Reflection

As I am unfamiliar with Javascript and the interaction of Javascript with HTML/CSS, I was bound to encounter many errors as I worked on this assignment. I faced some errors while dealing with different data types. Since I am more accustomed to working with programming languages like Java that require defining data types, I was a bit confused with Javascript's *var* and *let*. In one instance, I tried to update the total number of products on the page by adding the quantity of the selected product to the total number of products. However, I received an error every time I tried to add the two numbers together. This is because the quantity I received from the form data was actually a string. To fix this problem, I converted the form data to an integer and successfully added the two numbers together. In the future, I should pay more attention to dynamically typed variables. If I'm not sure what the contents of a certain variable are, I can print them to the console to double-check as well.

Programming Concepts

1. Constructing objects

   I created the Product object in Javascript to help store product-related information like type, quantity, and glaze. This helped me create an array of Products so that I could store products in my cart in one data structure along with their different properties. By extension, the object made it possible for me to store product properties in local storage.

2. Creating functions

   In this assignment, I created many Javascript functions to support the interactivity of the cart page and the product detail page. For instance, I created the *getCartNumber()* function to fetch the total number of items in the cart from local storage and update the navbar to reflect the total number of cart items. I call this function from the HTML file of each page because each page contains the navbar and displays the number of items in the cart within it.

3. Manipulating HTML DOM with Javascript

   As I mentioned from the previous concept, I learned how to manipulate the HTML DOM directly with Javascript. *removeItemFromCart()* is a function that I wrote for this assignment that illustrates this concept. This function is called from the button element in the HTML file. The function then traverses through the different element nodes until it reaches the element that contains the HTML that displays the item to be removed. Finally, it removes that element from the DOM.

4. Using local storage to pass data from page to page

   One challenge I faced was using JSON and local storage to pass data from page to page. I created the functions *sendToCart()*, *getCartNumber(), and getCart()* to send quantity and product data to local storage, receive quantity from local storage, and receive product data from local storage respectively. Once I learned the correct syntax and usage of *setItem* and *localStorage,* storing data became easy. I learned how to use *onload* and *onunload* to call those functions from HTML files to ensure that my webpages were displaying data that was located in local storage.

5. Linking Javascript functions to HTML elements

   Lastly, I learned how to trigger Javascript functions by linking them to HTML attributes. I briefly touched on this in my previous point. Learning how to use *onload* and *onunload* was important to send and receive information from local storage. Without using *onclick*, several key functions would not work. For instance, the function *addToCart()* only triggers when the "add to cart" button is clicked on the product detail page. If it triggered before, the user might not have updated the form to reflect the quantity and glaze type that they want to order and the wrong data would be sent to local storage.