

myMurdoch Learning Dashboard myMurdoch Advice My Units

Submission status	Submitted for grading
Grading status	Graded
Due date	Friday, 15 April 2022, 5:00 PM
Time remaining	Assignment was submitted 6 days 13 hours late
Last modified	Friday, 22 April 2022, 6:21 AM


File submissions

[33170193/CT375Assignment1.zip](#) 22 April 2022, 6:21 AM


Submission comments

[Comments \(0\)](#)

Feedback

Grade	85.00 / 100.00
Graded on	Friday, 15 July 2022, 6:36 AM
Graded by	 Hong Xie

Feedback comments

 This assignment is well done. However, you forgot to complete and include the assignment coversheet.
The photo display seems to work only with png format.

1) My assignment is considered a student management system where students and student details can be added to the system through the add student navigator bar. And student details can be retrieved in the search for student navigator bar. The way I guarantee that all new students added are unique and not duplicates is by validating whether the student ID is unique or not. Students can have all fields the same but the student ID must be different to indicate uniqueness.

Key findings

- On the ceto server if you upload a images file that are large in size (>70KB) there is a chance segmentation fault will occur when trying to upload. This could be because ceto has disabled any large uploads so the ceto server will not be full
- On non-ceto server (any other computer) you can upload any sized file without a segmentation fault
- This error is not related to my code since if you were to run the same code on local computer or any other computer it works
- Furthermore, this was confirmed by other students
- Issue can be related to node-formidable form.parse method

Assumption:

- Assume there will be a log in page. This makes it so displaying all students is not a security risk
- In the CSV file each line will have 7 cells meaning each line in CSV will be in this format
 - "",""
- Any other line format will be ignored (where there is no seven cells) such as-
 - "
 - Bob,
 - ""

The code to illustrate this-

```

11 function LoadCSVToDatabase(data) {
12
13     var dB = new StudentDatabase();
14     var studentArray;
15
16     studentArray = data.split('\n');
17
18     for(var i = 0; i < studentArray.length; i++) {
19         var newLine = studentArray[i].split(',');
20         if(ValidCSVLine(newLine)) { //SELF NOTE: Why not just add line if(typeof newCSVLineArray[0] === 'undefined' || typeof newCSVLineArray[1] === 'undefined' ...) to validate? Because ValidCSVLine is more robust
21             var tempStudent = new Student(newLine[0], newLine[1], newLine[2], newLine[3], newLine[4], newLine[5], newLine[6]); //SELF NOTE: Why not just execute this without validating? Because results in undefined properties for students
22             dB.AddStudent(tempStudent);
23         }
24     }
25
26     return dB;
27 }
28

```

```

63 function ValidCSVLine(newCSVLineArray) {
64
65     const NumberOfCellsInEachCSVLine = 7;
66
67     for(var i = 0; i < 7; i++) {
68         if(typeof newCSVLineArray[i] === 'undefined') {
69             return false;
70         }
71     }
72
73     return true;
74 }

```

This is a mechanism is a way to make sure the studentData is correct

2)

/home/student/accounts/33170193/assignment1/node_modules/				
Name	Size	Changed	Rights	Owner
..		20/04/2022 4:04:00 PM	rwxr-xr-x	33170193
asap		15/04/2022 9:05:55 AM	rwxr-xr-x	33170193
dezalgo		15/04/2022 9:05:27 AM	rwxr-xr-x	33170193
formidable		21/04/2022 6:12:15 AM	rwxr-xr-x	33170193
hexoid		15/04/2022 9:04:50 AM	rwxr-xr-x	33170193
once		15/04/2022 9:05:20 AM	rwxr-xr-x	33170193
qs		15/04/2022 9:06:48 AM	rwxr-xr-x	33170193
wrappy		15/04/2022 9:05:46 AM	rwxr-xr-x	33170193

All of the highlighted node.js modules are required to get the latest formidable working. I did not use any of them in my application they had to be installed in order to use formidable. If one of the highlighted modules were deleted then the application wouldn't run since formidable would be missing a dependency.

The node.js modules I developed include the following-

Student module- (student.js)

The responsibility of this module is to provide a constructor to allow for the creation of ES6 student objects. The benefit of this module is it provides a blue print for creating many student objects in a systematic controlled way. It also allows for easier handling of student information.

studentDatabase module- (studentDatabase.js)

This module provides a centralised structure to store the added students in the student management system. It is responsible for handling all the students in the student management system.

For instance- AddStudent(...) function is a typical function that all databases have as a feature. It is often referred as adding entries to the database. It is responsible for providing controlled addition of

students to the student management system. It validates the studentID to ensure no duplicate students will be added to the student management system

For instance- Search(...) function is also another typical function all database have. In this context the parameter search takes in is the search key and it returns all the students that match the search key. Searching is a perfect responsibility for the studentDatabase to have. Also this function makes sure searches can be case insensitive

Csv handling module- (csvHandling.js)

This module is responsible for any operations that involve CSV. For example writing newly added students to CSV and loading students.csv to the student database

addStudentBackend and searchStudentBackend- (addStudentBackend.js and searchStudentBackend.js)

From the request handlers, depending on the type of request, the addStudentBackend module is called to finish the adding student to database operation. Similarly, from the request handler the searchStudentBackend is called to complete the final steps in searching students' operation. Both files were created in order to not burden a requestHandler with too much responsibility (single responsibility principle). Also allows for better code readability and easier maintenance

3)

Submit student details

- addStudentForm.html
 - Provides the form to submit new student details to the student management system
- addStudentValidation.js
 - Responsible for client-side validation for the addStudentForm.html
 - Ensures the required fields- student ID, first name and last name are entered and all the fields are valid format
 - Provides inline feedback for client-side validation to addStudentForm.html
- RequestHandlers.js
 - Contains reqUploadStudent(...) this processes the addStudentForm.html
 - Calls the ProcessUploadStudentForm(...) to continue the processing of the addStudentForm.html

- Calls the DisplayFailMessage(...) or DisplaySuccessMessage(...) in the addStudentsBackEnd.js which displays success or failure message
- addStudentsBackEnd.js
 - Contains ProcessUploadStudentForm(...) function
 - Responsible for working out whether submission is successful or not
 - Responsible for calling the function to validate unique student ID
 - Creates the html page indicating success or failure with submission

Search student details

- searchStudentsForm.html
 - Provides the html form to search for students in the student management system
- searchStudentFrontEnd.js
 - Responsible for initiating the asynchronous JQuery post sending the search key to the backend through request handler
 - Responsible for capturing the matching students of the search key in JSON format
- RequestHandlers.js
 - Contains reqSearchStudentsSubmit (...) this request handler this processed the searchStudentsForm.html form
 - Calls the FindMatchesFromSearchKey(..)
 - Sends the matching student back to the front end (searchStudentFrontend.js)
- searchStudentBackEnd.js
 - Contains FindMatchesFromSearchKey(...) is responsible for calling the database and call it's function to find matching students
- studentDatabase.js
 - Contains the Search function and returns matching students to the client

4)

Why use a map in studentDatabase.js?

- StudentID is unique therefore guarantees that when searching for the key every element will not have to be iterated through so thus map is efficient
- Searching through the data structure for studentID is required every time a client tries adds a new student. This is to ensure unique students are entered
- O(1) time complexity
- Also logically adding a new entry should not take as much time as searching for something

Why have a structure studentDatabase.js?

- In many situations we need to be able to handle the stored group of students in the student management system
- The student database structure provides the client the opportunity to perform searching and adding operations

Why have an array in studentDatabase.js?

- The search operation needed to go through all the students in the data structure
- This means an array would be suitable since it is most ideal structure for iterating through all elements given it is stored in the contiguous memory side by side

The screenshot shows the Total Validator interface. At the top, it says 'Total Validator' with a logo and navigation links: 'Documentation', 'Reference', 'Website', and 'Contact Form'. Below this is a 'Page report' section with a 'Summary' tab selected. The summary indicates that the page checked is 'C:\Users\Admin\Desktop\studentDatabase.html' and that 0 errors were found. The 'Page layout' section shows the page structure with various elements like 'h1', 'h2', 'h3', 'h4', 'h5', 'h6', 'h7', 'h8', 'h9', 'h10', 'h11', 'h12', 'h13', 'h14', 'h15', 'h16', 'h17', 'h18', 'h19', 'h20', 'h21', 'h22', 'h23', 'h24', 'h25', 'h26', 'h27', 'h28', 'h29', 'h30', 'h31', 'h32', 'h33', 'h34', 'h35', 'h36', 'h37', 'h38', 'h39', 'h40', 'h41', 'h42', 'h43', 'h44', 'h45', 'h46', 'h47', 'h48', 'h49', 'h50', 'h51', 'h52', 'h53', 'h54', 'h55', 'h56', 'h57', 'h58', 'h59', 'h60', 'h61', 'h62', 'h63', 'h64', 'h65', 'h66', 'h67', 'h68', 'h69', 'h70', 'h71', 'h72', 'h73', 'h74', 'h75', 'h76', 'h77', 'h78', 'h79', 'h80', 'h81', 'h82', 'h83', 'h84', 'h85', 'h86', 'h87', 'h88', 'h89', 'h90', 'h91', 'h92', 'h93', 'h94', 'h95', 'h96', 'h97', 'h98', 'h99', 'h100'. The 'Errors' section is currently empty, showing '0 errors found'. The source code is visible at the bottom, showing a series of HTML tags and attributes, with some lines highlighted in red to indicate errors. The errors are listed as follows:

- Line 10: Error: The following element is not allowed within the container: <div>
- Line 11: Error: The following element is not allowed within the container: <div>
- Line 12: Error: The following element is not allowed within the container: <div>
- Line 13: Error: The following element is not allowed within the container: <div>
- Line 14: Error: The following element is not allowed within the container: <div>
- Line 15: Error: The following element is not allowed within the container: <div>
- Line 16: Error: The following element is not allowed within the container: <div>
- Line 17: Error: The following element is not allowed within the container: <div>
- Line 18: Error: The following element is not allowed within the container: <div>
- Line 19: Error: The following element is not allowed within the container: <div>
- Line 20: Error: The following element is not allowed within the container: <div>
- Line 21: Error: The following element is not allowed within the container: <div>
- Line 22: Error: The following element is not allowed within the container: <div>
- Line 23: Error: The following element is not allowed within the container: <div>
- Line 24: Error: The following element is not allowed within the container: <div>
- Line 25: Error: The following element is not allowed within the container: <div>
- Line 26: Error: The following element is not allowed within the container: <div>
- Line 27: Error: The following element is not allowed within the container: <div>
- Line 28: Error: The following element is not allowed within the container: <div>
- Line 29: Error: The following element is not allowed within the container: <div>
- Line 30: Error: The following element is not allowed within the container: <div>
- Line 31: Error: The following element is not allowed within the container: <div>
- Line 32: Error: The following element is not allowed within the container: <div>
- Line 33: Error: The following element is not allowed within the container: <div>
- Line 34: Error: The following element is not allowed within the container: <div>
- Line 35: Error: The following element is not allowed within the container: <div>
- Line 36: Error: The following element is not allowed within the container: <div>
- Line 37: Error: The following element is not allowed within the container: <div>
- Line 38: Error: The following element is not allowed within the container: <div>
- Line 39: Error: The following element is not allowed within the container: <div>
- Line 40: Error: The following element is not allowed within the container: <div>
- Line 41: Error: The following element is not allowed within the container: <div>
- Line 42: Error: The following element is not allowed within the container: <div>
- Line 43: Error: The following element is not allowed within the container: <div>
- Line 44: Error: The following element is not allowed within the container: <div>
- Line 45: Error: The following element is not allowed within the container: <div>
- Line 46: Error: The following element is not allowed within the container: <div>
- Line 47: Error: The following element is not allowed within the container: <div>
- Line 48: Error: The following element is not allowed within the container: <div>
- Line 49: Error: The following element is not allowed within the container: <div>
- Line 50: Error: The following element is not allowed within the container: <div>
- Line 51: Error: The following element is not allowed within the container: <div>
- Line 52: Error: The following element is not allowed within the container: <div>
- Line 53: Error: The following element is not allowed within the container: <div>
- Line 54: Error: The following element is not allowed within the container: <div>
- Line 55: Error: The following element is not allowed within the container: <div>
- Line 56: Error: The following element is not allowed within the container: <div>
- Line 57: Error: The following element is not allowed within the container: <div>
- Line 58: Error: The following element is not allowed within the container: <div>
- Line 59: Error: The following element is not allowed within the container: <div>
- Line 60: Error: The following element is not allowed within the container: <div>
- Line 61: Error: The following element is not allowed within the container: <div>
- Line 62: Error: The following element is not allowed within the container: <div>
- Line 63: Error: The following element is not allowed within the container: <div>
- Line 64: Error: The following element is not allowed within the container: <div>
- Line 65: Error: The following element is not allowed within the container: <div>
- Line 66: Error: The following element is not allowed within the container: <div>
- Line 67: Error: The following element is not allowed within the container: <div>
- Line 68: Error: The following element is not allowed within the container: <div>
- Line 69: Error: The following element is not allowed within the container: <div>
- Line 70: Error: The following element is not allowed within the container: <div>
- Line 71: Error: The following element is not allowed within the container: <div>
- Line 72: Error: The following element is not allowed within the container: <div>
- Line 73: Error: The following element is not allowed within the container: <div>
- Line 74: Error: The following element is not allowed within the container: <div>
- Line 75: Error: The following element is not allowed within the container: <div>
- Line 76: Error: The following element is not allowed within the container: <div>
- Line 77: Error: The following element is not allowed within the container: <div>
- Line 78: Error: The following element is not allowed within the container: <div>
- Line 79: Error: The following element is not allowed within the container: <div>
- Line 80: Error: The following element is not allowed within the container: <div>
- Line 81: Error: The following element is not allowed within the container: <div>
- Line 82: Error: The following element is not allowed within the container: <div>
- Line 83: Error: The following element is not allowed within the container: <div>
- Line 84: Error: The following element is not allowed within the container: <div>
- Line 85: Error: The following element is not allowed within the container: <div>
- Line 86: Error: The following element is not allowed within the container: <div>
- Line 87: Error: The following element is not allowed within the container: <div>
- Line 88: Error: The following element is not allowed within the container: <div>
- Line 89: Error: The following element is not allowed within the container: <div>
- Line 90: Error: The following element is not allowed within the container: <div>
- Line 91: Error: The following element is not allowed within the container: <div>
- Line 92: Error: The following element is not allowed within the container: <div>
- Line 93: Error: The following element is not allowed within the container: <div>
- Line 94: Error: The following element is not allowed within the container: <div>
- Line 95: Error: The following element is not allowed within the container: <div>
- Line 96: Error: The following element is not allowed within the container: <div>
- Line 97: Error: The following element is not allowed within the container: <div>
- Line 98: Error: The following element is not allowed within the container: <div>
- Line 99: Error: The following element is not allowed within the container: <div>
- Line 100: Error: The following element is not allowed within the container: <div>

Page report

Summary

Upgrade to **Total Validator Pro** to validate an entire site in one go
 Page checked: C:\Users\Admin\Desktop\dd\addStudentForm.html
 Errors found: 6

Page Layout

Display issue details: Hover Show all Hide all
 Short report

The line numbers refer to lines in the original source. Any with a line number of '0' are implicit tags added by Total Validator:

Go to first issue

- 6 E044 File not found: file://C:/Users/Admin/Desktop/dd/mainStyleSheet
- 8 E044 File not found: file://C:/Users/Admin/Desktop/dd/searchStudentsFrontEnd
- 13 E874 [WCAG21 3.3.2 (A)] The matching <label> text must appear after the control:
- 16 E046 Not a recognisable file:
- 17 E044 File not found: file://C:/Users/Admin/Desktop/dd/addStudent
- 18 E044 File not found: file://C:/Users/Admin/Desktop/dd/searchStudents
- 27 P892 [WCAG21 1.3.1 (A)] Use CSS for presentation effects:

Page report

Summary

Upgrade to **Total Validator Pro** to validate an entire site in one go
 Page checked: C:\Users\Admin\Desktop\dd\searchStudentsForm.html
 Errors found: 8

Page Layout

Display issue details: Hover Show all Hide all
 Short report

The line numbers refer to lines in the original source. Any with a line number of '0' are implicit tags added by Total Validator:

- 6 E044 File not found: file://C:/Users/Admin/Desktop/dd/mainStyleSheet
- 8 E044 File not found: file://C:/Users/Admin/Desktop/dd/searchStudentsFrontEnd
- 13 E874 [WCAG21 3.3.2 (A)] The matching <label> text must appear after the control:
- 16 E046 Not a recognisable file:
- 17 E044 File not found: file://C:/Users/Admin/Desktop/dd/addStudent
- 18 E044 File not found: file://C:/Users/Admin/Desktop/dd/searchStudents
- 27 P892 [WCAG21 1.3.1 (A)] Use CSS for presentation effects:

Page report

Summary

Upgrade to **Total Validator Pro** to validate an entire site in one go

Page checked: C:\Users\Admin\Desktop\dd\searchStudentsForm.html

Errors found: 8

Warnings found: 1

Page Layout

Display issue details: Hover Show all Hide all

Short report

The line numbers refer to lines in the original source. Any with a line number of '0' are implicit tags added by Total Validator:

Go to first issue

- 6 E044 **File not found: file://C:/Users/Admin/Desktop/dd/mainStyleSheet**

The page links to a file that could not be found. Check that the page referred to exists and is in the correct folder, and it, and its parent folders, have sufficient permissions.

 <link rel="stylesheet" type="text/css" href="mainStyleSheet" />
- 8 E044 **File not found: file://C:/Users/Admin/Desktop/dd/searchStudentsFrontEnd**
 <script src="searchStudentsFrontEnd" type="text/javascript">
- 13 E874 [WCAG21 3.3.2 (A)] **The matching <label> text must appear after the control:**
 <input type="checkbox" class="chkMobileNav" id="chkMobileNav" />
- 16 E046 **Not a recognisable file:**

- 17 E044 **File not found: file://C:/Users/Admin/Desktop/dd/addStudent**

- 18 E044 **File not found: file://C:/Users/Admin/Desktop/dd/searchStudents**

- 51 P892 [WCAG21 1.3.1 (A)] **Use CSS for presentation effects:**
 <u>
- 53 W892 [WCAG21 1.3.1 (A)] **Provide an accessible description for complex data tables:**
 E880 [WCAG21 1.3.1 (A)] **For complex data tables explicitly associate data cells and header cells:**
 <table id="formTable" title="studentTable">

Application Testing Features

5)

Test #1

Test Objective: (Client) Check whether there is client-side validation of user input and inline feedback when submitting new student details.

The client validation should include- student ID, first name, and last name not empty and valid format for non-empty fields.

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/addStudent> URL
- Submit form to show required fields
- Fill out non-empty fields with invalid formats
- Submit form to show valid format validation

Expected result: Client feedback that required fields need to be filled out and invalid input format should be displayed with inline feedback

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKKNlqDGE07Erhzrsd>

Test #2

Test Objective: (Client) Check whether client is able to enter 7 pieces of information for a student. These seven should include-student ID, first name, last name, age, gender, degree, and photo

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/addStudent> URL
- Fill out add student form with valid input
- Submit form
- Go to <http://ceto.murdoch.edu.au:40004/searchStudents>
- The submitted student should appear in the search table

Expected result: The 7 pieces of information entered for the student should appear in the search table indicating successful submission

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKKNlqDGE07Erhzrsd>

Test #3

Test Objective: (Client) Check whether feedback of successful submission from server is displayed

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/addStudent> URL
- Fill out add student form with valid input
- Submit form

Expected result: Displayed text in html page indicating successful submission

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKKNlqDGE07Erhzrsd>

Test #4

Test Objective: (Client) Check whether feedback of unsuccessful submission from server is displayed.
Due to student ID entered by client already been assigned to a student

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/addStudent> URL
- Fill out add student form with already assigned student ID
- Submit form

Expected result: Displayed error message indicated unsuccessful submission

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKKNlqDGE07Erhzrsd>

Test #5

Test Objective: (server) Check whether successful submission of student saves the selected photo in subdirectory photos with student ID as filename with extension

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/addStudent> URL
- Fill out add student form
- Submit form
- Go to photos/ sub-directory
- Validate photo with correct file name is in there

Expected result: photo with student id as filename and file extension should be in the sub-directory

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKkNIqDGE07Erhzrsd>

Test #6

Test Objective: (server) Check whether successful submitted student entry with all seven fields non-empty writes to CSV

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/addStudent> URL
- Fill out add student form with no non-empty fields
- Submit form
- Go to data subdirectory
- Validate entry in students.csv

Expected result: a csv line with all seven fields including link to photo to file

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKkNIqDGE07Erhzrsd>

Test #7

Test Objective: (server) Check whether successful submitted student entry with some non-empty fields writes to CSV

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/addStudent> URL
- Fill out add student form with some non-empty fields
- Submit form
- Go to data subdirectory
- Validate entry in students.csv

Expected result: a csv line with some empty fields

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKkNIqDGE07Erhzrsd>

Test #8

Test Objective: (server) Check whether duplicate submitted student entry is not written to CSV

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/addStudent> URL
- Fill out add student form that is exactly the same as another student
- Submit form
- Go to data subdirectory
- Validate entry in students.csv

Expected result: no new csv line is added

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKKNlqDGE07Erhzrsd>

Test #9

Test Objective: (server) Check whether server sends response (feedback) for successful submission feedback

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/addStudent> URL
- Fill out add student form
- Validate with terminal output

Expected result:

1. Calls reqUploadStudent request handler which deals with submissions
2. Calls DisplaySuccessMessage(..) method which writes the response to client

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKKNlqDGE07Erhzrsd>

Test #10

Test Objective: (server) Check whether server sends response (feedback) for unsuccessful submission feedback

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/addStudent> URL
- Fill out add student form that is exactly the same as another student
- Validate with terminal output

Expected result:

1. Calls reqUploadStudent request handler which deals with submissions
2. Calls DisplayFailMessage(..) method which writes the response to client

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKKNlqDGE07Erhzrsd>

Test #11

Test Objective: (Client) Check whether client can click a button to submit search key asynchronously

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/searchStudents> URL
- Fill out search form
- Validate asynchronous call with inspect element console to see client-side calls

Expected result: Calls the processSearchKey() method which executes the asynchronous JQuery \$.post(...) sending search key server side

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKKNlqDGE07Erhzrsd>

Test #12

Test Objective: (Client) Search key case insensitive and display matching students in table with clickable links to photos.

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/searchStudents> URL
- Fill out search form with search key letters different cases
- Submit search key
- Click on photos link

Expected result: Display matching students in table with clickable photo and search key is case insensitive

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKKNlqDGE07Erhzrsd>

Test #13

Test Objective: (Client) Search key can be applied to different combinations of student information fields

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/searchStudents> URL
- Fill out search form with search key letters different combination of fields
- Submit search key

Expected result: Display matching students in table

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKkNIqDGE07Erhzrsd>

Test #14

Test Objective: (Server) Check search key received from client side is used to finding matching(s) student in data file

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/searchStudents> URL
- Fill out search form
- Submit search key
- Validate with terminal output

Expected result:

1. Call reqSearchStudentsSubmit(...) request handler which deals with search submission
2. Call FindMatchesFromSearchKey(...) method which responsible for making sure all the students are in the database before a search is initiated
3. Call Search(...) method which returns all the matching student in the data structure/database

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKkNIqDGE07Erhzrsd>

Test #15

Test Objective: (Server) Server sends matching student to client in JSON format

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/searchStudents> URL
- Fill out search form
- Submit search key
- Validate with terminal output and inspect client-side console

Expected result: Matching students should appear on terminal output and client side console

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKkNIqDGE07Erhzrsd>

Test #16

Test Objective: (Client) Check whether client can click on a link to a photo asynchronously

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/searchStudents> URL
- Fill out search form
- Validate with terminal output

Expected result: Calls the DisplayStudentPhoto() client side method which executes the asynchronous JQuery \$.post(...) sending path to photo server side via reqShowStudentPhoto(...) request handler

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKkNIqDGE07Erhzrsd>

Test #17

Test Objective: (Client) Check whether can display photo from server on the same page

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/searchStudents> URL
- Fill out search form
- Click on photo

Expected result: Photo gets displayed on the same page

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKkNIqDGE07Erhzrsd>

Test #18

Test Objective: (Server) Check whether server can send different format photos to client

Test Steps:

- Go to <http://ceto.murdoch.edu.au:40004/searchStudents> URL
- Click on photo(s)

Expected result: Different photo format gets displayed on the same page

Passed/Failed: Passed

Actual result- <https://www.youtube.com/playlist?list=PLUktbenWQI5juPcBKkNIqDGE07Erhzrsd>

6)

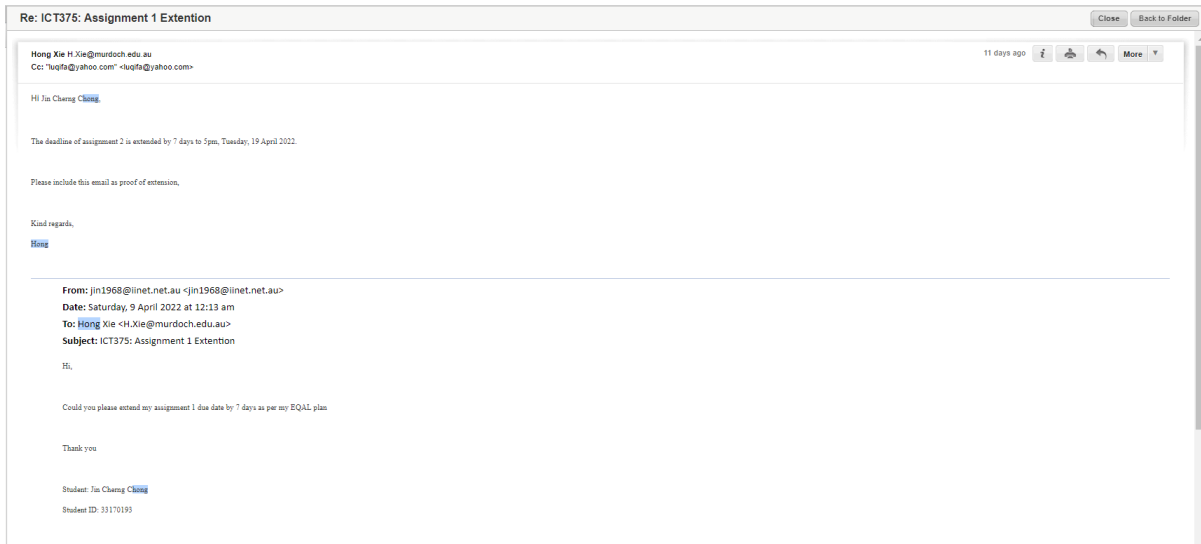
My assigned submission close to all the requirements set out. These include the following-

- Allow the user to enter 7 pieces of information including selecting a photo file
 - Test #2
- Validating user input and provide feedback validation
 - Test #1
- Submit the student details, including photo to the server asynchronously
- Display the feedback of the submission from the server
 - Test #3 and Test #4
- On receipt of the client request with student details save the photo to the subdirectory with correct filename
 - Test #5
- Save the seven fields to file as a single line and no duplication
 - Test #6, Test #7, Test #8
- Send the feedback response to client
 - Test #9 and Test #10
- Allow user to enter a search key and submit search key by clicking button
 - Test #11
- Search key case insensitive and display matching students in table with clickable links to photos.
 - Test #12
- Search key can be applied to different combinations of student information fields
 - Test #13
- Sever gets search key from client side and searches student.csv
 - Test #14

- Sever sends matching students client side in JSON format
 - Test #15
- Client can click on link of photo and send asynchronous request to server
 - Test #16
- Display photo from server on same page
 - Test #17
- Check photos can be uploaded in different formats and displayed
 - Test #18

The only requirement not fulfilled is submit key can be done by pressing enter

7 Day assignment extension granted as per my EQAL plan



Assignment 1 submission

A working version of the assignment must exist on ceto.murdoch.edu.au under your home directory. Also, you must submit an electronic copy via LMS. Both must be available/submitted by the due date and time. N.B, the version on ceto must be identical to the electronic copy submitted on LMS. Your LMS submission must be a single zipped file (including software components and the report), and should be submitted with the filename using the following naming convention:

unit code-assignment number-your last name-the initial of your first name-your student number.

e.g. ICT375-A1-Smith-J-87654321.zip for student John Smith with student number 87654321.

Note: zip compression only (no WinRar or 7z, etc.).

Submission status

Submission status	No attempt
Grading status	Not graded
Due date	Friday, 15 April 2022, 5:00 PM
Time remaining	Assignment is overdue by: 6 days 13 hours
Last modified	-
Submission comments	Comments (0)