COMP 9783 Front-end Development

Lab-1-3 - Express.js Routing and Middleware Lab. (4% of the course mark)

Name:

Student Number:

In this lab, participants will delve into the world of web development with Express.js, a powerful and flexible web application framework for Node.js. Through hands-on exercises, learners will explore how to rapidly build robust and scalable web applications using Express.js. Topics covered include setting up a basic Express.js application, defining routes, handling HTTP requests and responses, and implementing middleware.

Lab objectives:

1. Understand the role and benefits of Express.js in web development.
2. Set up a basic Express.js application and configure middleware.
3. Define routes to handle different HTTP requests and responses.

Create an Express.js app:

1. On **VSCode**, **create** a **folder** named **Express-Routing-Middleware-App**.
2. Open the **terminal** and change the directory to **Express-Routing-Middleware-App**.
1. Initialize the app by performing the following tasks:
3. **Type npm init -y** and **press enter**.

Note: npm init -y is a quick shortcut to initialize the package.json file. It will initialize the app with default values.

4. Type **npm install express** and **press enter**.

5. Create a file named **index.js** and enter the following code:

```
const express = require("express");
const app = express();
const port = 3000;
const APP_NAME = "Express-Routing-Middleware-App";

app.get("/", (req, res) => {
  res.send("GET request");
});

app.post("/", (req, res) => {
  res.send("POST request");
});

app.put("/", (req, res) => {
  res.send("PUT request");
});

app.delete("/", (req, res) => {
  res.send("DELETE request");
});

app.listen(port, () => {
  console.log(`${APP_NAME} listening on port ${port}`);
});
```

6. Save the changes.
7. On the terminal enter the **node index.js** and **press enter**. Leave this app running, as it will be used for testing HTTP methods later on.

Testing with a Browser:

1. Open your browser and enter the url: **http://localhost:3000/**
2. Take a screenshot and name it **BrowserGet.png**.

Testing with Postman:

1. Navigate to: **https://www.postman.com/**
2. **Download** and **install** the **Postman** version based on your **Operating System**.
3. Upon successful installation, open **Postman**.
4. Create a new **HTTP request**, choose **GET** from the drop down list and enter url: **http://localhost:3000** and click on **SEND**.
5. The response would show at the bottom of the screen. Capture the screenshot and name it **PostmanGet.png**.
6. Modify the previous Postman request and choose **POST** from the drop down list and click on **SEND**.
7. The response would show at the bottom of the screen. Capture the screenshot and name it **PostmanPost.png**.
8. Modify the previous Postman request and choose **PUT** from the drop down list and click on **SEND**.
9. The response would show at the bottom of the screen. Capture the screenshot and name it **PostmanPut.png**.
10. Modify the previous Postman request and choose **DELETE** from the drop down list and click on **SEND**.
11. The response would show at the bottom of the screen. Capture the screenshot and name it **PostmanDELETE.png**.

Adding Middleware

1. Search for this code: **const APP_NAME = "Express-Routing-Middleware-App";** and below it, place this JS code:

```
app.use(simpleMiddleware);
```

2. At the bottom of the app, place this JS code:

```
function simpleMiddleware(req, res, next) {
  console.log(
    `[Type: Request] [Url: ${req.url}] [Method: ${req.method}] [User Agent: ${req.headers["user-agent"]}]`
  );
  next();
  console.log(
    `[Type: Response] [Status Code: ${res.statusCode}] [Status Message: ${res.statusMessage}]`
  );
}
```

3. Save the changes.
4. **Press ctrl-c** to terminate the app.
5. On the terminal enter the **node index.js** and **press enter**. Leave this app running, as it will be used for testing the middleware later on.

MIddleware Testing

1. Using Postman Create a new **HTTP request**, choose **GET** from the drop down list and enter url: **http://localhost:3000** and click on **SEND**.
2. Modify the previous Postman request and choose **POST** from the drop down list and click on **SEND**.

3. Modify the previous Postman request and choose **PUT** from the drop down list and click on **SEND**.
4. Modify the previous Postman request and choose **DELETE** from the drop down list and click on **SEND**.
12. On the terminal where the node app is running, there should be console.log output of the previous Postman requests. Capture the screenshot and name it **MiddlewareEvents.png**.

Submission:

1. Use the html template: **index.html** and **write HTML codes for each screenshot**:
    a. Write a title and short description.
    b. Display the screenshot.
2. Create a **zip file** of all the **HTML**, **CSS** and **PNG** files.
3. Submit the **zip file** to **GBC - D2L**.