



Diseño y Programación de Software Multiplataforma

Trabajo de Investigación:

Carrito de Compras

Docente

Alexander Sigüenza

Integrantes

José Gerardo Marroquín Vásquez	MV230090
Enma Sofia López Rojas	LR230079
Kevin Enrique Martínez Martínez	MM230084
Roger Eduardo Vásquez Portillo	VP223250
José Ernesto Sorto González	SG202883

Link de Repositorio: <https://github.com/KevinEnriqueMartinezMartinez/Tienda-Javascript-DPS.git>

Link de Video: [video de Trabajo 1.mp4](#)

Diseño y Programación de Software Multiplataforma

DPS941 G01T

San salvador 03 marzo de 2023

Contenido

INTRODUCCIÓN	i
OBJETIVOS	ii
GENERAL.....	ii
ESPECÍFICOS	ii
DESARROLLO	1
PARTES DE LA TIENDA EN LINEA.	1
DETALLE DE TIENDA	2
SELECCIÓN DE PRODUCTO.	2
CARRITO DE COMPRAS.....	3
AGREGAR AL CARRITO	3
ELIMINAR PRODUCTO DE CARRITO.....	4
FACTURACION	4
ESTRUCTURA DE PROYECTO.....	5

MANUAL DEL CARRITO DE COMPRAS CON JAVASCRIPT

INTRODUCCIÓN

En el siguiente documento se detalla el manual sobre el carrito de compras de una tienda virtual creado con HTML, JavaScript y CSS, se mostrará la lógica implementada en nuestro proyecto de tienda virtual de Zapatos al detalle, en el cual se desarrolla un sitio web en el que se puntualizan características de los productos ofrecidos.

Dentro del desarrollo del siguiente manual se describirán las carpetas que son de utilidad para la ejecución del proyecto, así como hacer un detalle de los archivos que estas contendrán, para mostrar más adelante un detalle más conciso de las variables, funciones y métodos utilizados.

OBJETIVOS

GENERAL

Desarrollar un manual en el que se expliquen las principales partes de la tienda en línea, sus productos y carrito de compras y el código realizado por el equipo de trabajo, para que sirva de guía para próximos desarrollos.

ESPECÍFICOS

Ofrecer al usuario un manual que sirva de guía en la utilización de la app web de la tienda en línea de la variedad de productos mostrados.

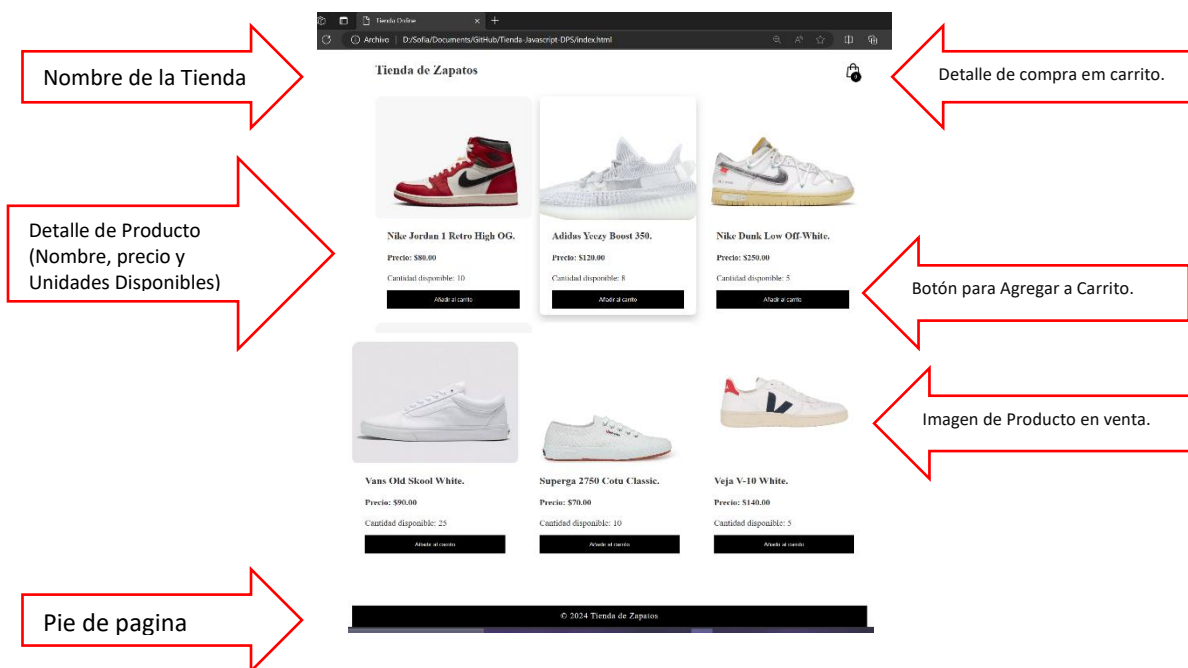
Mostrar la estructura utilizada en la codificación de la app web creada.

Establecer variables y funciones de estas para mostrar dinamismo en la tienda.

DESARROLLO

PARTES DE LA TIENDA EN LINEA.

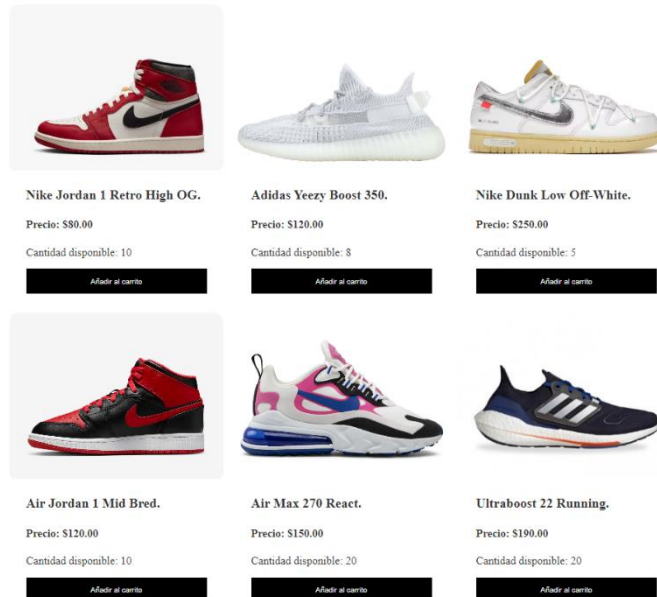
A continuación, se detallarán la parte visual de la tienda en línea, así como también cada una de las partes que la conforman y como el usuario de la aplicación podrá hacer uso de esta, se iniciará con una vista general de toda la pagina y sus principales partes, para que el usuario se familiarice con esta.



DETALLE DE TIENDA

Dentro del detalle se podrá observar el listado de los estilos de productos, así como el nombre de este y la cantidad disponible en el inventario para la venta y adicionalmente se muestra el precio por unidad de cada uno.

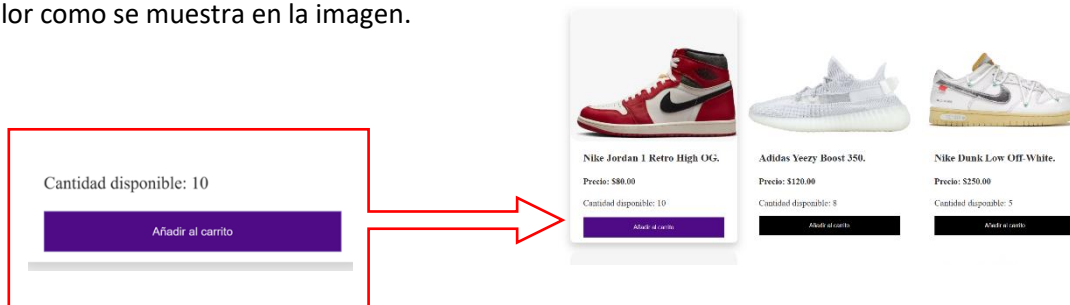
Tienda de Zapatos



SELECCIÓN DE PRODUCTO.

Si el usuario se posiciona sobre el producto de interés se vera un efecto el cual le indica sobre que producto se esta posicionando, adicionalmente si el usuario desea agregarlo al carrito deberá de acercar el puntero al botón que tiene la leyenda “Agregar al Carrito” el cual cambiara de color como se muestra en la imagen.

Tienda de Zapatos



En caso de proceder con la compra se deberá de seleccionar mediante un clic en el botón para ir agregando las unidades que desee siempre y cuando no se excedan de las que están en existencias, una vez dado el clic en el botón esta acción hará que se disminuyan las existencias del

producto tal y como se muestra en la siguiente imagen, en la cual inicialmente detallaba 10 unidades en existencias y mediante se ejecuten los clic de las unidades requeridas este ira disminuyendo tal y como se muestra en la imagen donde ya solo quedan 9 unidades disponibles.



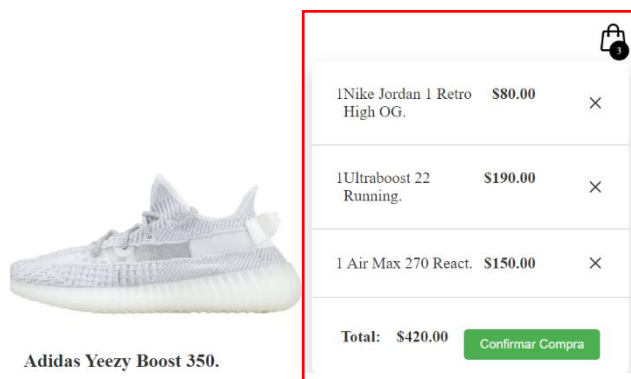
CARRITO DE COMPRAS

AGREGAR AL CARRITO

En la parte superior derecha se muestra un icono en el cual se podrá ver el detalle de lo agregado al carrito dando un clic sobre el icono de la bolsa



Se mostrará el siguiente detalle de los productos agregados al carrito, así como muestra la imagen a continuación detallando el nombre del producto la cuantía de los productos agregados, así como el total a cancelar, como muestra la imagen.



ELIMINAR PRODUCTO DE CARRITO

En caso de que ya no se desee un producto de los seleccionados, bastara con dar un clic en la “X” que aparece en la parte derecha del producto que desea eliminar, para que este sea anulado de los artículos a comprar, una vez realizada esa opción se mostrara el carrito como en la imagen, detallando solamente los artículos seleccionados y el total a cancelar.

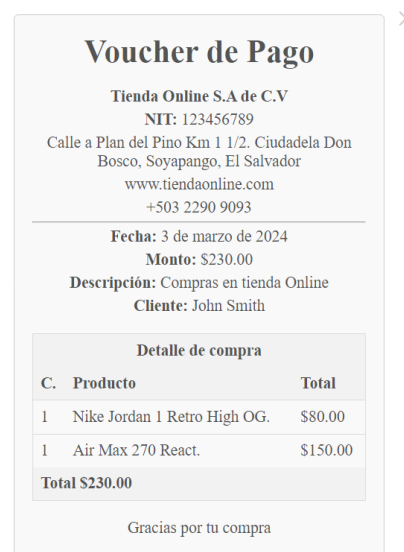


FACTURACION

Una vez seleccione el botón “Confirmar Compra” se procederá a mostrar un detalle de lo que se cancelara del carrito de compras, así como se muestra en la imagen.



Una vez se seleccione el botón “Pagar Ahora” se activará una alerta de procesamiento de los datos el cual concluirá con la emisión de un Recibo de la compra realizada a continuación, se mostrara el comprobante emitido por la app. Una vez terminada la compra se cierra el recibo y vuelve a la pagina inicial en la cual se puede proseguir con alguna otra compra que se desee.



Dentro de la estructura del JavaScript inicialmente definiremos una variable que servirá para captar los objetos que en este caso son los diferentes productos que se han ingresado en la

```

index.html x index.js x
1 let items = [
2   {
3     img: "https://letkicks.com/cdn/shop/products/StockXAndre3_86.png?v=1677087684",
4     title: "Nike Jordan 1 Retro High 06.",
5     price: "80.00",
6     stock: 10,
7   },
8   {
9     img: "https://i.ebayimg.com/00/s/HT4u4Fp0dDhu/z/PcUAA05wSepf07Ye/$_58.png",
10    title: "Adidas Yeezy Boost 350.",
11    price: "120.00",
12    stock: 8,
13  },
14  {
15    img: "https://images.stockx.com/360/Nike-Dunk-Low-Off-White-Lot-1/Images/Nike-Dunk-Low-Off-White-Lot-1/Lv2/img01.jpg?fm=avif&auto=compress&w=576&dpr=1&updated_at=1635618",
16    title: "Nike Dunk Low Off-White.",
17    price: "250.00",
18    stock: 5,
19  },
20  {
21    img: "https://res.cloudinary.com/brokenchains/image/upload/v1687587091/Broken-Chains-Air-Jordan-1-Mid-DQ8423-060-03.jpg",
22    title: "Air Jordan 1 Mid Bred.",
23    price: "120.00",
24    stock: 10,
25  },
26  {
27    img: "https://deportint.co/cdn/shop/products/CI3899-100_1080x.jpg?v=1678822268",
28    title: "Air Max 270 React.",
29    price: "150.00",
30    stock: 20,
31  },
32  {
33    img: "https://static.runnea.com/images/202306/adidas-ultraboost-22-zapatillas-running-400x400x90xX.jpg?1",
34    title: "Ultraboost 22 Running.",
35    price: "190.00",
36    stock: 20,
37  },
38  {
39    img: "https://static.nike.com/a/images/t_default/940a4f7c-6c23-481a-8c7b-f474a87a6d56/calzado-de-running-en-carretera-pegasus-39-jXTgc0.png",
40    title: "Air Zoom Pegasus 39 Running.",
41    price: "120.00",
42    stock: 25,
43  },
44 ],
45
46

```

aplicación, a continuación, se mostrara la fracción de código que muestra lo detallado

anteriormente:

Con el lenguaje de JavaScript se detallan los elementos que se ingresaran a las variables así como el recorrido que se realizaran a los arreglos de los objetos creados

```

index.html x index.js
90 stock: 5,
91 },
92 },
93 // Selecciona el elemento HTML con la clase "container-items" y lo almacena en la variable containerItems.
94 let containerItems = document.querySelector(".container-items");
95 // itera sobre cada objeto en el array items, usando el metodo foreach.
96 items.forEach(item => {
97   // crea un nuevo objeto div para representar un articulo.
98   let divItems = document.createElement("div");
99   // añade la clase items al nuevo div.
100  divItems.classList.add("items");
101  // arregla el contenido HTML del nuevo div creado anteriormente con la info del producto actual.
102  divItems.innerHTML = `
103    <figure>
104      
105    </figure>
106    <div class="info-product">
107      <h2>${item.title}</h2>
108      <p class="price">Precio: $${item.price}</p>
109      <p class="quantity-available">
110        Cantidad disponible: <span class="available-quantity">${item.stock}</span>
111      </p>
112      <button class="btn-add-cart">Añadir al carrito</button>
113    </div>
114  `;
115  // Añade el nuevo div al contenedor con la clase "container-items"
116  containerItems.appendChild(divItems);
117 });
118

```

Siguiendo con el lenguaje de JavaScript se han planteado la lógica para las funcionalidades del carrito de compra, ejemplo de esto es la siguiente codificación que se muestra a continuación:

```

160 // creamos un evento para añadir la logica de agregar productos al carrito cuando se hace click en el boton añadir al carrito.
161 productList.addEventListener('click', e => {
162
163     //Verifica si el elemento en el que se hizo clic tiene la clase CSS "btn-add-cart"
164     if(e.target.classList.contains('btn-add-cart')){
165         const product = e.target.parentElement;
166         const availableQuantity = parseInt(product.querySelector('.available-quantity').textContent);
167         //Verifica si hay suficiente cantidad disponible del producto para ser añadido al carrito.
168         if (availableQuantity > 0) {
169             // Actualiza la cantidad disponible del producto en el HTML
170             const quantityAvailableElement = product.querySelector('.quantity-available .available-quantity');
171             quantityAvailableElement.textContent = availableQuantity - 1;
172
173             const infoProduct = {
174                 quantity: 1,
175                 title: product.querySelector('h2').textContent,
176                 Price: product.querySelector('.price').textContent.replace('Precio: $', ''),
177                 availableQuantity: availableQuantity
178             }
179
180             // recorre todos los objetos que tenga el vector
181             const exist = allProducts.some(product => product.title === infoProduct.title)
182             // Verifica si el producto ya existe en el carrito
183             if(exist){
184                 //Utilizo el método map para crear un nuevo array (products) basado en el array existente de productos.
185                 const products = allProducts.map(product => {
186                     // Verifica si el título del producto actual en el array coincide con el título del producto que se está intentando añadir.
187                     if(product.title === infoProduct.title){
188                         //Si hay coincidencia, incrementa la cantidad del producto en el carrito .
189                         product.quantity++;
190                     }
191                     // nos retorna el producto actualizado en el nuevo array.
192                     return product
193                 })
194             } else{
195                 return product
196             }
197             allProducts = [...products]
198         } else{
199             //Crea un nuevo array que incluye todos los productos existentes en el carrito.
200             allProducts = [...allProducts, infoProduct]
201         }
202         // Llama a la funcion showhtml.
203     }
204 }

```

Toda la lógica utilizada esta detallada en cada uno de los archivos que se pueden observar en el repositorio, cuyo link esta en la portada de esta actividad, cada hoja utilizada esta comentada para que sea entendible por los usuarios externos como internos