

Task 6: Implementing Middleware-like Functionality

Objective:

Implement and use middleware-like functionality in a web application using plain JavaScript to log requests or modify responses. Ensure middleware-like functions log requests or modify responses as expected.

Prerequisites:

- Basic understanding of JavaScript.
- Node.js installation.

Concepts:

- **Middleware-like Functions:**
 - Middleware functions are functions that have access to the request object (`req`), the response object (`res`), and the next middleware function in the application's request-response cycle.
 - Middleware functions can perform a variety of tasks such as executing any code, making changes to the request and the response objects, ending the request-response cycle, and calling the next middleware function in the stack.

Example:

```
// server.js
const http = require('http');

const requestLogger = (req, res, next) => {
  console.log(`${req.method} ${req.url}`);
  next();
};

const handleRequest = (req, res) => {
  //your code here
};


const server = http.createServer((req, res) => {
  requestLogger(req, res, () => handleRequest(req, res));
});

server.listen(3000, () => {
  console.log('Server is running on port 3000');
});
```

Setup:



1. Install Node.js:

Ensure Node.js is installed on your machine. You can access the instructions [here](#):

- 

Detailed Instructions for Installing Node.js and NPM and integration...

For Windows: — Download Node.js Installer: — Visit the official Node.js website:...



w3o NFThing
 Last Edited 7/3/2024

Tasks:


1. Middleware-like Functionality:

- **Task:**
 - Create a file named `server.js` and write your own code that:
 - Creates a basic web server using the `http` module.
 - Implements middleware-like functionality to log requests.
 - Handles routes for `/`, `/about`, and `/contact`.
- **Outcome:**
 - Ensure middleware-like functions log requests or modify responses as expected when accessed via Postman or a browser.

Instructions:

- **Perform the following tasks:**
 - Write the required code in `server.js`.
 - Run the server using Node.js to ensure the code executes without errors and demonstrates the use of middleware-like functionality.

Resources:

- 

HTTP | Node.js v22.4.0 Documentation

This module, containing both a client and server, can be imported via `— require('node:http')` (Commo...

<https://nodejs.org/api/http.html>

GitHub Instructions:

- Open in Visual Studio Code:**
 - After clicking on the "Open in Visual Studio Code" button from the GitHub Classroom confirmation page, Visual Studio Code (VSCode) will open the repository directly.
 - If prompted, select "Open" or "Allow" to open the repository in VSCode.
- Complete the Task:**
 - Write your solution in `server.js`.
- Run and Test Your Code:**
 - Run your code to ensure it works correctly. Use the following command:

```
node server.js
```

4. Commit Your Changes:

- Commit your changes with a meaningful message:

```
git commit -m "Completed Middleware-like Functionality task"
```

5. Push Your Changes to Your Forked Repository:

- Push your changes to your forked repository:

```
git push origin main
```

6. Create a Pull Request:

- Go to your forked repository on GitHub.
- Click on the "Pull Requests" tab.
- Click the "New Pull Request" button.
- Ensure the base repository is the original template repository and the base branch is `main`.
- Ensure the head repository is your forked repository and the compare branch is `main`.
- Click "Create Pull Request".
- Add a title and description for your pull request and submit it.

Summary of Commands:

```
# Fork the repository on GitHub

# Clone the forked repository
git clone https://github.com/your-github-username/repository-name.git
cd repository-name

# Complete the task by writing and running the code in server.js

# Add, commit, and push your changes
git commit -m "Completed Middleware-like Functionality task"
git push origin main

# Create a pull request on GitHub
```