A yellow and pink seal with a black background

Description automatically generated

Name: Rishabh Singh

Asu id: 1224933294

Assignment: Module 1 Assignment: Web Basics

Date:8/26/2023

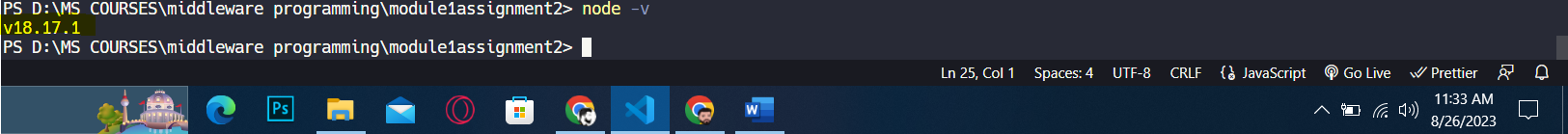
* **A summary of your understanding of Node.js**

I've had the opportunity to work with Node.js to create a web application that utilizes the Express framework and the EJS template engine. Through this experience, I've successfully built a form that collects student information from users. One crucial aspect of my work was setting up an Express application, defining routes to handle different parts of the application, and incorporating middleware like body-parser to manage form data effectively.

In my coding journey, I've gained familiarity with EJS templates, allowing me to dynamically render content on web pages. This involves embedding JavaScript within HTML, enabling me to display data seamlessly. I found this approach quite effective in creating interactive user interfaces.

Furthermore, I've developed the skill to manage form submissions. This includes processing user inputs on the server side, extracting relevant data from form fields, and incorporating this information into the application's logic. I found it intriguing how I could display the submitted data on the same page, demonstrating my ability to create dynamic views that adapt to user actions.

* **A screenshot showing the successful installation of Node.js on your local machine.**



* **A brief explanation of the additional functionalities Node.js offers over standard browser-side JavaScript.**

Node.js goes beyond regular browser-side JavaScript by extending JavaScript's capabilities to the server side. It provides the ability to build backend applications, handle network requests, and manage databases using JavaScript. This enables the creation of dynamic web servers, real-time applications, and APIs. Unlike browser-based JavaScript, Node.js operates outside the browser environment, utilizing the V8 engine for fast execution. It offers features like non-blocking I/O, event-driven architecture, and access to the file system. This makes Node.js an efficient and versatile choice for building server-side applications.

* **The Node.js script you created, along with a description of what it does and a screenshot of the script's output.**

A screenshot of a computer program

Description automatically generated

* Objective: This code sets up a basic Express web server that manages student information.
* Dependencies: It requires express, body-parser, and path modules for server and form handling.
* Server Setup: An Express app is created and configured.
* Form Handling: It uses body-parser middleware to parse form data.
* Template Engine: The view engine is set to ejs, allowing dynamic HTML rendering.
* View Location: EJS views are stored in a "views" directory.
* Student Data: An empty array called students holds student information.
* GET Route: When a user accesses the root ("/") route, it renders the "index" view, passing the students data.
* POST Route: When a form is submitted, student data is added to the students array, and the page is redirected to root.
* Server Start: The server listens on port 3000, displaying a console message on startup.
* Overall, this code establishes a web app that accepts and displays student information using a form and Express framework.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

* Form Setup: This HTML code creates a form to collect student information.
* Form Fields: It includes input fields for first name, middle name, last name, address, phone, email, and a description.
* Submit Button: A "Submit" button triggers the form submission.
* Server Interaction: The form action is set to "/" (root), using the POST method to send data to the server.
* Data Display: Below the form, a list displays student data using an EJS loop.
* Dynamic Rendering: EJS tags like <%= student.fname %> dynamically inject data into the HTML.
* User Interaction: An "Add New Student" link lets users return to the form.
* Overall Purpose: This HTML code creates an interactive web page to input, display, and manage student information using a form and dynamic rendering.

**OUTPUT:**

A screenshot of a computer

Description automatically generated

* **Your reflection on the experience of setting up and working with Node.js.**

Working with Node.js has been an enlightening experience for me. It's impressive how Node.js extends JavaScript beyond the browser and empowers me to build powerful server-side applications. The ability to utilize the same language on both the client and server sides brings a sense of cohesion to my development process. Setting up an Express web server felt intuitive, and the integration of middleware like body-parser streamlined form handling. Incorporating the EJS template engine allowed me to craft dynamic views effortlessly. What's particularly intriguing is the event-driven and non-blocking architecture that enhances the application's efficiency. This experience has deepened my understanding of full-stack development and reinforced the versatility of JavaScript in driving both front-end and back-end functionalities.