Intro to JavaScript Week 5 Coding Assignment

Points possible: 70

Category	Criteria	% of Grade
Functionality	Does the code work?	25
Organization	Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear.	25
Creativity	Student solved the problems presented in the assignment using creativity and out of the box thinking.	25
Completeness	All requirements of the assignment are complete.	25

Instructions: In VS Code, or an IDE of your choice, write the code that accomplishes the objectives listed below. Ensure that the code compiles and runs as directed. Take screenshots of the code and of the running program (make sure to get screenshots of all required functionality) and paste them in this document where instructed below. Create a new repository on GitHub for this week's assignments and push this document, with your JavaScript project code, to the repository. Add the URL for this week's repository to this document where instructed and submit this document to your instructor when complete.

Coding Steps:

1. Create a menu app as seen in this week's video. What you create is up to you as long as it meets the following requirements.
    a. Use at least one array.
    b. Use at least two classes.
    c. Your menu should have the options to create, view, and delete elements.

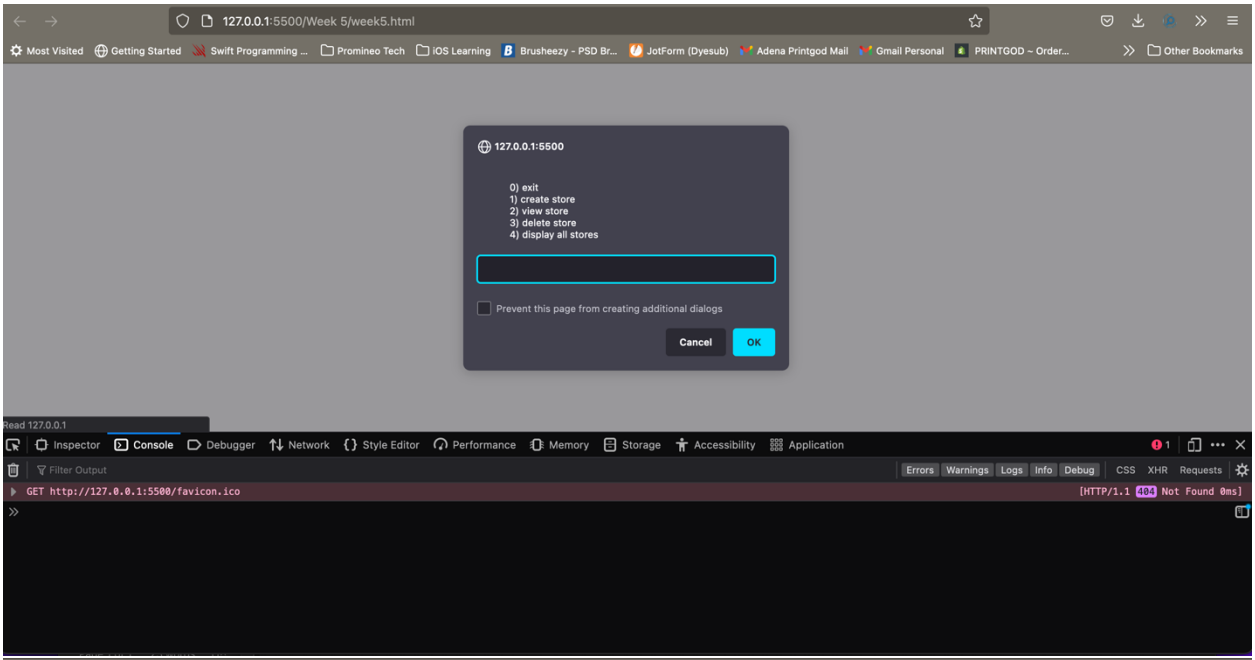
Screenshots of Code:

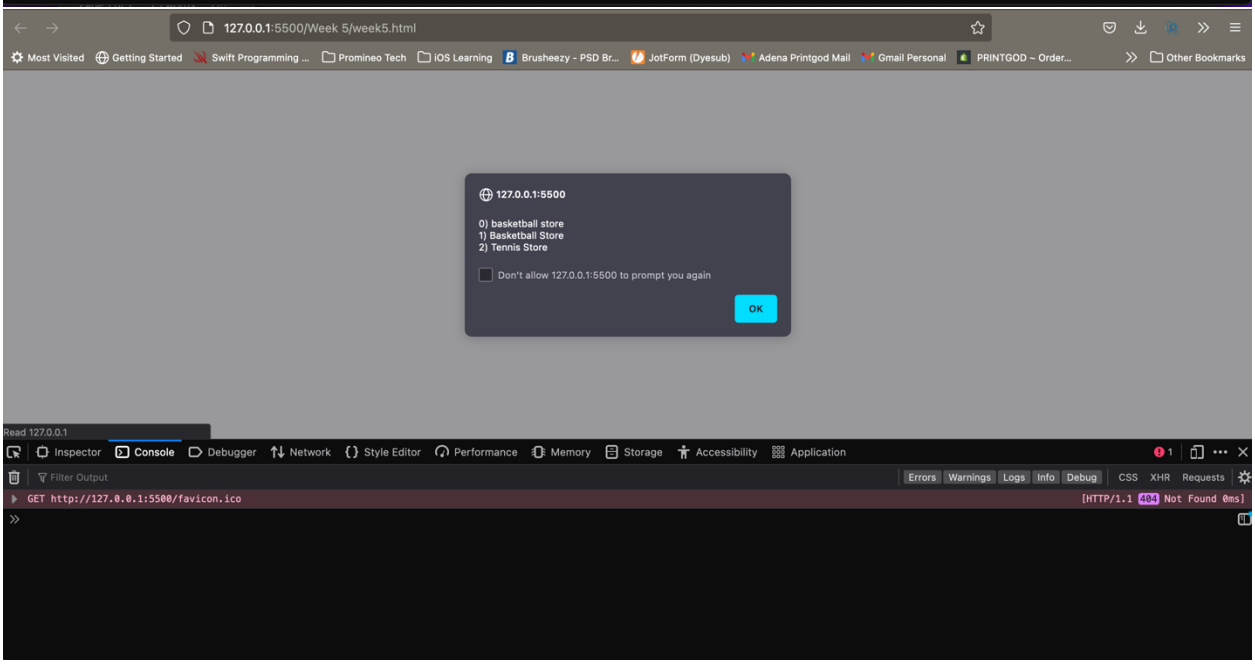
PROMINEO TECH

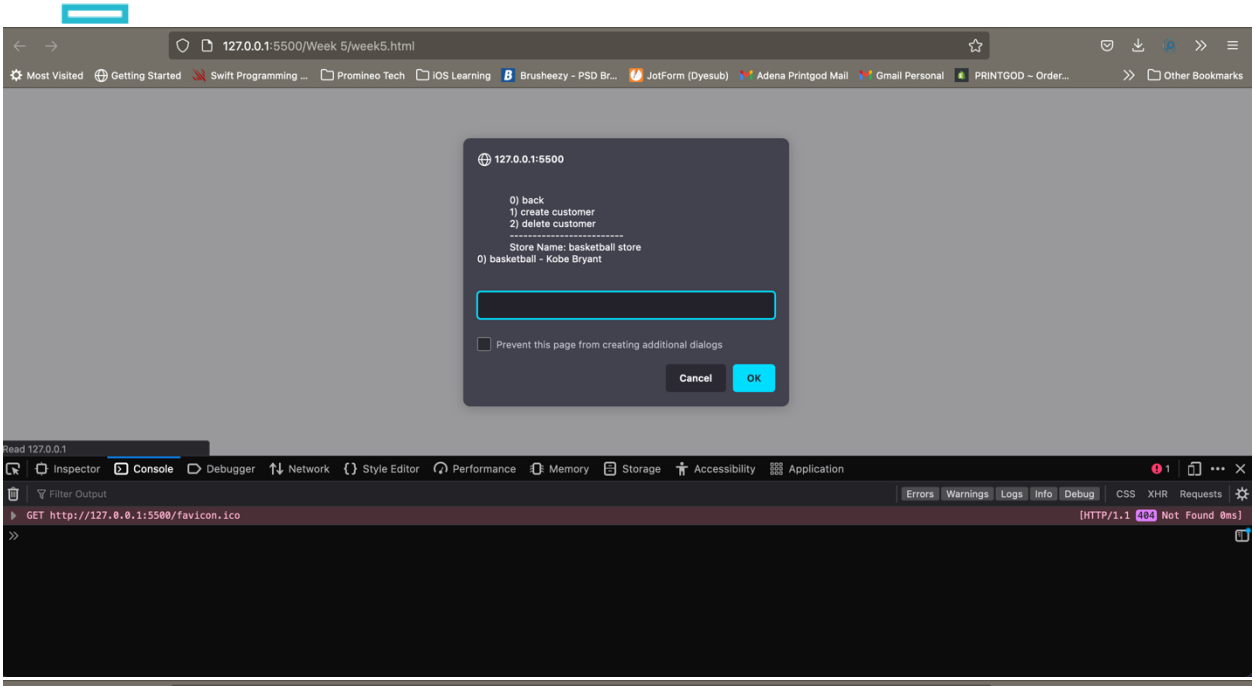
```
class Sneaker { //a class is needed - sneaker store is entered based on sneaker type that is associated with the sport played
    constructor(sport, shoe) {
        this.sport = sport;
        this.shoe = shoe;
    }

    describe() { //describes the sport being played and shoe they will wear - is this a made up method???
        return `If a person plays ${this.sport} they should wear this type of sneaker: ${this.shoe}`;
    }
}

class SportStore { //a class is needed - create stores that customers will enter
    constructor(sport) {
        this.sport = sport; //store type
        this.customers = []; //create empty array so each customer type can be push.ed into that store type
    }

    addCustomer(customer) { //method created to take in a customer
        if (customer instanceof Sneaker) { //this if checks to see if customer is an instance of Sneaker class - so no one can just enter in anything that doesn't fit that Sneaker Class
            this.customers.push(customer); //will push customer to the customers empty array in the SportStore
        } else {
            throw new Error(`You can only add an instance of Sneaker. Argument is not a player: ${customer}`);// exception or error if it is
        }
    }

    describe() {//method for our SportStore
        return `${this.sport} has ${this.customers.length} customers.`;//this will tell us how many customers are in this particular store
    }
}

class Menu { // actual menu itself and this class is what drives the application with all the options/choices
    constructor() {//wont take in anything/any arguments
        this.stores = []; //initialize our stores, an array of stores. We can multiple stores inside this application.
        this.selectedStore = null; //create a variable for the store selected. Managing one store at a time - we want to know what team is selected. Setting it equal to null to start because when we start, no store is selected.
    }

    start() { //a method that will start up our application - the entry point to our menu - the follow showMainMenuOptions will provide menu and user will select option.
        let selection = this.showMainMenuOptions(); //ShowMainMenuOptions method doesn't exist yet, building out our menu and what we want it to look like. Top-down development approach - build out methods and then implement after/later.

        while (selection != 0) { // selection is a variable we're using to get user input - what option in our menu has been selected by the user. showMainMenuOptions method will return what the option the user input.
            switch (selection) { // loop type to look at selection - all cases below haven't been created yet.
                case "1":
                    this.createStore()
                    break;
                case "2":
                    this.viewStore();
                    break;
                case "3":
                    this.deleteStore();
                    break;
                case "4":
                    this.displayStores();
                    break;
                default:
                    selection = 0;
            }
            selection = this.showMainMenuOptions(); //keep this in the while loop so it will keep looping if user does not input options provided: 0-4.
        }

        alert("Goodbye!"); //if user does enter 0 - we will exit loop/outside the loop. User input of 0 will make the while loop false (because 0 = 0) and then loop will be exited
    }

    showMainMenuOptions() { //program the methods above (cases). Return prompt is below to have Menu prompt display the cases in while loop we created above. & it will RETURN what the user selected/input as their option.
        return prompt(`
            0) exit
            1) create store
            2) view store
            3) delete store
            4) display all stores
        `);
    }

    showStoreMenuOptions(storeInfo) {
        return prompt(`
            0) back
            1) create customer
            2) delete customer
            ----------------------
            ${storeInfo}
        `);
    }

    displayStores() {
        let storeString = ""; // blank string - we need build a string that has all info for stores - this line will create blank string.
        for (let i = 0; i < this.stores.length; i++) { //this.stores.length will provide the list of all the stores that exist in the empty array we built in our MENU class. This line will iterate through each team, grab each team.
            storeString += i + ") " + this.stores[i].sport + "\n"; //concatenate all team info - i is the index of each team and grab current team we are looking at for this iteration (this.stores[i].name) - grab the name for each team and then new
        }
        alert(storeString); //once we get out the loop we can see all the stores
    }

    createStore() { //need to be able to display stores
        let sport = prompt("Enter sports store type the customer is entering:");
        this.stores.push(new SportStore(sport)); //array where we keep stores in the MENU class and sending to SportStore class
    }

    viewStore() {
        let index = prompt("Enter the index of the store you wish to view:");
        if (index > -1 && index < this.stores.length) { //validate the user input if they put 0- array of stores created then move onto the next
            this.selectedStore = this.stores[index]; //set our selectedStore class property to the store that was input by the user.
            let description = "Store Name: " + this.selectedStore.sport + "\n";

            for (let i = 0; i < this.selectedStore.customers.length; i++) { //selectedStore is the store and each store has a customer array and the length attached is going to iterate through the customer array not the selectedStore.
                description += i + ") " + this.selectedStore.customers[i].sport + " - " + this.selectedStore.customers[i].shoe + "\n"; //this will be a list of all the customers and the store they're in.
            }

            let selection = this.showStoreMenuOptions(description); //using top-down development again because we haven't built this yet. this is to provide menu for store option / sub menu - we need to create this list of menu options above, right
            switch (selection) {
                case "1":
                    this.createCustomer(); //create customer that entered store - method that needs to be created
                    break;
                case "2":
                    this.deleteCustomer(); //break isn't neede after this because there is nothing that will come after this case. - method that needs to be created again like above cases
            }
        }
    }

    deleteStore() {
        let index = prompt("Enter index of sport store that is being closed/deleted for the day:"); //all customers have left store and is now closed for the day
        if (index > -1 && index < this.stores.length) { //validate if user input is true
            this.stores.splice(index, 1); //delete the requested index/only 1
        }
    }
```

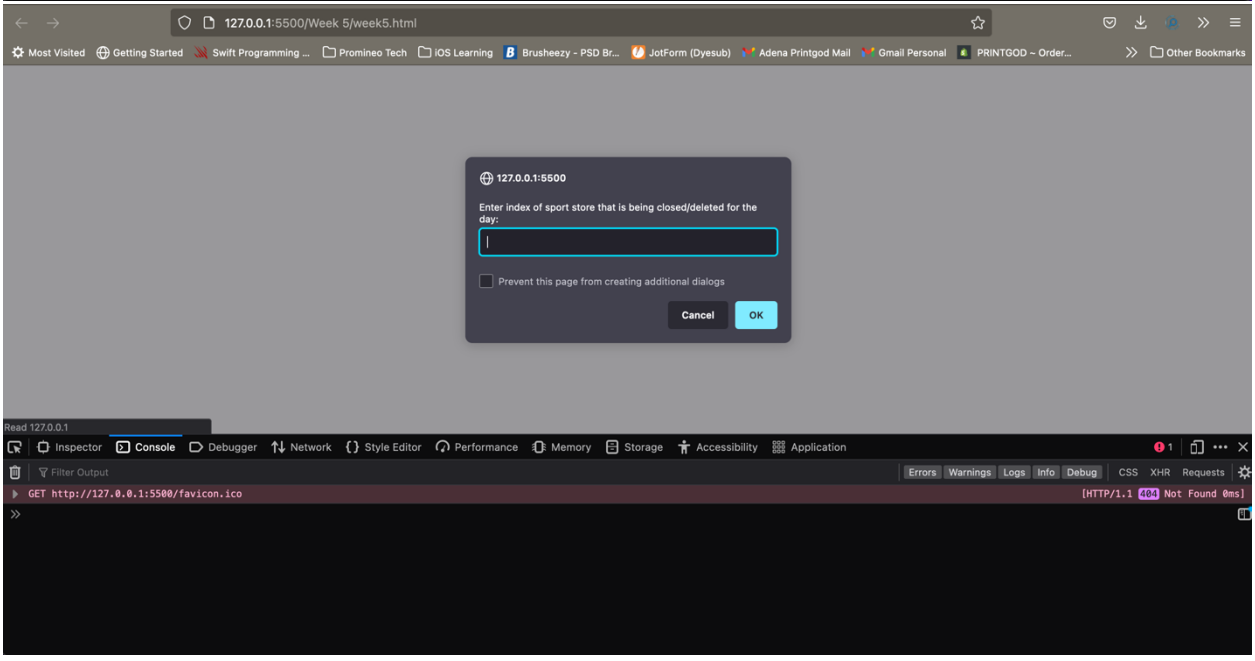
```
    createCustomer() { //creating customer profile in order to send customer to correct store
        let sport = prompt("Enter sport type for customer:"); //this will allow customer to be placed at correct store.
        let shoe = prompt("Enter Signature Player shoe customer is looking for:") // which icon player shoe is customer looking for?
        this.selectedStore.customers.push(new Sneaker(sport, shoe)) //selected store variable allows us to know what store is selected when we pass customer info into it. & push customer to correct store.
    }

    deleteCustomer() {
        let index = prompt("Enter index of the customer that is leaving store after purchase:"); //after purchase is made, customer is leaving store.
        if (index > -1 && index < this.selectedStore.customers.length) { //validate that customer is leaving/exiting - based on customers at the sport store
            this.selectedStore.customers.splice(index, 1); //this is the actual act of removing 1 element/index requested
        }
    }
}

let menu = new Menu();
menu.start();
```

Screenshots of Running Application:









URL to GitHub Repository:

https://github.com/Denaag/Week-5/blob/main/myMenuApp.js

https://github.com/Denaag/Week-5.git