Node.JS

Practice – 6

Overview

*In this hands-on session you will continue learning **Node.js**. Upon completion of this hands-on session, you should be able to:*

- *Configure package.json for*
  - *Express web-framework dependencies*
  - *Nodemon for dev-dependencies*
  - *The script for starting npm app*
- *Create a simple Express Web Application*
- *Perform different routes*
  - *Use appropriate status codes*
  - *Handle all routes*
- *Realize the power of Express middleware*
  - *To serve static files*

Hands-on

1. Create a new directory by name '**Express_Demo**' and setup the following:

   - Initialize the **package.json** file.
   - Install **express** module locally.
   - Install '**nodemon**' module as **Development Dependency**
     Check your '**package**.**json**' file.
   - In the '**script**' section of the '**package**.**json**' file include the following instruction:

   ```
   "scripts" : {
      "start":  "nodemon app.js"
   }
   ```

   Save the **package**.**json** file

2. Your challenge now is to build the first Web Application using the Express Web Framework which does the following:

   - Listens at port 3000
   - Display '**Hello from Express**' in h1 element
   - Express tag line: ***Fast, unopinionated, minimalist web framework*** in h3 element.

NOTE: The source code should be written in **'app.js'** file.

Open the terminal within VS Code and execute the command '**npm start**'

This will invoke the script with the **'start'** key, which will invoke **'nodemon'** and your express app should be started.

Open the browser and enter the URL with port 3000 and observe the output.

3. Your next challenge now is to design Express Web App which will have multiple routes as detailed below:

   [a] The **'/' route** should display
   Welcome from Express! Using h1 element
   Fast, unopinionated, minimalist web framework
   Using h3 element

   [b] The **'/api'** route should display about the APIs as follows:
   APIs Using h1 element

   And using the <p> element the following text:

   With a myriad of HTTP utility methods and middleware at your disposal, creating a robust API is quick and easy.

   [c] The **'/performance'** route should display about the Performance of Express as:

   Performance Using h1 element

   And using the <p> element the following text:

   Express provides a thin layer of fundamental web application features, without obscuring Node.js features that you know and love.

   [d] The **'/frameworks'** routes should display the popular web frameworks build using Express as an un-ordered list.

   You can refer to the following **URL** for info:
   http://expressjs.com/en/resources/frameworks.html

   Ensure that the app return appropriate **Status Codes.**

   Ensure that if the URL does not match to the above given URLs appropriate message with the right Status Code should be returned and displayed.

   Check your web for all the routes.

4. Your next challenge is use the already existing JS application

   We had built a complete '**future_value**' application in JavaScript.
   Hope you have the source code.

   NOTE: If the source code for the **'future_value'** JS application is not available with you, ask the trainer.

   Copy the **'future_value'** JS application in a folder by its name.

   In the Express Web Application's '**app.js**' file, use the **res.sendFile()** and for **the '/'** URL redirect the '**index.html**' available in the '**future_value**' application.

   NOTE: You can use the 'path' module if you desire to resolve the path name for the 'index.html' available in 'future_value' folder/directory.

   Check the Express Web Application in the web-browser.

   Record your observation
   [a] Is the style applied
   [b] Is it functional

5. The application which you have executed above would have not been functional and properly formatted although the required JS file and CSS file are available in the folder/directory.

   You next challenge is to use the Express web framework's '**express.static()**' middleware to server static files.

   Create a folder/directory by name **'public'** under your project root directory and copy all the files from '**future_value**' directory to the **'public'** directory.

   Run your Express Web Application and check the following:
   - Styling
   - Functionality