HW 6b
Cydney Vicentina
Javascript
Reflection

Live Site Link: https://cydneyjv.github.io/assignment_6/productbrowsingpage.html
Github Repo Link: https://github.com/cydneyjv/cydneyjv.github.io

Reflection:

The first issue I encountered while completing this assignment was the addition of a local storage system. My experience with Javascript is pretty limited (only from the summer programming class which mainly focused on Python). I actually didn't understand the lab example code for a long while. The first time I implemented the local storage, I was only able to print "[object Object]" with no text. I was so confused because I knew I was pushing a simple string object into the local storage. One thing I learned was just how useful console.log() is. I never had the need to use it before since document.getElementbyId() is pretty stable. But, when I used to code in R, I used to use print() functions all the time. After learning this trick, the code was much easier to debug and also to see exactly what the computer was interpreting from my code. I was definitely scared to learn a new set of functions using Javascript. But, after messing with the code for a long time now, I feel much more confident, especially with the help of console.log(). Let me also say that taking a break from looking at your code is majorly effective. There were several days where I started on the wishlist function (or the cart function too) and worked on the code for several hours and nothing was working or progressing. Then, I took a break (sometimes a few hours, sometimes a few days) and when I looked at the code again, it magically started working. While I know this definitely wasn't magic, looking at the code with fresh eyes (and a rested mind and a full stomach) helped more than I thought.

The next set of issues had to do with the bug of setting a null object's innerHTML (or any other attribute). I couldn't understand why my code wasn't working, especially after reading the code over and over again. So, I re-read the instructions and decided to look into Chrome's Web Developer Tools. Complete game-changer. The console there was able to provide an actual error message (which didn't happen before, things just got broken and/or didn't work). These error messages also point you back to the exact line of your code where the issue is. That was super helpful for debugging and just understanding what I was trying to do and why the computer didn't like it. One of my favorite aspects of the Web Development Tool was the Explore function where, as your mouse hovers over different objects, a tag will pop up with the element's ID (if applicable), class, and name (innerHTML). This was absolutely necessary as I was formatting my shopping cart page since the way I set it up was that each press of "Add to Cart" (on the product browsing or product detail pages) appended a new child item. In order to change each child element, I had to add unique IDs to each child and change the innerHTML from there. One time, after adding in a delete function, my ID numbers were messed up (didn't

delete the element from the array) so things weren't updating. Only from the use of the Web Developer Tools was I able to figure out this error.

Finally, a small issue that I had to overcome was just the length and complexity of my files. Compared to the first coding assignments, I now had hundreds of lines of code to scrape through and try and understand. Mentally, it's a lot to remember. I never really appreciated the use of comments until this assignment. I was reading comments from my old code (all the way back to my summer programming class!) to help me interpret code that I wrote. It literally felt like someone else wrote the code and I was playing catch-up. But, the comments helped immensely especially because I tend to write weird function or variable names like "add1o()" for the add to cart button on the o-riginal product detail page. I am sure in a few weeks, I'll forget that because it's so obscure so adding the comments was and will be a life saver. It's an annoying process to go through but it's better to do it now while I still have a grasp of my code versus the next time I need the code and can't use it because I can't understand it.

Programming Concepts:
a. Demonstrate 5 programming concepts that you learned in Javascript and used in this assignment with an example.

1. Initializing and Using Local Storage between Web Pages
    a. I never used the local storage feature on Javascript and I didn't know that it was even an option to be used (especially since Patrick recommended using AJAX or JSON functions).
    b. You can see the use of getting and setting local storage items all over my script.js file ("add1o()" - line 507, "addonec()" - line 487, etc). I basically used it everytime I needed to add an item to the cart and remember to show it on the shopping cart page.
2. Stringify + JSON.parse() - Converting Between Types (String + Numeric)
    a. In order to use the local storage feature, I also had to learn about these two functions since local storage only stores strings (though you usually want to store objects so you have to convert them to a string then parse it back to an object when you want to use it).
    b. You can see the use of stringify and JSON.parse() all over my script.js file in the same functions ("add1o()" - line 507, "addonec()" - line 487, etc) as well as on the top of the cart.js file to get the arrays back as objects.
3. For Loops + object.id = i - Loops and New Element Attributes
    a. As I mentioned before, in order to add unique cart items, I needed to add unique IDs to each element (quantity, name, and price of the items) in order to update the information. To do this, I had to find all the elements I wanted to update (i.e. with the class "quantity") and add an ID to each. This was also one of the first times that I used classes in HTML because I don't really find classes useful for CSS (in most of the cases I've coded for). But, to select all the items with a specific class and attach specific IDs to them worked really well for me.

    b. You can see this application as early as line 32 in cart.js but it shows up in many places throughout cart.js. In line 32, I was adding IDs to the quantity items (leftmost div of the shopping cart page).
4. clickedElement + this - Using New Function Parameters
    a. As I was trying to figure out how to change attributes of an object, I realized that I needed to be able to get the ID of a particular element. In this case, I needed to get the ID of a button when it was pressed. For example, when people want to delete a shopping cart item, I need to know which specific item the user wants to delete. Through a Google search, I found that this can be done by setting function(**this**) in the HTML onclick() and function(**clickedElement**) in the Javascript file.
    b. You can see one example of this on line 120 of cart.js in the subtract() function. This function deals with subtracting a quantity (i.e. if you want to buy 3 cinnamon rolls instead of 4, you can use the minus button to lower the quantity). To get the ID of the item to update, I used clickedElement and this in the Javascript and HTML files.
5. Number() - Using Numeric Functions in Javascript
    a. Finally, to update the quantity and price elements, I needed to use a function to convert strings to numbers. It's kind of funny because I'm used to R where numerical types are the default and strings require quotation marks. But, in Javascript, numbers are interpreted as strings. I was confused for the longest time because when I tried to add prices, I always got "NaN" and console.log() was showing me the correct number. Took another Google search to figure that one out.
    b. You can see this in action in line 130 of cart.js in the subtract() function. It also shows up in the add() function. I used it mainly to update the quantity and price elements as well as to calculate the total price.