Intro to JavaScript Week 4 Coding Assignment

Sean Flaherty

**Points possible:** 70

|  |  |  |
| --- | --- | --- |
| Category | Criteria | % of Grade |
| Functionality | Does the code work? | 25 |
| Organization | Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear. | 25 |
| Creativity | Student solved the problems presented in the assignment using creativity and out of the box thinking. | 25 |
| Completeness | All requirements of the assignment are complete. | 25 |

**Instructions:** In VS Code, or an IDE of your choice, write the code that accomplishes the objectives listed below. Ensure that the code compiles and runs as directed. Take screenshots of the code and of the running program (make sure to get screenshots of all required functionality) and paste them in this document where instructed below. Create a new repository on GitHub for this week’s assignments and push this document, with your JavaScript project code, to the repository. Lastly, in the Learning Management System, click the “Add Submission” button and paste the URL to your GitHub repository.

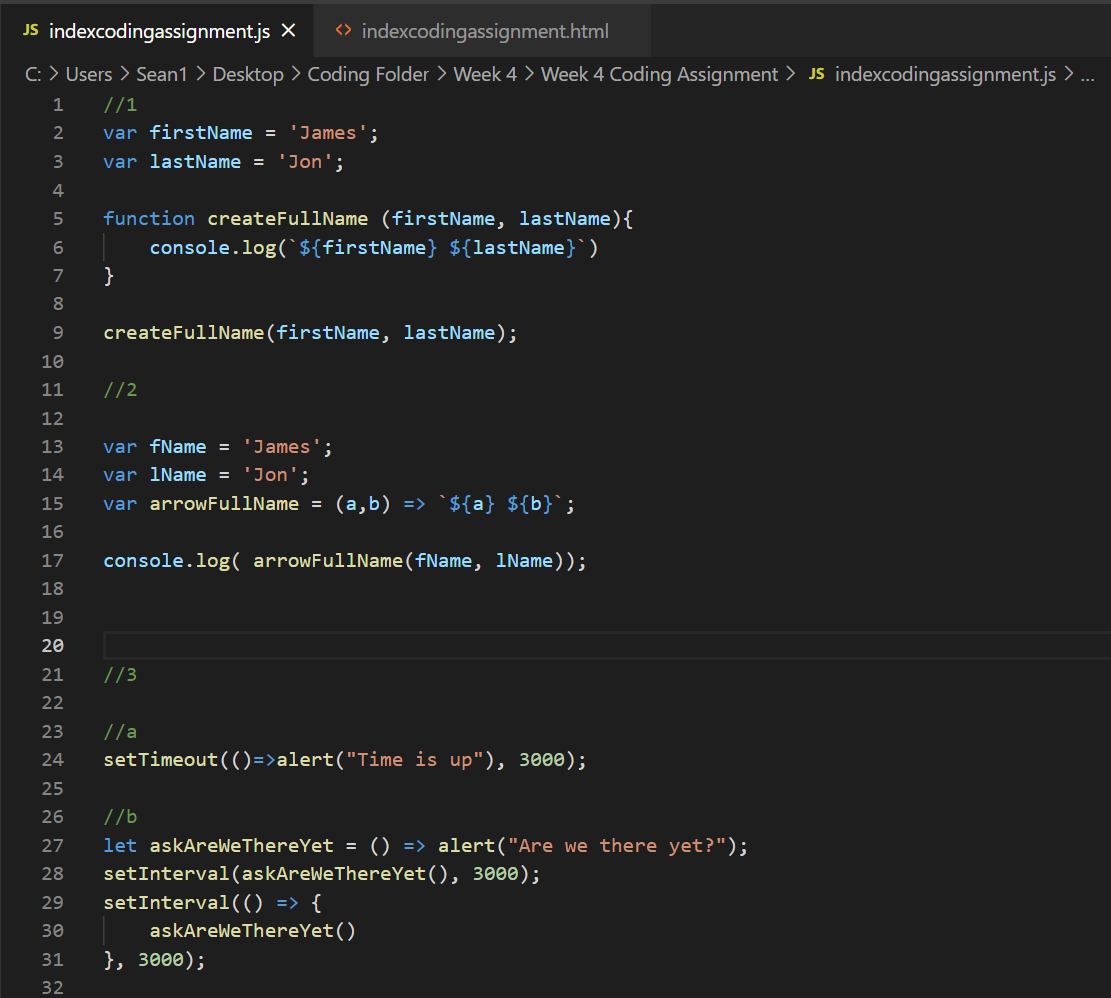
**Coding Steps:**

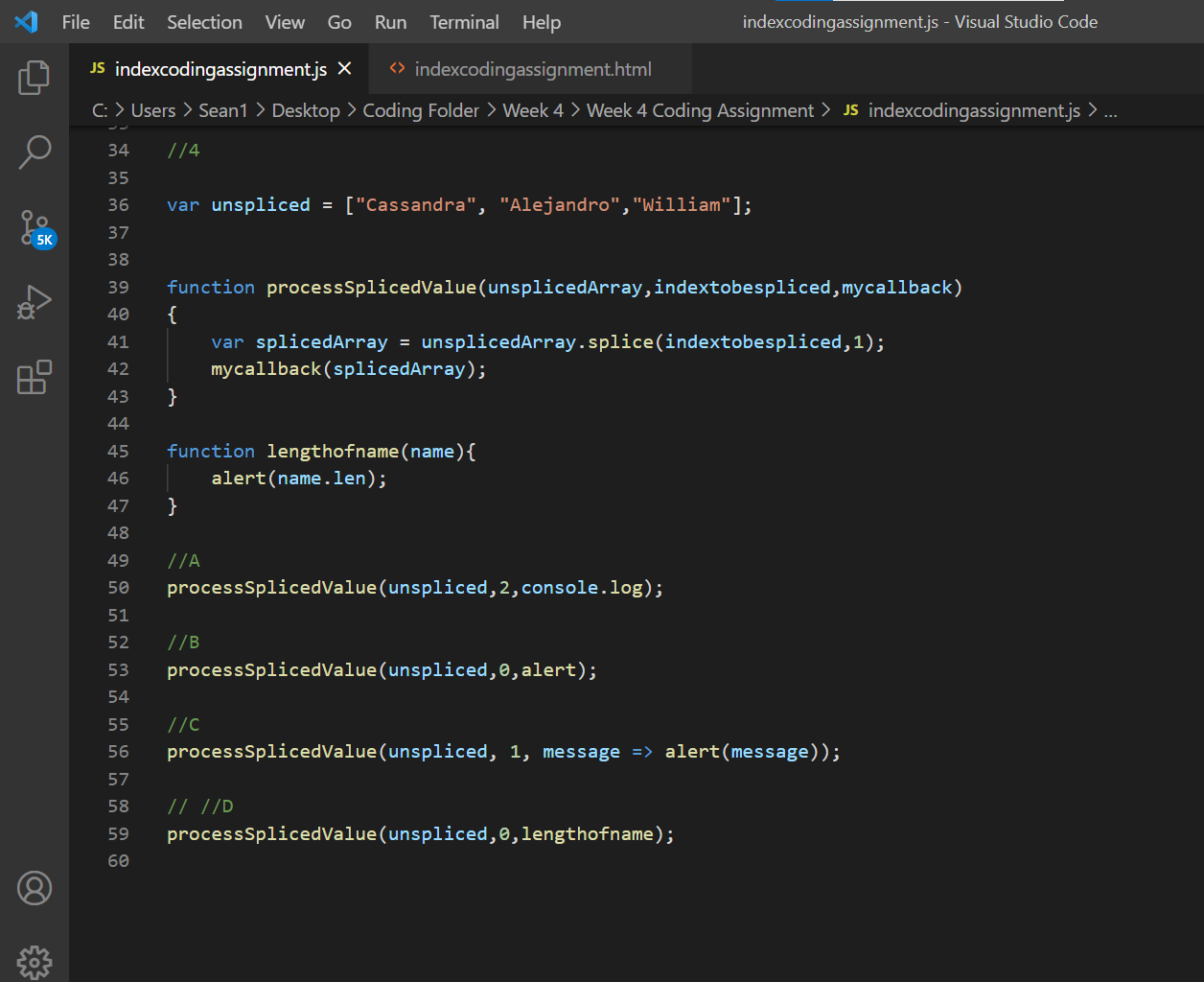
1. Using template literals instead of concatenation, write a function that takes firstName and lastName and returns ‘fistName lastName’
2. Write the same function as above as an arrow function with a different name.
3. Look up the JavaScript functions setTimeout() and setInterval(). Notice how they each take a callback.
   1. Using setTimeout, write an anonymous (has no name assigned to it) arrow function in the callback parameter position. The function should alert ‘Time is up!’. Choose whatever length of time you want for the timeout.
   2. Write an arrow function named askAreWeThereYet that alerts ‘Are we there yet?’. Using setInterval, pass askAreWeThereYet into the callback parameter position. Choose whatever length of time you want for the interval.
4. In this step you are going to write a function that takes a callback to better understand how callbacks work.
   1. Write a new function named processSplicedValue that takes 3 parameters – an array, the index of the element to be spliced from the array, and a callback function that will process the sliced element.   
        
      Inside the function, use the first two parameters to splice an element from the array.  
        
      Inside the function, call the callback function and pass the spliced value into it.

Outside of the function, create an array of strings, call processSplicedValue, and pass in the array you just created, an index number, and console.log into it.   
  
For example: processSplicedValue(arrayName, 2, console.log);

* 1. Call the processSplicedValue function again but this time pass in the alert method instead of console.log.
  2. Call the processSplicedValue function again, but this time pass in an anonymous arrow function that alerts the spliced value.
  3. Call the processSplicedValue function one more time, but this time, pass in a custom function of your choice that you should create and name.

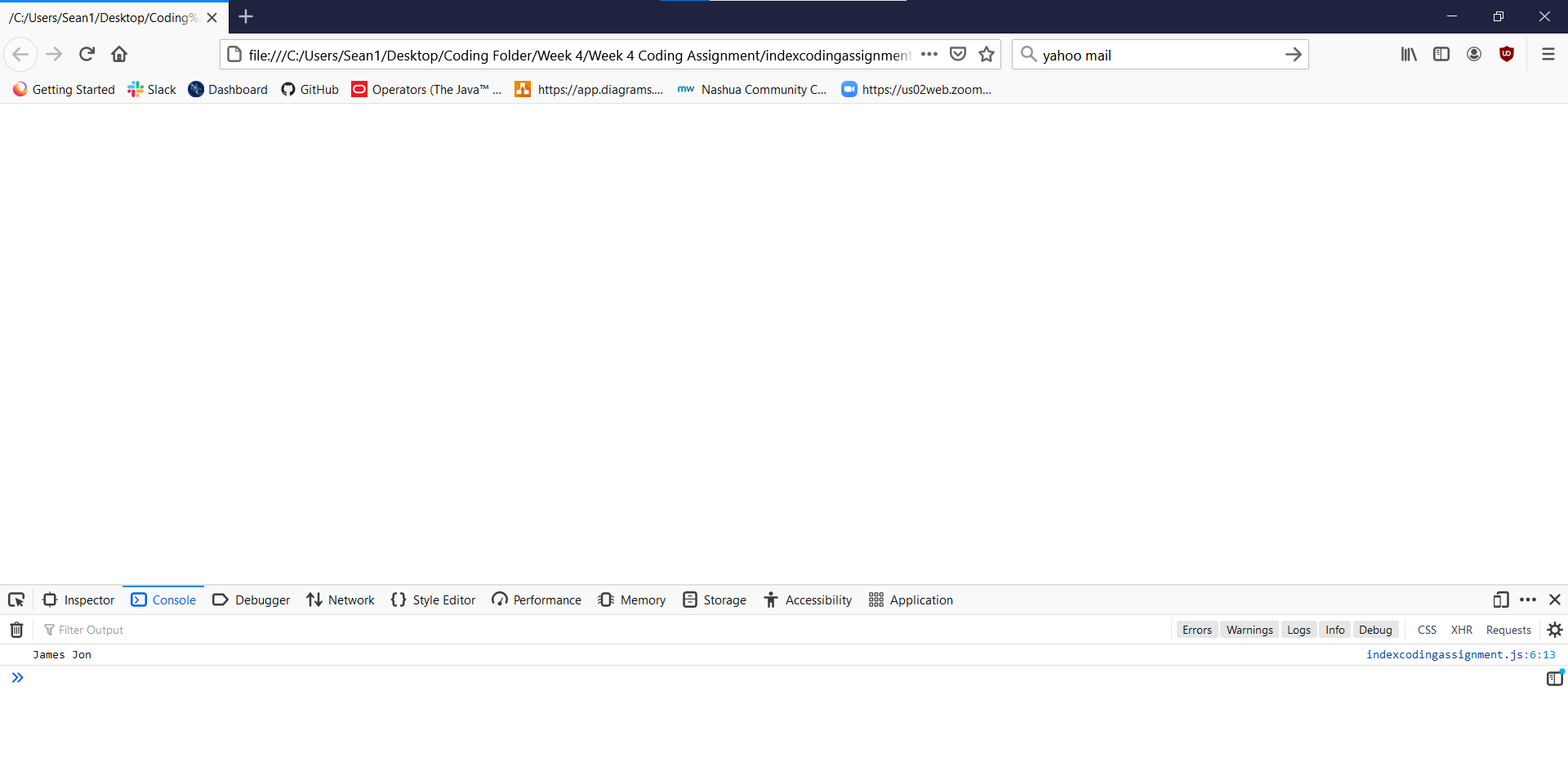
**Screenshots of Code:**

****

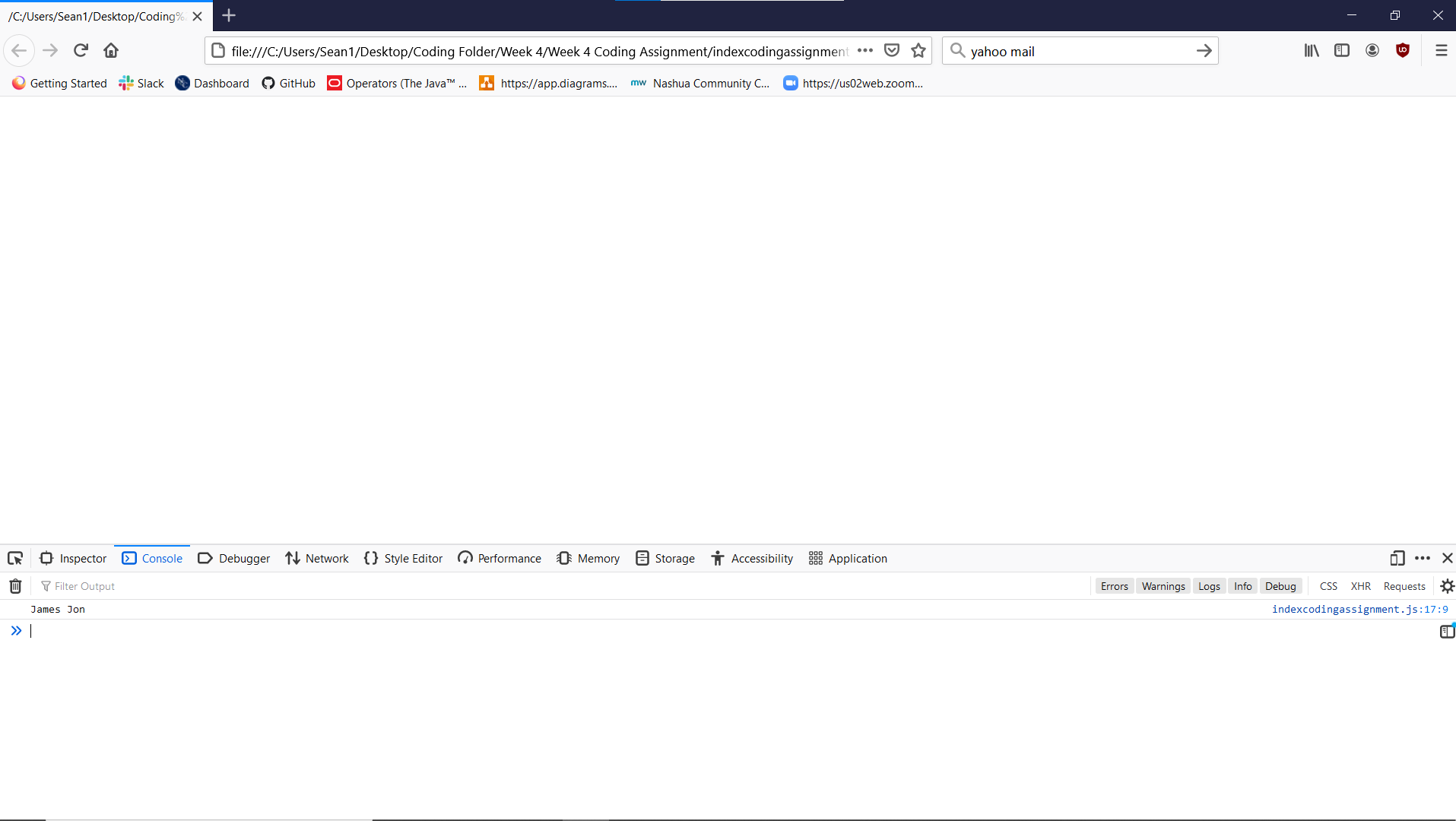
****

**Screenshots of Running Application:**

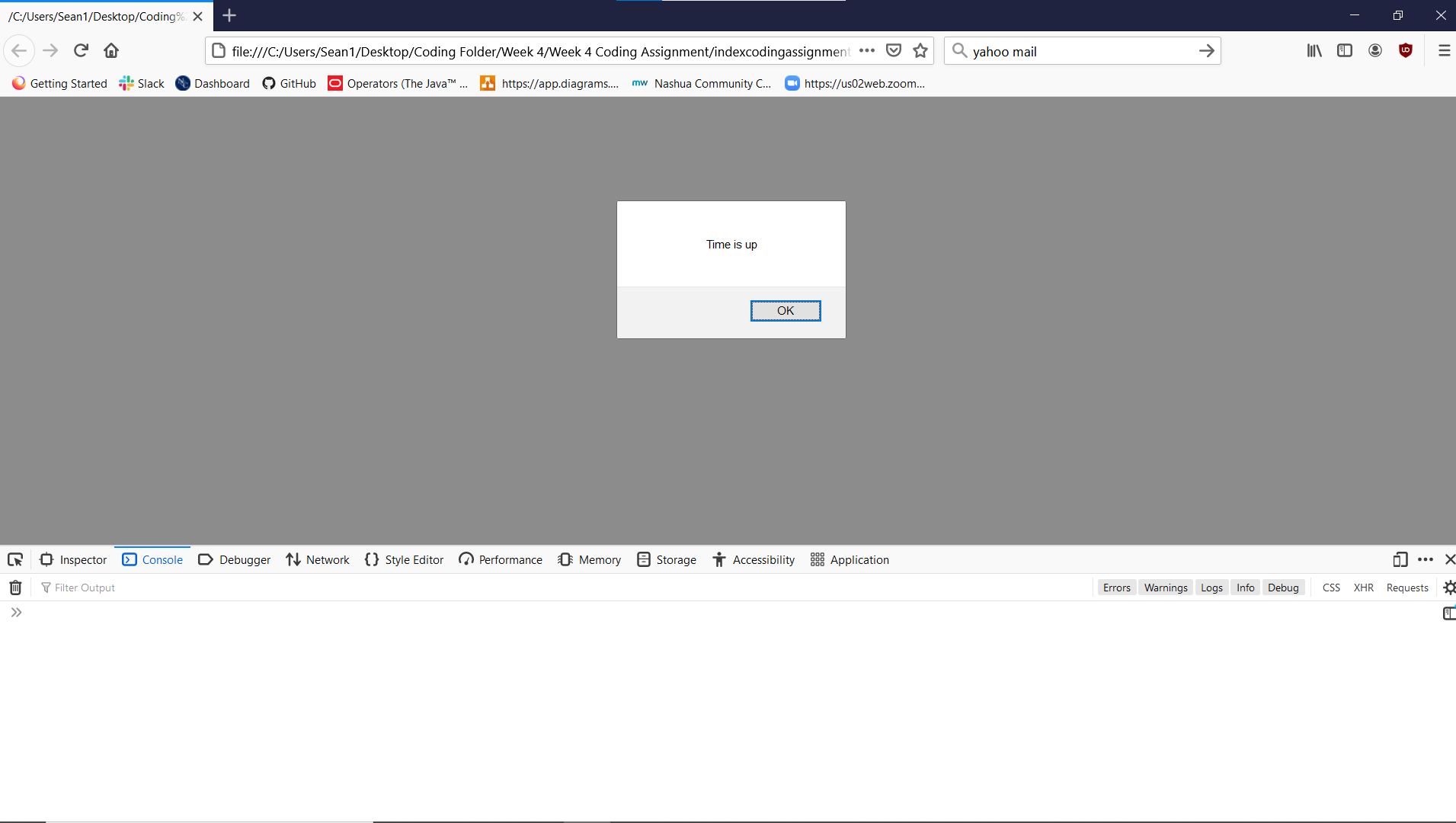
**1.**

****

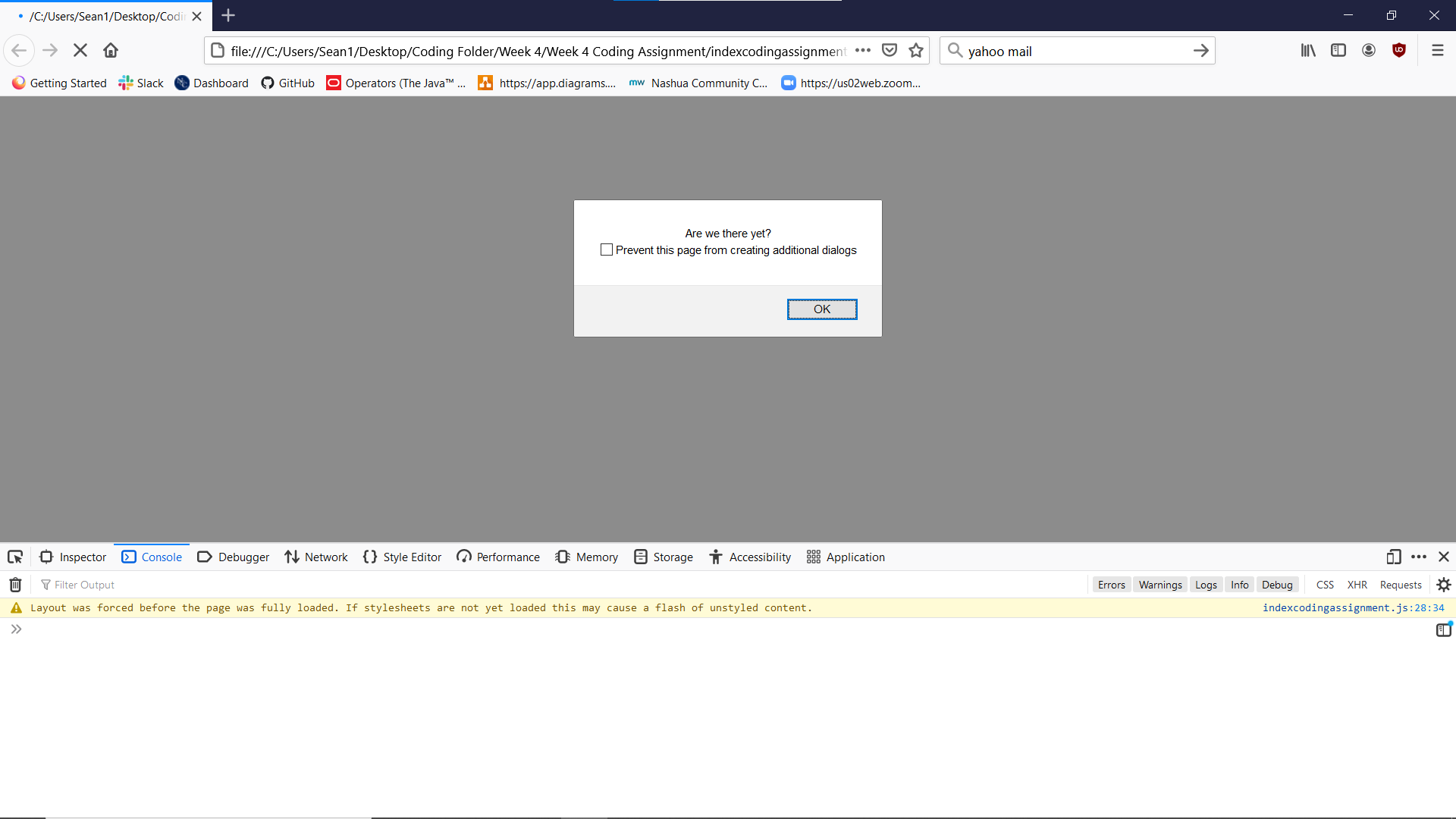
**2.**

****

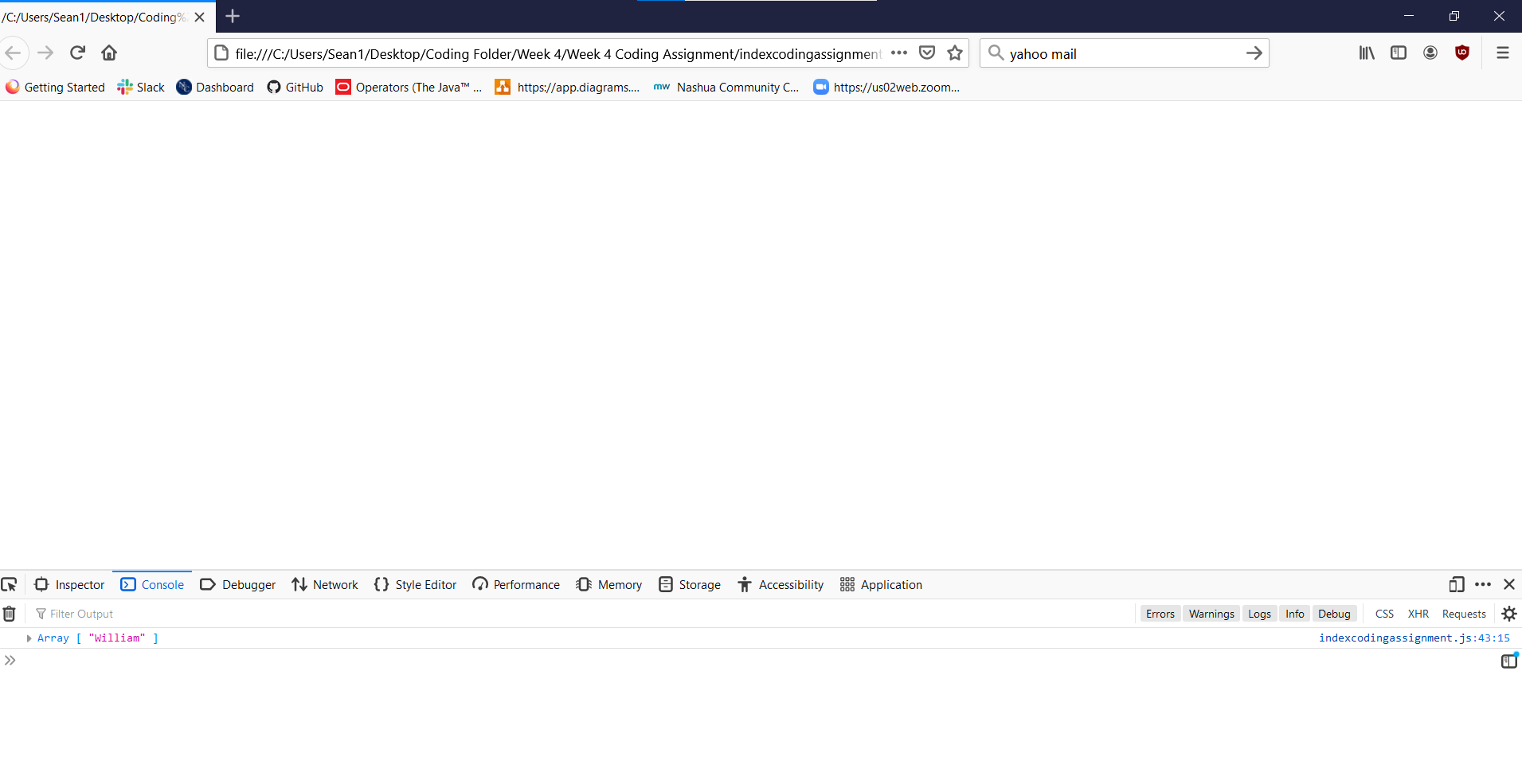
**3a.**

****

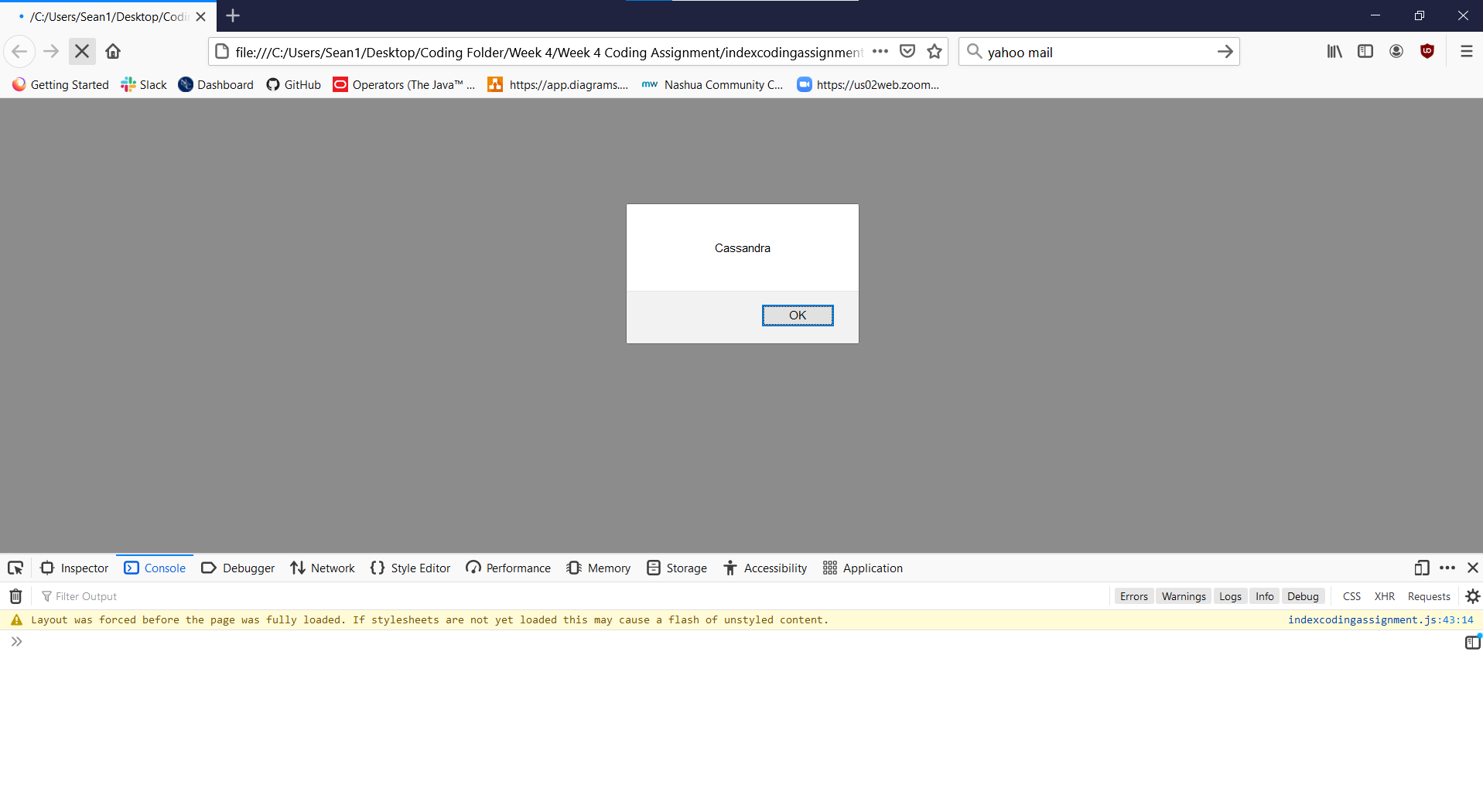
**3b.**

****

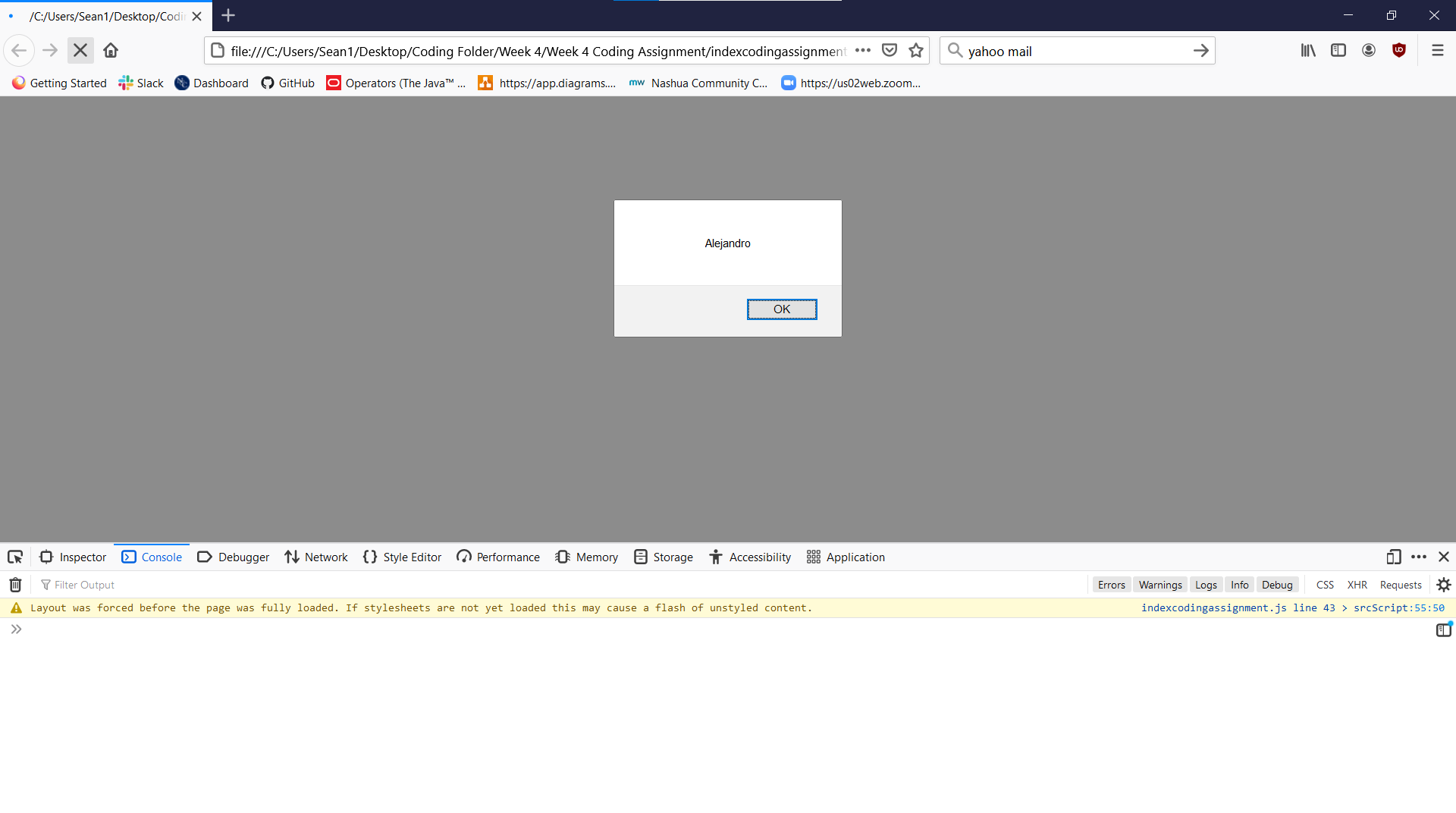
**4a.**

****

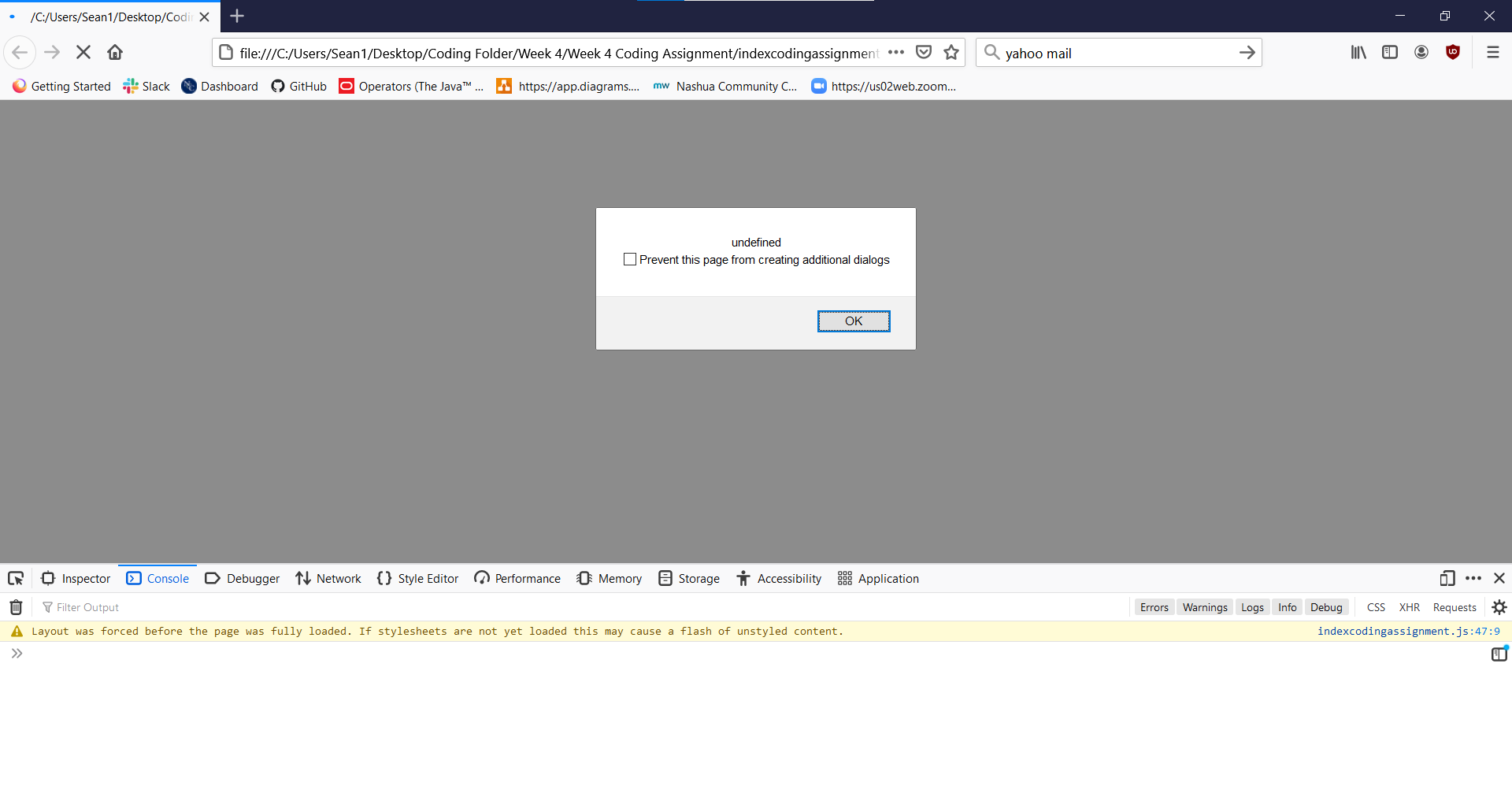
**4b.**

****

**4c.**

****

**4d.**

****

**URL to GitHub Repository:**

<https://github.com/sflaherty50/IntroductiontoJavaScript-Week-4>