# CPSC 2030 – Web Development II

**Lab7: [40 marks] Form Validation**

|  |
| --- |
| **Objectives** |
| * Do basic form validation on the client-side * Implement form validation on the server-side through PHP |
| **Preparation** |
| 1. Review [jQuery](https://www.w3schools.com/jquery/). |
| **What to do** |

**Setup:**

* Download Lab7StarterFiles.zip from D2L and unzip it.
* Copy the folder lab7 to C://xampp/htdocs

**Task 1: [10 marks] Client-Side Validation.**

* Examine the form set up in **lab7.php** to see how it works.
* The following validation should be done:
  + The email address is required and should have a valid email format.
  + Exactly three checkboxes should be checked.
  + The date is required and should be formatted as in the label text. (for the client-side validation, it's okay if the form accepts slightly different date formats (like yyyy-mm-dd instead of yyyy/mm/dd.)
* A folder called **jquery-validation-1.19.3** is providedwith the necessary files to get the client-side validation to work. The necessary scripts are imported in the head of the document. If you would like to download the files yourself, you can do so here: <https://github.com/jquery-validation/jquery-validation/releases/tag/1.19.3>
* Some starter code is written in the script block. The code sets up some validation error messages and a few other little things. Your job is to fill in the **rules** object, which defines how the form fields will be checked and validated. A few resources to help you do this:
  + This is a pretty good tutorial on how to use the plugin. Start at Step 5: <https://www.sitepoint.com/basic-jquery-form-validation-tutorial/#step5createthevalidationrules>
  + From the official documentation:
    - How **rules** works: <https://jqueryvalidation.org/validate/#rules>
    - How to handle form fields with complex names (like array fields for checkbox groups) <https://jqueryvalidation.org/reference/#link-fields-with-complex-names-(brackets-dots)>
    - A few validation methods:
      * required: <https://jqueryvalidation.org/required-method/>
      * maxlength: <https://jqueryvalidation.org/maxlength-method/>
      * minlength: <https://jqueryvalidation.org/minlength-method/>
      * dateISO: <https://jqueryvalidation.org/dateISO-method/>
      * email: <https://jqueryvalidation.org/email-method/>

**Task 2: [30 marks] Server-Side Validation**

* Since the **jQuery Validation** prevents a form from being sent if it has validation errors, we need to temporarily disable it while we build and test our PHP validation functionality; temporarily comment out the necessary JavaScript code.
* The file **validation.php** is imported into **lab7.php** by using a **require\_once()** function. This file has some starter code that you can use to structure your app. Take a look at the PHP functions that are called from **lab7.php** to get an idea of how the server-side validation will be done.  
    
  If you want, you can disregard the starter code provided and write your own code from scratch. The only requirement is that there is some separation between the logic and presentation. However, it is recommend sticking to this template since it will probably be a bit easier.  
  + **validation.php** contains two global variable declarations:
    - **$valid:** this variable will be set to true or false depending on whether the submitted form contains valid input or not.
    - **$val\_messages:** this array should contain a validation message for each field in the form. If a field is valid, the message should simply be "", and if it's invalid, it should contain a message string explaining the error.
  + **validation.php** also contains three function declarations:
    - **validate():** this function takes no parameters, but checks to make sure the user has submitted the form with data that is in the proper format. If all the inputs are valid, the **$valid** variable should be set to **true**.
      * **Email:** to make sure a valid address has been entered, you can use the regular expression pattern provided in the comments of the file. (Or come up with your own pattern.) The function used for checking regular expressions in PHP is **preg\_match**($pattern, $submission) which will return **1** for a match and **0** for no match.
      * **Date:** A regular expression pattern is provided for this.
      * **Animals:** for this checkbox group, you'll have to count the number of boxes that have been checked. It must be exactly three animals.
    - **the\_results():** If the form is all valid, this function outputs the data entered by the user. If the form contains any *invalid* info, this function should output nothing.
    - **the\_validation\_message($type):** This function should display the validation message for the given type of input. If the input is valid, it should display nothing. If the input is invalid, it should display a descriptive and helpful error message. The class "failure-message" should be applied to the output HTML in order for the correct CSS styling to be applied. Notice that this function is called in **lab7.php** wherever a validation message should be output.

|  |
| --- |
| **What to hand in** |
| * Zip your lab7 folder under htdocs and upload it to D2L. |
| **When to hand in** |
| * By 11:59pm, Sunday, October 30, 2022 |