軟體人才培訓學院 / Back-End Class

Remote Learning Assignment - Week 3

Assignment 1: Your First Web Server

To build your first web server for development, follow the steps below:

1. Install java 17
2. Create a Spring Boot project (via IDE/ Spring initializr)
    a. with **Maven** as dependency management
    b. with **Java as** programming language
    c. with **3.2.1** or later as Spring boot version
    d. with **Jar** as packaging
    e. with **17** as Java version
3. Write a simple web server program and start it
4. Show an HTML page when you enter http://localhost:3000/ in a browser's address bar (For example a simple page including "Hello, My Server!" is an acceptable result.)

You may refer to getting started documents on Spring Boot official website to complete this assignment.

Reminders:

1. You have to learn how to use a command-line interface on your computer.
2. Set up your GitHub repository to **ignore unnecessary files**, Refer to Ignoring Files document.
3. All the assignments this week should continue with the same Spring boot project built in this assignment.
4. You don't need to split folders for each assignment this week, your folder structure could be like remote-assignments/Week3/Assignments

Assignment 2: Build Backend API for Front-End

Now, try to modify your code executed on the server-side to build a simple API. Your server should fulfill the following client requests:

1. When the user enters http://localhost:3000/data in a browser's address bar, show a "Lack of Parameter" message on the page.
2. When the user enters http://localhost:3000/data?number=xyz in a browser's address bar, it shows a "Wrong Parameter" message on the page. (xyz means any non-integer value)
3. When the user enters http://localhost:3000/data?number=5, they should get the result of 1+2+....+5 on the page.
4. Generally speaking, when the user enters http://localhost:3000/data?number=N, they can get the result of 1+2+....+N on the page. (N is any positive integer)

Note:

1. handle HTTP GET method and parameters with Java Spring Boot on the server-side.
2. (Optional) Think about what will happen when N is very large.

Assignment 3: Connect to Backend API by AJAX

You have built your first API in the backend, then let’s get back to the front-end. Follow the steps below to send an HTTP request to your backend API by AJAX.

1. Serve a static HTML file named sum.html. It means you can enter http://localhost:3000/sum.html in a browser’s address bar to get this HTML page.
2. Write some JavaScript code in sum.html to make an HTTP request by AJAX to http://localhost:3000/data?number=10, and get the result 55 from the server.
3. Write a simple user interface to let users enter a number and get results from the server. (For a simple example, a text input and a button.)

Hint: refer to W3Schools or MDN for learning more about AJAX.
Hint: using async function or promise is also an acceptable answer.

Assignment 4: HTTP Cookie

Cookies are an important mechanism for storing a small piece of data in the browser. Modify your code executed in the backend to use cookies for user tracking.

1. Serve a URL http://localhost:3000/myName by your server.
2. When the user connects to http://localhost:3000/myName, check cookies for the user's name in the backend.
    a. If you can get the user's name from cookies, show it on the web page. **Done.**
    b. If you cannot get the user's name from cookies, show an HTML form including a text input and a button on the web page. **Go to step 3.**
3. The user can enter his name in the text input, and then click the button to submit the form to a URL http://localhost:3000/trackName?name=使用者的輸入 which should be served from your server, too.
4. When the user submits the form to http://localhost:3000/trackName?name=使用者的輸入, you should get the user's name from the HTTP parameter and store it in the cookies.
5. Redirect user to http://localhost:3000/myName, user can see his name, finally. **Done.**

References:

1. Document for getting/checking cookies in the backend.
2. Document for setting cookies in the backend.

Assignment 5: Algorithm Practice (Advanced Optional)

Given an array of integers, return indices of the two numbers such that they add up to a specific target. You may assume that each input would have exactly one solution, and you may not use the same element twice.

you can either use Java or javascript to hand in this assignment.

```
function twoSum(nums, target) {
 // your code here
}

/*
 For example:
 twoSum([2, 7, 11, 15], 9);
 Should returns:
 [0, 1]
 Because:
 nums[0]+nums[1] is 9
*/
```