

# Code Review

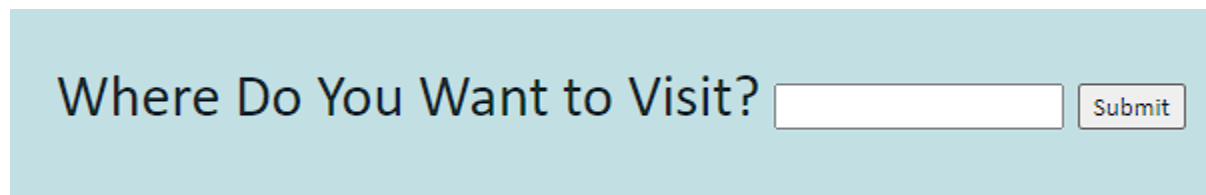
## Code #1:

### a) Code Snippet:

```
<label for="rating">Where Do You Want to Visit?</label>  
<input type="text" id="rating" name="input">  
<input type="submit" value="Submit">
```

```
.experience form {  
  font-size: 30px;  
  position: absolute;  
  margin-left: 150px;  
  margin-top: 470px;  
}
```

### b) Current Outcome:

A screenshot of a web form on a light blue background. The text "Where Do You Want to Visit?" is displayed in a large, dark font. To the right of the text is a white text input field with a thin grey border. Further to the right is a rectangular button with a thin grey border and the word "Submit" in a dark font.

### c) Desired Outcome:

The desired outcome for the new code snippet is to allow the user to feel more welcomed and rewarded when they put in the effort to type in a response and press submit. When the submit button is pressed, I want a few things to happen.

Firstly, I want a message to be printed under the input box and submit button that thanks the user for submitting their response. I desire this because it is a better solution than giving no feedback to the user when they submit a response. With a proper thank you message being printed, the user will feel as though their response matters, which they may appreciate. Printing a thank you message also ensures that the response actually went through and that there was no error that occurred. Additionally, if numbers are submitted, a error message will be printed instead.

Secondly, I want to integrate a way of locking the input field when the “Submit” button is pressed. This will be useful as it indicates to the user very clearly that their response has been received and that there was no error in between the time of them inputting their answer and submitting it. Not only will the input field become locked, which will prevent any further input being submitted, I will change the colour of the field to green if the input is all words, which will print a thank you message, however, if numbers are submitted, the field will turn red and an error message will be printed.

Lastly, I want to incorporate a “Reset” button next to the “Submit” button. This Reset button will act as a way to reset the input to default so that if the user wants to submit another answer, they can easily do so by just pressing the reset button. When the button is pressed, the input fields will go back to the default colour of white, and the thank you message that appears below the input field will also disappear. Furthermore, the input fields will be unlocked, allowing the user to type into the field once again.

## d) Code Snippet For Desired Outcome:

Javascript code:

```
//Function is called when the user submits their answer
function getSubmission() {
  // Get the user's answer input element
  const userInput = document.getElementById( 'answer');

  // Check if the input contains only letters and spaces
  if (/^[a-zA-Z\s]+$/i.test(userInput.value)) {
    // Set the background color of the input field to light green
    userInput.style.backgroundColor = "lightgreen";

    // Get the result message and set it to a 'thank you' message
    const resultMessage = document.getElementById( 'result-message');
    resultMessage.innerHTML = 'Thank you for your submission!';

    // Set the input field to read-only so that it can't be changed by the user
    userInput.readOnly = true;
  } else {
    // Set the background color of the input field to red
    userInput.style.backgroundColor = "red";

    // Get the result message and set it to an error message
    const resultMessage = document.getElementById( 'result-message');
    resultMessage.innerHTML = 'You cannot enter numbers!';

    // Clear the input field
    userInput.value = '';

    // Set the input field to read-only so that it can't be changed by the user
    userInput.readOnly = true;
  }
}
```

```

// Function is called when the user clicks the reset button
function reset() {
    //Get the user's input answer input element
    const userInput = document.getElementById( elementId: 'answer');

    //Background color is reset to white
    userInput.style.backgroundColor = "white";

    //Input field is set to blank
    userInput.value = " ";

    //Set the result message to be blank
    const resultMessage = document.getElementById( elementId: 'result-message');
    resultMessage.innerHTML = " ";

    userInput.readOnly= false;
}

```

```

// Recieve the submit and reset button
const submitButton = document.getElementById( elementId: 'submit-button');
const resetButton = document.getElementById( elementId: 'reset-button');

// Adding event listeners so that the functions are activated when the buttons are clicked on
submitButton.addEventListener( type: "click", getSubmission);
resetButton.addEventListener( type: "click", reset);

```

HTML code:

```
<div class="user">
  <h2>Where Do You Want to Visit?</h2>

  <label for="answer">Enter Here:</label>
  <input type="text" id="answer">

  <input type="submit" value="Submit" id="submit-button">
  <input type="submit" value="Reset" id="reset-button">

  <p id="result-message"></p>
</div>
```

CSS code:

```
.user {
  margin-top: 1400px;
  font-size: 30px;
}

#submit-button{
  font-size: 25px;
}

#reset-button{
  font-size: 25px;
}

#answer{
  height:30px;
  width: 300px;
}

#result-message{
  margin-left: 150px !important;
}
```

### e) New Outcome of the Code

When the “Submit” button is pressed, and the input is valid:

**Where Do You Want To Visit?**

Enter Here:

Thank you for your submission!

When the “Submit” button is pressed, and the input has numbers:

**Where Do You Want To Visit?**

Enter Here:

You cannot enter numbers!

When the “Reset” button is pressed:

**Where Do You Want to Visit?**

Enter Here:

## Code #2:

### a) Code Snippet:

```
.lake {  
  position: absolute;  
  margin-right: 60%;  
  padding: 1px;  
  border: 2px solid black;  
}
```

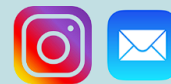
```
<div class="lake">  
    
</div>
```

### b) Current outcome:



## Welcome!

Hi! My name is Muhammad Hassan, and I'm a first year Computer Science student at [Dalhousie University](#). The goal of this website is to allow people to get me know better through my background and some of my personal interests, such as music and photography, and allow people to understand how I plan to use those interests/hobbies as a potential future career that I would pursue, alongside a job in the Computer Science field.



### c) Desired outcome:

The issue with the current outcome is that it is a little stale. My website is designed so that the home page is meant to be the first page that a user visits, and I believe that it is important to allow the webpage to have some characteristics that will make the webpage pop out and thus, attract the user to the website. One of the ways to implement such characteristics is through javascript.

Firstly, I would like to add a transition effect to the image that is located to the left of the paragraph (see b)). I would like to add a transition where it starts on the left side of the page and transitions into place from the right, thus creating a 'slide-in effect'. I feel like a sliding effect will be best on an image since the image size is relatively big, the sliding effect will be best noticed on the image. Furthermore, this effect will be loaded every time the webpage is loaded/reloaded, in order for the effect to begin as quickly as possible.

Secondly, the two icons that are located under the paragraph will also have an effect. I would like to add a transition in combination with a sliding effect where the two icons will fade into existence onto the page, thus creating a 'fading' effect in order to provide some further visual attraction. I believe a fade effect will work well on these smaller images as it will be more subtle than doing it on the main image.

### d) Code Snippet For Desired Outcome:

CSS code:

```
.lake {  
    position: absolute;  
    margin-right: 60%;  
    padding: 1px;  
    border: 2px solid black;  
}  
  
.lake.slide-in {  
    margin-top: 0;  
    transform: translateX(20%);  
    transition: margin-bottom 1s ease-out, transform 1s ease-out;  
}
```

HTML code:

```
<div class="lake">
  
</div>
```

Javascript code:

```
//Slide in image to the right
window.addEventListener( type: 'load', listener: () =>{
  const lakeDiv = document.querySelector( selectors: '.lake');
  lakeDiv.classList.add('slide-in');
})

//Declare variables for both icon sized images
const instaDiv = document.querySelector( selectors: '.insta');
const mailDiv = document.querySelector( selectors: '.mail');

// Set initial opacity to 0
instaDiv.style.opacity = 0;
mailDiv.style.opacity = 0;

// Fade in the icon sized images when the page is loaded
window.addEventListener( type: 'load', listener: () => {
  instaDiv.style.transition = 'opacity 3s';
  mailDiv.style.transition = 'opacity 3s';

  instaDiv.style.opacity = 1;
  mailDiv.style.opacity = 1;
});
```



### **e) New Outcome of the Code**

<https://gyazo.com/36f6be63a70c881da037d4e376f12d96>

The outcome of the code turned out exactly how I wanted it to. The sliding effect works on the main image and the two smaller icon-size images fade in as intended. Both effects work simultaneously when the webpage is loaded, and the effects work properly every time it is reloaded/loaded.