Kristel Grace Yu kty

REFLECTION

From this assignment, I believe my greatest takeaway was how much websites are currently being developed with numerous features to help the user out as much as possible. For the websites I use, I can easily see how more complicated features were implemented in place of easy but difficult to understand features. More specifically, some complicated features that I personally found difficult to create was the editability of my shopping cart page. I had trouble cloning the items to create more boxes for each item, and this was eventually resolved by creating a “deep” copy to ensure that each copy was not dependent on each other. I also had a lot of trouble with sessionStorage. It was at first difficult to determine where to set the storage to ensure that changes made are saved. In addition, the types that sessionStorage accepts is very limited so using JSON.stringify was a key element that helped in the following things that needed to be saved in storage. This was also eventually solved by finding the elements that were creating issues by using print statements (console.log). Using Chrome Developer Tools was essential in creating the website. I initially used localStorage, but sessionStorage was eventually the better choice for building my website because it allows me to reset whenever I close the tab. Moving forward, I believe that I will gain a lot from watching YouTube tutorials and reading up on stackoverflow on concepts I find particularly difficult when looking into outside resources; I also think it is very important to continue as creating print statements to mitigate the time used for debugging and find the source of the error immediately.

PROGRAMMING CONCEPTS

1. Setting and getting storage
   I learned how to set and retrieve items stored in session storage during this assignment. I learned to first set items and update these each time necessary, especially when the page loads. A specific object or string is first set in session storage, then when changes to it are necessary, it is set once again to save the updated version. This is retrieved when needed, especially for logic and for reloading the page, to ensure that data is saved, much like users may refresh the page and expect data to stay. I used this for the shopping bag count and displaying the items in the shopping cart, as shown below.

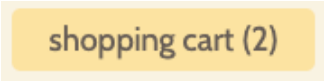


Image 1: Shopping cart count. The count is triggered when the user clicks “add to cart” in the product detail page and can be viewed across pages.

Image 2: Item in shopping cart. When a user adds an item to cart, it is stored in an array and displayed on the shopping cart page individually. Changes to the item is also saved to sessionStorage.

2. Creating and using functions
   I also learned how to create functions that are triggered by certain events, such as when the page loads, when a dropdown menu selection changes, when “add to cart” is clicked, etc. Implementing each one was very different, from updating the price when the index of the dropdown menu changes, or even removing an item from the shopping cart page. I learned that functions are triggered by certain events, and these events should make the users’ interaction with the website easier and faster.

Image 3: Item in shopping cart. When the user changes selection in the dropdown menu, the price automatically updates and the changes for the item is saved to storage. The dropdown menu is triggered onchange.

3. Getting elements from HTML
   I also learned how to get elements from HTML pages. Retrieving elements from HTML differed slightly when using classes and ids, and I initially did not know this. After knowing this, retrieving classes and ids was much easier. This understanding has also led me to further understand the difference between classes and ids, that ids are meant to be unique and classes are meant to be used multiple times. Getting elements from HTML also helped with styling when necessary, retrieving the value from dropdown menus and input fields, changing text to reflect updated prices, etc.

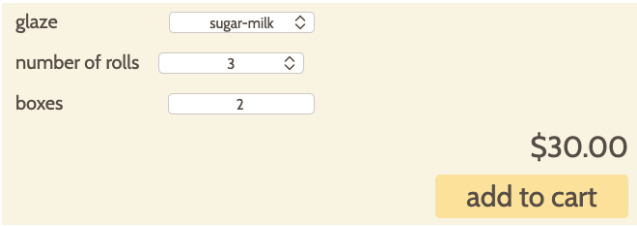


Image 4: Partial screenshot of product detail page. Prices were updated by retrieving the text in the HTML by using retrieved HTML on the quantity for rolls and boxes. Getting the text for the dropdown menu on the selected glaze enabled the switching of images depending on the selection.

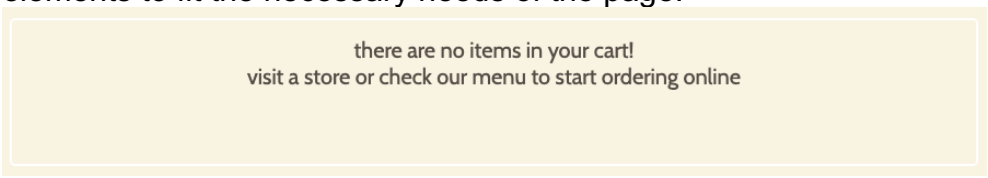
4. Changing index of dropdown menu
   I was initially inexperienced working with dropdown menus and thought that while it looked neater on a page, it was more difficult to implement than checkboxes or radio buttons. In the shopping cart page, I implemented the page so that users can easily edit their desired quantities and selections just as easily as they did

when adding items to their cart. This included updating the index based on the items in the cart.

Image 5: Item in shopping cart. I retrieved the item in the cart from an array saved within sessionStorage. I then used this to update the index number of the attributes of the item to reflect the selection. This change is immediately saved to sessionStorage.

5. Adding HTML elements

I also learned how to add HTML elements from JavaScript only. This was most important for creating the items in the cart, as we cannot assume how many items a customer will order. This said, I added, removed, and edited HTML elements to fit the necessary needs of the page.



Item 6: Shopping cart page when no items in the cart. I edited the HTML code to show when there are no items in the cart by changing the innerHTML of the div.

Item 7: Item in shopping cart page. Each item is individually shown. I added HTML elements by cloning a sample item depending on the number of items and updated each element manually.