WEB Test 2

[16 marks]

Time period: 100 minutes (in class)

Instructions

It is a computer-based open-resource test. You can refer to course materials on blackboard, or internet. However, you CANNOT communicate with others. It is an INDIVIDUAL test. You need to work on your own and CANNOT discuss it with others, or copy from others, or distribute your work to others.

You are going to make the Academic Integrity declaration in the work. See below.

Read and follow the instructions CAREFULLY.

Specifications

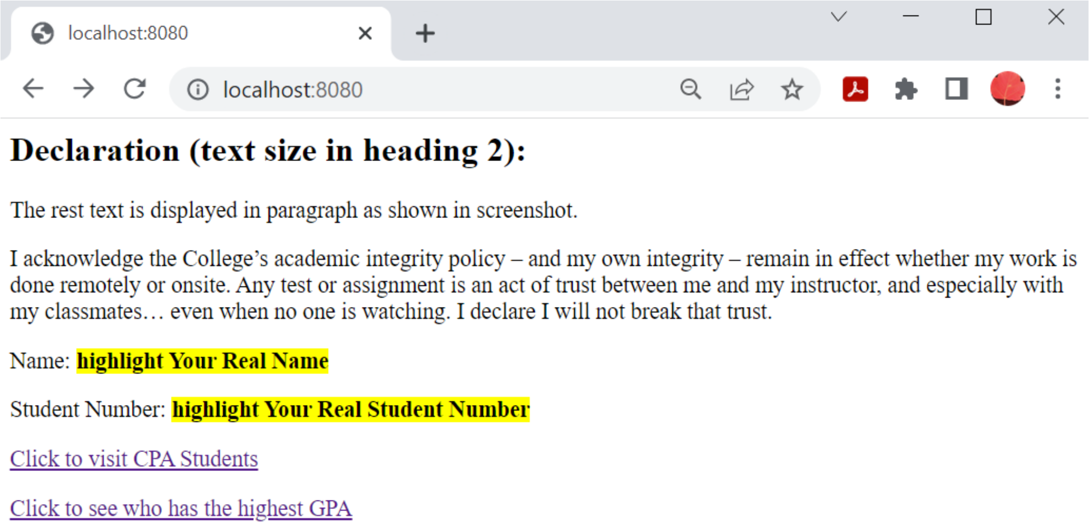
You are going to create a web server program using Node.js and Express.js module. Setup the development environment and install the required module(s) as instructed in class and in Assignments.

This server includes the following routes: home route **"/"**, **"/CPA"**, and **"/highGPA"**. If user requests **other routes**, your server should respond with status code: 404, and return the message: "Error 404: page not found."

Deploy it to Cyclic.

Name your server file **server.js** . Implement it according to the following specifications.

1) The server must make use of the "**express**" module
2) The server must listen on port number **process.env.PORT or 8080**
3) When start the server, it must output: "Express http server listening on ***port***" - to the console, where ***port*** is the port the server is currently listening on (ie: 8080)
4) The home route "**/**" must return the **text in HTML statement**, which will be displayed as the following screenshot. If the user requests to this route using GET method, your server will respond with **EXACTLY** the same page. Instructions are specified on the following screenshot. Please read and follow carefully.



Declaration (text size in heading 2):

The rest text is displayed in paragraph as shown in screenshot.

I acknowledge the College's academic integrity policy – and my own integrity – remain in effect whether my work is done remotely or onsite. Any test or assignment is an act of trust between me and my instructor, and especially with my classmates… even when no one is watching. I declare I will not break that trust.

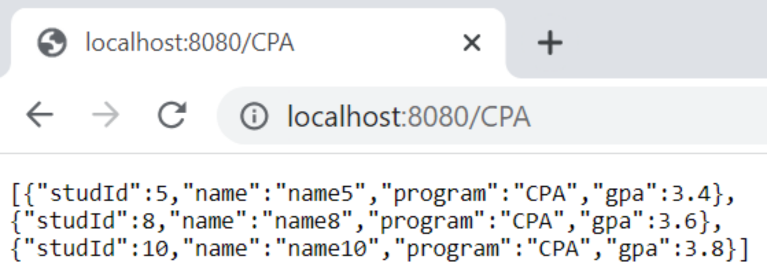
Name: **highlight Your Real Name**

Student Number: **highlight Your Real Student Number**

Click to visit CPA Students

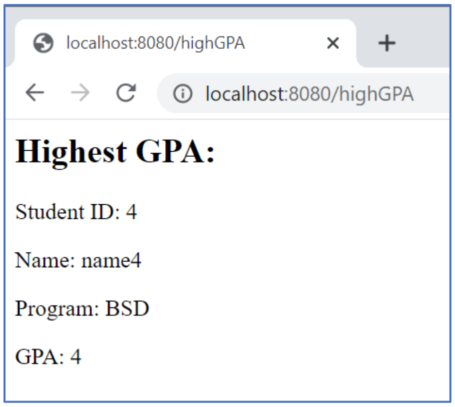
Click to see who has the highest GPA

5) When user click the link “Click to visit CPA Students”, your server responds with all CPA students’ information in JSON format from route **“/CPA”**. Refer to the following screenshot.



```
[{"studId":5,"name":"name5","program":"CPA","gpa":3.4},
{"studId":8,"name":"name8","program":"CPA","gpa":3.6},
{"studId":10,"name":"name10","program":"CPA","gpa":3.8}]
```

6) If user clicks the link “Click to see who has the highest GPA”, your server responds from route **“highGPA”**, with that student’s information as the following screenshot.



Highest GPA:

Student ID: 4

Name: name4

Program: BSD

GPA: 4

Data, folders, files, and functions

1) Create file **students.json** file using the following link and store it in the same folder as **server.js**, name the foler: **test2**.
https://seneca-my.sharepoint.com/:u:/g/personal/sunny_shi_senecacollege_ca/ESoPP9Yh_RdPmglq0JqwYacBwzncBeKHgMpDOaVlS0LNCQ?e=Wgtpej
2) Create a module, name it: **data_prep.js**. Store it in folder **test2**, which is the same folder as server.js & students.json.
3) In the module **data_prep.js,** create and **EXPORTS** the following functions that ALL return **PROMISE** objects. See below for details.
prep():
- This function reads the contents of the data file **"students.json"** and converts the file's contents into an array of objects. Name the array **students**.
- If the operations have finished successfully, invoke the **resolve** method for the promise to communicate back to **server.js** that the operation was a success. Method resolve() does not pass any data.
- If there are errors at any time during this process, invoke the **reject** method for the promise and pass an appropriate message, ie: **reject**("unable to read file").

cpa():

- This function will **resolve** an array of all CPA students.
- If for any reason, no CPA students are returned, this function must invoke the **reject** method and pass a meaningful message, ie: "no results returned".

highGPA():

- This function will **resolve ONE** student who has the highest GPA. If more than one student has the same highest GPA, this function returns the first student with the highest GPA.
- If for any reason, cannot find the student with the highest GPA, this function must invoke the **reject** method and pass a meaningful message, ie: "Failed finding the student with the highest GPA".

4) In **server.js**, you **only call app.listen()** if the call to the **prep()** method is successful. If the **prep()** method invoked **reject**, then you should not start the server (since there will be no data to fetch) and instead display (console.log) the error message.

Deploy to Cyclic **[1 mark]**

- Once you are satisfied with your application, deploy it to Cyclic as you did in Assignment 1.
- **NOTE:** if you have been experimenting on Cyclic and have created multiple apps already, you may delete the unused ones. If you have received a grade for Assignment 1, it is safe to delete this app also. (login to the Cyclic, click on your app, go to “settings”, … **Delete app**).

Submission:

- TEST & Place your app’s URL (from Cyclic)
  - in the beginning of server.js
  - on Blackboard as text submission
- If your app doesn’t work, please DON’T include this Cyclic URL. Put a note on Blackboard submission mentioning your problems instead.
- Submit the following zip file.
- Compress (**.zip**, do NOT use rar or other formats) your “test2” folder. Name it as **test2_yourName.zip** and submit it to Blackboard under **Assignments** -> **Test2**.

Evaluation:

Server.js basic structure OK	**[1 mark]**
Routes (**“/”**, **“/cpa”**, **“/highGPA”**, **no matching** routes) are defined properly	**[8 marks]**
prep() is implemented as expected	**[2 marks]**
CPA() is implemented as expected	**[2 marks]**
highGPA() is implemented as expected	**[2 marks]**
Successfully published to Cyclic	**[1 mark]**