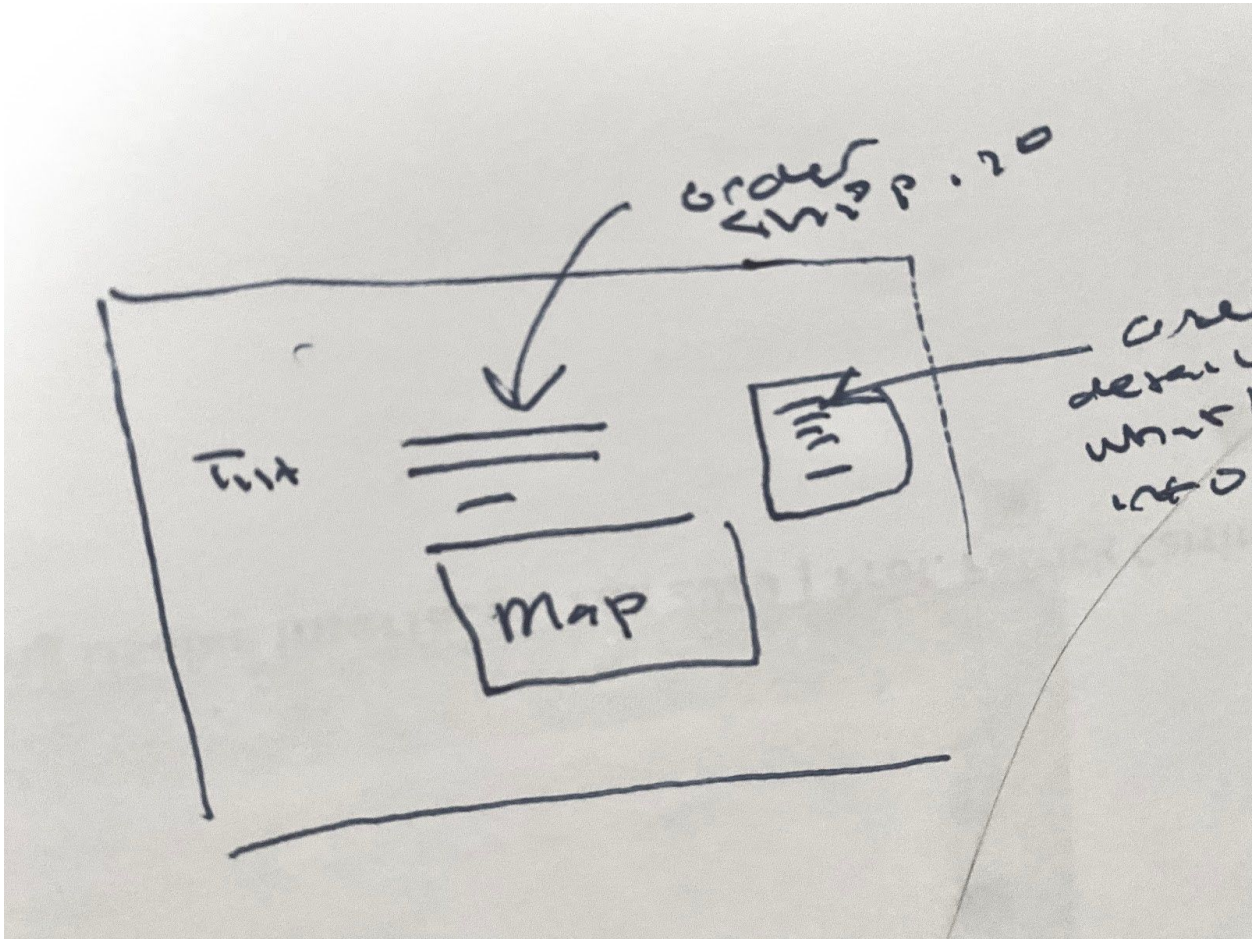
PUI HW 6A

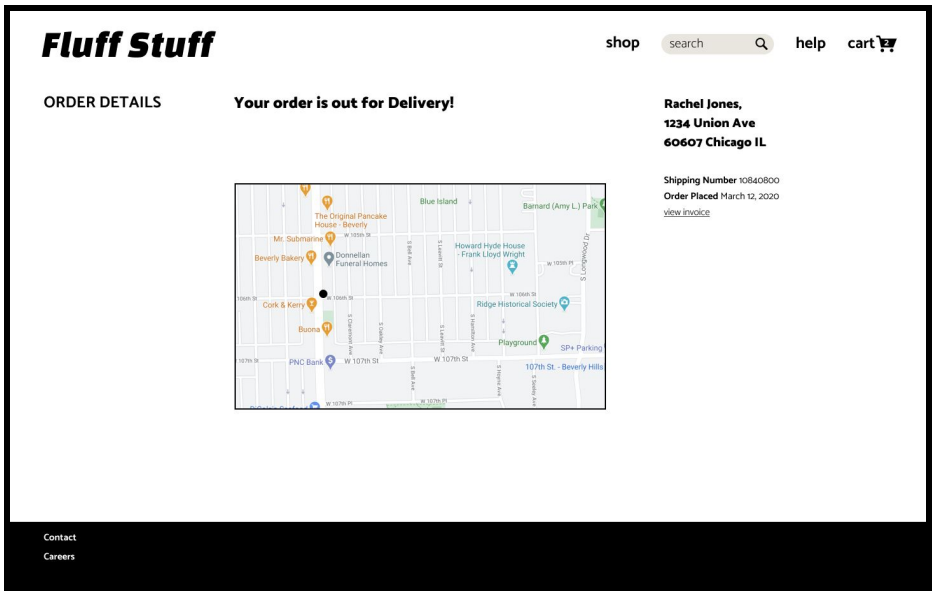
New Page: Order Tracking

LowFi



I had already created an ordering page on Figma, so I decided to create an order tracking page. To create a tracking page, I first thought about what I am looking for when I visit them. I came up with 2 things. The first is to validate my order and the second is to obsessively check how long it will take until it reaches me. With that in mind. I decided to add a large map view of where the package was as well as large descriptive language about the nature of the order in relation to the customer, and those are the most useful info for such a task.

Hi-Fi



Live site: https://ralajo.github.io/Tracking.html **- This page can be found on main via a link from the help page, but If used normally, this should really only be accessed via an outside email key.**
HTML = Tracking.html
CSS = Tracking.css

New Features: Product Page

Live site: https://ralajo.github.io/FluffProduct.html
HTML = https://github.com/ralajo/ralajo.github.io/blob/main/FluffProduct.html
CSS = https://github.com/ralajo/ralajo.github.io/blob/main/FluffProduct.css
JS = https://github.com/ralajo/ralajo.github.io/blob/main/FluffStuff.js

- Changes that occur on the product page include:
    - Change Fill Label When Selected
    - Change Price to Reflect Quantity
    - Change Cart Count When Click to Add
        - I had problems with this and could not get it to actually add up to a total in a way that made sense in time, I would either get a long string or a function that did not work.

PUI HW 6B

Links

Active Site = https://ralajo.github.io/FluffProduct.html

JavaScript (all) = https://github.com/ralajo/ralajo.github.io/blob/main/FluffStuff.js

Cart HTML = https://github.com/ralajo/ralajo.github.io/blob/main/FluffCart.html
Cart CSS = https://github.com/ralajo/ralajo.github.io/blob/main/FluffCart.css

Product HTML = https://github.com/ralajo/ralajo.github.io/blob/main/FluffProduct.html
Product CSS = https://github.com/ralajo/ralajo.github.io/blob/main/FluffProduct.css

Reflection

This was a mess, obviously. I was really confused by most of the concepts we had to learn, which I have explained later, but the one I feel like I should warn you about, as I think it will be useful to future students is that AdobeDreamweaver does not actually run a new URL to preview the next page so NONE OF MY FUNCTIONS THAT CUT ACROSS THE PAGES WORKED AND NEITHER DID ANYTHING WITH LOCAL STORAGE FOR MONTHS BECAUSE OF THE PREVIEWER I WAS USING. This left me absolutely miserable, confused as to why everything I did, just didn't work, even if it was nearly identical to what we had learned in class.

This was a mess, but working on this cart every day did leave me with a stronger understanding of the material, and learning why nothing I did worked helped me feel a little bit more confident in what I was presenting.

Programming Concepts

1. Constructor Functions: Here, is the construction of the "Pillow" Class, a constructor is a group that has attributes and is created without specific links, but those links can be added later and they can be built new in a different spot. This demonstrates the

```
class Pillow {
    constructor(shape, color, fill, quantity, price, image) {
        this.shape = shape;
        this.color = color;
        this.fill = fill;
        this.quantity = quantity;
        this.price = price;
        this.image = image;
    }
    getPillowName() {
        return this.shape + " in " + this.color + " with " + this.fill + " filling";
    }
}
```

2. Local Storage: The concept was really confusing to me basically local storage is where you can put something into your website you can get it out on another page. You can see that an array is added to the local storage here. It requires the use of JSON.

```
function addToCart() {
    let shape = document.getElementById("pshape").innerHTML;
    let color = document.getElementById("pcolor").innerHTML;
    let fill = document.getElementById("fillinsert").innerHTML;
    let quantity = document.getElementById("pquantity").value;
    let price = document.getElementById("pprice").innerHTML;
    let image = document.getElementById("MHRBig").src;

    let fish = new Pillow(shape, color, fill, quantity, price, image);
    fish.name = fish.getPillowName();
    pillowArray.push(fish);
    window.localStorage.setItem("pillowArray", JSON.stringify(pillowArray));
```

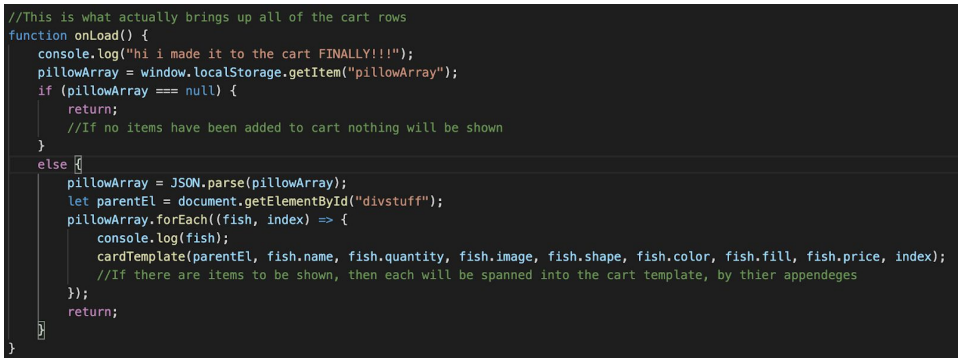
3. If Else: Here you can see an if-else function. It yields nothing for an empty array but runs a for each function that I will explain below if the array is not empty/ This concept of if-else functions is the foundation of creating arguments and this was the first time I've done it in an actual programing language.

```
//This is what actually brings up all of the cart rows
function onLoad() {
    console.log("hi i made it to the cart FINALLY!!!");
    pillowArray = window.localStorage.getItem("pillowArray");
    if (pillowArray === null) {
        return;
        //If no items have been added to cart nothing will be shown
    }
    else {
        pillowArray = JSON.parse(pillowArray);
        let parentEl = document.getElementById("divstuff");
        pillowArray.forEach((fish, index) => {
            console.log(fish);
            cardTemplate(parentEl, fish.name, fish.quantity, fish.image, fish.shape, fish.color, fish.fill, fish.price, index);
            //If there are items to be shown, then each will be spanned into the cart template, by thier appendeges
        });
        return;
    }
}
```

4. forEach: A for each function is sort of like a shortcut for understanding and depending on elements. Here we use a for each function to get all the main categories out of a pillow array and into the card template.

```
let parentEl = document.getElementById("divstuff");
pillowArray.forEach((fish, index) => {
    console.log(fish);
    cardTemplate(parentEl, fish.name, fish.quantity, fish.image, fish.shape, fish.color, fish.fill, fish.price, index);
    //If there are items to be shown, then each will be spanned into the cart template, by thier appendeges
});
return;
```

5. Index & Splice: An index is a number and which item is in an array that concept is being used to remove the rest card from the cart.



```
//This is what actually brings up all of the cart rows
function onLoad() {
    console.log("hi i made it to the cart FINALLY!!!");
    pillowArray = window.localStorage.getItem("pillowArray");
    if (pillowArray === null) {
        return;
        //If no items have been added to cart nothing will be shown
    }
    else {
        pillowArray = JSON.parse(pillowArray);
        let parentEl = document.getElementById("divstuff");
        pillowArray.forEach((fish, index) => {
            console.log(fish);
            cardTemplate(parentEl, fish.name, fish.quantity, fish.image, fish.shape, fish.color, fish.fill, fish.price, index);
            //If there are items to be shown, then each will be spanned into the cart template, by thier appendeges
        });
        return;
    }
}
```