**Portfolio (Week-4)**

This week’s lab session centred on creating a web application using **Express**, a popular Node.js framework for building web applications. Below is the detailed explanation of the tasks performed and the lessons learned.

1. **Setting Up the Project**

To begin, I created a new directory for the Week 4 task. Inside this directory, I initialized a Node.js project by running the following command in the terminal:

This created a package.json file, which is essential for managing project dependencies and scripts.

Next, I installed Express using the command:

This ensured that the Express framework was available for use in the project.

 **Creating a Simple Express Application**

To test the setup, I created a simple Express application that outputs a "Hello" message. Below are the steps:

* Created an index.js file in the project directory.
* Wrote the code to create a basic Express server that responds with a simple message when accessed.

**Output**

After running the application using the command node index.js, I accessed it through a web browser. The output displayed a "Hello" message as expected.

**Exercise 1: Building a Simple Web Application**

In this exercise, I extended the functionality of the Express app to handle different routes and respond to HTTP requests. I added code to:

* Respond to GET requests on various endpoints.
* Send appropriate messages based on the route accessed.

A screen shot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

**Exercise 2: Handling Dynamic Data**

This exercise introduced the concept of handling dynamic data in Express applications. I used route parameters and query strings to dynamically respond to user input. For example, the application could greet users by their names based on the URL path.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Exercise 3: Working with JSON Files**

In this exercise, I learned how to work with JSON files in an Express application. Below are the steps I followed:

1. **Creating a student.json File** I created a student.json file to store student data, including attributes like name, age, and grade.

A screenshot of a computer

Description automatically generated

1. **Reading the JSON File in index.js** I wrote code in index.js to read and parse the JSON file. This allowed the application to retrieve and display the stored data.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

1. **Adding Search Functionality** To make the application more interactive, I added code to allow searching for specific students based on their attributes. For instance, users could search for students by name or grade using the API endpoints.

A screen shot of a computer program

Description automatically generated

**Output**

After running the application, I tested the API endpoints by sending requests through a web browser or tools like Postman. The application successfully retrieved and displayed the requested data.

A screenshot of a computer

Description automatically generatedA screen shot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screen shot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Learning Objectives**

* To understand the basics of building web applications using Express.
* To learn how to handle routes and dynamic data in an Express app.
* To practice working with JSON files for storing and retrieving data.
* To explore the process of creating RESTful APIs.

**Key Points**

* Express simplifies the process of creating web applications with its intuitive and minimalistic design.
* JSON is a lightweight and versatile format for storing and exchanging data.
* Organizing project files and dependencies is crucial for maintaining clarity and efficiency.

**Problems Faced**

1. **Issue Installing Express:** Initially, I encountered an error while trying to install Express using npm install.
   * **Solution:** I checked my internet connection and cleared the npm cache using the command npm cache clean --force. Re-running the installation command resolved the issue.
2. **Error Reading JSON File:** While trying to read the student.json file, I faced an error due to incorrect file path references.
   * **Solution:** I reviewed the code and corrected the file path to ensure the application could locate the JSON file.
3. **Route Handling Issues:** At first, the application did not respond correctly to certain routes due to overlapping route definitions.
   * **Solution:** I reorganized the route definitions and added specific conditions to handle each route separately.

By overcoming these challenges, I enhanced my understanding of Express and gained confidence in building web applications. This week’s lab provided valuable insights into server-side development and API creation.