**JavaScript Menu**

A menu is used to display items within a webpage such as food, items at a grocery store and an online shop. The structure of the webpage (HTML) plays an integral part in the functionality of JavaScript. The whole page is styled using CSS. The menu is created using a section element as a container which then have title container, filter buttons container and the menu-items container. A

Title Container

The title container uses <div> is used to center the title and the underline

The title preferably placed using <h2> while the underline is created from an empty <div> and CSS is used to center, color, and assign height and width it will occupy.

Filter Button Container

<div> is used as a container and a sample structure is provided to allow styling of the buttons which would be added dynamically using JS. All <button> uses the same class to allow uniform styling. Additionally, dataset <data-id=’’> is used to assign IDs to each button; same as names assigned to the buttons in this case - ‘all’, ‘breakfast’, ‘lunch’, ‘shake’, and ‘dinner’.

Menu-Items Container

The container has a class of ‘section-center’. The section center then has another container of <article> with a ‘menu-item’ class. It has an image with a class of ‘photo’ and a <div> with ‘item-info’ class which is a container for the header and the a <p> element with ‘item-text’ class for description of the item. The header is also a container for the title of the item and the price which are later styles {display: flex}.

<div class= ‘section-center’> is styled {display: grid}.

**JavaScript Functionality**

JavaScript already has the menu array-like-object which is used to contain each menu item where each property is has several variables including the id, image (img), title, price, and description.

To be able to transverse DOM (document object Model) variables are declared using query paths such as document.querySelector() and document.querySelectorAll().

The best approach to edit or to add javascript functions to dom is using the containers. The most important containers include the section-center (for menu items) and btn-container (for buttons) and the buttons using the assigned classes.

const sectionCenter = document.querySelector(‘.section-center’);

const container = document.querySelectot(‘.btn-container’)

The whole webpage will be loaded using DOMContentLoaded event. The following syntax is used:

window.addEventListener(‘DOMContentLoaded’, function (){

// other functions are loaded here

displayMenuItems (menu)

displayMenuButtons ()

})

***Loading the array with the javascript***

Functions are the most appropriate approach towards creating a clean code. For the menu website, several array methods are used to iterate through the provided Menu array-like-object.

Dynamic addition of menu items function

The first function is used to replace the items created in the section-center class such as one template can be used to iterate though the number of items within the menu array-line-object.

Within the function, map () method is used to iterate through the menu, access each item and display the items’ variables as they are called through interpolation. The following syntax is developed

function displayMenuItems (menuItems) {

let menuContent = menuItem.map(function (item) {

return `<article class = ‘menu-item’>

<img src = ${item.img} class = ‘photo’ alt = ${item.title}/>

<div class = ‘item-info’>

<header>

<h4>${item.title}</h4>

<h4 class = ‘price’>$${item.price} </h4>

</header>

<p class = ‘item-text’> ${item.desc}</p>

</div>

</article>`

});

});

***Loading the buttons into DOM using JavaScript***

The category variable has varying values and some are repetitive and loading them using JavaScript ensure that even if other categories are added into the description the buttons shall be added into the DOM.

The function for adding buttons include using reduce () method to return values once and then use filter the unique categories.

function displayMenuButtons () {

//getting the unique categories

const categories = main.reduce (function(values, item){

if (!values.includes(item.category)){

values.push(item.category); }

return values;

},

[‘all’];

)

// getting the categories into buttons

Const categoryBtns = categories.map(function (category){

return` <button class = ‘filter-btn’ type = ‘button’ data-id = ${category}> ${category}</buttton>`

}).join(‘’);

//the join() method ensures all btn are joined into one string

// adding the buttons into the DOM

container.innerHTML = categoryBtns;

const filterBtn = container.querySelectorAll(‘.filter-btn’);

//filter according to button categories

filterBtn.forEach(function (btn){

btn.addEventListener(‘click’, function(e){

const category = e.currentTarget.dataset.id;

const menuCategory = menu.filter(function (menuItem){

if(menuItem.category === category){

return menuItem;

};

});

if(category = ‘all’){

displayMenuItems(menu);

} else {

displayMenyItems(menuCategory);

};

});

});

};