

CSCI 4131 – Internet Programming
Spring Semester 2020
Homework Assignment 2 – Version 2, Updated February 3rd

Due Date: Friday February 7th at 2:30pm
Late Submission Date (with Penalty): Saturday, February 8th at 2:30pm
No submissions are accepted after the late submission date.

1. Description

The objective of this assignment is to expose you to client-side JavaScript, CSS, the Document Object Model, and jQuery. You will refactor and build upon your first homework assignment to accomplish this. Specifically, when you complete this your website will consist of the following 3 pages:

1. A Contacts Page (You should add your own contact(s), and their info/picture)
2. A Form Input Page
3. A Widgets Page

This assignment description is 10 pages long.

The following tasks are required by this assignment

- i.) The navigation bar should be styled and updated to enable navigation between all three pages of your updated website.
- ii.) Update your Contacts page (*MyContacts.html*) as follows:
A thumbnail image of the contact and a pop-up image of the contact should be displayed when your mouse hovers over the corresponding row in the contacts table. *Use your best judgment in selecting the pictures and do not use copyrighted material. Note, you can use pictures you create (take).* The larger image should be displayed to the center right of the contacts, and the contacts and image should be displayed on one page without a scrollbar.
- iii.) Create a new page named *MyForm.html* consisting of a contact input form that will be used to add contacts to your contacts table in later assignments.
 - a. The form should use HTML5 input elements and/or JavaScript to perform basic validation on the form's input files from the contacts.
 - b. Upon successfully filled out data into the form, the form can be successfully submitted (i.e., submitted without displaying an error) to a URL provided which displays the data in the form.
- iv.) Update your widgets page (*MyWidget.html*) as follows
 - a. Add a Twitter **Stream** (that should replace the tweet on your page from assignment 1).
 - b. Add an Instagram widget
 - c. Add a Password Strength validator written solely in JavaScript (it cannot use jQuery) that behaves the same as the jQuery example we provide.
 - d. Organize the widgets side by side.

- v.) Style all your webpages with CSS. Specifically, create and use a single external style file, named **style.css** to style the pages on your Website (with one exception specified in the Constraints section on page 9) . Your webpages should not contain any inline or embedded styling. Your styling should be accomplished using CSS commands (i.e., CSS code) that are specified in your **style.css** file.

2. Required Functionality

Updates to the Contacts Page

Once the Contacts page is loaded into the browser, it should display the contacts' information and the image of the contact as shown in the screenshots below.

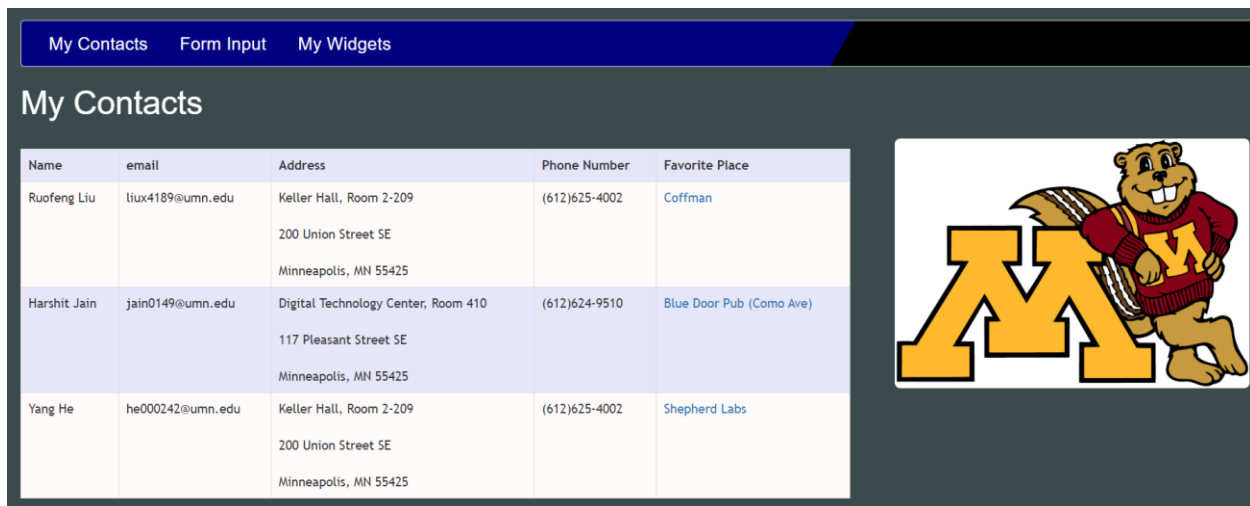


Figure 1. Contact information displayed when page is initially displayed

- The rows of your contacts table should be displayed in alternating colors in a manner similar as displayed in figure 1 below. Note, you can choose the colors, they don't have to be the same as in figure 1, but they should **enhance** the readability of your contacts.
- The placeholder image shown (provided with the assignment) should be displayed as pictured next to your contacts table. The table and picture should fit on a single web-page without scroll-bars.
- When you hover over a row in the contacts table with a pointing device (e.g., a mouse or your finger on a touch screen), a small (thumbnail) image of the corresponding contact should be displayed and the larger version of the image should be displayed next to the table. The larger image should persist even on mouse-out.

Specifically, as depicted in Figure 2, a small thumbnail image is displayed under the *name* cell, and a larger version of the image is displayed next to the table. If the pointing device is moved to another row on the contacts table, a thumbnail and large image of that contact should be displayed. If the mouse moves off the table, the last large image displayed should persist whereas the thumbnail image should disappear.



Figure 2. Contacts page as displayed when pointing device hovers over a row of contact information on contacts table

Add a Form Page:

Using HTML, CSS and JavaScript as necessary, create and add a Web page with a form for adding new contacts to your contacts table (we will not implement its functionality in this assignment).

The form should use HTML-5 input elements wherever possible to accept the following **mandatory** input:

- Name;
- Email;
- Address;
- Favorite Place, and
- A uniform resource locator (URL) specifying information about the contact.

Name	<input type="text"/>
Email	<input type="text"/>
Address	<input type="text"/>
Favorite Place	<input type="text"/>
Enter URL for the Favorite Place	<input type="text"/>
	<input type="submit" value="Submit"/>

Figure 4 an example what your form might look like (the final styling is up to you).

On the form page, you must ensure all data entered on the form is syntactically legal input (using HTML-5 elements and/or JavaScript and regular Expressions). Specifically,

- i) All the fields are required to be filled.
Upon the submission, an error message should be displayed if any field is left blank.

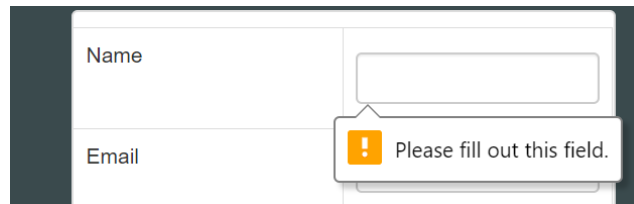


Figure 5: Example error message displayed if there is an unfilled field.

- ii) The name and favorite Place should accept only alphanumeric characters. Acceptable Alphanumeric characters are specified in the following ranges: number: 0-9, lower case characters: a-z, upper-case characters: A-Z and spaces.

Upon the submission, an error message should be displayed if non-alphanumeric characters are included in the name and/or favorite place fields.

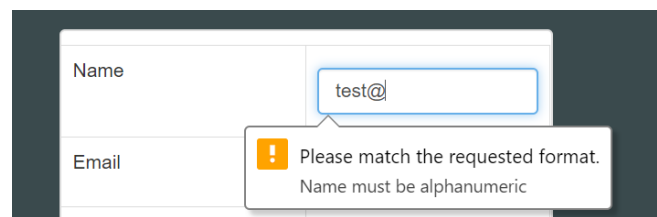


Figure 6: Example error message displayed if there is an error in the contact name or favorite place input to the form.

- iii) The email should be syntactically validated as to ensure it is formed as a legal email address (it may not really exist, but the form should be correct). As shown in Figure 7 below, an error should be displayed if a syntactically incorrect email address is entered.

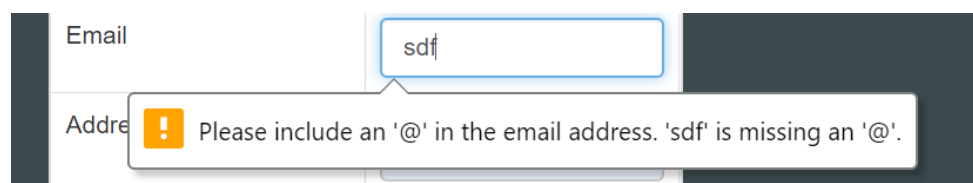
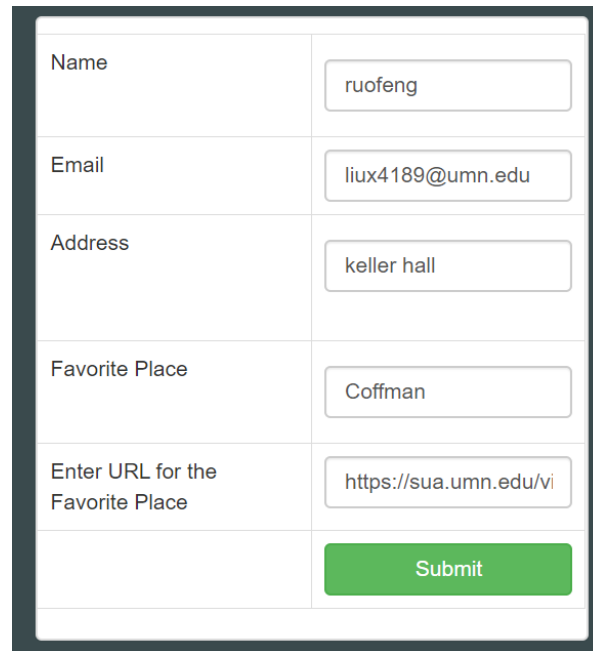


Figure 7: An error should be displayed to the user if they enter syntactically incorrect URL

- iv) Finally, the URL should be syntactically validated to ensure it is formed as a syntactically legal URL (it may not really exist, but the form should be correct). An error should be displayed if a syntactically incorrect URL is entered.

Once the form is filled out with legal input (as pictured in Figure 8 below) and submitted (by left-clicking when your mouse is on the submit button), a HTML page with a table of your input data should be displayed, indicating that the form was successfully submitted (as pictured in Figure 9 below).



Name	ruofeng
Email	liux4189@umn.edu
Address	keller hall
Favorite Place	Coffman
Enter URL for the Favorite Place	https://sua.umn.edu/vi
<input type="submit" value="Submit"/>	

Figure 8: A form that has been successfully filled out by the user – it can be submitted without errors, so the action specified in the form action attribute can be executed

In specific, the form is submitted to the following server script we created with URL <http://www-users.cselabs.umn.edu/~liux4189/echo.php>. To achieve this, the action attribute on your form should specify the URL of the server script (above) - see https://www.w3schools.com/tags/att_form_action.asp for a discussion on how to accomplish this. **Note - you should not develop or deploy the server script for this assignment, use the one we have provided.**

The id field of each input items in the form must be as specified in the table below:

Input Field	Id
Name	name
Email	email
Address	address
Favorite Place	place
URL	URL

Double check that the information in the returned html tables are the same as the input data.

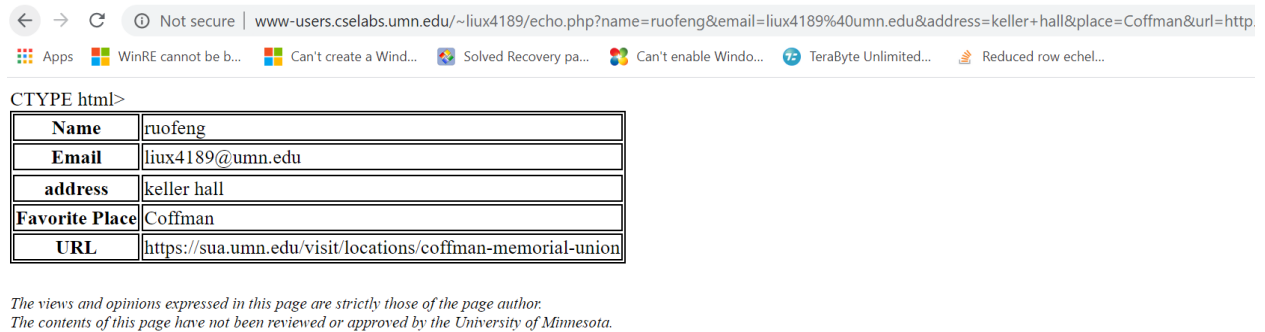


Figure 9: Web page that is displayed upon successful submission of the form.

Updates to the Widgets Page

1. **First**, replace the Twitter Tweet with a Twitter Timeline Stream widget on the **My Widgets** Page, as pictured below in Figure 10. See the link <https://developer.twitter.com/en/docs/twitter-for-websites/timelines/overview.html> to get details about how to accomplish this.

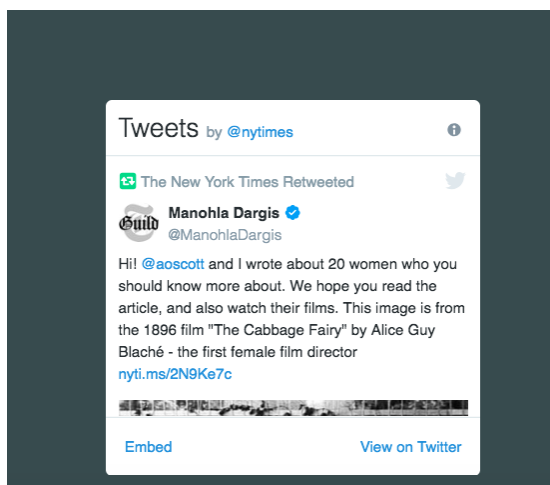


Figure 10: A twitter stream – this should replace the tweet from assignment 1

2. **Second**, Embed an Instagram widget as pictured in Figure 11 below onto the My Widgets page using following link as reference.
<https://www.instagram.com/developer/embedding/>
Note – you should add “http” to the source link you obtain from Instagram (see: <https://stackoverflow.com/questions/37322148/why-is-instagram-embed-code-only-showing-an-instagram-icon-not-t>)

[he-image](#). for an explanation.

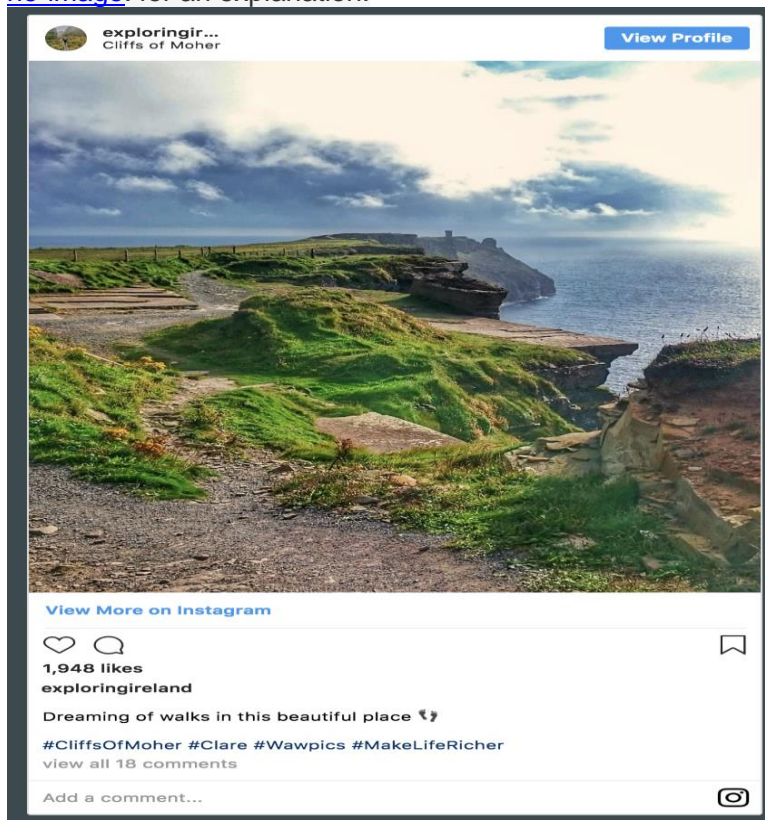


Figure 11: An Instagram widget should be added to your widgets page

3. **Third**, add a password strength check Widget on the My Widgets page. This JavaScript widget will check the strength of a password entered by the user. Use the jQuery example given in the link to help you write a plain JavaScript code without using the jQuery library (in other words, you must not use jQuery to implement this widget).

Your JavaScript code should display the strength of the password as you type it using a progress bar layout as illustrated in Figures 12 and 13 below.

<https://www.formget.com/password-strength-checker-in-jquery/>

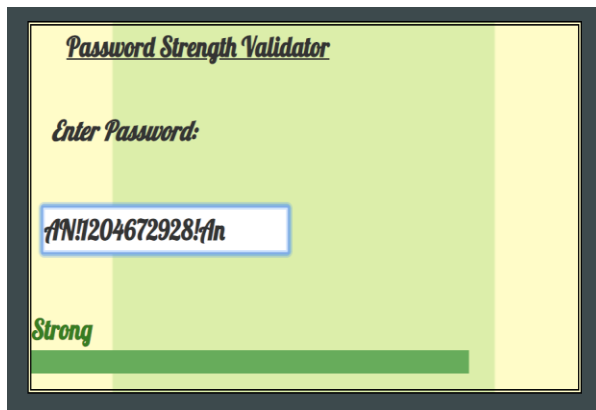


Figure 12: Example of the password checking widget indicating a strong password

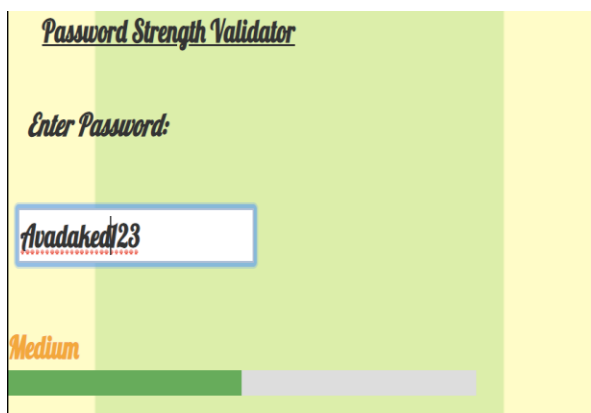


Figure 13: Example of the password checking widget indicating a medium strength password

Finally, you should style your widgets page, so the widgets are displayed side by side and displayed toward the center of the page. Scroll-bars are permitted on this page as indicated in Figure 13 below:

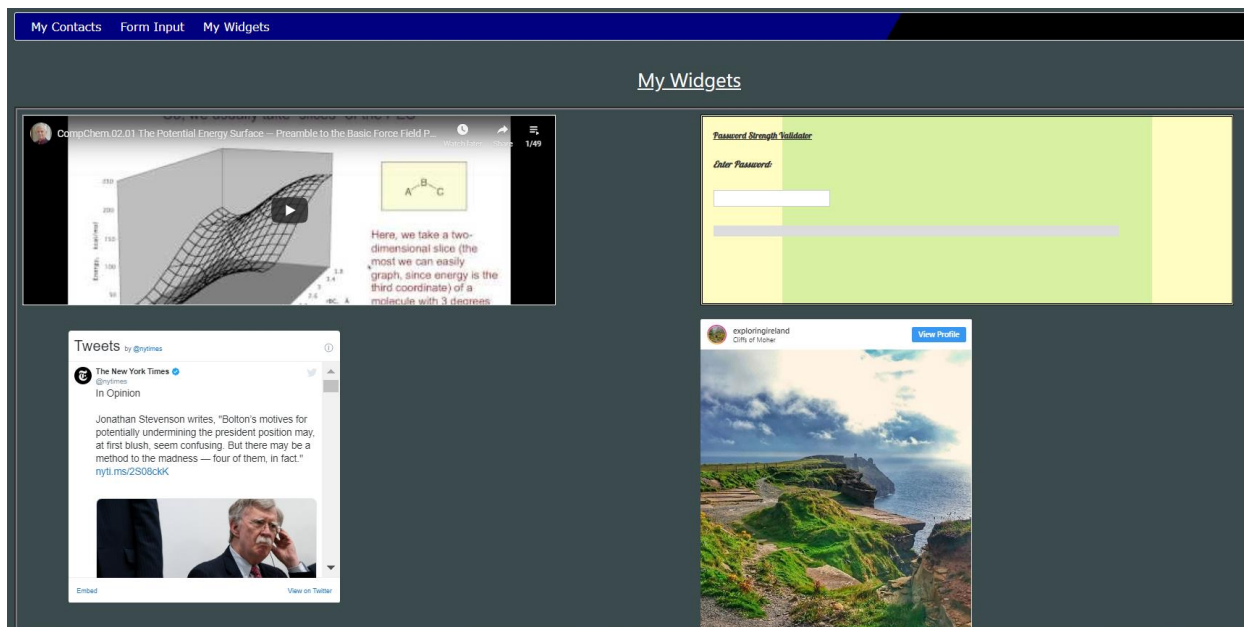


Figure 14: Widgets as displayed on the widgets page

Hint: Side-by-side organization of the widgets can be achieved with grid layout.

https://www.w3schools.com/css/css_grid.asp

Constraints:

You must develop and use your own CSS styling for all items that are displayed. Do not use any style sheet (file) other than the one you have not developed yourself – with the exception of the style sheet that accompanies the password checking widget. Also, do not use embedded or inline styling in your final deliverable.

Violation of the Constraints is a 15 point penalty.

Grading Criteria

- a. The navigation bar should be styled and updated to enable navigation between all three pages of your updated website. **(5 points)**
- b. Update your Contacts page as follows:

A thumbnail image of the contact should be displayed when your mouse hovers over the corresponding row in the contact table. **(5 points)**

A larger pop-up image of the thumbnail image should be displayed to the center right of the contacts table, and the table and image should be displayed on one page without a scrollbar. **(15 points)**

When the mouse moves off the table, the last large image displayed should persist whereas the thumbnail image should disappear. **(5 points)**

- c. Create a new page named MyForm.html consisting of a contact input form that will be used to add contact in later assignments. **(5 points)**
 - a. The form should use HTML5 input elements and/or JavaScript to perform basic validation on the form's input files from the contacts as specified in the required functionality discussion above **(10 points)**
 - b. Upon successfully entering data into the form, the form can be successfully submitted (i.e., submitted without displaying an error) to the URL provided. The returned page should display a table displayed the input information in the form. **(10 points)**
- d. Update your widgets page according to the requirements discussed previously as follows
 - a. Add a Twitter **Stream** (that should replace the tweet on your page from assignment 1). **(5 points)**
 - b. Add an Instagram widget. **(5 points)**
 - c. Add a Password Strength validator written solely in JavaScript (it cannot use jQuery) that behaves the same as the jQuery example we provide **(15 points)**
 - d. The widgets are displayed side-by-side. **(5 points)**
- e. Style all your webpages with CSS. Specifically, create and use a single external style file, entitled **style.css** to style the pages on your Website. Your webpages should not contain any inline or embedded styling. Your styling should be accomplished using CSS commands (i.e., CSS code) that are specified in your **style.css** file. **(10 points)**

- f. With the exception of the embedded widgets on the updated widget page, HTML and CSS files pass w3schools validators (<http://validator.w3.org> and <https://jigsaw.w3.org/css-validator>) without errors. Warnings are accepted. (5 points)

SUBMISSION INSTRUCTIONS:

Your submission should be packaged in a tar file or zip file. When opened, it must create a directory titled '<UMN internet ID>' containing all of your files (HTML, CSS, Pictures, and additional external JavaScript file (If used). You will lose 10 points if you do not do this correctly.

Submit your homework via the class Canvas item with the Homework 2 Submission Link.

Additional:

We have provided some images you can use with the assignment in the images folder with the assignment on Canvas. You are responsible for legally obtaining any other images you use.