1. Tell us which pages we should look at for your form and response, including the version that doesn't have client-side validation.
    a. The form with client-side and server-side validation is connect_Client_Validation.php
    b. The form with only server-side validation is connect.php
    c. The form pages update once the inputs have been updated to display the user input and give a "thanks for submitting" message (this is my response page)
2. List the requirements that you've implemented on the client and the server side to validate your form. If there are differences between the client and server side, justify them. (1)
    a. Client-Side Validation
        i. I used the HTML5 form validation to:
            1. Require that every field is filled out
            2. Check that the email is a valid email
            3. Check that a phone number is valid
            4. Check that the Organization URL is a valid URL
    b. Server-Side Validation
        i. I used PHP form validation to:
            1. Require that every field is filled out
            2. Check that each field is valid
                a. Name is only letters and white space
                b. Email is in the format: name@email.com
                c. Phone number is 10 digits
                d. Check that the Organization URL is a valid URL
                e. Message is less than 500 characters
    c. I split up the validation in this way because if a browser does not support the client-side validation, then there needs to be a check for every case on the server-side. That is why the server-side is inclusive of every input case. I tried to also add all of the input validation techniques to the client-side to improve the user experience when providing an error message. On the server side, I decided not to add the 500 message character limit, because there is not easy way to do that using HTML5 form validation (will need to use javascript to do this on the client-side)
3. How does your design present errors to the user and why is this a good choice that makes it easy and clear for your user. (2)
    a. Client-Side Validation
        i. This validation sequence presents the errors to the user through the native html5 error rending of the browser (usually a dialog box under the input field with the error displayed)
    b. Server-Side Validation
        i. This validation sequence finds all of the mistakes in the input and then posts an error message under each of these fields once the form has been submitted. The error messages are in sync with the design of my website, so they are recognizable. I made the errors displayed below the input field, so improve the visibility of the error (and also the user can deduce where the error is).
    c. When I made my design choices, I made sure to have the client-side validation catch the errors first. This way, the user experience is improved with the native HTML5 errors

that the browser renders. Once the form passed these errors, then if there were any more errors (only possibility was message character limit) then the server-side would catch it. If the client-side validation is not supported, the server-side validation can always catch the error. This double-checking system was an important choice because client-side browser versions and modifications are not in my control as a developer. If there is any intent to perform a form injection attack, the client-side validation can be tampered with. However, this is why my server-side validation was implemented to first parse the inputs and then throw an error if the fields do not correspond to the correct format.

4. Include any additional information, justifications, or comments we should be aware of. If you have questions, please limit them to only 2. We have office hours where you can get your questions answered. =)
    a. I used PHP in the:
        i. Menu – because the navigational menu bar was the same in all of my webpages, so my creating a menu.php file, I could link to it in ever webpage file.
            1. Because I linked to this php menu bar file in all of my webpages, every file was saved as a .php
        ii. Confirmation webpage – because here I had to process the form and let the user know that the form had been processed. In my php code, after processing the variables, I echoed (printed) them to the webpage to show the user what he/she inputted
        iii. I decided to include a form under the contact me webpage. This way if anyone saw my website and wanted to reach out to me, they have the ability to do so. To remind you, I designed this website to be a professional portfolio of my experiences. There are two target audiences that I have in mind: job recruiters and my fellow colleagues/students. If my target audience wanted to learn more about my work experiences, education, and projects they could contact me through the form that I created.
    b. I had 7 distinctive elements (name, gender, email, phone number, organization website, intent, and message). I used 3 distinct input types which were text, radio buttons, and textarea for the form with only server-side validation. I used 6 distinct input types which were text, radio, email, tel, url, textarea for the form with client-side and server-side validation
        i. I wanted to make the contact form more than just a simple form. That is why I asked the user to let me know who they represent (organization website), what the intent is of their outreach (recruiting, project question, or general inquiry), their message to me, and contact information for them.
    c. When validating my code, I get the error of having a duplicate ID for my radio buttons – I have spoken with TAs and I am fairly certain that I have implemented this correctly. I think the validator will throw an error because it sees multiple radio buttons that correspond to the same variable/input.