WEBD6201 – Client-Side Scripting

Lab 2
jQuery & Form Validation

Due: Week #8 (Friday March 6, 2020) @ midnight.

Value: 7.5%

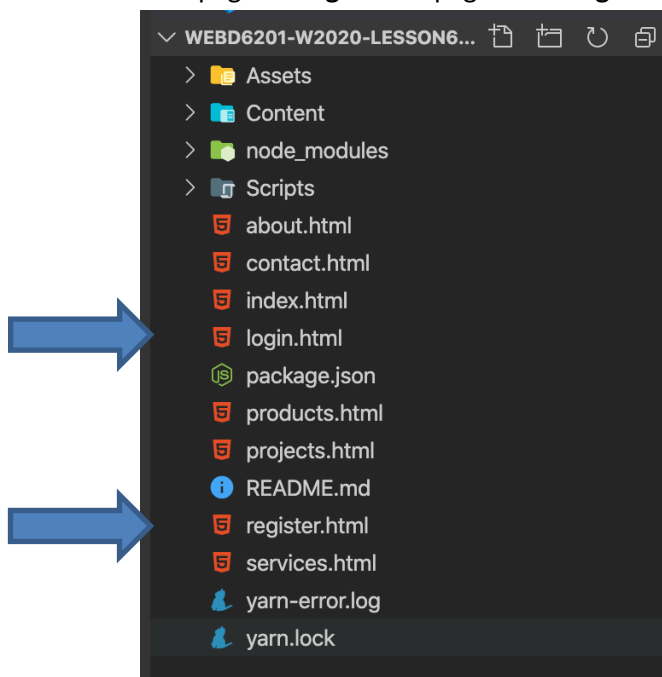jQuery & Form Validation   **Maximum Mark: 37**

Overview: This lab was made to practice and reinforce the use of the jQuery JavaScript library as well as JavaScript form Validation.
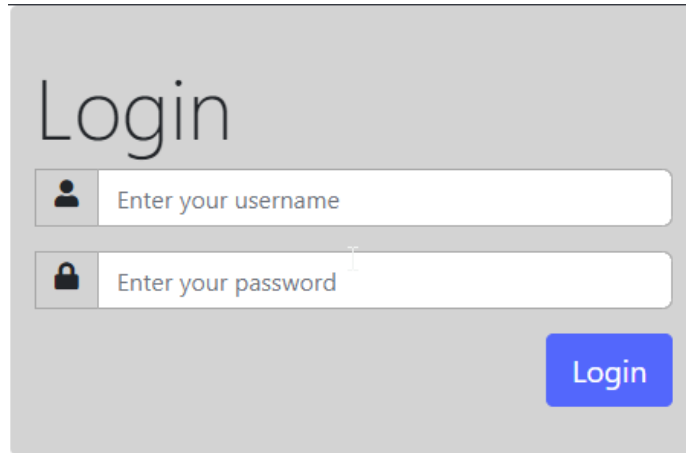
Instructions :
(3 Marks: Site Structure, 26 Marks: Functionality, 4 Marks: Internal Documentation, 4 Marks: Version Control)
1. **Project Setup (3 Marks Site Structure)**
    a. For the most part, your folder structure should follow the same structure shown in class. You may use WEBD6201-W2020-Lesson6 GitHub repo as a starter. This will include two new pages: a **login.html** page and a **register.html** page (1 Marks: Site Structure)
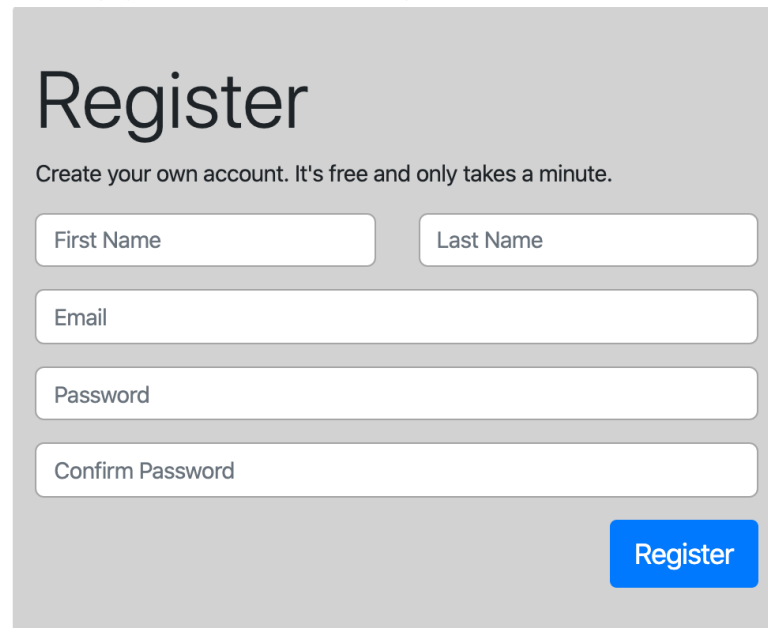
b. The **login.html page** must include **username** and **password** input elements. Use the following image as a guide to create the form (if needed):(1 Marks: Site Structure)

c. The **register.html** will include **First Name**, **Last Name**, **Email**, **Password** and **Confirm Password** input elements. Use the following image as a guide to create the form (if needed): (1 Marks: Site Structure)

2. **Form Validation with jQuery (26 Marks: Functionality)**
   a. **Login Page.** Using **only JavaScript and jQuery**, when the user enters a username and password and then clicks the Login button, insert their username between the **Contact Us** link and the **Login/Logout** link. Ensure you allow for appropriate spacing between the two other list items (Hint: take a look at the **navbar-text** class on the bootstrap website for appropriate styling) (2 Marks: Functionality).
   b. **Register page.** Using **only JavaScript and jQuery**, create a **div** element with an **id** of "ErrorMessage". This **div** element should be **hidden** when the user first navigates to the **register.html** page. This area will be used to **display errors** if the user enters invalid data in the input fields of the registerForm. When the error clears, this div element should be hidden (4 Marks: Functionality).
   c. **Register Page.** Using **only JavaScript and jQuery**, ensure when the user enters their First Name and Last Name that the **minimum length** that each input element will allow is 2 characters. If the user violates this check, display an error message in the div element with an id of "ErrorMessage" (4 Marks: Functionality).
   d. **Register Page.** Using **only JavaScript and jQuery**, when the user enters their email address ensure that the **minimum length** is 8 and that an **@ symbol** is present. If the user violates this check, display an error message in the div element with an id of "ErrorMessage" (4 Marks: Functionality).
   e. **Register Page.** Using **only JavaScript and jQuery**, ensure that when the user selects a password and enters it in both the **Password** and **Confirm password** input elements that it is the same string. Also, passwords should be at least **6 characters in length** (4 Marks: Functionality)
   f. **Register Page.** Using **only JavaScript and jQuery**, when the user clicks on the Register button prevent the default form behaviour (which submits the form). (2 Marks: Functionality).
   g. **User Class.** Create a JavaScript Class named **User**, in the same file (app.js) but above the IIFE that includes firstName, lastName, username, email and password properties. (2 Marks: Functionality)
   h. **Register Page.** Using **only JavaScript and jQuery**, when the Register button is clicked, create an instance of the **User** class and display it in the console. Finally, clear the form. (4 Marks: Functionality).

3. Include **Internal Documentation** for your program **(4 Marks: Internal Documentation):**
   a. Ensure you include a comment header at the top of your app.js file that indicates: Your **Full Name**, **StudentID** and **Date Completed** (2 Marks: Internal Documentation).
   b. Ensure your program uses contextual variable names that help make your code human-readable (1 Marks: Internal Documentation).
   c. Ensure you include inline comments as required. As a rule the code should be descriptive but sometimes some information is required, especially before any function you include (1 Marks: Internal Documentation)

4. Share your files on **GitHub** to demonstrate Version Control Best Practices **(4 Marks: Version Control).**
   a. Your repository must include **your code** and be well structured (2 Marks: Version Control).
   b. Your repository must include **commits** that demonstrates the project being updated at different stages of development – each time a major change is implemented (2 Marks: Version Control).

Evaluation Criteria

Feature	Description	Marks
Site Structure	Your Project adheres to the site structure described (including Assets, Content and Scripts folders)	3
Functionality	The program's deliverables are all met and the program functions as it should. No errors appear as a result of execution. User Input does not crash the program.	26
Internal Documentation	A comment header is present and includes the name of the student, studentID, and date completed.	4
Version Control	GitHub commit history demonstrating regular updates.	4
Total		**37**

SUBMITTING YOUR WORK

Your submission should include:
1. A zip archive of your project uploaded to DCConnect
2. A link to your complete project files on GitHub

Please zip all files in to a single project archive.

This assignment is weighted **7.5%** of your total mark for this course.

Late submissions:
- 20% deducted for each additional day.

External code (e.g. from the internet or other sources) can be used for student submissions within the following parameters:
1. The code source (i.e. where you got the code and who wrote it) must be cited in your internal documentation.
2. It encompasses a maximum of 10% of your code (any more will be considered cheating).
3. You must understand any code you use and include documentation (comments) around the code that explains its function.
4. You must get written approval from me via email.