

Code Review

1. Contact Me Form

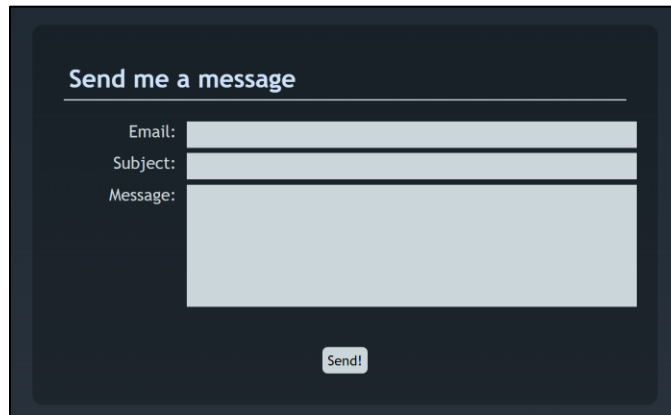
```
<div class="flex-item">
  <h2>Send me a message</h2>
  <form id="email-form">
    <div id="email-label-div">
      <label for="email-input">Email:</label>
    </div>
    <div id="email-input-div">
      <input id="email-input" type="email" required/>
    </div>

    <div id="subject-label-div">
      <label for="subject-input">Subject:</label>
    </div>
    <div id="subject-input-div">
      <input id="subject-input" type="text" required/>
    </div>

    <div id="message-label-div">
      <label for="message-input">Message:</label>
    </div>
    <div id="message-input-div">
      <textarea id="message-input" rows="7" required></textarea>
    </div>

    <div id="subit-div">
      <input id="submit" type="submit" value="Send!"/>
    </div>
  </form>
</div>
```

a.



b.

- c. The desired outcome would be nicer and more responsive contact form validation. It should turn red and prompt an error message above the send button that says the error.

```
<div class="flex-item">
  <h2>Send me a message</h2>
  <form id="email-form">
    <div id="email-label-div">
      <label for="email-input">Email:</label>
      <p id="email-warn" class="warning-text">Invalid email</p>
    </div>
    <div id="email-input-div">
      <input id="email-input" type="email" class="error-input"/>
    </div>

    <div id="subject-label-div">
      <label for="subject-input">Subject:</label>
      <p id="subject-warn" class="warning-text">Subject is empty</p>
    </div>
    <div id="subject-input-div">
      <input id="subject-input" type="text" class="error-input"/>
    </div>

    <div id="message-label-div">
      <label for="message-input">Message:</label>
      <p id="message-warn" class="warning-text">Message is empty</p>
    </div>
    <div id="message-input-div">
      <textarea id="message-input" rows="7" class="error-input"> </textarea>
    </div>

    <div id="subit-div">
```

d.

```
.error-input {
  background-color: darkred;
}

.warning-text {
  color: darkred;
  margin: 0;
```

```

function validate(event) {
  let emailInput = document.getElementById("email-input");
  let subjectInput = document.getElementById("subject-input");
  let messageInput = document.getElementById("message-input");

  let emailWarning = document.getElementById("email-warn");
  let subjectWarning = document.getElementById("subject-warn");
  let messageWarning = document.getElementById("message-warn");

  let emailValid = validateEmail(emailInput.value);
  let subjectValid = !isEmptyString(subjectInput.value);
  let messageValid = !isEmptyString(messageInput.value);

  if (!emailValid) {
    emailInput.classList.add("error-input");
    emailWarning.innerHTML = "Invalid email";
  }

  if (!subjectValid) {
    subjectInput.classList.add("error-input");
    subjectWarning.innerHTML = "Subject is empty";
  }

  if (!messageValid) {
    messageInput.classList.add("error-input");
    messageWarning.innerHTML = "Message is empty";
  }

  if (emailValid && subjectValid && messageValid) {
    subjectInput.classList.remove("error-input");
    subjectInput.classList.remove("error-input");
    subjectInput.classList.remove("error-input");

    emailWarning.innerHTML = "";
    subjectWarning.innerHTML = "";
    messageWarning.innerHTML = "";
  } else {
    event.preventDefault();
  }
}

function validEmail(email) {
  atIndex = email.indexOf('@');

  if (atIndex <= 0) {
    // Email does not contain '@' or name before '@'
    return false;
  }

  if (email.lastIndexOf('.') < atIndex) {
    // No '.' after '@'
    return false;
  }

  if (email.length < 5) {
    // Min size if 5 chars (a@b.c)
    return false;
  }

  return true;
}

function isEmptyString(text) {
  return text == null || text.trim() == "";
}

const formElement = document.querySelector("form");
formElement.addEventListener("submit", validate);

```

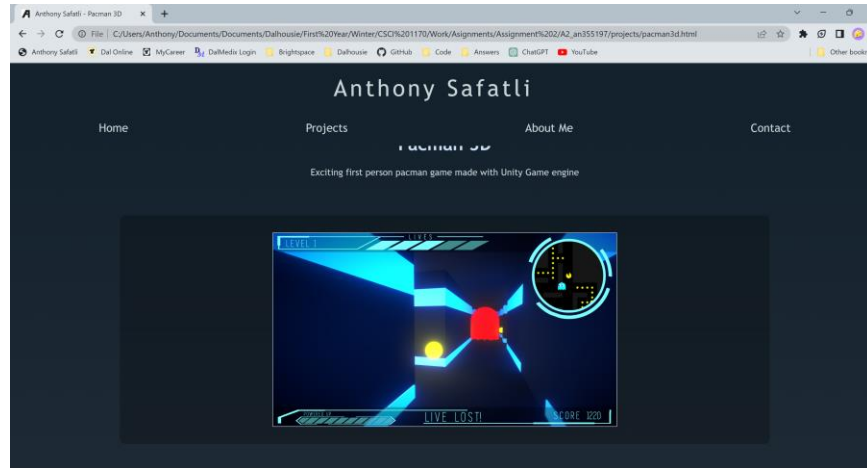
- e. The updated outcome tells the user the problem with the form instead of sending an incomplete form, displaying the colour and messages as well.

The image shows a contact form with a dark blue background. The title 'Send me a message' is at the top. Below it are three input fields: 'Email:', 'Subject:', and 'Message:'. Each field has a red error message: 'Invalid email' for Email, 'Subject is empty' for Subject, and 'Message is empty' for Message. A 'Send!' button is at the bottom.

2. Image Expanding

```
<div class="flex-item">  
    
</div>
```

a.



b.

- c. The desired outcome is for the ability for the user to click in the image, and the image to expand and take up the majority of the screen. Clicking off the image should remove the enlarged image. It should also gray out all the elements in the background.

```

body {
  height: 100vh;
  padding: 0;
  margin: 0;
}

.fullscreen {
  width: 100%;
  height: 100%;
  background-color: black;
  opacity: 70%;
  z-index: 10;
  position: fixed;
}

#main-media {
  position: fixed;
  width: 70vw;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
  z-index: 11;
}

```

d.

```

function addFullScreen(event) {
  let bodyElement = document.querySelector("body");
  let divElement = document.createElement("div");
  divElement.classList.add("fullscreen");
  divElement.addEventListener("click", removeFullScreen);

  let newMedia = event.target.cloneNode();
  newMedia.id = "main-media";
  newMedia.addEventListener("click", removeFullScreen);

  bodyElement.appendChild(divElement);
  bodyElement.appendChild(newMedia);
}

function removeFullScreen(event) {
  let divElement = document.querySelector(".fullscreen");
  let mainMedia = document.getElementById("main-media");
  divElement.remove();
  mainMedia.remove();
}

const mediaElements = document.querySelectorAll(".media");
for (let i = 0; i < mediaElements.length; i++) {
  mediaElements[i].addEventListener("click", addFullScreen);
}

```

- e. The new outcome is the ability to click on the media element, and be able to enlarge to take 70vw. If you click anywhere on the screen it will delete these elements which lets

you continue looking at the page.

