Flickr Image Grabber Coding Challenge

Introduction:

While I was a Software Engineer Intern with Kapsch TrafficCom, I participated in a coding challenge to showcase my ability to be a front-end developer.

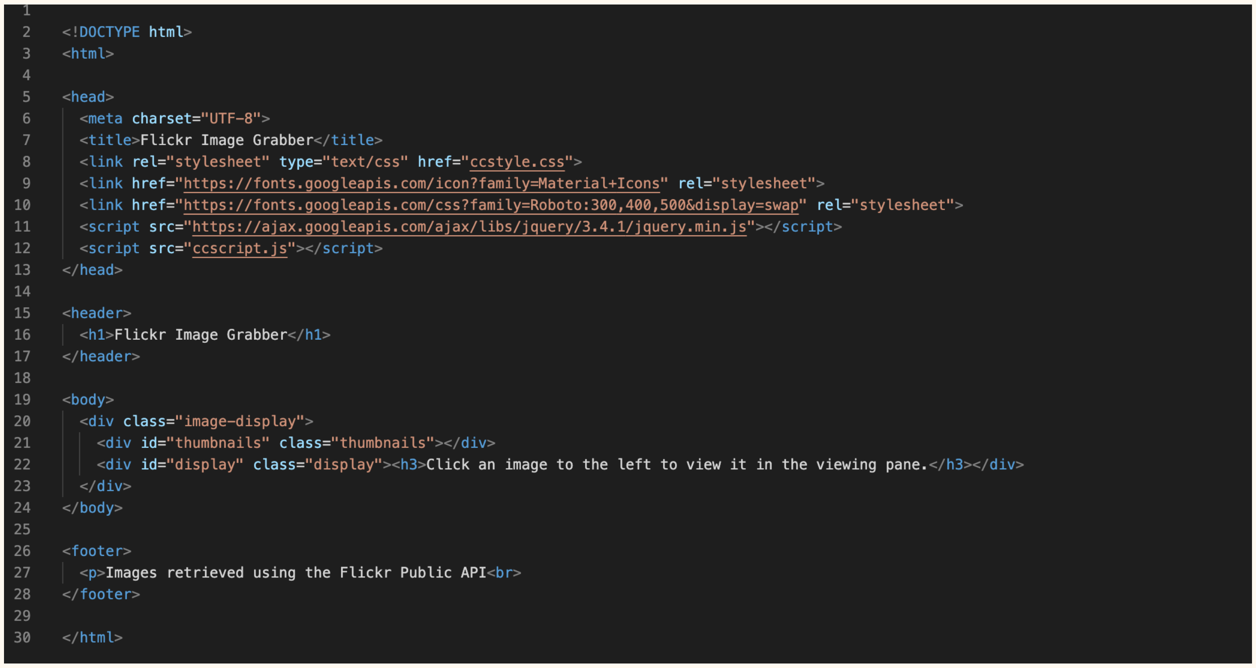
Here was the prompt given to interns:

Please construct a "Flicker Image Grabber". We are looking for a fully functioning webpage that allows a user to browse images (with or without a search function). Using the Flickr API is mandatory, but feel free to design the layout how you see fit. You only have 1 hour, good luck.

P.S. We highly suggest using JQuery.

Using HTML in VSCode:

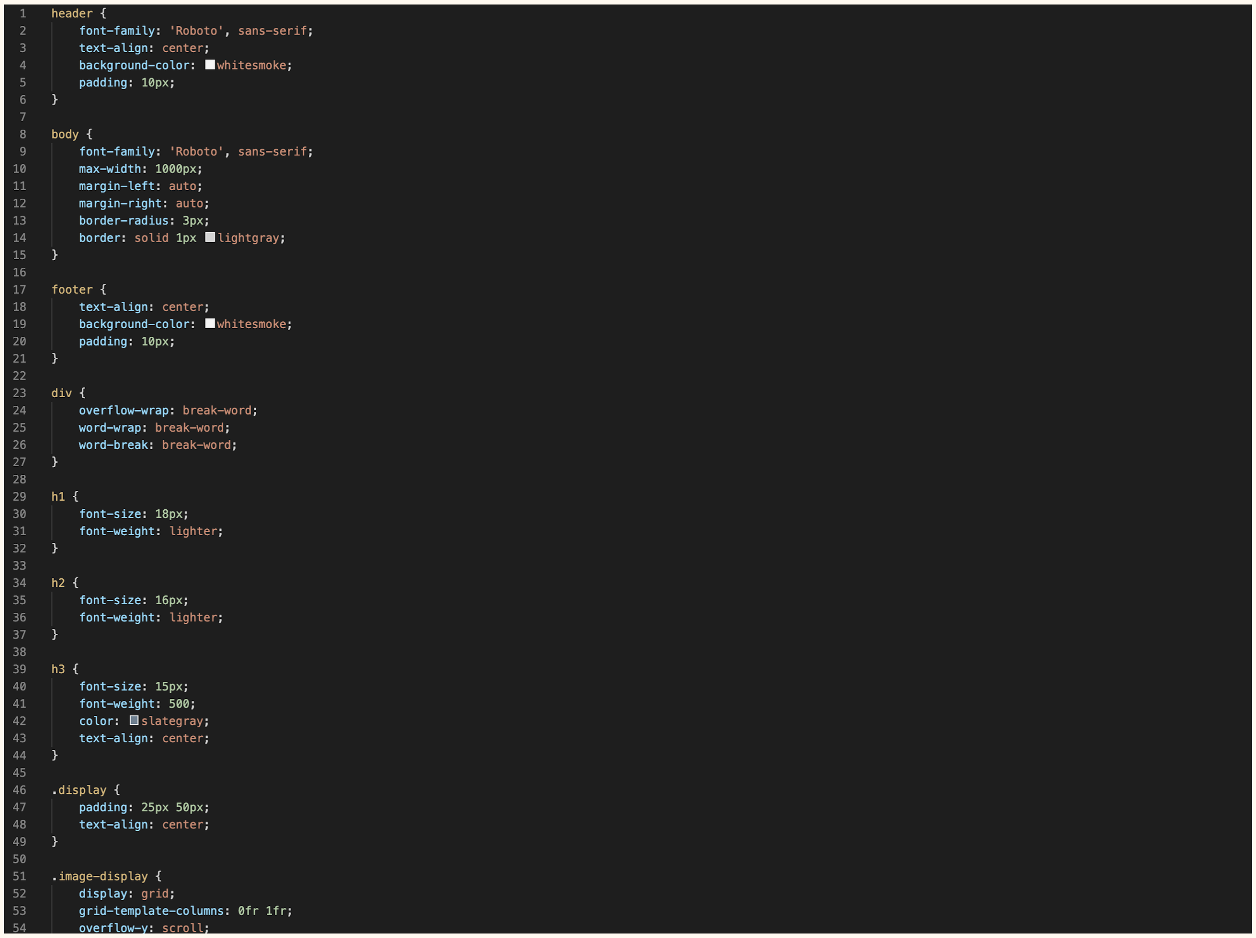
I ended up using JQuery to help streamline my coding process. Considering that we only had an hour to complete the challenge, I created my code to be very simple and easy to understand. I also emphasized organization in my code structure (mainly so I could understand what *I* was looking at).



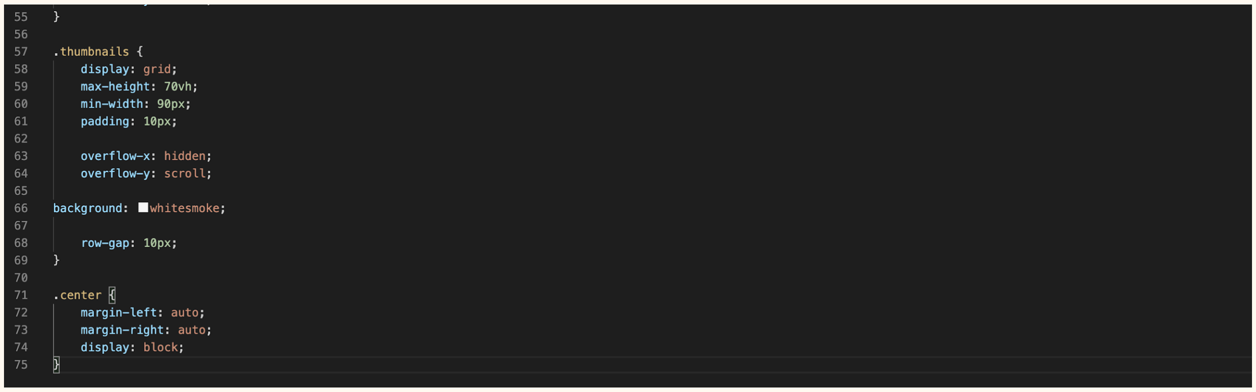
```html

<!DOCTYPE html>
<html>

<head>
  <meta charset="UTF-8">
  <title>Flickr Image Grabber</title>
  <link rel="stylesheet" type="text/css" href="ccstyle.css">
  <link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
  <link href="https://fonts.googleapis.com/css?family=Roboto:300,400,500&display=swap" rel="stylesheet">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
  <script src="ccscript.js"></script>
</head>

<header>
  <h1>Flickr Image Grabber</h1>
</header>

<body>
  <div class="image-display">
    <div id="thumbnails" class="thumbnails"></div>
    <div id="display" class="display"><h3>Click an image to the left to view it in the viewing pane.</h3></div>
  </div>
</body>

<footer>
  <p>Images retrieved using the Flickr Public API<br>
</footer>

</html>
```

Using CSS in VSCode:

It took me a while to toggle with spacing and image placement, but it ended up looking professional. I wanted the webpage to have neutral colors (whites and greys) to better highlight the images the user was looking at.



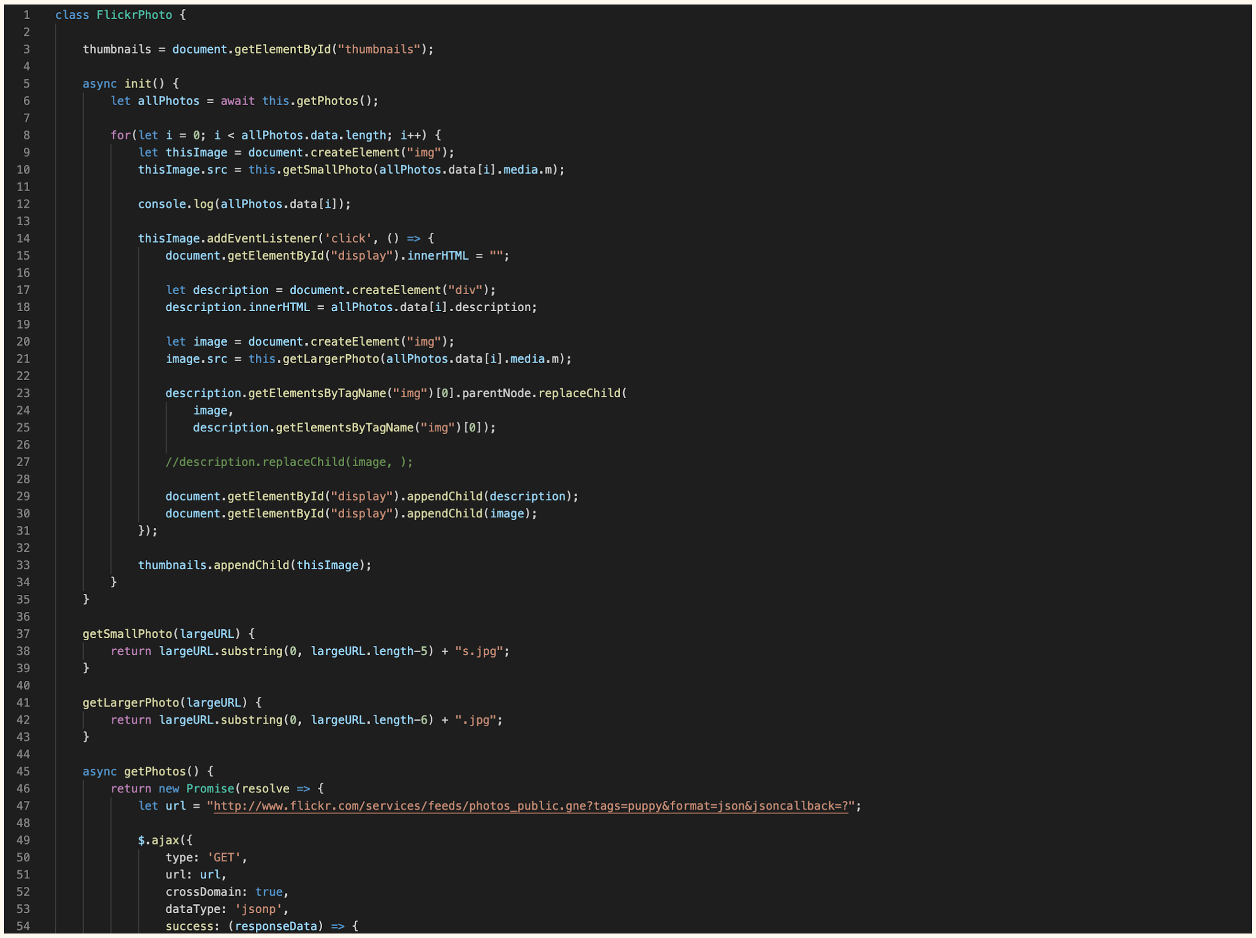
```css
header {
    font-family: 'Roboto', sans-serif;
    text-align: center;
    background-color: whitesmoke;
    padding: 10px;
}

body {
    font-family: 'Roboto', sans-serif;
    max-width: 1000px;
    margin-left: auto;
    margin-right: auto;
    border-radius: 3px;
    border: solid 1px lightgray;
}

footer {
    text-align: center;
    background-color: whitesmoke;
    padding: 10px;
}

div {
    overflow-wrap: break-word;
    word-wrap: break-word;
    word-break: break-word;
}

h1 {
    font-size: 18px;
    font-weight: lighter;
}

h2 {
    font-size: 16px;
    font-weight: lighter;
}

h3 {
    font-size: 15px;
    font-weight: 500;
    color: slategray;
    text-align: center;
}

.display {
    padding: 25px 50px;
    text-align: center;
}

.image-display {
    display: grid;
    grid-template-columns: 0fr 1fr;
    overflow-y: scroll;
```



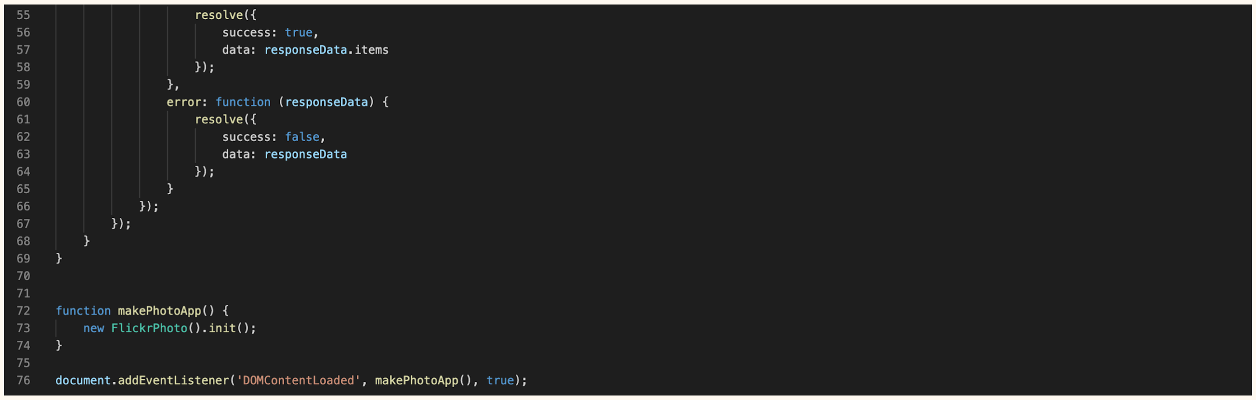
```css
}

.thumbnails {
    display: grid;
    max-height: 70vh;
    min-width: 90px;
    padding: 10px;

    overflow-x: hidden;
    overflow-y: scroll;

background: whitesmoke;

    row-gap: 10px;
}

.center {
    margin-left: auto;
    margin-right: auto;
    display: block;
}
```

Using JavaScript in VSCode:

Using JQuery helped really streamline my work and reduced the amount of code I would've had to have used (especially with the loops). I also implemented AJAX when utilizing the API to help synch the photos in real time. To add a personal touch - the Flickr tag is displaying puppies :) .



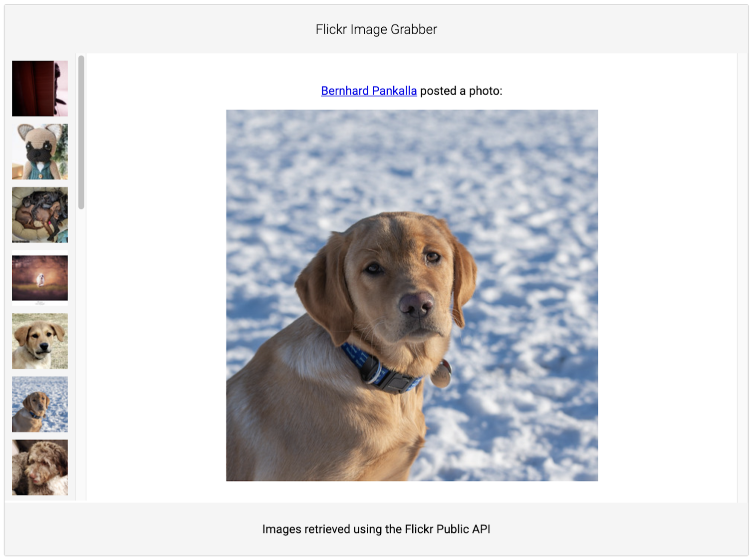
```javascript
class FlickrPhoto {

    thumbnails = document.getElementById("thumbnails");

    async init() {
        let allPhotos = await this.getPhotos();

        for(let i = 0; i < allPhotos.data.length; i++) {
            let thisImage = document.createElement("img");
            thisImage.src = this.getSmallPhoto(allPhotos.data[i].media.m);

            console.log(allPhotos.data[i]);

            thisImage.addEventListener('click', () => {
                document.getElementById("display").innerHTML = "";

                let description = document.createElement("div");
                description.innerHTML = allPhotos.data[i].description;

                let image = document.createElement("img");
                image.src = this.getLargerPhoto(allPhotos.data[i].media.m);

                description.getElementsByTagName("img")[0].parentNode.replaceChild(
                    image,
                    description.getElementsByTagName("img")[0]);

                //description.replaceChild(image, );

                document.getElementById("display").appendChild(description);
                document.getElementById("display").appendChild(image);
            });

            thumbnails.appendChild(thisImage);
        }
    }

    getSmallPhoto(largeURL) {
        return largeURL.substring(0, largeURL.length-5) + "s.jpg";
    }

    getLargerPhoto(largeURL) {
        return largeURL.substring(0, largeURL.length-6) + ".jpg";
    }

    async getPhotos() {
        return new Promise(resolve => {
            let url = "http://www.flickr.com/services/feeds/photos_public.gne?tags=puppy&format=json&jsoncallback=?";

            $.ajax({
                type: 'GET',
                url: url,
                crossDomain: true,
                dataType: 'jsonp',
                success: (responseData) => {
```



```javascript
                    resolve({
                        success: true,
                        data: responseData.items
                    });
                },
                error: function (responseData) {
                    resolve({
                        success: false,
                        data: responseData
                    });
                }
            });
        });
    }
}


function makePhotoApp() {
    new FlickrPhoto().init();
}

document.addEventListener('DOMContentLoaded', makePhotoApp(), true);
```

The Visual Result:

Once the hour was up, I was extremely happy with how the web page turned out. Though I wish I could've made it a bit more visually pleasing, it met all of the functional aspects of the prompt.


Flickr Image Grabber
Bernhard Pankalla posted a photo:
Images retrieved using the Flickr Public API

Conclusion:

This challenge was a lot of fun to participate in! Reflecting back on my work it taught me a lot about:

- Integrating an API
- Working with JQuery and AJAX
- Working under a time crunch

After completion, a lot of other interns recommended using Angular. I've never used Angular but I've started the "Tour of Heros" exercise that the creators have added to their website. (https://angular.io/tutorial)