Assignment 6 Section A Xiangzhu Chen

InVision prototype: <https://invis.io/6ZOUENL2RWV>

For reviewing the live website, please open index first, then go to browsing page, then click through every product so that the data can be loaded.

Reflection

**The challenges of coding and how I overcame them**

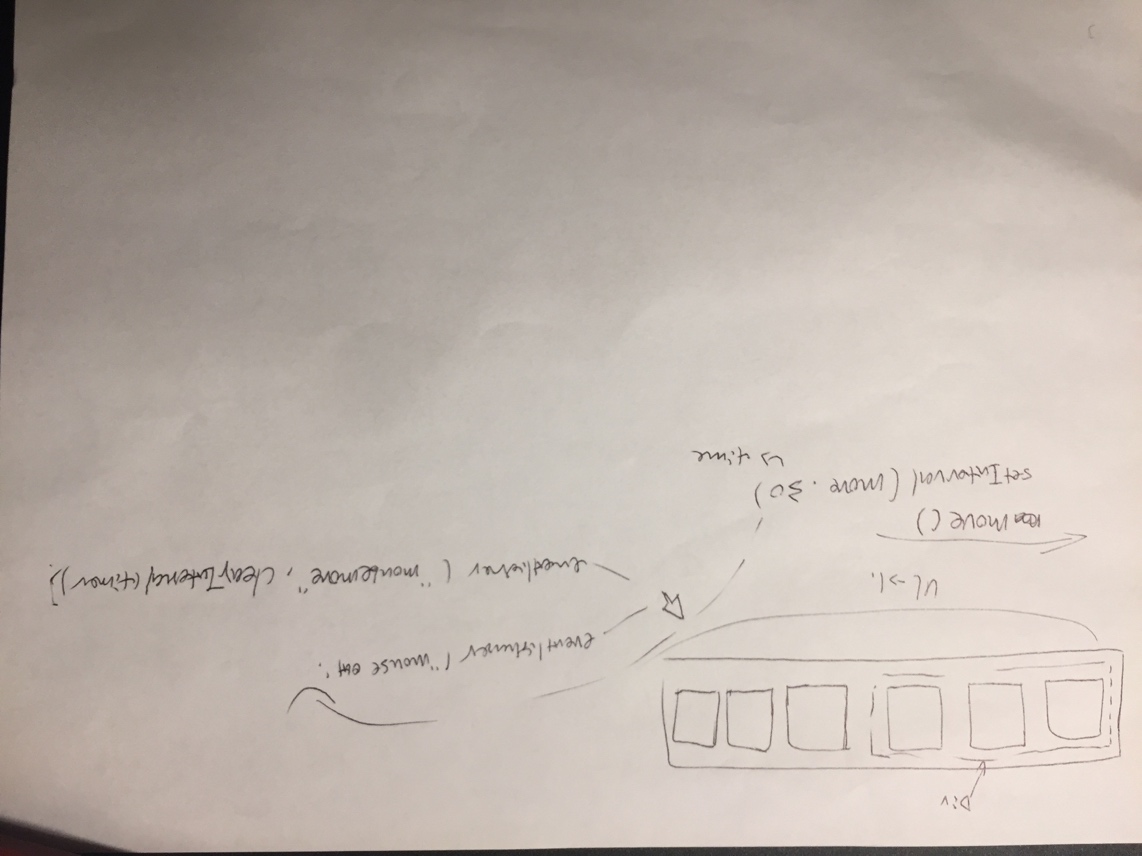
The first challenge I encountered was how to grab the value of sizes and colors from the product detail page because they were images initially, which did not have values. To overcome this problem, I changed them from images into divs, and then assign them with text values in the productLsit array later when I tried to combine multiple detail pages into one, which will be explained in more details later.

The next bug that gave me a headache was that after grabbing the values I needed and writing the “add to cart” function, which created new items and pushes them into the shopping cart array, I found in the console that all the objects in the array had undefined properties. This meant that none of the values I grabbed were passed in. I became so confusing about the disappearance of these values, so I reached out to my friends, who pointed out that the values needed to be hold somewhere before passing them into the addtocart() function. After discussing this with my peers, I solved this problem by creating a current cart array first, and then store all the properties such as name, size, price, etc. to it, then pass them while the user clicks add to cart button. While looking back, this seems like an easy fix, however at that time it really took me some time to understand where the values were at between the “click color/size” event, and “click add to cart button” event.

Store the shopping cart array to the local storage was not very difficult, however, removing them after rendering the list on shopping cart page was hard to tackle. The major problem was, how could I make sure that the website takes out the item the user selected? How to locate it in the shopping cart array? I overcame this challenge by assigning index to each item while rendering the list using renderList() function, and then pass in the index to the remove() function. This fixed the problem quite well because the index could add on to itself. It was also amazing that I could write html structure under the renderList() using jQuery. It was beautiful to see different codes could work together seamlessly.

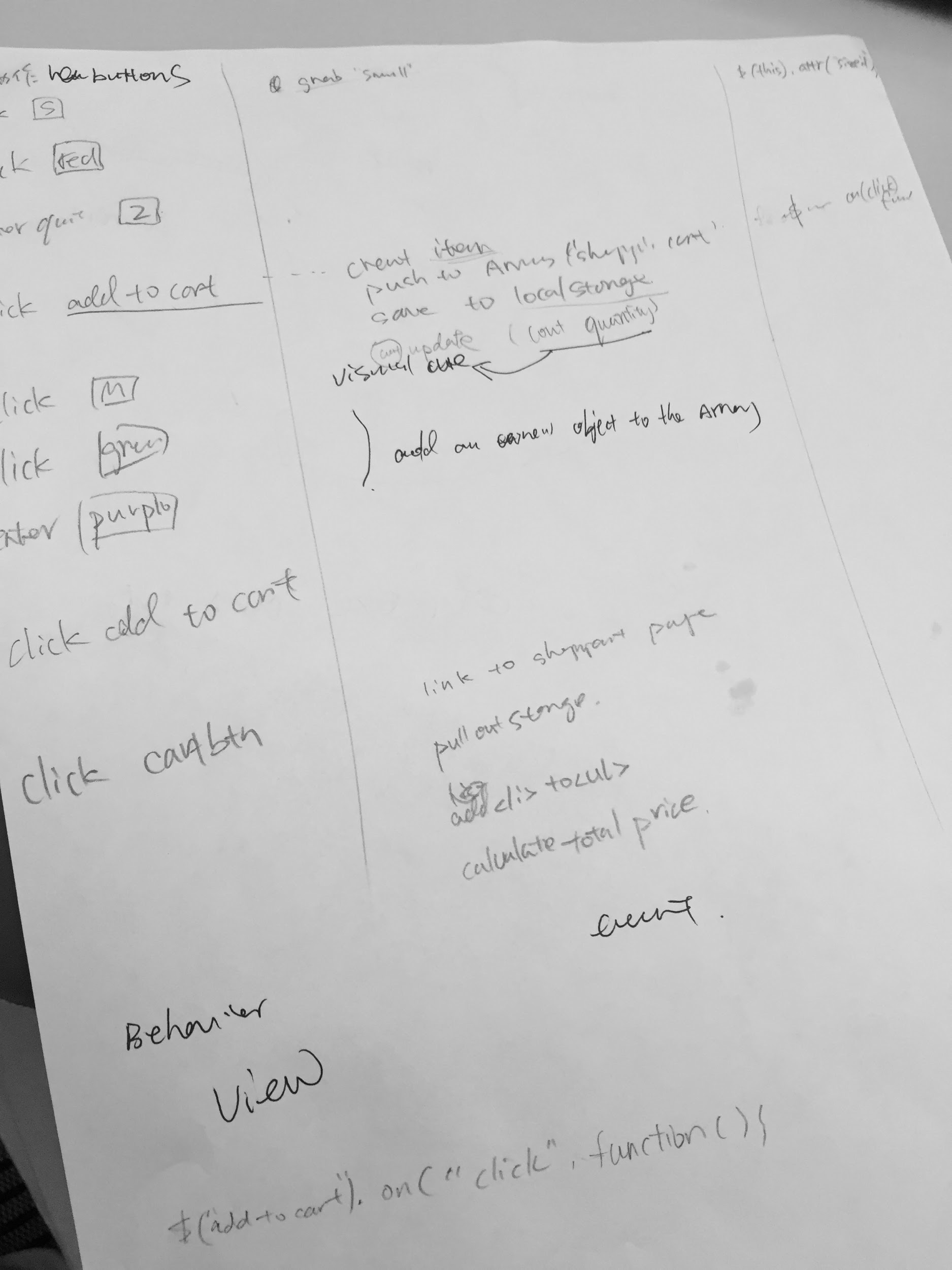
After finishing up most of the functions, I realized that having individual pages for each of the product caused repetitive codes in js, this when dealing with the next big challenge, which was to combine all the detail htmls into one, so that there were not repetitive function codes for each of the detail page. I did this by creating an array and put all the products, associated with their name, image, size, color(inside color there was another array), into this array, and get them whenever I needed them by passing in productid every time the user opened up the detail page from the browsing page.

For implementing the carousel, I watch a free class of it to figure it out. Basically, the big idea was to move a ul at a certain speed with a certain amount of time interval. Additional to that, eventlistener helped to detect user’s mouse behavior and thus stop/resume the scrolling.



As for bugs related to CSS, how to place the visual cue red dots above the shopping cart button took me a while to figure out. I successfully overlaid them on top of the button by using negative value for the position.

**Takeaways**

1. Start early: I found myself spend much more time debugging than writing up the initial individual functions. The real challenges came when I tried to link these functions together and make the overall logic flow. Next time I would first draw out logic clearly on paper, then use comments to layout them in the codes, then dive into the actual coding process.
2. MVC is very helpful. I unconsciously used this framework while trying to make connections between user’s behaviors, in other words the events happen on the page, to the codes behind them. After I draw out links on the page, I got a better understanding of how event-based programming is like, and the separation of concern principles. (on the right is my sketch of relationship between behavior, events, codes)

**Citations of tutorials**

<https://www.youtube.com/watch?v=ES1jky86JLQ&list=PLoN_ejT35AEhzNoPStBzAkpqAu3YQwPj7&index=9>

This is a major tutorial of shopping cart js codes I watched. It was really helpful, and after watching this list of videos every day for one week, I successfully learned using objects, local storage, array, and major functions associated with them, such as counting the quantity and price.

<https://love2dev.com/blog/javascript-remove-from-array/>

<https://stackoverflow.com/questions/47032704/multiple-conditions-using-an-and-if-statement-javascript>

<https://www.imooc.com/learn/386> (This is a free class of how to make a carousel)

<https://stackoverflow.com/questions/1735230/can-i-add-custom-attribute-to-html-tag>

<https://stackoverflow.com/questions/554273/changing-the-image-source-using-jquery>

<https://stackoverflow.com/questions/11563638/how-do-i-get-the-value-of-text-input-field-using-javascript>

<https://www.w3schools.com/html/html_tables.asp>

<https://www.w3schools.com/jquery/sel_last.asp>

<https://www.oreilly.com/library/view/javascript-cookbook/9781449390211/ch12s13.html>

<https://www.w3schools.com/js/js_loop_for.asp>

<https://www.w3schools.com/jsref/met_storage_getitem.asp>

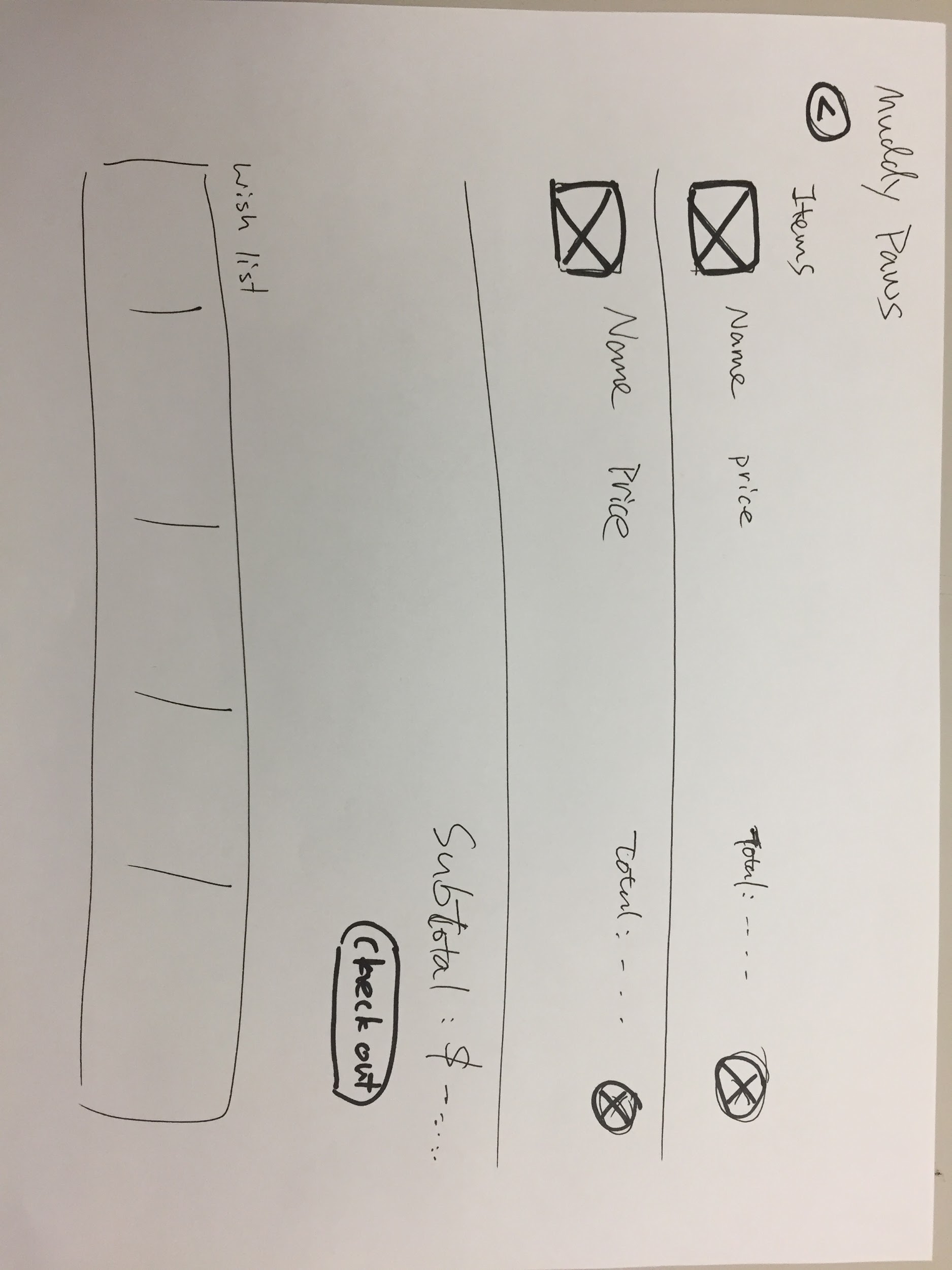
Lo-fi and Hi-fi Prototype Write Up

**The shopping cart**

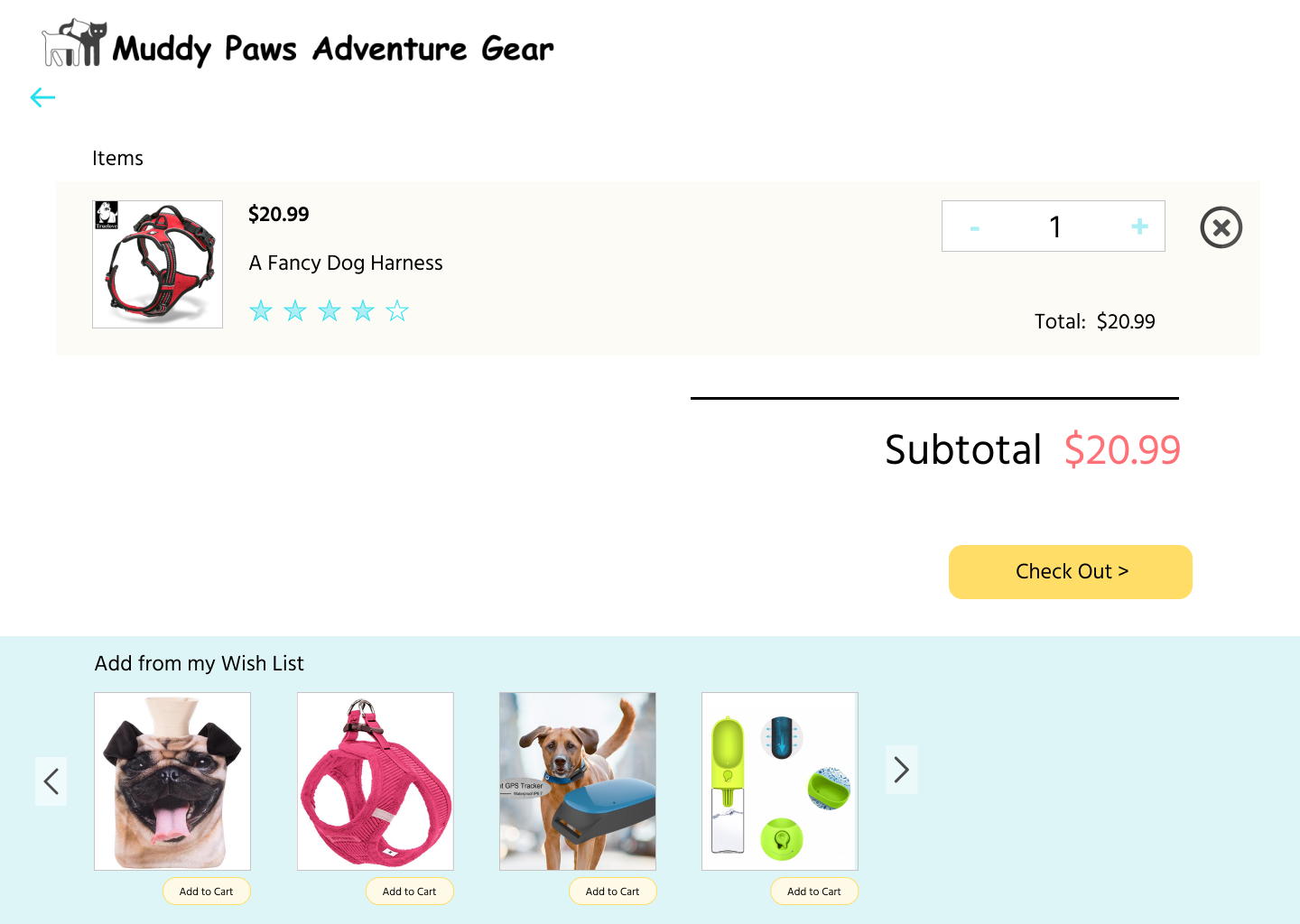
**Iteration 1**

I designed a page of the shopping cart list page last time for the InVision prototype, however it was a very rough one. There was a list of wish list items for customers to select, however, after testing this iteration with peers, I got few negative feedbacks of the wish list at bottom. Users felt that there were too many buttons which were overwhelming together with the information above. They wanted to be able to concentrate on the products they had selected. They also felt it created hassles when going back and forth between product detail page and shopping cart list page to change the quantity of a certain product.

There were also lack of important information on this iteration. Users wanted to see more details of the product they selected, for example the size and color.



Lo-fi sketch of shopping cart list



hi-fi prototype of shopping cart list (added in a quantity change box, so that users can easily change the quantity without going back)

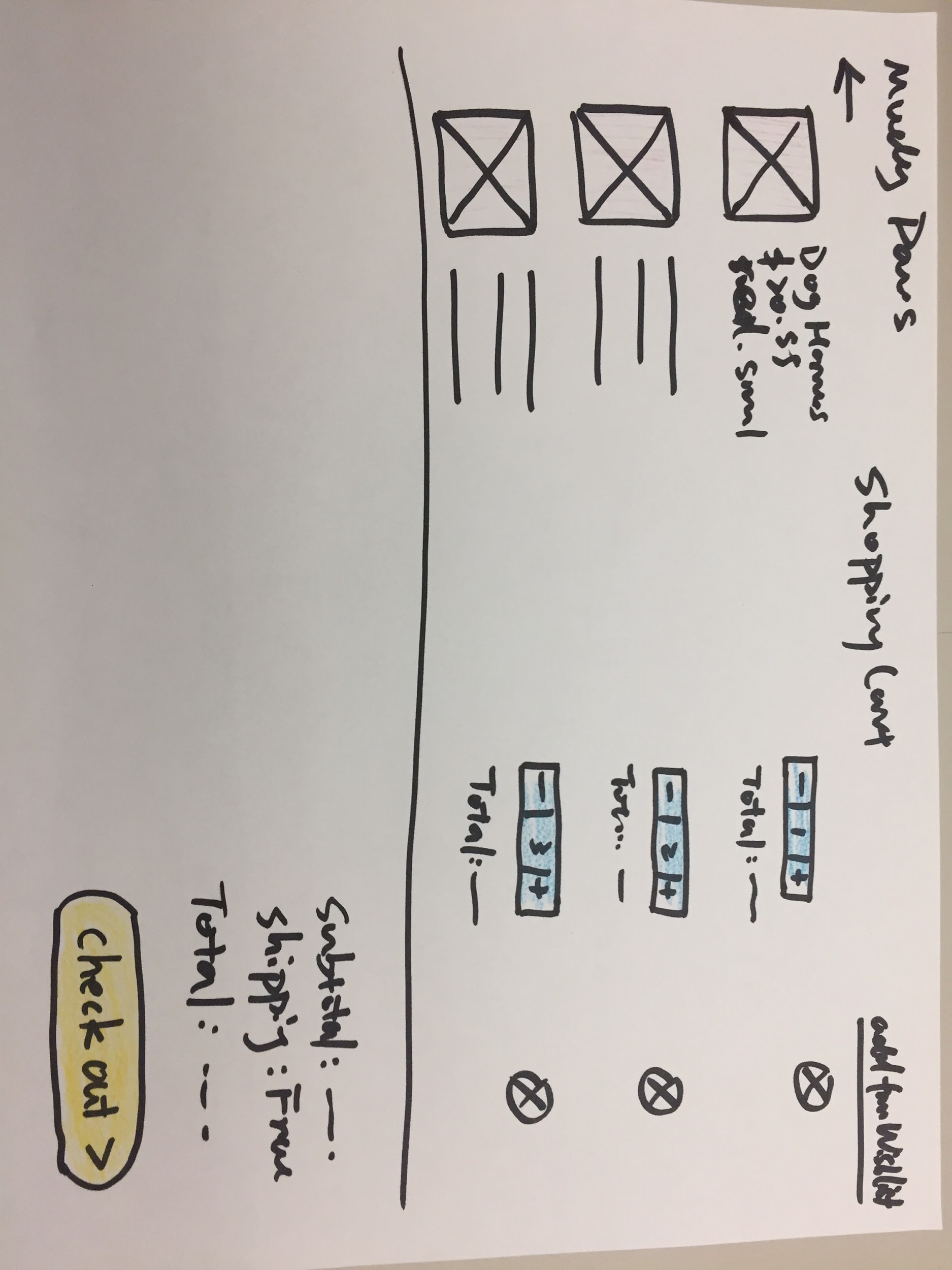
**Iteration 2**

In this iteration, I got rid of the wish list scroll at the bottom, and made it into a separate page. The reason for this change is to reduce distractions and help users focus on checking out their selected products.

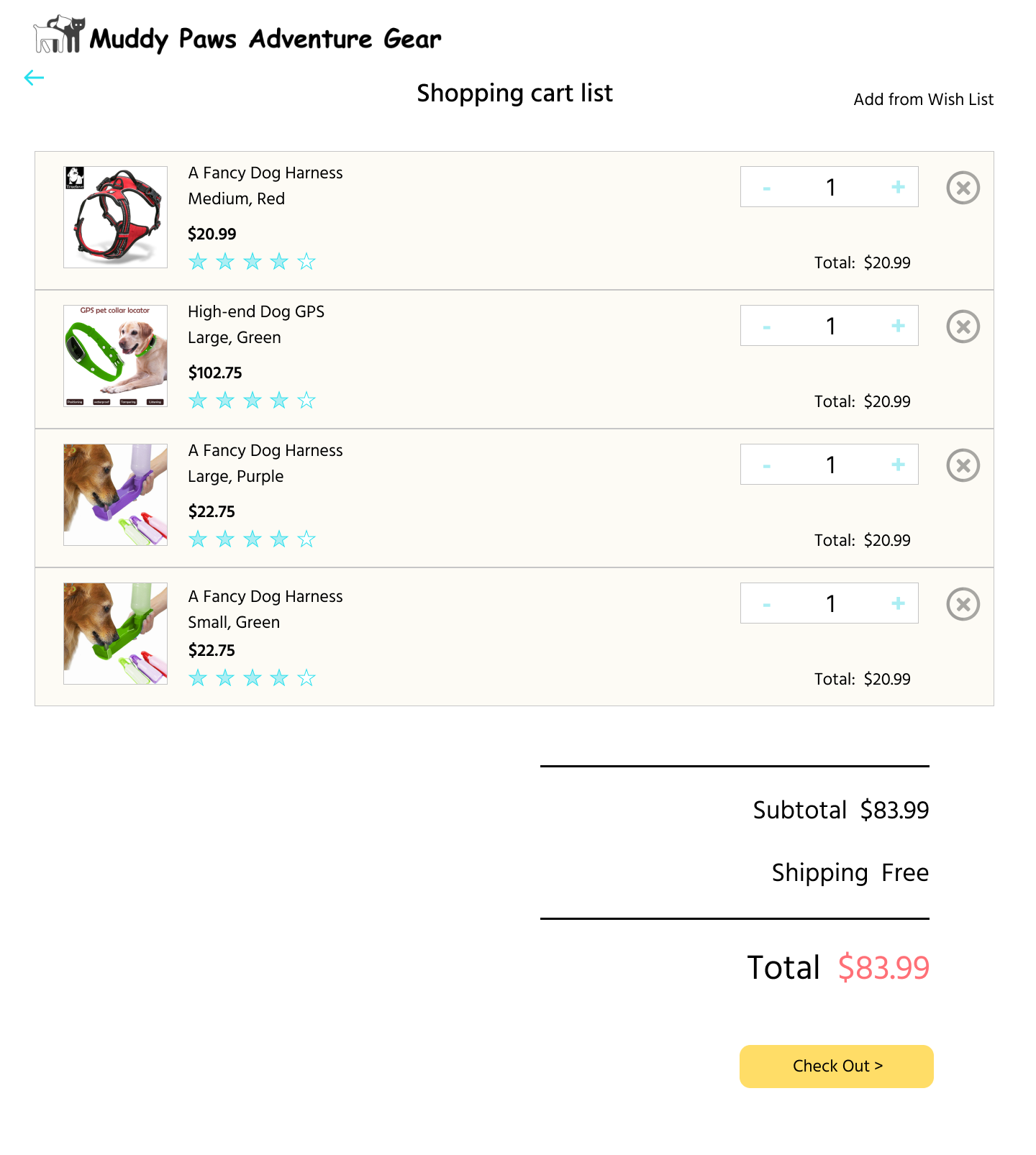
In addition to that, I added size, color, shipping price (in this case free) under each row of selected products to give more information to users for them do better review their choices and make a checkout decision.

After adding in the wish list, I also added an entry to it on the shopping cart page, so that users can add in their favorite items when needed.

Above design decisions were made largely due to the purpose of providing users a clean, easy, and flexible way of reviewing/adding/deleting selected products.



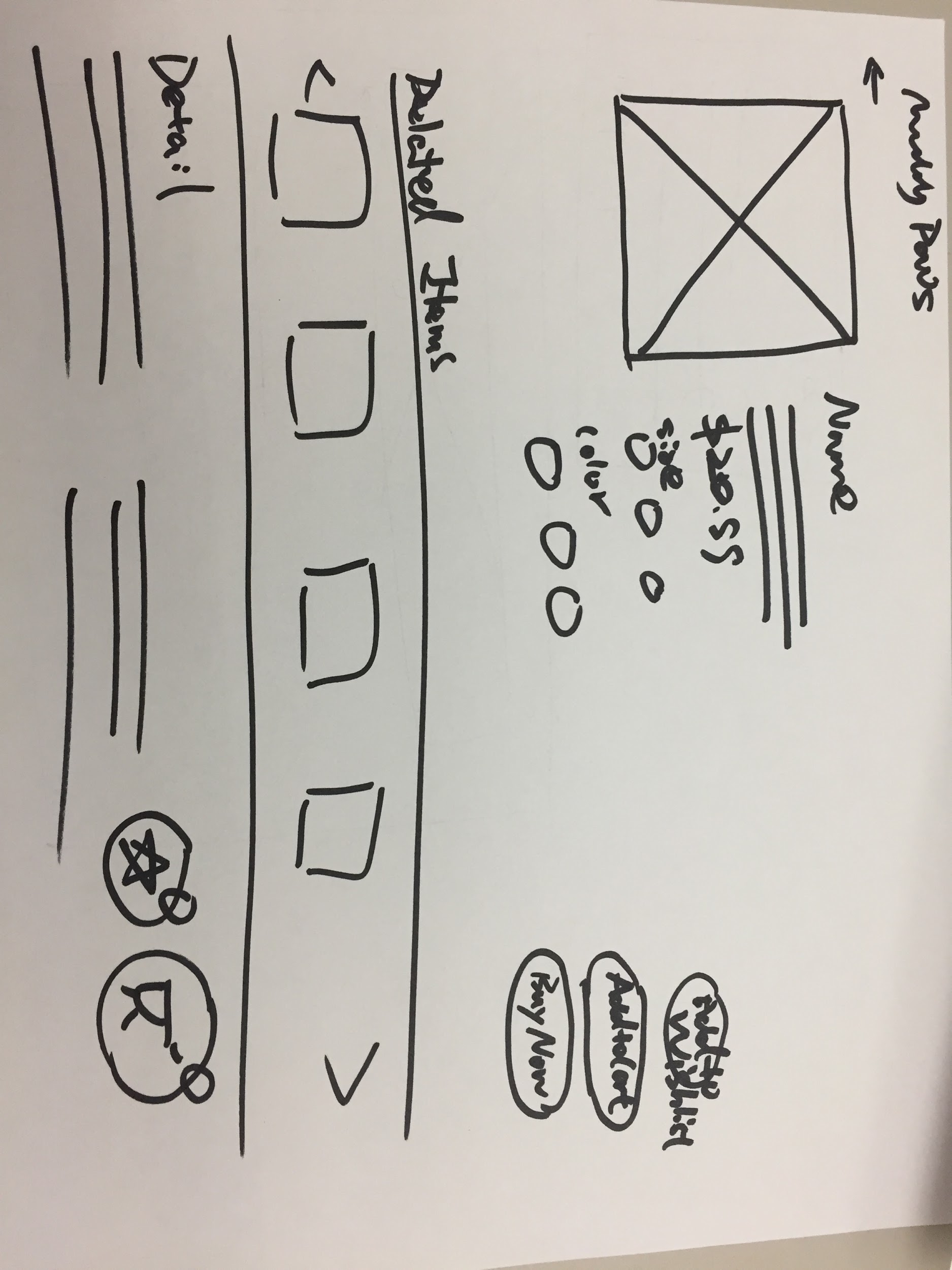
Shopping cart page lo-fi prototype



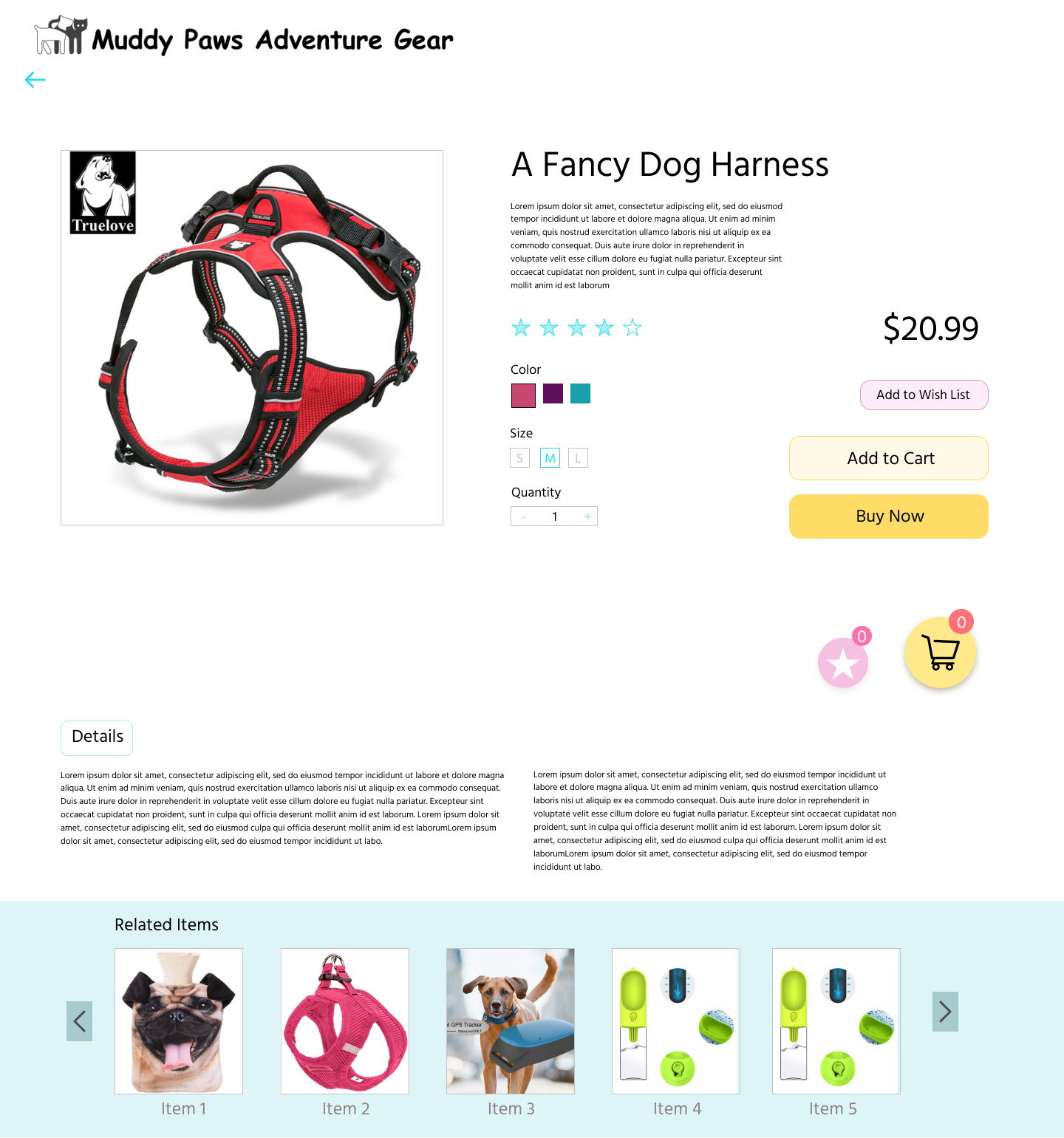
Shopping cart hi-fi prototype

**The product detail page**

In this iteration, the major changes were 1) adding “add to wish list” button and wish list FAB(floating action button, the button that floats on the page and provides additional actions) for the wish list feature, and 2) a carousel of related items to provide customers more choices.



Revised product detail page lo-fi prototype



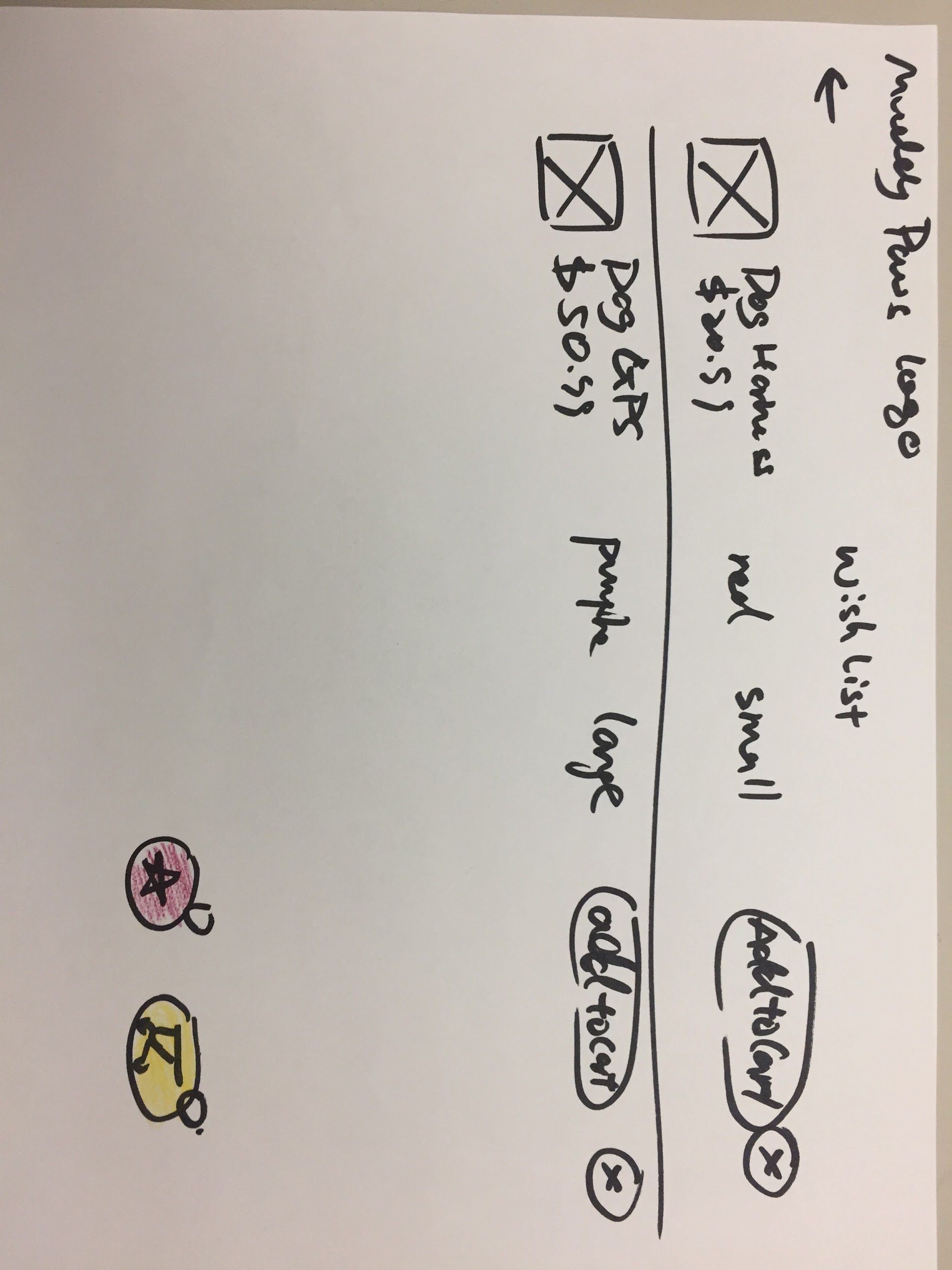
Revised product detail page hi-fi prototype (I changed the sequence of the carousel and the detail section, because the detail section and the product information should be one group of information, which needs to be visually grouped together as well)

**The Wish List**

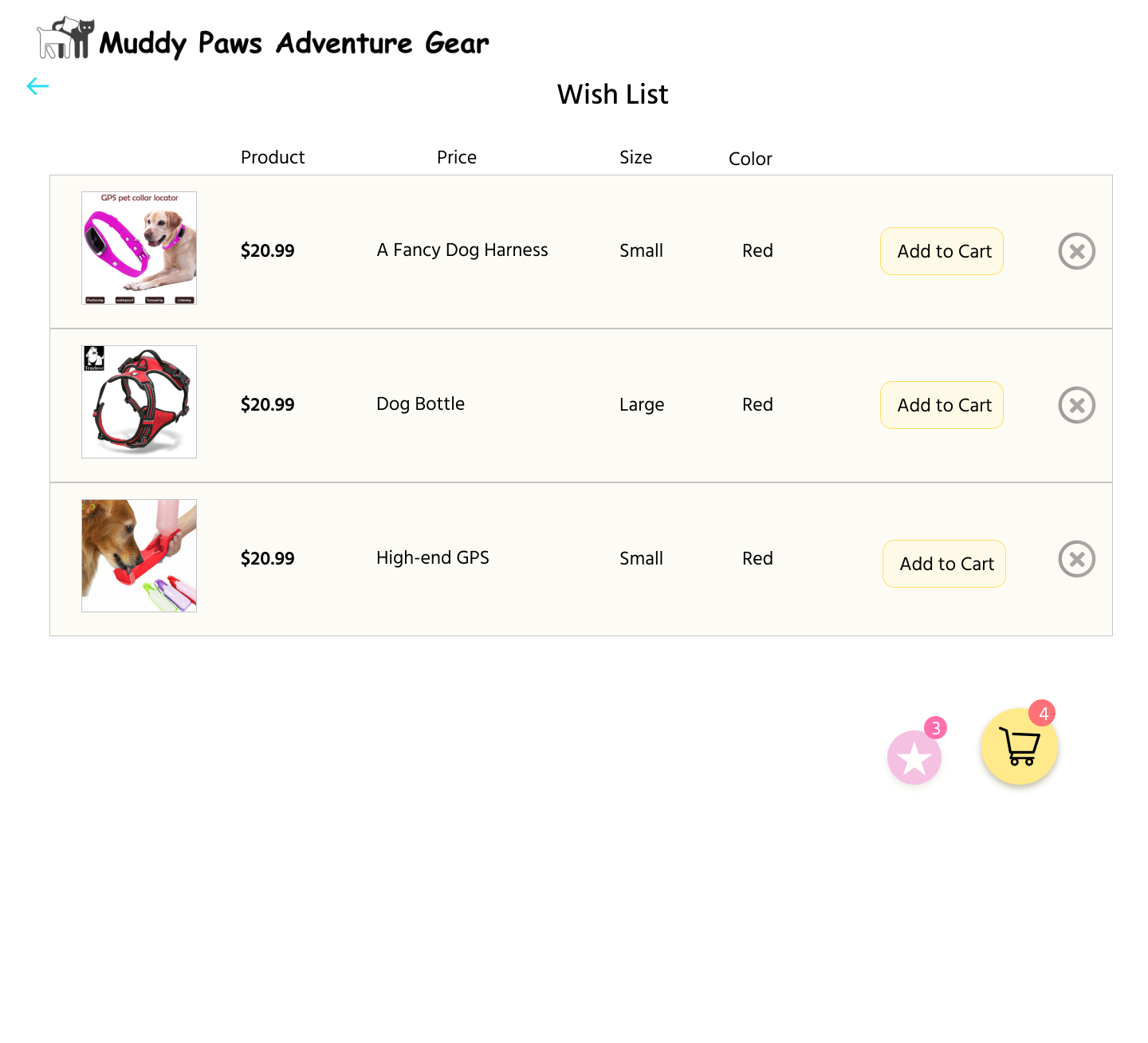
The wish list is similar to the shopping cart list, however, there’s no quantity nor total prices based on that. Instead, I added “add to cart” button for each of the product, so that customers can easily add their favorites to the cart.

Beneath the list of products, there are buttons of wish list and shopping cart for customers to track the quantity in each of them and get access to shopping cart page conveniently.

When showing the prototypes to peers, the design of two FABs were welcomed due to their bright colors that match the overall color theme, and



The wish list lo-fi prototype



The wish list hi-fi prototype