.

**Control Structures:**

This .ejs file employs a straightforward approach without the use of loops or conditionals. It primarily focuses on visualizing input numbers and displaying calculation results.

**Variable Display and Access:**

To retrieve and display variables within the HTML content, the syntax <%= variableName %> is utilized. For instance, it showcases operations like addition, subtraction, multiplication, and division involving variables such as num1, num2, and others.

**Unique EJS Features:**

The main EJS feature utilized here is variable interpolation through <%= %> tags. EJS enables the creation of dynamic content by synchronizing JavaScript variables with HTML templates.

**Dynamic HTML:**

The static structure of the webpage, including headings, labels, and form elements, is defined in the HTML file. When the form action is set to "/calculate," it signifies that the server will handle the input.

**Adaptable Content:**

At certain points, the HTML incorporates dynamic content generated by the .ejs file. Dynamic content is seamlessly integrated into the HTML by replacing placeholders with real values.

By replacing placeholders with actual data, the HTML obtains dynamic content from the .ejs file. This approach helps in separating the presentation (HTML) from the logic (calculations), making the code more maintainable.

EJS templates offer a flexible and efficient solution for creating dynamic HTML content. They streamline the integration of logic and presentation, simplifying web development.

**Code Explanation:**

This JavaScript code configures an Express.js web server to utilize the EJS templating engine. It establishes routes for handling both GET and POST requests, performs computations based on user input, and generates dynamic HTML templates. It serves as a clear example of how Express.js and EJS can be used to create a web application for conducting computations and displaying results.

**Browser Analysis:**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

The .ejs code defines the structure of the HTML page and includes placeholders for dynamic content. When the server processes a request and utilizes this .ejs template with specific data, it generates an HTML page with the same structure but filled with real information. This approach enhances code maintainability and enables the creation of dynamic content within web applications.