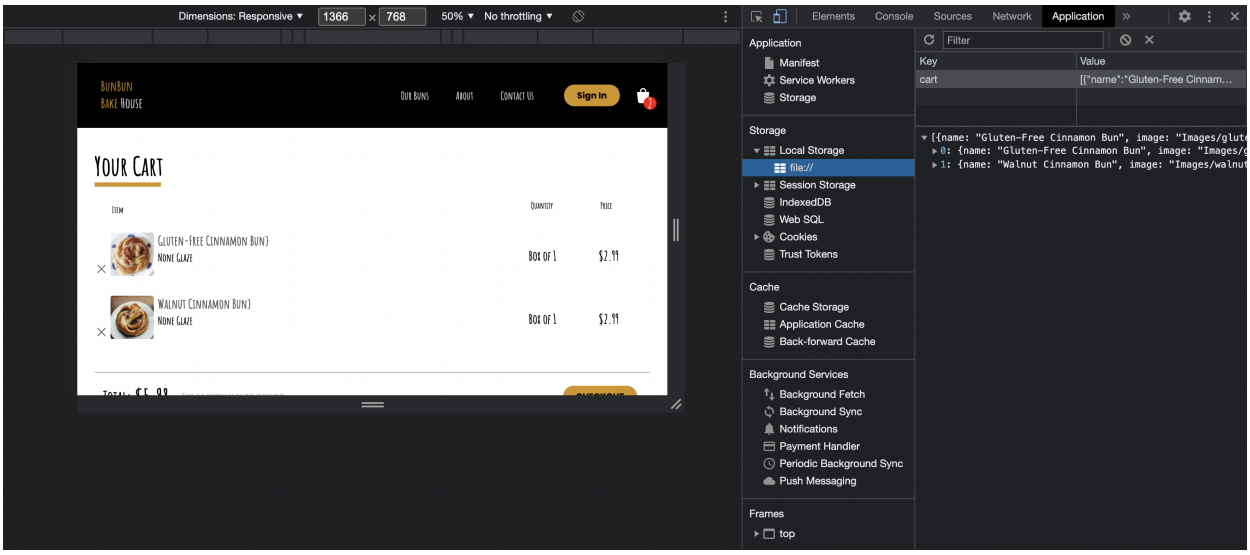
Assignment 6b

Adding Functionality to a Website with JavaScript

Errors and debugging

- **Adding items to localstorage:** When I first created a variable to store my objects and added it to the localstorage, it didn't seem to work. I then learnt that I could go to the A*pplications* tab in Inspect mode and check if the localstorage is populated. Upon seeing that it was, I then realized that the issue was with getting the items from the localstorage into my cart. I figured out that I had to retreive my cart array from within localstorage and then use each element in the array to set as an individual cart item in my cart. I learnt that in the future, I could always go to the *Applications* tab and check if my localstorage has data and see how the data is structured.
- **Removing items from the cart and localstorage:** Adding items was easy, but removing items needed some thinking. At first, when I hit the 'remove' icon on one element, the entire cart would disappear. I added a *console.log* in my remove function to figure out that all the HTML elements were vanishing. Some *Googling* and *StackOverflowing* showed me that I had to have a unique id for each of my elements in order to remove them individually from my cart. So I added a counter which would increment in my for loop for each new item which would be added to the array of items which reflected in the cart. This allowed me to easily remove each cart item based on its unique id(count). From this error, I learnt that I could use *console.log* as a checkpoint in multiple points of my code to debug and see how each function works.

Checking localstorage



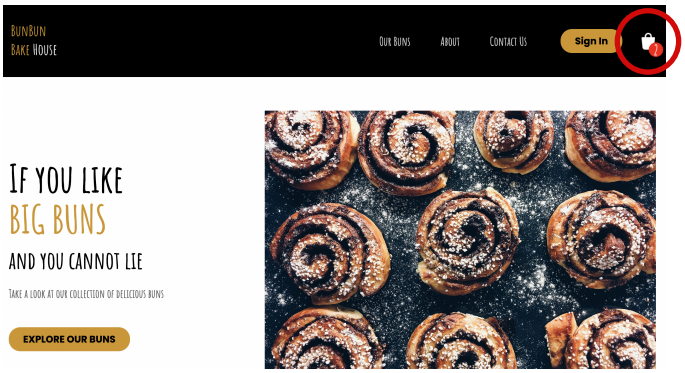
Localstorage contains the cart array with all object items in it

All images were taken from Unsplash.com

- **Implementing the Google Map feature:** I decided to have a Google Map feature in the Contact Us page for users to easily find out where the bakery is and to quickly get directions to the bakery. As I set about trying to implement this feature, I added an image tag with a Google Maps screenshot of a location. But that wasn't intuitive enough. So I tried to use an API. I copied the embed code from Google's map API website into my HTML but it wasn't displaying the map. There was no error on the console either. I tried to debug this issue using some standard techniques like checking the code again, the syntax, using console.log but none of them seemed to show me what the error was. Upon further research, I learnt that I need a specific API key to use the API. I figured out the process of getting an API key- I had to sign up and create a Google Cloud account. This process taught me how I could use APIs in the future and it's something I think I will always remember.

Programming Concepts

1. OnLoad: In order to check the state of my shopping cart and get the length of the array in my localstorage for the cart counter, I wrote a function which would find out if the *cart* variable I had in localstorage was empty or not and appropriately fetch items from it. However, this had to be done everytime the page loaded. I found out about the **onLoad()** function and learnt that onload is most often used within the <body> element to execute a script once a web page has completely loaded all content. Now, the cart counter automatically updates with the latest value even when the user moves across pages.

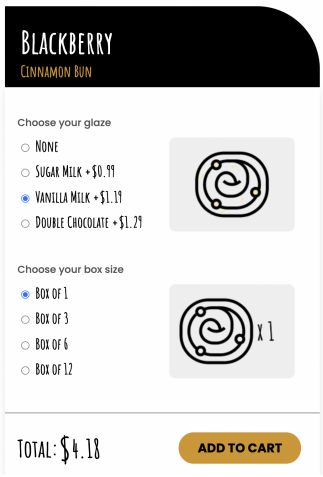


2. Nested functions: There were many instances where I created a function and had to call it but through another function on an **onClick** event. At first, I tried calling two functions on an onClick event like <button onClick="function1(); function2()"></button>. But then I learnt that we can invoke a function from inside another function.

All images were taken from Unsplash.com

This made it so much simpler to manage my HTML code and separate the HTML from the JavaScript. Here, I'm calling getDefaultPrice() within selectGlaze() based on the bun selected.

```
JS main.js > displayAlert
1   var defaultPrices = [1.29, 2.99, 2.99, 2.99, 1.99, 1.99];
2   //parseFloat(document.getElementById('price-calc').lastChild.innerHTML); //"7.99"
3   var defaultPrice;
4   var updatedPrice = document.getElementById('price-calc').lastChild.innerHTML;
5   var count;
6
7   function getDefaultPrice(){
8       if(document.getElementById('bun-name').innerHTML=='Original'){
9           defaultPrice=defaultPrices[0];
10      }else if(document.getElementById('bun-name').innerHTML=='Blackberry'){
11          defaultPrice=defaultPrices[1];
12      }else if(document.getElementById('bun-name').innerHTML=='Walnut'){
13          defaultPrice=defaultPrices[2];
14      }else if(document.getElementById('bun-name').innerHTML=='Gluten Free'){
15          defaultPrice=defaultPrices[3];
16      }else if(document.getElementById('bun-name').innerHTML=='Pumpkin Spice'){
17          defaultPrice=defaultPrices[4];
18      }else if(document.getElementById('bun-name').innerHTML=='Caramel Pecan'){
19          defaultPrice=defaultPrices[5];
20      }
21      return defaultPrice;
22  }
23
24  function selectGlaze() {
25      getDefaultPrice();
26      var price = defaultPrice;
27      var image;
28      if(document.getElementById('none').checked){
29          image = 'Images/Glaze-none.png';
30      }
```

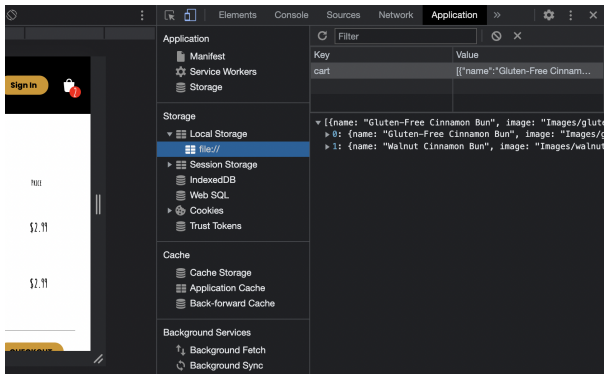


3. Storage: To implement the cart and make it functional, I had to learn about storage. There are two kinds of storages- local and session. In order to maintain the data across all pages and even on page refreshes, I decided to use localstorage. I had to learn about how I could set and retreive information from the localstorage. As I dove deeper, I also learnt about concepts like **JSON.stringify** and **JSON.parse** which I used extensively to work with localstorage. These essentially structure data in a way which makes it easy to operate on (append to cart variable and read from cart variable)

```
}

function removeItem(count){
    document.getElementById(`cartItem${count}`).remove();
    cart = JSON.parse(localStorage.getItem("cart"))
    cart.splice(count,1)
    localStorage.setItem("cart", JSON.stringify(cart));
    if(cart.length == 0){
        document.getElementById("empty-cart-text").style.display = 'block';

    }
}
```



4. Delays: While I was trying to implement the automatic slider for bun recommendations, I didn't know how I could get the slider to automatically move every x seconds. I implemented the buttons to manually click and move them and thought that I could find a work-around by having them be clicked automatically. Upon some exploring of JavaScript documentation, I learnt about the delay parameter which can be added to a function.

All images were taken from Unsplash.com

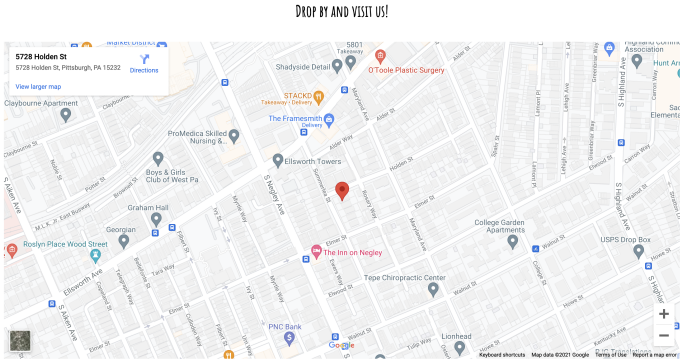
By adding the delay parameter to my function, I was able to set the slider to move to the right every 3000ms (3 seconds).

```
function nextSlide() {
  if(count < items) {
    slideList.style.left = "-" + count * sliderWidth + "px";
    count++;
  }
  else if(count = items) {
    slideList.style.left = "0px";
    count = 1;
  }
};

setInterval(function() {
  nextSlide()
}, 3000);
```

5. APIs: I decided to have a Google Map feature in the Contact Us page for users to easily find out where the bakery is and to quickly get directions to the bakery. To get started, I copied the embed code from Google's map API website into my HTML but it wasn't displaying the map. Upon further research, I learnt that I need a specific API key to use the API. I figured out the process of getting an API key- I had to sign up and create a Google Cloud account. This process taught me how I could use APIs specifically for HTML or JavaScript. The Google Cloud documentation has various kinds of APIs for each language.

```
<div class="map">
    <iframe
        width="1280"
        height="600"
        frameborder="0" style="border:0"
        src="https://www.google.com/maps/embed/v1/place?key=AI
    </iframe>
</div>
```



All images were taken from Unsplash.com