CSCI 4131 – Internet Programming
Assignment 8

Posted 12/4/2017 – **Due 12/15/2017 at 11:55 PM**

1 Description

The objective of this assignment is to provide an introduction to web-development with Node.js. Node.js is basically a webserver, written in JavaScript. It uses an event-driven, non-blocking I/O model. So far, in this course we have used JavaScript for client-side scripting. For this assignment, we will use JavaScript for server-side scripting. Essentially, instead of writing the server code in PHP or Python, we will develop a basic server using JavaScript.

2 Preparation and Provided Files

<u>The first step will be to get Node.js running on CSE lab machines. Please follow the following instructions carefully:</u>

1. Log into a CSE lab machine.
2. The CSE lab machines run version 6.12.0 of Node.js (as of 12/01/2017). The current stable version is 8.9.1. The version of Node.js on the CSE lab machines is sufficient for this introductory assignment.
3. Open the terminal and type the following command to add the Node.js module to your path:
   module add soft/nodejs

4. The next step is to check the availability of Node.js. Type the following command to check the version of Node.js on the machine:
   node -v

5. This will display the current installed version

<u>The second step is to create a Node.js project for this assignment. Please follow the following instructions carefully:</u>

1. Open the terminal on a CSE lab machine.
2. Create a directory named <x500id_hw08> by typing the following command:
   mkdir yourx500id_hw08

3. Go inside the directory by typing the following command:
   cd yourx500id_hw08

4. Having a file named *package.json* in Node.js project makes it easy to manage module dependencies and makes the build process easier. To create *package.json* file, type the following command:
   npm init

5. Executing step 4 above will result in a prompt for you to enter information. Use the following guideline to enter the information (The things that you need to enter are in bold. The fields with the text **<Leave blank>** should be left blank.):

   name: (yourx500id_hw08) **yourx500id_hw08**

   version: (1.0.0) **<Leave blank>**

   description: **Assignment 8**

   entry point: (index.js) **<Leave blank> (**We will provide an index.js file for your use**)**

   test command: **<Leave blank>**

   git repository: **<Leave blank>**

   keywords: **<Leave blank>**

   author: **yourx500id**

   license: (ISC) **<Leave blank>**

6. Next, copy all the that are provided for this assignment to the directory: *yourx500id_hw08*

7. Listing (via the command: **`ls -al`**) all the available files in the directory should display the following:

```
-rwxrwxrwx 1 yourx500id CSEL-student 1076 Dec  1 10:00 404.html

-rwxrwxrwx 1 yourx500id CSEL-student 1157 Dec  1 10:00 405.html

-rwxrwxrwx 1 yourx500id CSEL-student 1591 Dec  1 10:00 course.json

-rwxrwxrwx 1 yourx500id CSEL-student 6241 Dec  1 10:01 index.js

-rwxrwxrwx 1 yourx500id CSEL-student  229 Dec  1 09:40 package.json
```

8. The project setup is now complete and you are ready to start the server.

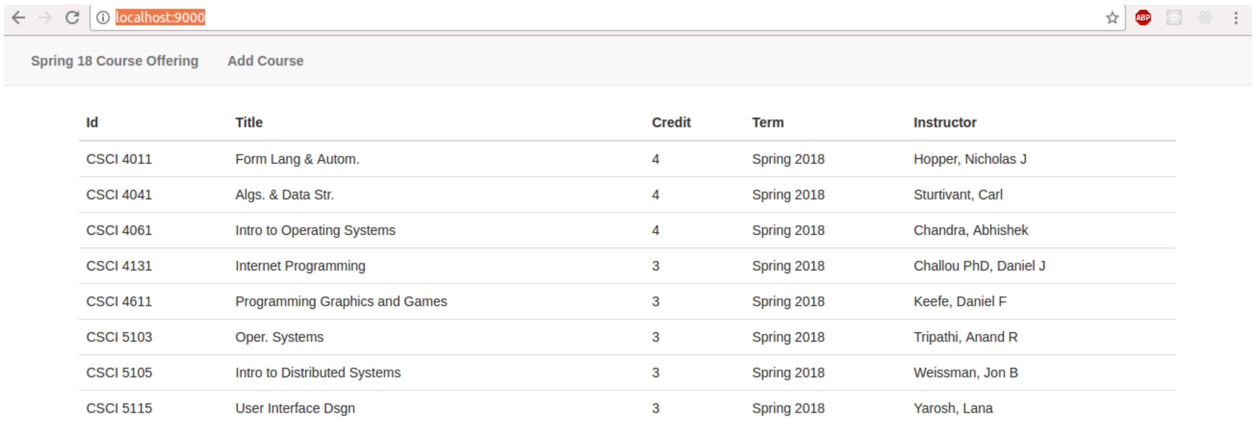
To start the server, type the following command:

node index.js

Executing step 8 above starts the server and binds it to port 9000.

Now, using the browser running on the same machine as your node.js server, open: **localhost:9000**

The following page should be displayed:



Id	Title	Credit	Term	Instructor
CSCI 4011	Form Lang & Autom.	4	Spring 2018	Hopper, Nicholas J
CSCI 4041	Algs. & Data Str.	4	Spring 2018	Sturtivant, Carl
CSCI 4061	Intro to Operating Systems	4	Spring 2018	Chandra, Abhishek
CSCI 4131	Internet Programming	3	Spring 2018	Challou PhD, Daniel J
CSCI 4611	Programming Graphics and Games	3	Spring 2018	Keefe, Daniel F
CSCI 5103	Oper. Systems	3	Spring 2018	Tripathi, Anand R
CSCI 5105	Intro to Distributed Systems	3	Spring 2018	Weissman, Jon B
CSCI 5115	User Interface Dsgn	3	Spring 2018	Yarosh, Lana

The following files are provided for this assignment:

1. `index.js`: This file contains the partially complete code for the server.
2. `404.html`: This file should be sent to client when the requested file is not found.
3. `405.html`: This file should be sent to client when the requested method is supported.
4. `course.json`: This file contains the course list in JSON format.

A description of the functionality that you need to add for this assignment is on the next page. Note, the functionality you need to add is highlighted in red font.

3 Functionality

When the server starts, it listens for incoming connections on port 9000. This server is designed to handle only GET and POST requests.

GET requests:

1. Requests for two different pages can be handled by this server: the *homepage*, and the *addCourse.html* page.
2. GET request for the homepage: The code for this has already been provided to you in *index.js* file. The function *getHomePage* reads the *course.json* file and constructs the HTML page which is returned to the client requesting this page. **You do not need to add any code for this.**
3. GET request for the addCourse.html page:
   a. You need to develop your own *addCourse.html* page. This will be very similar to the other forms that you have developed for other assignments in this course. A screenshot for this page has been provided in the screenshots section. The form contained in this page should send the posted form data to the server by using: *action="/postCourse"*

   b. You need to complete the *getAddCoursePage* function in index.js file to return your addCourse.html page to the client. Return the appropriate status code (200).

4. GET request for any page other than homepage and addCourse.html: If the client requests for any page other than the homepage and the addCourse.html page, the server should return the 404.html page provided to you. For this, complete the *get404* function in index.js file. Return the appropriate status code (404).

POST requests:

5. The server should process the form data posted by client. The form present in *addCourse.html* should allow the user to enter the contents of a new course and update the course offering.
6. Details for few courses are pre-populated in course.json file. Your job is to add the capability to add a the new course to this file and redirect the user to the homepage after successful addition of the new course.
7. For this, you need to complete the *addCourseFunction* function present in index.js file. This function should read the form data, add the new information to course.json file, and redirect the user to homepage. The code for redirection is 302. Ensure that the newly added data does not change the format of the course.json file.

Requests other than GET/POST:

8. For such requests, server should return the 405.html file provided to you. Complete the *get405* function in index.js file. Return the appropriate status code (405).

4 Submission Instructions

1. Include the following two files in one zipped folder for your submission:
    a. index.js
    b. addCourse.html
2. You do not need to provide any extra files.
3. The name of the zipped folder should be yourx500id_nodejs.
   PLEASE ENSURE TO TEST YOUR CODE ON CSE LAB MACHINES.

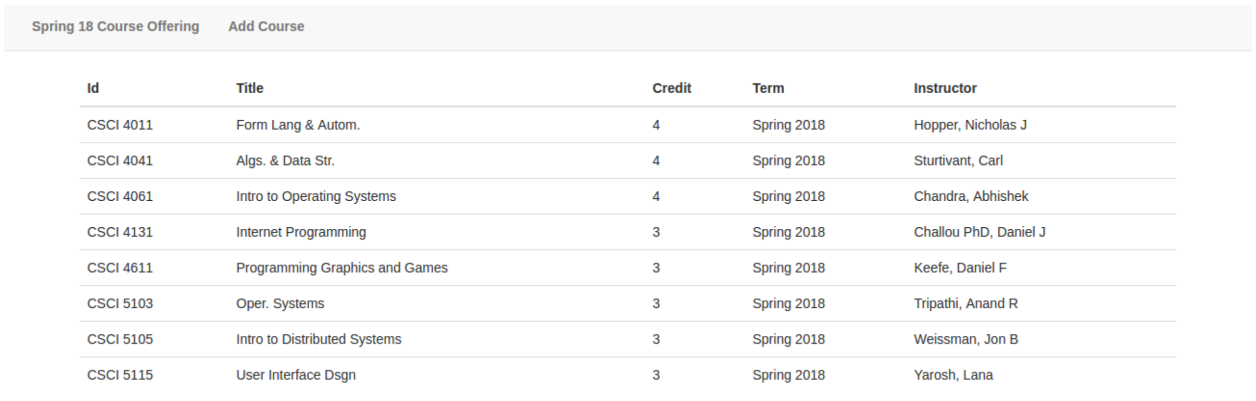
5 Evaluation

Your submission will be graded out of 50 points on the following items:

1. *addCourse.html* is present and can send form data to the server (**25 points**).
2. *getAddCoursePage* function returns the addCourse.html to the client with proper response code (**5 points**).
3. *get404* function returns the 404.html to the client with proper response code (**5 points**).
4. *get405* function returns the 405.html to the client with proper response code (**5 points**).
5. *addCourseFunction* function adds the content of the new course to course.json file (**5 points**).
6. The user is redirected to the home page after successful addition of a new course (**5 points**)

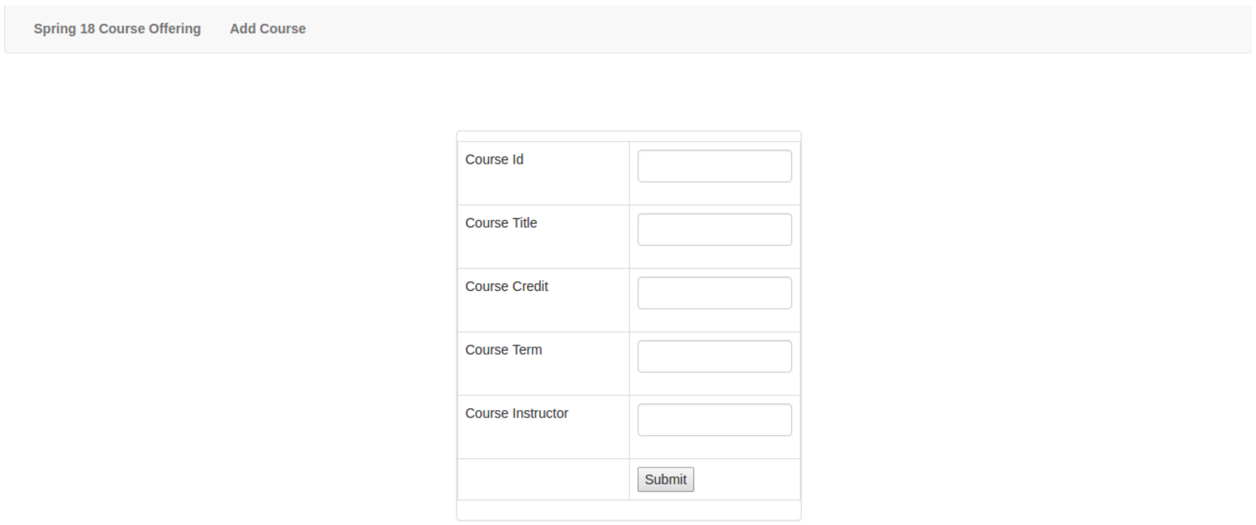
6 Screenshots

HomePage



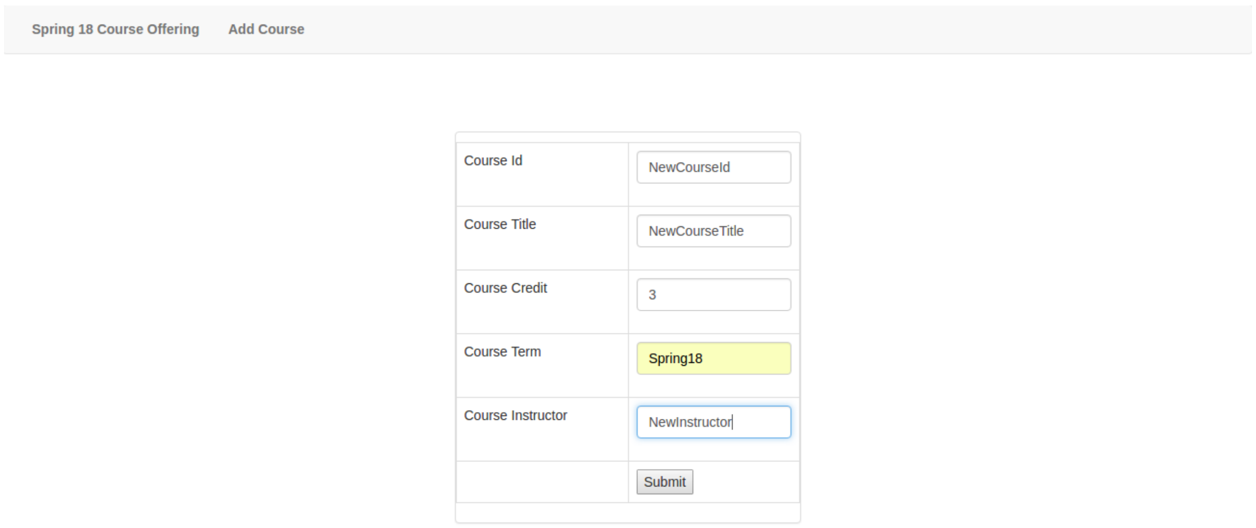
Id	Title	Credit	Term	Instructor
CSCI 4011	Form Lang & Autom.	4	Spring 2018	Hopper, Nicholas J
CSCI 4041	Algs. & Data Str.	4	Spring 2018	Sturtivant, Carl
CSCI 4061	Intro to Operating Systems	4	Spring 2018	Chandra, Abhishek
CSCI 4131	Internet Programming	3	Spring 2018	Challou PhD, Daniel J
CSCI 4611	Programming Graphics and Games	3	Spring 2018	Keefe, Daniel F
CSCI 5103	Oper. Systems	3	Spring 2018	Tripathi, Anand R
CSCI 5105	Intro to Distributed Systems	3	Spring 2018	Weissman, Jon B
CSCI 5115	User Interface Dsgn	3	Spring 2018	Yarosh, Lana

addCourse.html



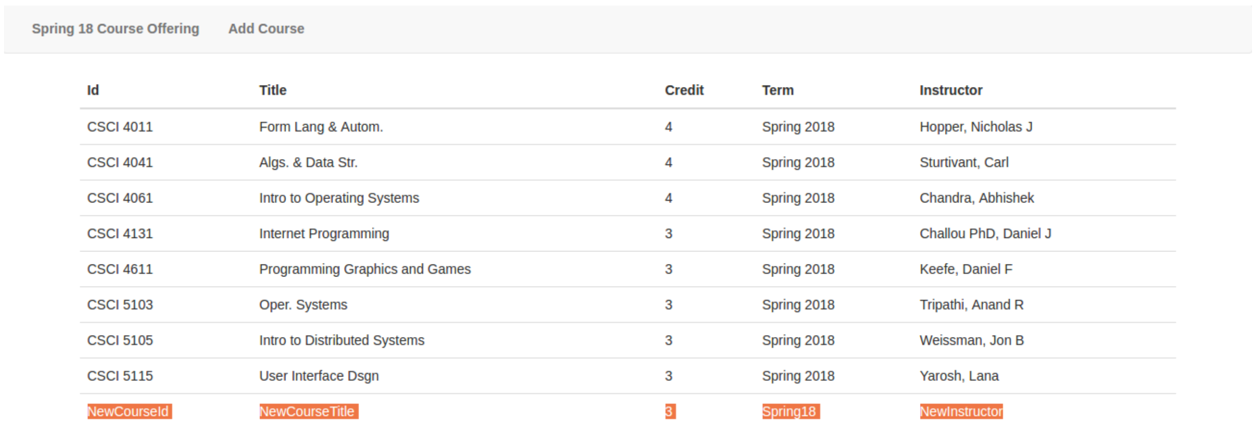
Course Id	
Course Title	
Course Credit	
Course Term	
Course Instructor	

addCourse.html - 2



Course Id	NewCourseId
Course Title	NewCourseTitle
Course Credit	3
Course Term	Spring18
Course Instructor	NewInstructor

HomePage after adding a new course



Id	Title	Credit	Term	Instructor
CSCI 4011	Form Lang & Autom.	4	Spring 2018	Hopper, Nicholas J
CSCI 4041	Algs. & Data Str.	4	Spring 2018	Sturtivant, Carl
CSCI 4061	Intro to Operating Systems	4	Spring 2018	Chandra, Abhishek
CSCI 4131	Internet Programming	3	Spring 2018	Challou PhD, Daniel J
CSCI 4611	Programming Graphics and Games	3	Spring 2018	Keefe, Daniel F
CSCI 5103	Oper. Systems	3	Spring 2018	Tripathi, Anand R
CSCI 5105	Intro to Distributed Systems	3	Spring 2018	Weissman, Jon B
CSCI 5115	User Interface Dsgn	3	Spring 2018	Yarosh, Lana
NewCourseId	NewCourseTitle	3	Spring18	NewInstructor