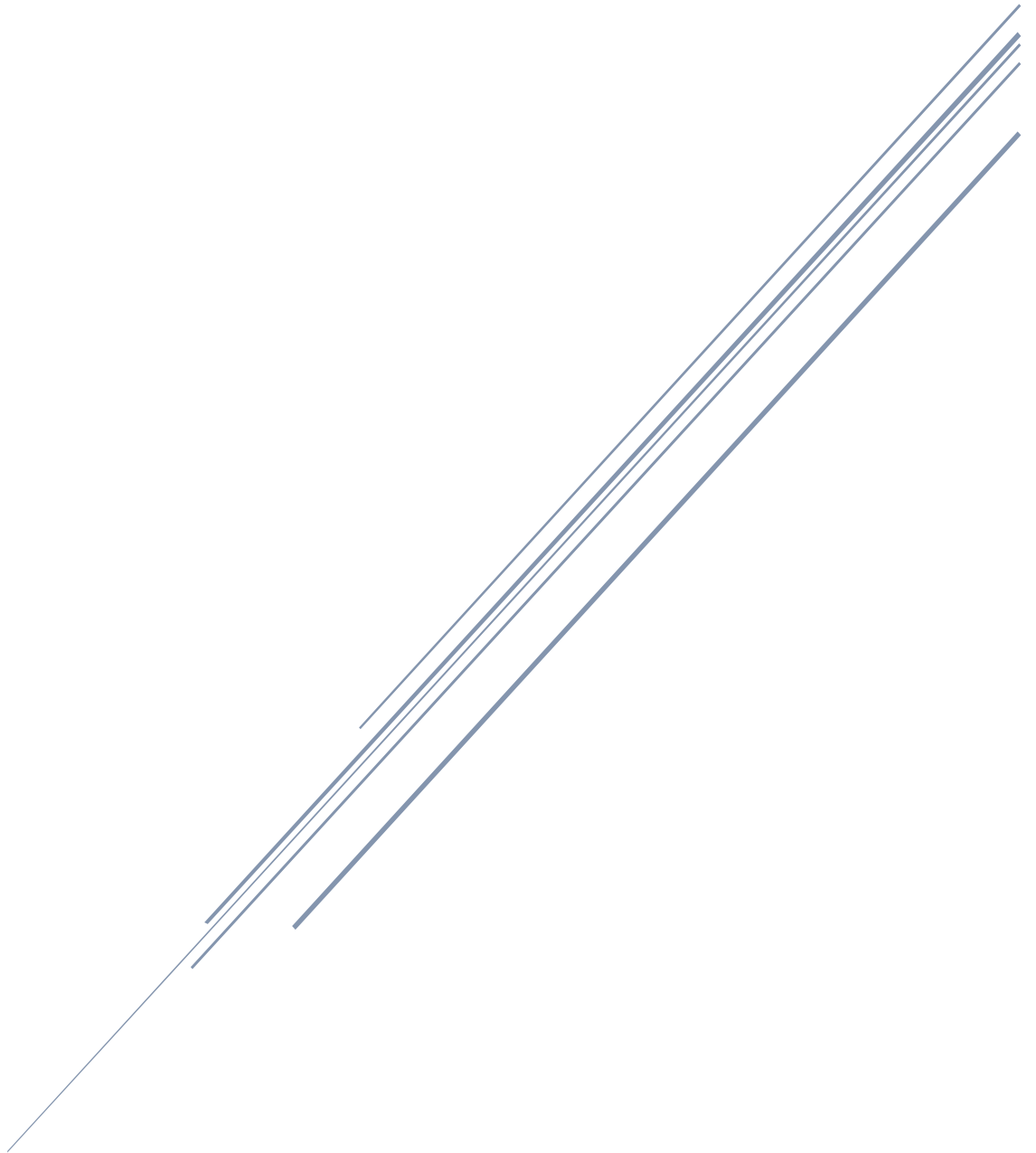


LAB#4WEB PROGRAMMING

CST8285 W2025



LAB OBJECTIVE

This lab will introduce you to the useful functionality of JavaScript.

The objective of this lab is to get familiar with the following:

- 1- Using client-side form validation using DOM manipulation.
- 2- Tracing submitted form data using a GET header (for testing purposes).
- 3- Understand how the client-side form validation works

Learning Resources

Lecture Slides and resources (week 5 - week 9).

Earning

To earn your mark for this lab, each student should finish the lab's requirements, submit your lab on the Brightspace and demonstrate the working code to the instructor.

Description

Weekly Kitten Pictures has hired you to create the functionality for their website idea. They would like to see how you do with form validation. They have provided you a simple sign-up form which they would like you to customize. You are tasked to add validation with JavaScript to ensure a user cannot submit the form without filling the requirements.

Requirements

Create a "*script.js*" file within a folder named "js" which should be found in the same location as the "*index.html*" file. Inside the script.js file, create a function called "validate". The HTML form is set to execute that specific function when the form is submitted. You will perform client-side verification according to the following criteria:

1. Email textbox value is a valid email structure (xyx@xyz.xyz).
2. Login name should be non-empty (do not use a required tag in

- HTML) and less than 30 characters long. When you send this data (on successful validation) convert the login name to all lower-case alphabetic characters (you will confirm this in the get header).
3. Password should be at least 8 characters long.
 4. Ensure that **both** the password fields have the same value and are not blank.
 5. If the user selects to receive a newsletter, immediately alert them about possible spam by setting an event on this field.
 6. Ensure that the terms and conditions are accepted

The form submission should only be successful if the listed requirements are all met. On a successful submission the form should call itself clearing its contents and providing the submitted data in the URI. You should verify that all of the data sent is in the appropriate format. Your HTML form will contain no standard HTML form validation attributes – **all validation will be handled through events tied to JavaScript functions.**

When the user attempts to submit the form then you will provide an inline (either under or beside the incorrect fields) feedback message and you will highlight the fields that are in error. You should use a colour that stands out when it appears as shown in **figure 2**. Do not use a pop-up alert message for displaying form feedback.

When a user selects to receive a newsletter, the immediate response should be in an **alert pop-up** which they have to dismiss to continue with the form. (Note that pop-ups are often disabled by users, so it is not a reliable way to do form feedback, but we do sometimes use them for informational purposes like this case.)

When a user enters valid data in a particular field, the **error message associated with that field should be cleared, providing immediate feedback without affecting other non-valid data fields** i.e. When a user corrects an error in a specific field and the input becomes valid, dynamically clear the error message associated with that field. Importantly, ensure that error messages in other fields, which may still contain non-valid data, remain visible until corrected.

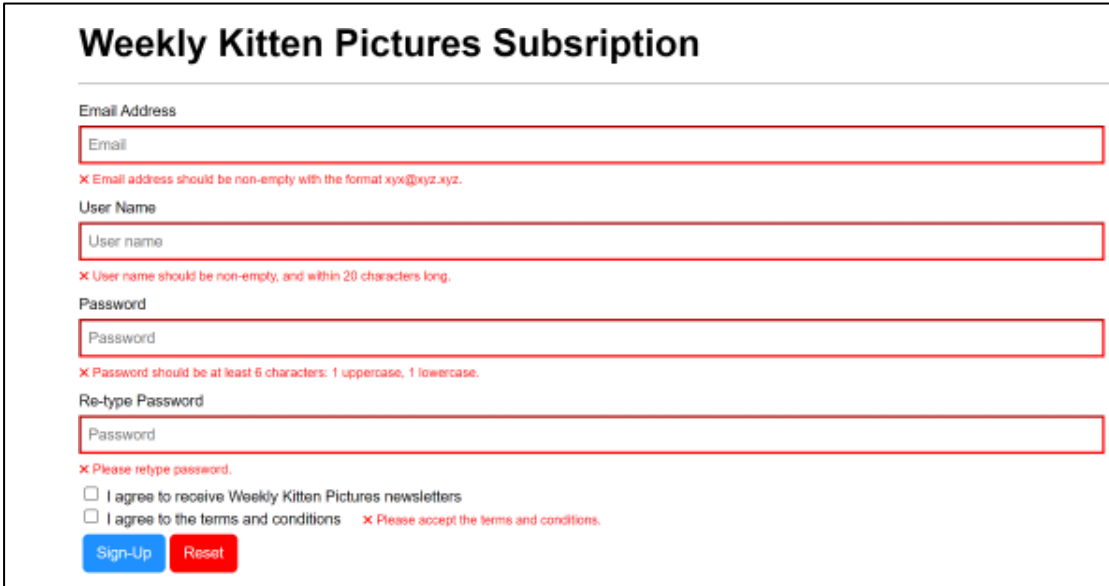
Create a CSS file and format the form to look like the form below:



The screenshot shows a web browser window with the title 'Lab 5 - JavaScript 1'. The address bar shows the file path 'D:/Dropbox/Courses/Algonquin%20College/CST8285%20Web%20Dev/registration.html'. The form is titled 'Weekly Kitten Pictures Subsription' (note the typo). It contains the following fields and elements:

- Email Address: Input field with placeholder 'Email'.
- User Name: Input field with placeholder 'User name'.
- Password: Input field with placeholder 'Password'.
- Re-type Password: Input field with placeholder 'Password'.
- Two checkboxes: 'I agree to receive Weekly Kitten Pictures newsletters' and 'I agree to the terms and conditions'.
- Two buttons: 'Sign-Up' (blue) and 'Reset' (red).

Figure 1.



This screenshot shows the same registration form as Figure 1, but with red error messages displayed below each input field:

- Below Email: **✖ Email address should be non-empty with the format xyz@xyz.xyz.**
- Below User Name: **✖ User name should be non-empty, and within 20 characters long.**
- Below Password: **✖ Password should be at least 6 characters: 1 uppercase, 1 lowercase.**
- Below Re-type Password: **✖ Please retype password.**
- Below the checkboxes: **✖ Please accept the terms and conditions.**

Figure 2.

Other Important Requirements

- Demo and justify your work and answer your lab professor's questions.
- The work will be graded zero if you do not demo it on time, even if uploaded.
- Explain in a simple Plain English terms and diagram to show how the HTML interact with javascripts and perform expected tasks such as producing error messages just under the text box using red colors (25% of the marks). Use a software tool (Ex. MS word) to create the

diagram. See Figure 2 for reference

Submission

To submit the lab on the Brightspace, you must upload “registration.html” file, your supporting CSS and JavaScript files, and a validation screenshot image in a .zip file.

The zipped file must be named the following: <First Name>_<Last Name>.zip
Note that <First Name> and <Last Name> should be replaced with your first and last name.

Grading Criteria:

- Documentation, Indentation (25%)
- Diagram that shows at least 4 interactions(errors) for email, user name, password, retype password & one error free case (25%)
- Validate () function (40%)
- Code running without any errors (10%)