Reflection:

One of the issue I met is when comparing the style attribute of an object to a value, I cannot directly compare using '===' and object.style.attribute. Instead, I need to use window.getComputedStyle(object).attribute === value. Since the style attribute is not directly attached to the html object, (it is placed in the CSS file), there needs to be window.getComputedStyle(object) for getting the attribute, otherwise, it would return null. However when assigning the attribute a new value, the window.getComputedStyle(object) is not needed and can just use object.style.attribute for a new value because when assigning a new value, the original value of the attribute is not need. The other issue I met is when I was trying to display the amount of items in the bag once the page is loaded (the grey dot with number in it on top of the shopping bag icon). At first I would add a global value and see if the dot needs to be be displayed (if the amount is 0, the dot should not display). However, when trying to get the dot object by id, it always returns null. Later I found out that, by doing it in a global environment, it will try to retrieve the object before DOM is rendered, so such object would be non-existent. I figured out I can use "onload=function()". In the function, I can successfully get the dot object when the page is loaded, and decide whether to display the dot object with the correct amount.

Programming Concepts:

1. Constant
   - I assigned the objects I got from getElementById() to constant (ie. const bag = document.getElementById("shopping-bag")) because later I don't need to change assignment so it is better for me to assign it to a constant.
2. Conditional
   - I used conditional to execute whether or not the shopping bag is displayed (ie. if (bag.style.visibility === 'visible')).If it is visible, the condition is true, then it would execute to toggle it to be 'hidden'.
3. Class
   - I created the class 'pillow' for storing the information of pillows. It has five properties (style, color, filling, quantity, pricepc, pricetl), and some functions for changing the properties more elegantly.
4. Array
   - I used array to store the pillow objects that are being added to the shopping bag. If there is already an object in the array, that has the same style, color and filling, it will just update the quantity, and pricetl of that object.
5. Loop
   - I used to loop to go through the array and create HTML nodes that respond to the objects in the array, which would be displayed in the shopping bag.