Assignment 6 – Adding Functionality to a Website with JS

Yuxiang (Eric) Guo

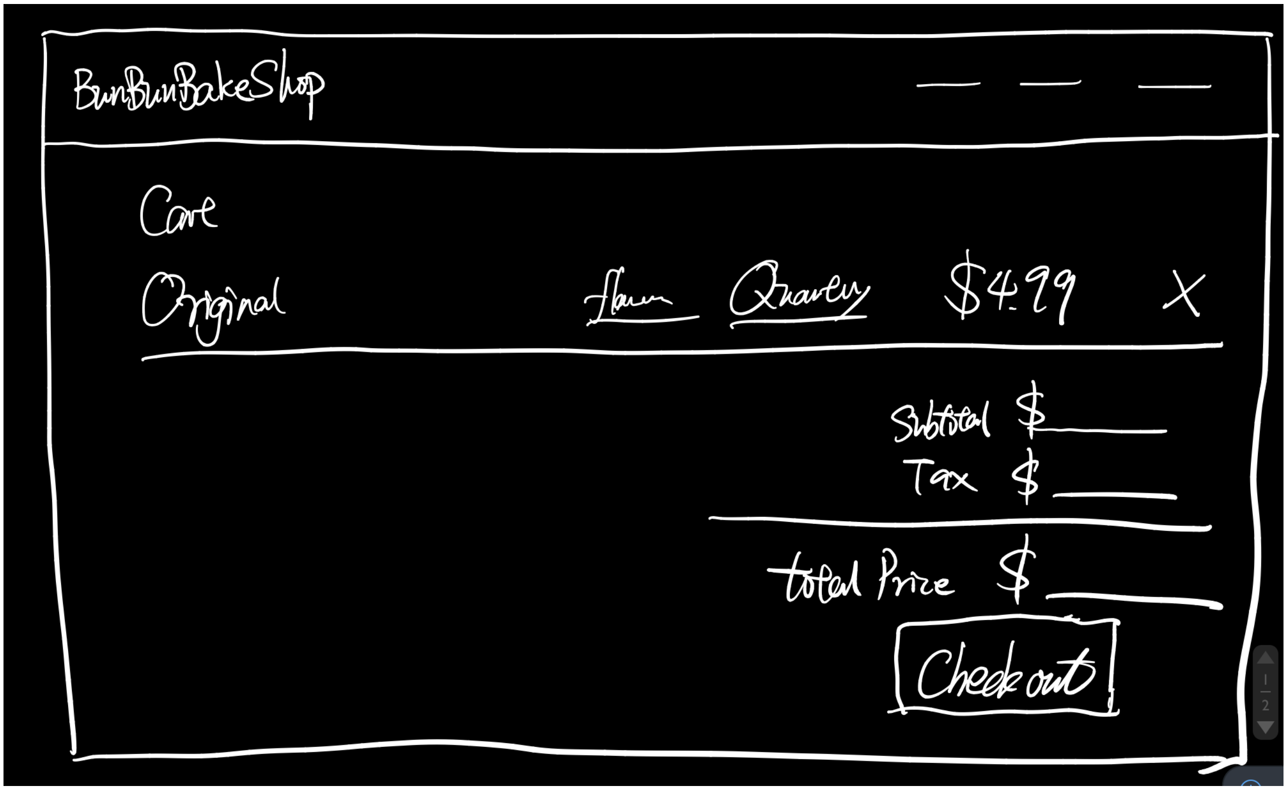
Scetion B Cole

Andrew ID: yuxiangg

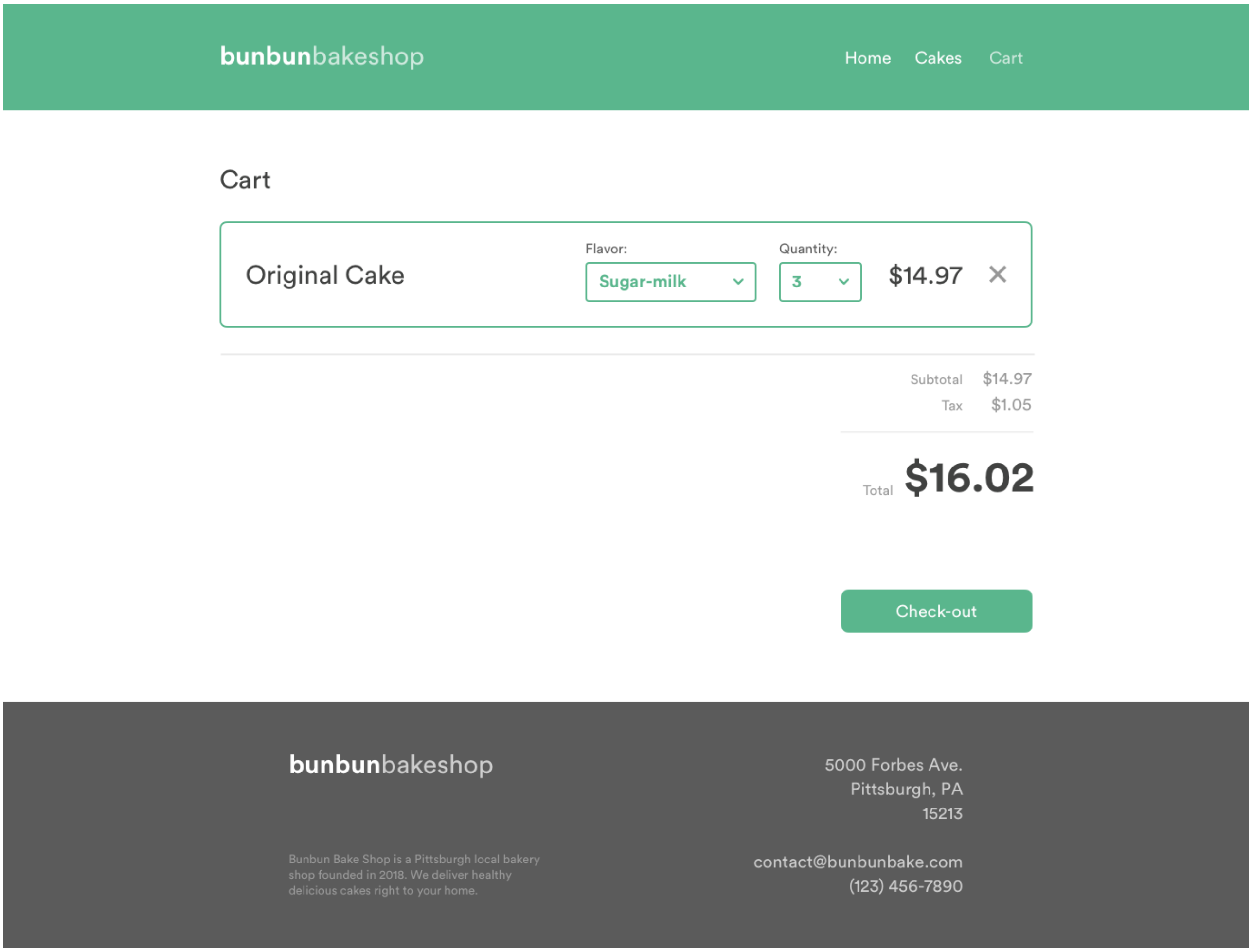
For the ease of grading, you can visit http://bunbunbakeshop.surge.sh/ (Best in Chrome)

New Shopping Cart Page

Low-fidelity Prototype for Shopping Cart Page



High-fidelity Prototype for Shooping Cart Page



Cart

Original Cake

Flavor: Sugar-milk

Quantity: 3

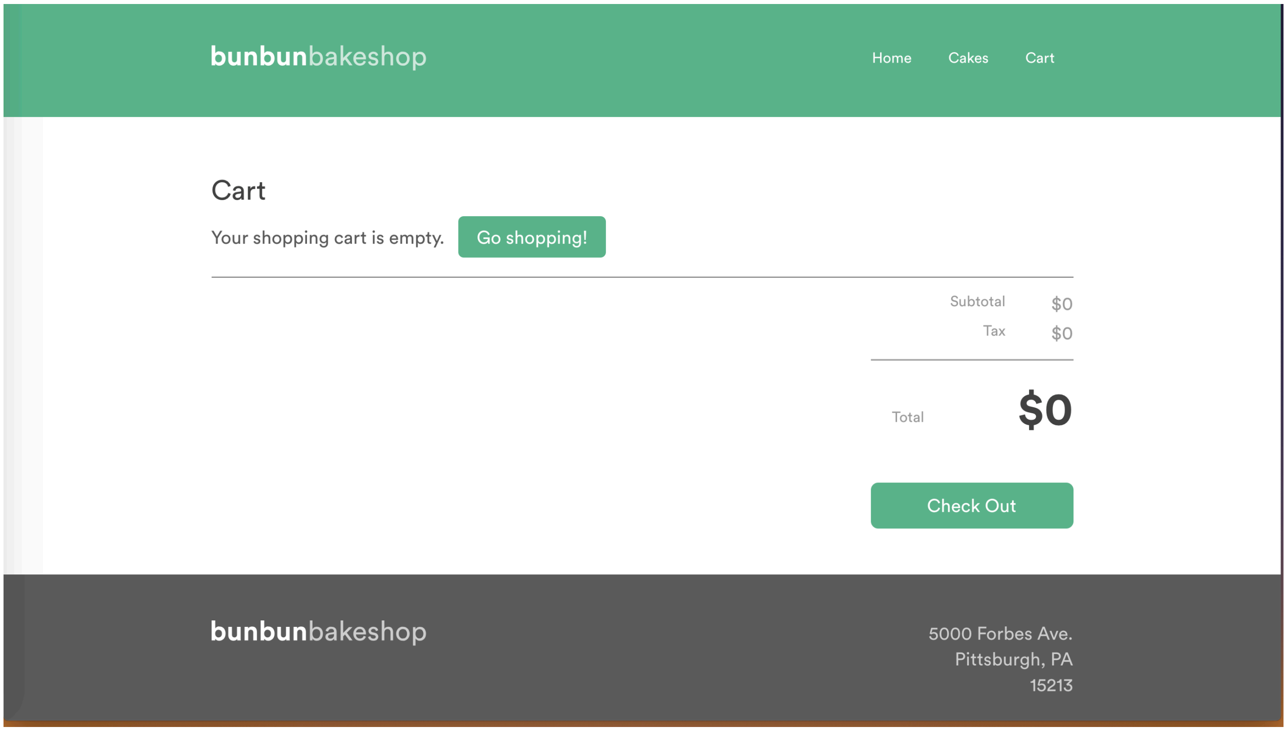
$14.97

Subtotal	$14.97
Tax	$1.05
Total	$16.02

Bunbun Bake Shop is a Pittsburgh local bakery shop founded in 2018. We deliver healthy delicious cakes right to your home.

5000 Forbes Ave.
Pittsburgh, PA
15213

contact@bunbunbake.com
(123) 456-7890



Cart

Your shopping cart is empty.

Subtotal	$0
Tax	$0
Total	$0

5000 Forbes Ave.
Pittsburgh, PA
15213

Design Rationale:

- To faciliate sales, I provide dropdown menus for users to change shopping cart items without leaving the shopping cart page.
- One difference between the high-fidelity prototype in Homework 4 and Homework 6 is that I removed the `Continue Shopping` button on the shopping cart page. The reason is to reduce distraction and close sale.
- The empty cart page display a `Go Shopping` button to further facilitate shopping behaviors.

Reflection

What challenges or bugs did you encounter? and How did you overcome these challenges?

1. When I am constructing the shopping cart page, one problem keep occuring was that the total cart price keep displaying as `$NaN`. I went back and checked my code for a long time and eveything looks fine. I then started to insert multiple `console.log()` to track different variables. And it turns out that the problem was: when the code is executing `totalPrice = (productPrice * productQuantity).toFixed(2)`, the variable `productPrice` is a string, while `productQuantity` is a number. So as a result, the `totalPrice` is a string, and string doesn't have `.toFixed(2)` (keeping only two decimal). I solved this problem by adding `Number()` :

```
 totalPrice = (Number(productPrice) *
productQuantity).toFixed(2);
```

2. Originally, I constructed my button in this way:

```
<button id="cart-item-delete"
onclick="cartDelete()">...</button>
```

The problem with this is that: when the `cartDelete()` is being called, it is very hard to know which current element is being called. So I searched on StackOverflow and came up with the following solution:

```
<button id="cart-item-delete"
onclick="cartDelete(this)">...</button>
```

In this way, when the `cartDelete()` is being called, I can directly use that element instead of using another `getElementById()`:

```
function cartDelete(element) {
    ...
    element.id = ...
    ...
}
```