Sprints Bootcamp JS Course Final Project

# Introduction

This document describes the requirements of the JS course final project. The main purpose of the project is to create a front-end application for an online shop webpage.

The project template files exist in [online-shop] folder.

## Template files

You must implement the functionalities of the following 6 pages inside the template:

1. Index.html
2. Shop.html
3. Checkout.html
4. Cart.html
5. Login.html (the page is not included in the template, you have to create it)
6. Register.html (the page is not included in the template, you have to create it)

## Index.html

You must implement the following functions inside the index.html page.

* Get the categories from [http://localhost:5000/api/categories/] API and fill the category list in the main menu. (Done)
* Fill the categories section in the home page with the first 4 categories only (sorted by products count in descending order) (Done)
* Fill the featured products section in the with the first 8 products filled from [http://localhost:5000/api/products/getFeatured] API (Done) but stars
* Fill the recent products section in the with the first 8 products filled from [http://localhost:5000/api/products/getRecent] API (Done) but stars
* In each product inside featured or recent sections, implement the (add to cart) and the (love) buttons. The buttons should to work on the client side without any post back (refresh). The love button should add +1 in the love\_counter variable (the variable should be stored in the local storage) and the add to cart button behavior should work as follow:
  + When you click on the add to cart button the cart counter (in the menu) have to be incremented by 1
  + **Products data should be stored in the localstorage.**
* Each product should have the price before discount and the price after discount.
* Each product should have the rating (represented as start) and rating count.

## Shop.index

You must implement the following functions inside the shop.html page.

* You must display all products fetched from [http://localhost:5000/api/products/] API.
* Implement the filters functions (filter by price, color, and size)
* Implement the sorting functions by (popularity (rating count), best rating, and price (ascending order)
* Implement the pagination functionality and showing menu.

## Checkout.html

* When the checkout page is loaded you have to get the products from the localstorage and display them as cart lines (product name and quantity) and display the subtotal and the tax and the total price
* The tax should be 10% when you select pay by paypal, 15% on the cheque and 5% with the bank transfer option
* You have to implement the common validations on the billing address form
* When you click place order button you have to post a request to [http://localhost:5000/api/orders/] API, remember thar add order API required a token first (you have to be logged in first)

## Cart.html

* When the cart page is loaded you have to get the products from the localstorage and display them as cart lines (product name and quantity) and display the subtotal price
* Implement the increment and decrement buttons on each cartline and save the result to the localstorage after each action.
* Implement the remove button and save the result to the localstorage after each action.
* You can access the cart.html page by clicking the cart icon in the main menu.

# Tips and Tricks

* Use OOP when you implement the project, create Cart, CartLine, Product, Category classes
* Implement the increment and decrement functions inside the CartLine class
* Implement remove and addProductToCart in the Cart Class
* Implement getProductsCount inside the Cart class
* Implement getProductHomeHTML, getProductHTML inside the Product class
* Separate the shop page js logic from the html (for the filter, sorting, and pagination functions)
* The filter by price should be implemented using a little bit complex function (as it has price from – to)
* All shop page functions should be implemented client side
* Redirect the checkout page to the login page if you are not logged in (check the access token saved in the localstorage)
* Disable the buttons that interact with any APIs till you get a response.

# Scoring items

1. Index.html functionalities (10%)
2. Shop.html functionalities (20%)
3. Checkout.html functionalities (20%)
4. Cart.html functionalities (10%)
5. Login.html functionalities (5%)
6. Register.html functionalities (5%)
7. Using OOP (20%)
8. Code quality (10%)