1. Gestión del carrito de compras

Variables y funciones principales:

let cart = []: Declara un arreglo vacío que servirá como el carrito de compras, donde cada elemento será un objeto con información del producto (nombre, precio, cantidad, subtotal).

Función: addToCart(productName, productPrice)

Descripción: Añade un producto al carrito. Si el producto ya existe, incrementa su cantidad y actualiza su subtotal. Si no, crea un nuevo objeto y lo añade al carrito.

Pasos principales:

Verifica si el producto ya está en el carrito usando Array.find.

Si ya está:

Incrementa la cantidad del producto.

Actualiza el subtotal (cantidad \* precio).

Si no está:

Crea un nuevo objeto del producto.

Lo añade al carrito con Array.push.

Almacena el carrito en localStorage para persistencia entre recargas de página.

Llama a updateCart para reflejar los cambios en la interfaz.

Función: updateCart()

Descripción: Actualiza la interfaz del carrito, mostrando los productos en una tabla y calculando el total.

Pasos principales:

Limpia el contenido actual de la tabla (cartBody.innerHTML).

Itera sobre los productos del carrito:

Crea filas con información del producto.

Agrega botones para incrementar/disminuir cantidad o eliminar productos.

Calcula el total sumando los subtotales.

Actualiza el texto que muestra el total del carrito.

Función: updateQuantity(productName, action)

Descripción: Permite aumentar o disminuir la cantidad de un producto en el carrito.

Pasos principales:

Busca el producto por nombre.

Modifica la cantidad:

Incrementa o decrementa según la acción.

Evita que la cantidad sea menor que 1.

Actualiza el subtotal del producto.

Llama a updateCart para reflejar los cambios.

Función: removeFromCart(productName)

Descripción: Elimina un producto del carrito.

Pasos principales:

Filtra el carrito excluyendo el producto especificado.

Guarda el carrito actualizado en localStorage.

Llama a updateCart.

Función: checkout()

Descripción: Confirmar la compra.

Pasos principales:

Verifica si el carrito está vacío. Si lo está, muestra un mensaje de alerta.

Si no está vacío, redirige al usuario a una página de ventas (ventas.html).

Función: loadCart()

Descripción: Carga el carrito desde localStorage al abrir la página y muestra los productos en una tabla de la página de ventas.

Pasos principales:

Recupera el carrito desde localStorage o inicializa un arreglo vacío.

Limpia y llena la tabla con los productos.

Calcula y muestra el total en la interfaz.

2. Validación de datos de pago

Función principal: validateAll()

Descripción: Valida todos los campos relacionados con el pago (número de tarjeta, fecha de expiración y CVV).

Pasos:

Llama a funciones de validación individuales para cada campo.

Retorna true solo si todos los campos son válidos.

Validaciones específicas:

validateCardNumber():

Verifica que el número de tarjeta tenga 16 dígitos (usando una expresión regular).

Cambia el color del borde del campo según sea válido o no.

validateExpiryDate():

Valida que la fecha de expiración esté en el formato MM/AA.

Comprueba que la fecha no sea anterior a la actual.

validateCVV():

Verifica que el CVV sea un número de 3 o 4 dígitos.

Función: generateInvoice()

Descripción: Genera una factura mostrando los detalles del cliente y del pago.

Pasos principales:

Captura la información del cliente y del total.

Construye un HTML con la factura.

Lo inserta en un contenedor (messageDiv) para mostrarlo al usuario.

3. Eventos del DOM

DOMContentLoaded

Configura las validaciones en tiempo real y maneja el evento de envío del formulario de pago.

window.onload

Llama a loadCart para cargar el carrito cuando la página termine de cargar.

4. Persistencia con localStorage

localStorage.setItem: Guarda el carrito en formato JSON.

localStorage.getItem: Recupera el carrito almacenado.