**22AIE457 - Full Stack Development**

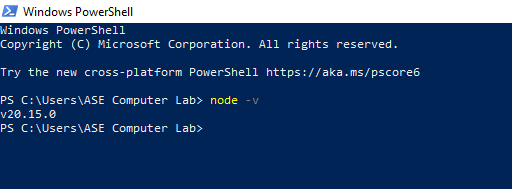
**Lab Report II – Node JS on Windows**

**Introduction**

This lab session aims to introduce the fundamentals of Node.js—a powerful JavaScript runtime built on Chrome’s V8 engine. It enables the development of fast and scalable network applications. The objective of this lab is to set up a Node.js environment on a Windows system and execute basic server-side scripts. Through this, students will gain hands-on experience with creating HTTP servers and using core Node modules.

1. **Initial Environment Setup**

Before writing any code, ensure that Node.js is installed on your system. This can be verified using the following command. These commands return the installed versions of Node.js and npm respectively, confirming the environment setup.

****

1. **Creating a Basic HTTP Server**

**Code:**

const { createServer } = require('node:http');

const hostname = '127.0.0.1';

const port = 3000;

const server = createServer((req, res) => {

res.statusCode = 200;

res.setHeader('Content-Type', 'text/plain');

res.end('Hello World');

});

server.listen(port, hostname, () => {

console.log(`Server running at http://${hostname}:${port}/`);

});

1. The createServer function is used to create an HTTP server.
2. The server responds with a plain text message "Hello World".
3. It listens on localhost (127.0.0.1) at port 3000.
4. When run, the server can be accessed via the browser or curl at <http://127.0.0.1:3000>.

**Running the Server Script**

1. Save the above code in a file named server.js.
2. Open the terminal in the same directory.
3. Run the command: node server.js. You should see the message about the server running
4. Open a browser and visit http://127.0.0.1:3000 to see the output: Hello World.

**Output:**

**A computer screen shot of a blue screen

AI-generated content may be incorrect.**

A screenshot of a computer

AI-generated content may be incorrect.

1. **Understanding JavaScript Output in Node**

**Code:**

const x= "My first web page"

console.log(x)

1. A basic script that declares a constant x and prints it using console.log.
2. When executed using node filename.js, it displays the string.
3. This illustrates how Node.js can be used not only for server-side operations but also for executing general-purpose JavaScript code on the backend.

**Output:**

A screenshot of a computer program

AI-generated content may be incorrect.

**Conclusion:**

The basics of Node.js setup and usage on Windows is explored here. We created a simple HTTP server and executed a basic JavaScript script. It demonstrated how Node.js handles backend tasks efficiently. This lays the foundation for more advanced full stack concepts.