**Lea Emerlyn**
**Assignment 6B Reflection**

**Reflection**

Initially I set up the HTML by putting the script tag as a child of the head tag. When I ran my javascript code, nothing happened. I resolved the problem by looking at HTML templates to see where they put the script tag. Then, I moved it to the bottom, inside the body tag. Later, I learned that I could actually put the tag in the head by using the async keyword.

Several times, I ran into runtime errors, where the code did not execute. I resolved these errors by checking the console in the browser. Often, the errors were due to trying to get an element that did not exist on the current page. To help with this, I split my code into several different files, one for each page with distinct functionality.

I was stuck for a little bit, while making the purchase list on the cart page. Each element in the purchase list contained many sub-elements such as price, type of item, image, etc. Also, I had to generate a different series of purchase displays depending on what the customer bought. I started by creating a sample purchase display in HTML/CSS. Based on that, I created a javascript function to generate the purchase information using the key data such as product name and amount. Using this function, I looped through the saved purchase data to generate all the purchase displays. I learned that it helps to split up a big task into smaller functions.

**Programming Concepts**

1. I learned how to use the DOM tree structure and append children in javascript to generate complicated HTML objects for the cart purchase list. I looked at how I coded the HTML sample. Then, looking at which element is a child of which element and paying attention to what class each element is in, I was able to use javascript to generate the cart list.
2. I learned how to save data in session storage. I used this to save the amount of product in the cart as well as the name and amount of each specific product that the customer added to the cart.
3. I learned how to register event functions on buttons and dropdowns such as onclick and onchange. In addition to onclick for buttons, I used onchange to have a text display automatically update when the user changed the dropdown on the product details pages.
4. I learned how to use a dictionary to map product names to product prices in order to calculate the total checkout price using the data about purchased products saved from previous pages.
5. I learned how to use an array to map the dropdown indices from the glazing dropdown to a unique description string for each glazing type to help with automatically updating the description based on the drop down index.

6. I learned how to use nested loops to test every combination of product name and glazing that might be saved in the session storage. Using loops, I combined the name and glazing strings to create the keys to index into the session storage. This was much faster than enumerating all of the possible combinations manually.