Display The Image On The Page

We're making our request to Unsplash, it's returning a response that we're then converting to JSON, and now we're seeing the actual JSON data. Fantastic! All we need to do now is display the image and caption on the page.

Here's the code that I'm using:

```
function addImage(data) {
    let htmlContent = '';
    const firstImage = data.results[0];

    if (firstImage) {
        htmlContent = `<figure>
            <img src="${firstImage.urls.small}" alt="${searchedForText}">
            <figcaption>${searchedForText} by ${firstImage.user.name}</figcaption>
        </figure>`;
    } else {
        htmlContent = 'Unfortunately, no image was returned for your search.'
    }

    responseContainer.insertAdjacentHTML('afterbegin', htmlContent);
}
```

This code will:

- get the first image that's returned from Unsplash
- create a `<figure>` tag with the small image
- creates a `<figcaption>` that displays the text that was searched for along with the first name of the person that took the image
- if no images were returned, it displays an error message to the user

Handling Errors

Our app is now done with getting the image from Unsplash!...almost. We're requesting the image and adding it to the page, but this is only *one* possible outcome. Granted, it's the most likely way that the app will end up, but we're not handling any errors. What errors could possible happen you ask? A couple I can think of are:

- Issues with the network
- Issues with the fetch request
- Unsplash not having an image for the searched term

We're handling this last one in the `addImage` function. For the other two, we can use chain on a `.catch()` method to the Fetch request!

Again, because a Fetch request returns a Promise `.catch()` is available to us from the Promise API.

So let's add a `.catch()` method to handle errors:

```
fetch(`https://api.unsplash.com/search/photos?page=1&query=${searchedForText}`, {
    headers: {
        Authorization: 'Client-ID abc123'
    }
}).then(response => response.json())
.then(addImage)
.catch(e => requestError(e, 'image'));

function addImage(data) {
    debugger;
}

function requestError(e, part) {
    console.log(e);
    responseContainer.insertAdjacentHTML('beforeend', `<p class="network-warning">Oh no! There was an erro
r making a request for the ${part}.</p>`);
}
```

This code adds the `requestError` function and adds a `.catch()` request to the end of the Promise chain. The `.catch()` function will receive an error object (that we're storing in the `e` variable) and in turn calls `requestError` passing along the error object and the request that failed. If the Promise rejects anywhere

Mentorship
Get support and stay on track
1