**Unit 4 Learning Dairy**

I integrated a small client‑side validation script into my contact form to support my Unit 1 personas (recruiter, collaborator contacting me). The script from the geeksforgeeks tutorial (<https://www.geeksforgeeks.org/javascript/form-validation-using-javascript/>) validates several inputs with regular expressions and show inline error messages.

**What the code does:**

* It grabs the form by ID (myForm) and attaches a submit listener. On submit it calls e.preventDefault() so the browser doesn’t post the form until validation succeeds. I wrapped the code around DOMContentLoaded listener to make sure it runs only after the full HTML document has loaded.
* It then reads seven input values from the DOM (first and last name, username, email, password, phone, and date of birth) and stores them in local constants so they can be validated. Before validating, it resets the text content of all <span> error placeholders so old error messages are removed.
* For each field, it defines a regex pattern or rule and tests the user’s value against the pattern.
* If any test fails, it writes a friendly message into the corresponding -error span and flips isValid to false.
* If no rules fail (isValid remains true), it shows a success alert() (but still prevents submission in the original snippet—there’s no form.submit() call).

**What’s good:**

* The code uses dedicated <span> elements to display error messages instead of intrusive alerts().
* Each field report its own reason for error making it easier for the user to figure out what caused the error.
* The code is clearly written, properly indented and follows a single naming convention throughout.

**What can be Improved:**

* Adding a DOMContentLoaded listener to make sure the code only runs after the full html document is loaded so form is not null.
* The code could benefit from some inline comments in addition to the tutorial notes.
* Ensuring the form is submitted once all validations are completed.

**My Changes:**

* Wrap the code in DOMContentLoaded listener so form is never null.
* I removed all the fields that were not related to my page and repurpose the first name validation to the name validation. Updated the regex to also allow special characters.
* Added the message validation regex to ensure the message is never null.
* Submitted the form once all validations where completed.
* Styled the error message in my style.css file.

**Unit 4 Reflection**

In Unit 4, I focused on enhancing the functionality and usability of my portfolio website by incorporating JavaScript form validation. I began by searching the internet for ideas and decided to implement form validation because it ensures that recruiters or potential clients submit complete and accurate information when contacting me.

To implement this feature, I adapted code from GeeksforGeeks(<https://www.geeksforgeeks.org/javascript/form-validation-using-javascript/>) and customized it to fit my project.

While working on this I used w3schools to refresh my knowledge on JS.