

# Part 1 - Code Review

## Improvement 1 - Introductory Text

The introductory section for my home page is satisfactory. However, I originally had an idea for this section that could only be implemented using JavaScript. I only want to display one list element at a time, and then after a few seconds, switch to another one.

```
<!--Quick introduction for user-->
<div id="intro">
  <div id="content">
    
    <div id="text">
      <h3>Hi, my name is</h3>
      <h1><mark>Kevin Kuruvilla</mark></h1>
      <ul>
        <li>First-year BCS student at Dalhousie University</li>
        <li>Passion for deep learning and NLP</li>
        <li>Aim to become a DL engineer</li>
      </ul>
      <a href="#contact" class="underline">
        Always open to new opportunities and collaborations
      </a>
    </div>
  </div>
</div>
```

In the code snippet of my current HTML code, we can see right now all the list elements are static. They don't change relative to anything. Because of this, I can only add so many list items without making the home page look too cluttered and ruining the aesthetic of my website. This means I have to be very selective of what I want to present to myself, as I only have so many spots to display it to the user.

Hi, my name is

**Kevin Kuruvilla**

- First-year BCS student at Dalhousie University
- Passion for deep learning and NLP
- Aim to become a DL engineer

Always open to new opportunities and collaborations



Here's the current outcome of the code. As you can see, this layout only enables me to keep three points about myself without cluttering the viewport too much. Regardless, I still find this to feel a tad overwhelming to read from a user's perspective, and I would much prefer present only one list item at a time to the user.

Hi, my name is

**Kevin Kuruvilla**

- First-year BCS student at Dalhousie University

Always open to new opportunities and collaborations



This is the desired outcome. When the user enters the page, I want to only display the first list element and hide the rest.

Hi, my name is

**Kevin Kuruvilla**

- Passion for deep learning and NLP

Always open to new opportunities and collaborations



After a few seconds have past, I want to hide the current list element and display the next list element.

Hi, my name is

**Kevin Kuruvilla**

- Aim to become a DL engineer

Always open to new opportunities and collaborations



This process repeats until the last list element is displayed, in which case it should display the first element again where it continues this process.

```

<main>
  <!--Quick introduction for user-->
  <div id="intro">
    <div id="content">
      
      <div id="text">
        <h3>Hi, my name is</h3>
        <h1><mark>Kevin Kuruvilla</mark></h1>
        <ul>
          <li>First-year BCS student at Dalhousie University</li>
          <li>Passion for deep learning and NLP</li>
          <li>Aim to become a DL engineer</li>
          <li>Proficient in Java and Python</li>
          <li>Experience with Tensorflow</li>
          <li>Done some robotics projects</li>
          <li>Notetaker for Dalhousie</li>
          <li>Love chess and Rubik's cubes</li>
        </ul>
        <a href="#contact" class="underline">
          Always open to new opportunities and collaborations
        </a>
      </div>
    </div>
  </div>
</div>

```

In the HTML file, I first added all the list elements that I wanted to add originally but could not due to the limited space I had available in the front page. Now that I am switching between list elements, I can have as many as I want!

```

#text ul li:not(:first-child) {
  display: none;
}

```

In the CSS file, I hid all the list elements from view except the first element.

```

// Nodes
const listItems = document.querySelectorAll("#text ul li");

// Variables
let currentIndex = 0;

// Switches introductory text that is being displayed
function changeListItem() {
  // Hide current li element, then increment index
  listItems[currentIndex++].style.display = "none";

  // Ensure index remains within range of list elements
  currentIndex = currentIndex % listItems.length;

  // Unhide next li element
  listItems[currentIndex].style.display = "list-item";
}

// Calls changeListItem every 3 seconds
setInterval(changeListItem, 3000);

```

In a newly created JS file, I created a function called `changeListItems()` that simply hides the current element index, then increments the index, and then displays the next element index. I then use `setInterval()` to call the function every 3 seconds, so that it displays a new list element every three seconds.

...

[Home](#) [About Me](#) [Projects](#)

[Email](#) [LinkedIn](#) [Resume](#)

Hi, my name is

**Kevin Kuruvilla**

- Love chess and Rubik's cubes

Always open to new opportunities and collaborations



The resulting outcome of this code creates a webpage that looks far cleaner. The list elements are not cluttering up the entire viewport, and I was able to fit double the number of points about myself than before. My intention for parsimony is much more apparent to the user, allowing a more vivid sense of branding for my website.

## Improvement 2 - HTML forms

For assignment, I felt pretty happy with the look of my contact form. It felt welcoming, clearly communicated its purpose, and gave direction to users to enable more of them to reach out to me. However, there were some default behaviors by the form that felt too stern and artificial, which is not the kind of tone I want with my website.

```
<footer>
  <!--Contact form-->
  <div id="contact">
    <h2>Get in Touch</h2>
    <form action="https://example.com/" method="post">
      <p>
        <label for="email">Email: <br></label>
        <input type="email" id="email" name="email" placeholder="example@domain.com" required>
      </p>
      <p>
        <label for="message">Message: <br></label>
        <textarea id="message" name="message" placeholder="How can I help?" required></textarea>
      </p>
      <input type="submit" value="Send">
    </form>

    <!--Simple link that brings user back to top of page-->
    <a href="#"><br>Back to Top</a>
  </div>
</footer>
```

In the code snippet of my current HTML code, we can see I placed the form within the footer section. The main problem I face has to do with the `required` keywords for both the input field and the text area.



**Get in Touch**

Email:

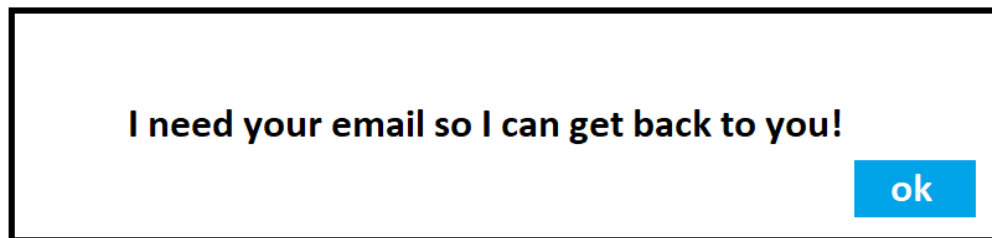
example@domain.com

Please fill out this field.

Send

Back to Top

This is the current outcome of the code above. When the user submits an empty form, they will get this warning. However, the writing feels lifeless and it does not provide any incentive to the user to provide me with their email. Furthermore, this hardly looks like a warning, but more like a suggestion. The warning feels detached from the rest of the form and does not fit the aesthetic of the rest of the website.

A contact form titled "Get in Touch" in a bold, black, sans-serif font. Below the title, there are two input fields. The first is labeled "Email:" and contains the text "example@domain.com". The second is labeled "Message:" and contains the text "How can I help?". Below the message field is a black button with the text "Send" in white. At the bottom of the form is a link that says "Back to Top". The email input field is highlighted with a thick red border.

This is the desired outcome. I want to create a custom alert to better communicate with the user about using the form, while also providing more solid reasoning as to why I am asking for an email address. I also want a thick red border around the field in question, which feels more connected with the form than the overlaid warning.

```

</footer>
<!--Contact form-->
<div id="contact">
  <h2>Get in Touch</h2>
  <form action="https://example.com/" method="post" onsubmit="return validateForm()">
    <p>
      <label for="email">Email: <br></label>
      <input type="email" id="email" name="email" placeholder="example@domain.com">
    </p>
    <p>
      <label for="message">Message: <br></label>
      <textarea id="message" name="message" placeholder="How can I help?"></textarea>
    </p>
    <input type="submit" value="Send">
  </form>

  <!--Simple link that brings user back to top of page-->
  <a href="#"><br>Back to Top</a>
</div>
</footer>

```

In the HTML file, I added an `onsubmit` handler to call and return the `validateForm()` function in my JS file. Furthermore, I removed the `required` keyword from both input fields, as I will make my own custom version.

```

// Checks if input is valid
function validateForm() {
  // Get input fields
  const email = document.getElementById("email");
  const message = document.getElementById("message");

  // Initialize both fields to have a 1.5px thick solid black border
  email.style.border = "1.5px solid black";
  message.style.border = "1.5px solid black";

  // Checks if email field is left empty
  if (email.value == "") {
    // Give field a 3px thick solid red border
    email.style.border = "3px solid red";

    // Alert user of the problem
    alert("I need your email so I can get back to you!");

    return false;
  }

  // Checks if message field is left empty
  if (message.value == "") {
    // Give field a 3px thick solid red border
    message.style.border = "3px solid red";

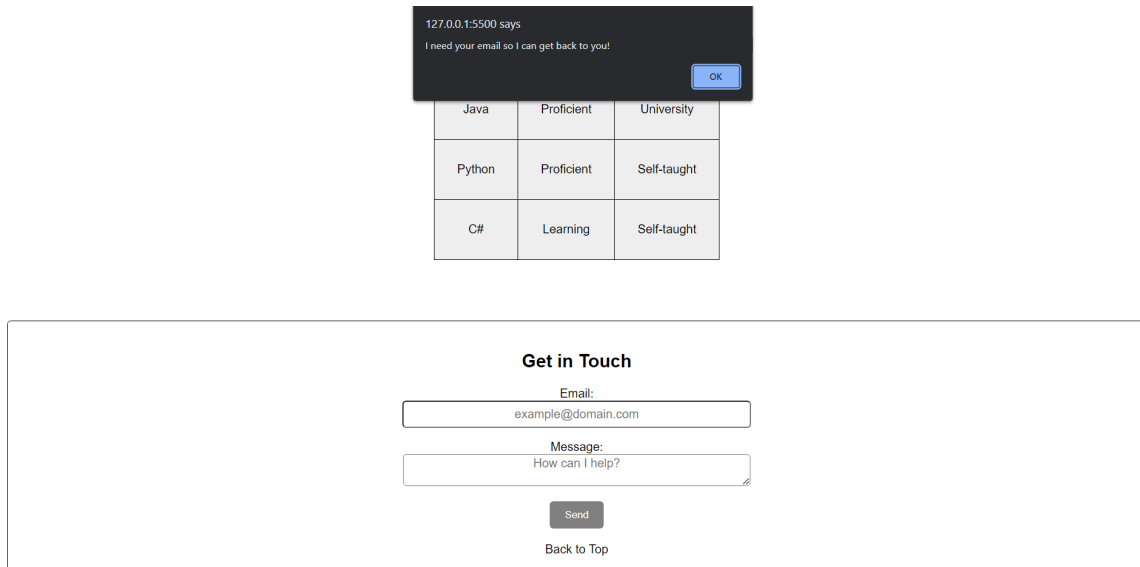
    // Alert user of the problem
    alert("You should probably type something!");

    return false;
  }
}

```



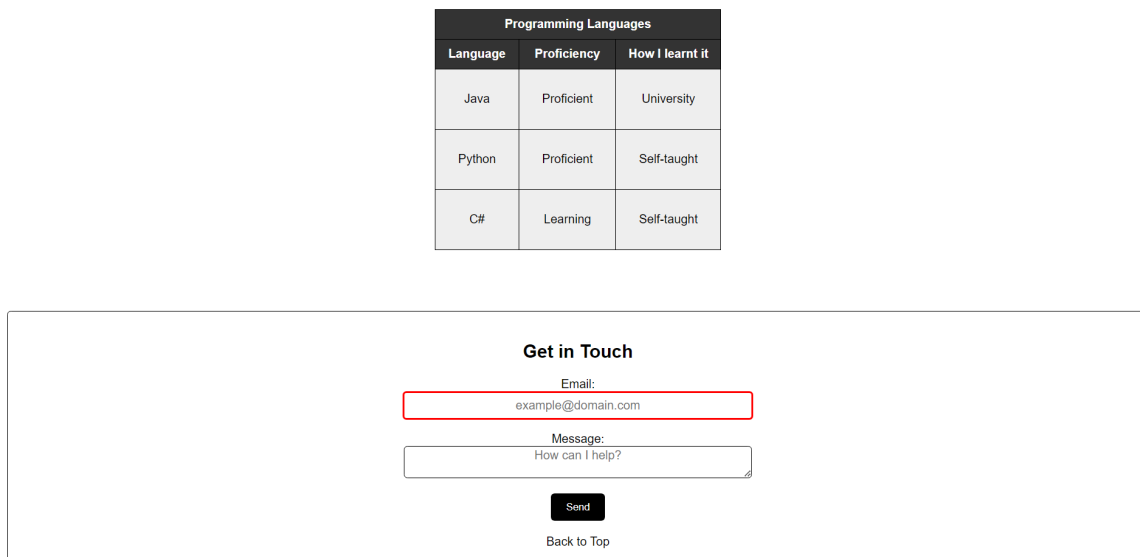
In the JS file, I defined a `validateForm()` function. I first reset both input fields to have a 1.5 pixel thick solid black border. Then, I made two if statements that checks if the email or the message input fields are empty. In which case, I add a 3 pixel thick solid red border around the input field and call an `alert()` to inform the user about the problem. Finally, I return false to each of these as to not submit any invalid form data.



The screenshot shows a web form titled "Get in Touch". It contains an "Email:" input field with the text "example@domain.com" and a "Message:" input field with the text "How can I help?". Below the input fields is a "Send" button and a "Back to Top" link. An alert box is displayed over the form, showing the message "127.0.0.1:5500 says I need your email so I can get back to you!" with an "OK" button. A table is also visible in the background, showing programming languages and proficiency levels.

Language	Proficiency	How I learnt it
Java	Proficient	University
Python	Proficient	Self-taught
C#	Learning	Self-taught

This is the resulting outcome of the code. When I leave the email field blank, I receive an alert that says "I need your email so I can get back to you!".



The screenshot shows the same web form as before, but with a red border around the "Email:" input field. The "Message:" input field is still empty. The "Send" button and "Back to Top" link are still present. The table is also visible in the background.

Language	Proficiency	How I learnt it
Java	Proficient	University
Python	Proficient	Self-taught
C#	Learning	Self-taught

After closing the alert, the empty email input field is highlighted by the red border. From a user perspective, these small indicators really help ensure users know how to interact with my website!

127.0.0.1:5500 says  
You should probably type something!

OK

Java	Proficient	University
Python	Proficient	Self-taught
C#	Learning	Self-taught

**Get in Touch**

Email:  
example.email@domain.com

Message:  
How can I help?

Send

Back to Top

When I input an email, but omit a message. I now get an alert that says “You should probably type something!”. This feels more lighthearted than the default warning message, and persuades users to finish the form.

Programming Languages		
Language	Proficiency	How I learnt it
Java	Proficient	University
Python	Proficient	Self-taught
C#	Learning	Self-taught

**Get in Touch**

Email:  
example.email@domain.com

Message:  
How can I help?

Send

Back to Top

Closing this alert yields a similar result. A red border appears around the empty message field. With this, my form ensures users know what to do at every step of the way!

