COMP 9783 Front-end Development

Lab-1-2 - Express.js 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.

Lab objectives:

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

Create an Express.js app:

1. On **VSCode**, **create** a **folder** named **Express-App**.
2. Open the **terminal** and change the directory to **Express-App**.
3. Initialize the app by performing the following tasks:
    a. **Type npm init** and **press enter**.
    b. **Press enter** to accept the default package name.
    c. **Press enter** to accept the default version.
    d. Enter a **simple description** and **press enter** to set the description.
    e. **Press enter** to accept the default entry point: (index.js).
    f. **Press enter** to accept the default test command.
    g. **Press enter** to accept the default git repository.

h. **Press enter** to accept the default keywords.

i. Enter your **first** and **last name** and **press enter** to set the author.

j. **Press enter** to accept the default license: (ISC).

k. Enter **yes** and **press enter** to accept the previously entered 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-App";

app.get("/", (req, res) => {
  res.send("Hello World!");
});

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**.
8. Open your browser and enter the url: **http://localhost:3000/**
9. Take a screenshot and name it **HelloWorld.png**.
10. **Press ctrl-c** to terminate the app.

11. Make changes to the file **index.js** and write the following JS codes:
    a. Create a new route by copying the code below:

```
app.get("/json", (req, res) => {
  const userObject = { firstName: "Elon", lastName: "Musk" };
  res.send(userObject);
});
```

12. Save the changes.
13. On the terminal enter the **node index.js** and **press enter**.
14. Open your browser and enter the url: **http://localhost:3000/json**
15. Take a screenshot and name it **NewRouteJSON.png**.
16. **Press ctrl-c** to terminate the app.
17. Make changes to the file **index.js** and write the following JS codes:
    a. Create a new route by copying the code below:

```
app.get("/html", (req, res) => {
  const htmlObject = `<!doctypehtml><meta charset=utf-8><meta
content="width=device-width,initial-scale=1"name=viewport><title>Simple
HTML</title><h1>Simple HTML</h1>`;
  res.send(htmlObject);
});
```

18. Save the changes.
19. On the terminal enter the **node index.js** and **press enter**.
20. Open your browser and enter the url: http://localhost:3000/html
21. Take a screenshot and name it **NewRouteHTML.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**.