UNIVERSIDAD AUTÓNOMA DE SINALOA

Presentación de código del MundoPC

Profe:

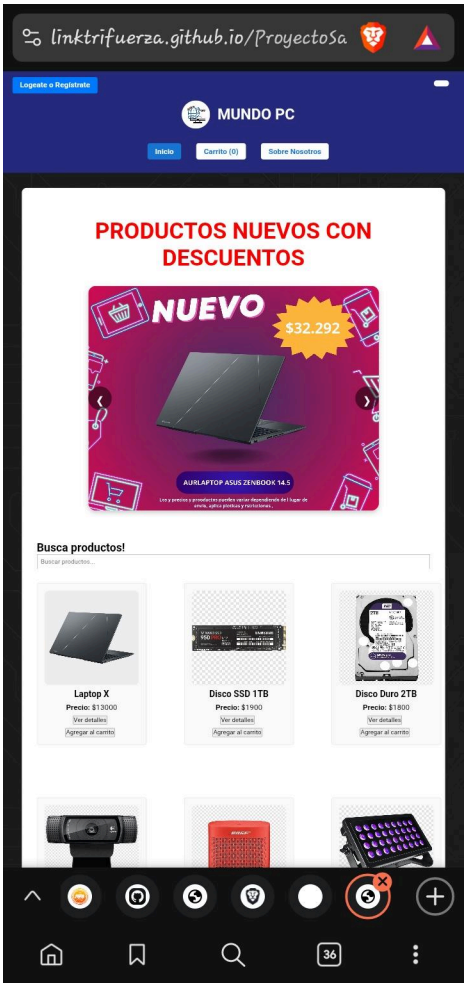
- Dra. Sandra Luz Lara Devora

Integrantes:

- Orduño Angulo Gustavo
- Vega Garcia Susana Valentina
- Fernandez Villa Sergio David

Explicacion del codigo:

En pagina inicio en index.html



Los estilos que usamos para la página de inicio serían los siguientes.

```
<style>
      /* Global */
// Los estilos de cuerpo de la página ( Fondo)
      body {
        font-family: Arial, sans-serif;
        margin: 0;
        background: image-set("fondo.png");
        display: flex;
        flex-direction: column;
        min-height: 100vh;
      }

// Los estilos del header (cabecera de la página)
      header {
        width: 100%;
```

```
    background-color: #262b7e;
    padding: 12px 0 12px 20px;
    box-sizing: border-box;
    position: relative;
  }

// Ajustar contenido que se encuentra dentro del header.
  .left-header {
    display: flex;
    align-items: center;
    margin-right: auto;
  }

// Los estilos de botón del login
  .btn-login {
    display: inline-block;
    padding: 8px 16px;
    background-color: #007bff;
    color: white;
    text-decoration: none;
    font-weight: bold;
    border-radius: 4px;
    transition: background-color 0.3s ease;
  }
  .btn-login:hover {
    background-color: #0056b3;
  }
  /* Logo */
  .link__to__home__logo {
    display: inline-flex;
    flex-direction: row;
    align-items: center;
    gap: 0.5em;
  }
  .logo_img {
    margin-right: 0.5em;
  }
  .link__to__home__logo h1 {
    margin: 0;
  }
  /* Navegación */
  nav {
    display: flex;
    justify-content: center;
    gap: 2em;
    margin: 1em 0;
  }
  nav a {
```

```
  color: #1976d2;
  text-decoration: none;
  font-weight: bold;
  background: #fff;
  padding: 0.5em 1em;
  border-radius: 5px;
}
nav a.active,
nav a:hover {
  background: #1976d2;
  color: #fff;
}
#usuario-info {
  position: absolute;
  right: 2em;
  top: 1.2em;
  background: #fff;
  color: #1976d2;
  padding: 0.4em 1em;
  border-radius: 6px;
  font-weight: bold;
}
#logout-btn {
  position: absolute;
  right: 5em;
  top: 4em;
  background: #f44336;
  color: #fff;
  border: none;
  padding: 0.4em 1.2em;
  border-radius: 6px;
  cursor: pointer;
  display: none;
}
#logout-btn:hover {
  background: #b71c1c;
}
main {
  max-width: 900px;
  margin: 2em auto;
  background: #fff;
  padding: 2em;
  border-radius: 10px;
  flex: 1;
}
/* Sección Inventario */
#inventario-section {
  display: block;
```

```
}
.section__producto__carrosel__ {
  margin-bottom: 2em;
}
.h2__carrosel__ {
  text-align: center;
}
.carrousel__container {
  position: relative;
  overflow: hidden;
}
.carrousel__track {
  display: flex;
  transition: transform 0.3s ease;
}
.carrousel__item {
  min-width: 100%;
}
.carrousel__btn {
  position: absolute;
  top: 50%;
  transform: translateY(-50%);
  background: rgba(0, 0, 0, 0.5);
  color: #fff;
  border: none;
  padding: 0.5em;
  cursor: pointer;
  font-size: 1.5em;
}
.carrousel__btn.left {
  left: 0.5em;
}
.carrousel__btn.right {
  right: 0.5em;
}
.producto {
  border: 1px solid #ddd;
  border-radius: 5px;
  padding: 1em;
  margin: 1em 0;
  background: #fafafa;
  display: flex;
  flex-direction: column;
  gap: 0.5em;
  align-items: center;
}
.producto h3 {
  margin: 0;
```

```
}
.producto img {
  width: 200px;
  height: 200px;
  object-fit: cover;
  border-radius: 10px;
  background: #eee;
}
.producto button {
  margin-right: 0.5em;
}
#productos-container {
  display: grid;
  grid-template-columns: repeat(3, 1fr);
  gap: 5em;
}
/* Sección Carrito */
#carrito {
  display: none;
}
#carrito-items div {
  margin-bottom: 0.5em;
}
.carrito-btn {
  float: right;
  background: #43a047;
  color: #fff;
  border: none;
  padding: 0.5em 1em;
  border-radius: 5px;
  cursor: pointer;
}
.carrito-btn:hover {
  background: #388e3c;
}
.cerrar-carrito {
  float: right;
  background: #f44336;
  color: #fff;
  border: none;
  padding: 0.2em 0.7em;
  border-radius: 5px;
  cursor: pointer;
}
/* Modal de pago */
#modal-pago {
  display: none;
  position: fixed;
```

```
  top: 0;
  left: 0;
  width: 100vw;
  height: 100vh;
  background: rgba(0, 0, 0, 0.5);
  justify-content: center;
  align-items: center;
  z-index: 1000;
}
#modal-pago .modal-content {
  background: #fff;
  color: #222;
  padding: 2em;
  border-radius: 10px;
  max-width: 350px;
  width: 90%;
}
/* Sección Sobre Nosotros */
#sobre-nosotros {
  display: none;
}
#sobre-nosotros h2 {
  text-align: center;
  color: #1976d2;
}
#sobre-nosotros p {
  font-size: 1.1em;
  color: #444;
  text-align: justify;
  margin: 1em 0;
}
footer {
  margin-top: auto;
  text-align: center;
  margin: 2em 0 0 0;
  color: #1976d2;
}
@media (max-width: 600px) {
  main {
    padding: 1em;
  }
  #productos-container {
    grid-template-columns: 1fr;
  }
  #usuario-info,
  #logout-btn {
    position: static;
    display: block;
```

```
        margin: 0.5em auto;
      }
    }
  </style>

// header de la página
<header class="header__mundo__pc">
    <div class="left-header">
      <a href="login.html" class="btn-login">Logeate o Regístrate</a>
    </div>
    <!-- Logo con imagen a la izquierda del texto -->
    <a
      href="#"
      class="link__to__home__logo"
      onclick="mostrarSeccion('inventario-section'); mostrarProductos(); return false;"
    >
      <img src="assets/icon/logo.jpg" alt="Mundo PC" class="logo_img" />
      <h1 style="margin: 0;">MUNDO PC</h1>
    </a>
    <!-- Navegación -->
    <nav>
      <a href="#" id="nav-inicio" class="active">Inicio</a>
      <a href="#" id="nav-carrito" style="position: relative">
        Carrito (<span id="carrito-cantidad">0</span>)
      </a>
      <a href="#" id="nav-nosotros">Sobre Nosotros</a>
    </nav>
    <span id="usuario-info"></span>
    <button id="logout-btn">Cerrar sesión</button>
  </header>

  <!-- Sección Carrito de Compras -->
    <section id="carrito">
      <h2>
        Carrito de Compras
        <button class="cerrar-carrito" onclick="cerrarCarrito()">X</button>
      </h2>
      <div id="carrito-items"></div>
      <button id="vaciar-carrito" style="margin-top: 1em">
        Vaciar Carrito
      </button>
      <button id="btn-pagar" style="margin-top: 1em; margin-left: 1em">
        Pagar
      </button>
    </section>

  <!-- Sección Sobre Nosotros -->
    <section id="sobre-nosotros">
```

```
        <h2>Sobre Nosotros</h2>
        <p>
          Bienvenidos a <strong>Mundo PC</strong>, tu mejor opción en tecnología y computadoras.
          Somos una empresa dedicada a la venta de productos relacionados con computadoras, ofreciendo desde laptops y smartphones hasta componentes y accesorios de última generación.
        </p>
        <p>
          Nuestra misión es proporcionar soluciones tecnológicas innovadoras y de alta calidad que se adapten a las necesidades de profesionales, gamers y entusiastas de la informática.
          Con años de experiencia en el mercado, nos hemos consolidado como líderes en el sector, comprometidos con la excelencia y la satisfacción de nuestros clientes.
        </p>
        <p>
          En Mundo PC, creemos que la tecnología impulsa el progreso y nuestro compromiso es estar siempre a la vanguardia para ofrecerte lo mejor del mundo digital.
        </p>
      </section>

<!-- Modal de pago -->
    <div id="modal-pago">
      <div class="modal-content">
        <h3>Pagar con Tarjeta</h3>
        <form id="form-pago">
          <input type="text" placeholder="Nombre en la Tarjeta" required />
          <input type="number" placeholder="Número de Tarjeta" required />
          <div style="display: flex; gap: 1em">
            <input type="text" placeholder="MM/AA" required style="width: 50%" />
            <input type="number" placeholder="CVV" required style="width: 50%" />
          </div>
          <div style="margin-top: 1em; text-align: right">
            <button type="button" onclick="cerrarModalPago()">Cancelar</button>
            <button type="submit">Confirmar</button>
          </div>
        </form>
      </div>
    </div>

// Utilizamos la librería toastify js para manejar las notificaciones y mejorar la experiencia del usuario
  <script src="https://cdn.jsdelivr.net/npm/toastify-js"></script>

<script>
      let db;
      const DB_NAME = "MundoPcDB";
      const DB_VERSION = 1;
```

```
 function openDB() { // FUNCION PARA UTILIZAR LA API INDEXDB
       return new Promise((resolve, reject) => {
         const request = indexedDB.open(DB_NAME, DB_VERSION);
         request.onerror = () => reject("Error abriendo la base de datos"); // Tirar error si fallo
en abrir la base de datos
         request.onsuccess = (event) => { // en caso que funcione, que ejecute con lo siguiente
           db = event.target.result; // Obtiene el resultado de la bd
           resolve(db);
         };
         request.onupgradeneeded = (event) => {
           db = event.target.result;
           if (!db.objectStoreNames.contains("usuarios")) // Sino existe usuarios en indexb lo
crea
             db.createObjectStore("usuarios", { keyPath: "email" });
           if (!db.objectStoreNames.contains("productos")) // Sino existe productos en indexb lo
crea
             db.createObjectStore("productos", { keyPath: "id" });
           if (!db.objectStoreNames.contains("carritos"))
             db.createObjectStore("carritos", { keyPath: "email" });
         };
       });
     }

function buscarUsuario(email) { // FUNCION PARA BUSCAR USUARIO POR MEDIANTE
EL EMAIL
       return openDB().then((db) => {
         return new Promise((resolve) => {
           const tx = db.transaction("usuarios", "readonly"); // Solo de lectura
           const store = tx.objectStore("usuarios");
           const request = store.get(email); // Obtener el correo mediante usuarios en indexdb
           request.onsuccess = () => resolve(request.result); // Traer el resultado en caso que
sea exista la operacion
         });
       });
     }

 function guardarProductosIniciales(productos) { // Guardar productos
       return openDB().then((db) => {
         const tx = db.transaction("productos", "readwrite");
         const store = tx.objectStore("productos");
         productos.forEach((producto) => store.put(producto)); // Por cada producto que exista
que lo añada en indexb en “productos”
       });
     }

function obtenerProductos() { // Obtener productos
       return openDB().then((db) => {
         return new Promise((resolve) => {
```

```
          const tx = db.transaction("productos", "readonly"); // filtro y solo de lectura
          const store = tx.objectStore("productos");
          const request = store.getAll(); // Traer todos los productos que estén en indexb que
sean “productos”
request.onsuccess = () => resolve(request.result);
        });
      });
    }

 function guardarCarrito(email, carrito) { // Guardar en carrito
      return openDB().then((db) => {
        const tx = db.transaction("carritos", "readwrite"); // Buscar “carritos” en indexb y que
sean “readwrite”
        const store = tx.objectStore("carritos");
        store.put({ email, carrito }); // Añadir correo y carrito al indexb
      });
    }

function obtenerCarrito(email) { // Obtener carrito mediante correo/email
      return openDB().then((db) => {
        return new Promise((resolve) => {
          const tx = db.transaction("carritos", "readonly");
          const store = tx.objectStore("carritos");
          const request = store.get(email);
          request.onsuccess = () => // En el caso que si funcione, ejecute lo siguiente.
            resolve(request.result ? request.result.carrito : []); // Si existen datos en carrito
entonces los trae, pero si no hay ningún dato en result en carrito entonces que sean un
array vacío
        });
      });
    }

  // Inicialización de productos en la base de datos, si aún no han sido guardados
    openDB().then(() => {
      obtenerProductos().then((p) => {
        if (!p || p.length === 0) guardarProductosIniciales(productosEjemplo);
      });
    });
 // ========== Manejo de sesión ==========
    function getUsuarioLogueado() {
      return JSON.parse(localStorage.getItem("usuarioLogueado") || "null");
    }
 function mostrarUsuarioLogueado() {
      const usuario = getUsuarioLogueado();
      const info = document.getElementById("usuario-info");
      const logoutBtn = document.getElementById("logout-btn");
      const loginBtnContainer = document.querySelector(".left-header");
      if (usuario) {
```

```
        info.textContent = `Bienvenido, ${usuario.nombre}`;
        logoutBtn.style.display = "inline-block";
        loginBtnContainer.style.display = "none";
      } else {
        info.textContent = "";
        logoutBtn.style.display = "none";
        loginBtnContainer.style.display = "block";
      }
    }


 // Al cerrar sesión se elimina el usuario, se muestra un Toast y se actualiza la vista a
inventario
    document.getElementById("logout-btn").onclick = function () {
      localStorage.removeItem("usuarioLogueado");
      Toastify({
        text: "Sesión cerrada",
        duration: 2000,
        gravity: "bottom",
        position: "right",
        style: { background: "#1976d2" },
      }).showToast();
      setTimeout(() => {
        mostrarUsuarioLogueado();
        mostrarSeccion("inventario-section");
        mostrarProductos();
      }, 1200); // Despues de 1200 ms se desactiva.
    };
```


```
// ========== Carrito por usuario ==========
    let carrito = []; // Arreglo carrito
    async function cargarCarrito() { // funcion asíncrona para cargarCarrito
      const usuario = getUsuarioLogueado(); // variable para obtener el usuario logueado
      if (usuario) { // Si existe usuario que haga lo siguiente
        carrito = await obtenerCarrito(usuario.email);
      } else {
        carrito = [];
      }
      actualizarCarritoDOM(); // Actualizar el contenido, si se añade algo nuevo al carrito
    }
```

</script

Crea y maneja una base de datos IndexedDB (`MundoPcDB`) con tres almacenes: `usuarios`, `productos`, y `carritos`.

Guarda productos de ejemplo si la base de datos está vacía.

Gestiona usuarios registrados y logueados con `localStorage`.

Muestra productos en el inventario.

Agrega productos al carrito (por usuario logueado).

Guarda el carrito en IndexedDB por `email`.

Usa `Toastify` para notificaciones amigables.

En Auth.js tenemos lo siguiente.

```
// Auth.js
import { toast } from 'react-toastify';
import localforage from 'localforage';

// Tenemos una funcion que registra los usuarios y usando la libreria Localforage para
guardar los datos del usuario
export const registerUser = async (userData) => {
  try {
    await localforage.setItem(userData.email, userData);
    toast.success('Registro exitoso'); // En el caso que sea exitoso la operacion, entonces
envia una notificacion usando toastify
  } catch (error) {
    toast.error('Error en registro'); // En el caso que falle tire una notificacion dando el
mensaje
  }
};


// Aquí tenemos una función de para logear los usuarios
export const loginUser = async (email, password) => {
  const user = await localforage.getItem(email); // Obtenemos el correo mediante la librería
localfogare
  return user?.password === password ? user : null; // Si la contraseña es igual que tiene
éxito, en el caso que no sea igual va tirar un nul
};

// Buscador.js
const [resultados, setResultados] = useState([]);

const handleBusqueda = async (termino) => { // Para buscar productos
  const productos = await localforage.getItem('productos'); // Hace un filtrado que solo quiere
que “productos” que esten en localforgae
  const filtrados = productos.filter(p =>
    p.nombre.toLowerCase().includes(termino.toLowerCase())
   );
  setResultados(filtrados); // añadir los productos filtrados
};
```

```
const track = document.querySelector(".carrousel__track");
const items = document.querySelectorAll(".carrousel__item");
const btnLeft = document.querySelector(".carrousel__btn.left");
const btnRight = document.querySelector(".carrousel__btn.right");

let currentIndex = 0; // El número actual del producto que se encuentra en el carrousel

function updateCarousel() { // función para actualizar el carrousel y añadir una animación,
dependiendo si se cambio el número actual del producto
  const width = items[0].clientWidth;
  track.style.transform = `translateX(-${currentIndex * width}px)`;
}

btnRight.addEventListener("click", () => {
  currentIndex = (currentIndex + 1) % items.length;
  updateCarousel();
});

btnLeft.addEventListener("click", () => {
  currentIndex = (currentIndex - 1 + items.length) % items.length;
  updateCarousel();
});

window.addEventListener("resize", updateCarousel);
console.log("funcionado SI"); // Si jala


// indexeddb.js
let db;
const DB_NAME = 'CoolCenterDB'; // Nombre indexb
const DB_VERSION = 1; // Version indexb

function openDB() { // Funcion para ejecutar
  return new Promise((resolve, reject) => { // Returnar una nueva promesa
    const request = indexedDB.open(DB_NAME, DB_VERSION);
    request.onerror = (event) => reject('Error abriendo la base de datos'); // Tirar error si falla
    request.onsuccess = (event) => {
      db = event.target.result;
      resolve(db);
    };
    request.onupgradeneeded = (event) => {
      db = event.target.result;
      if (!db.objectStoreNames.contains('usuarios')) {
        db.createObjectStore('usuarios', { keyPath: 'email' });
      }
      if (!db.objectStoreNames.contains('productos')) {
        db.createObjectStore('productos', { keyPath: 'id' });
      }
```

```
    };
  });
}

// Guardar usuario
function guardarUsuario(usuario) {
  return openDB().then(db => {
    return new Promise((resolve, reject) => {
      const tx = db.transaction('usuarios', 'readwrite');
      const store = tx.objectStore('usuarios');
      const request = store.add(usuario); // añadimos el usuario en el indexb
      request.onsuccess = () => resolve();
      request.onerror = () => reject('Usuario ya existe'); // Tirar error si el usuario ya existe
    });
  });
}

// Buscar usuario por mediante el correo
function buscarUsuario(email) {
  return openDB().then(db => {
    return new Promise((resolve) => {
      const tx = db.transaction('usuarios', 'readonly');
      const store = tx.objectStore('usuarios');
      const request = store.get(email); // obtener el correo del usuario
      request.onsuccess = () => resolve(request.result); // Traer el resultado
    });
  });
}

// Guardar productos (solo una vez al iniciar)
function guardarProductosIniciales(productos) {
  return openDB().then(db => {
    const tx = db.transaction('productos', 'readwrite');
    const store = tx.objectStore('productos');
    productos.forEach(producto => store.put(producto)); // Por cada producto que se añada al
indexb “productos”
  });
}

// Obtener todos los productos
function obtenerProductos() {
  return openDB().then(db => {
    return new Promise((resolve) => {
      const tx = db.transaction('productos', 'readonly');
      const store = tx.objectStore('productos');
      const request = store.getAll(); // Obtener todos los productos en indexb “productos”
      request.onsuccess = () => resolve(request.result);
    });
```

```
  });
}


// app.js
// Encriptar contraseña simple
function hashPassword(pass) {
    return btoa(pass); // Hash basico
  }

  // Registro
  window.registrarUsuario = function() {
    const nombre = document.getElementById('reg-nombre').value.trim();
// Obtener el nombre del usuario desde el input por mediante el Element ID y agarrar el texto sin espacios

    const email = document.getElementById('reg-email').value.trim();
// Obtener el email del usuario desde el input por mediante el Element ID y agarrar el texto sin espacios

    const pass = document.getElementById('reg-pass').value.trim();

// Obtener el contraseña del usuario desde el input por mediante el Element ID y agarrar el texto sin espacios

 // Si No se ingresaron datos correspondientes, tirar una notificación que todos los campos son obligatorios


    if (!nombre || !email || !pass) {
Toastify({text: "Todos los campos son obligatorios", duration: 2000, gravity: "bottom", position: "right", style: {background: "#f44336"}}).showToast();
      return;
    }

//Expresión regular para validar correos electrónicos, si el correo es invalido va tirar una notificación “Email invalido”

    if (!/^[\w-\.]+@([\w-]+\.)+[\w-]{2,4}$/.test(email)) {
      Toastify({text: "Email inválido", duration: 2000, gravity: "bottom", position: "right", style: {background: "#f44336"}}).showToast();
      return;
    }

// Guardar usuario el nombre, correo, contraseña hasheada
    guardarUsuario({nombre, email, pass: hashPassword(pass)}).then(() => {
      Toastify({text: "Registro exitoso", duration: 2000, gravity: "bottom", position: "right", style: {background: "#4caf50"}}).showToast();
```

```
      setTimeout(() => window.location.href = "login.html", 2000); // Después de 2000 ms se va desaparecer la notificación
    }).catch(err => {
      Toastify({text: err, duration: 2000, gravity: "bottom", position: "right", style: {background: "#f44336"}}).showToast();
    });
  };

  // Login
  window.loginUsuario = function() {
    const email = document.getElementById('login-email').value.trim();
    const pass = document.getElementById('login-pass').value.trim();
```

```
 // Si no se ingresan los datos, tirara una notificación “Todos los campos son obligatorios”

    if (!email || !pass) {
      Toastify({text: "Todos los campos son obligatorios", duration: 2000, gravity: "bottom",
position: "right", style: {background: "#f44336"}}).showToast();
      return;
    }

// Buscar usuario mediante el correo
    buscarUsuario(email).then(usuario => {

          // Si la contraseña es igual a la guardada, es un login existoso
      if (usuario && usuario.pass === hashPassword(pass)) {
        Toastify({text: "Login exitoso", duration: 2000, gravity: "bottom", position: "right", style:
{background: "#4caf50"}}).showToast();
        setTimeout(() => window.location.href = "index.html", 2000);
      } else {
        Toastify({text: "Usuario o contraseña incorrectos", duration: 2000, gravity: "bottom",
position: "right", style: {background: "#f44336"}}).showToast();
      }
    });
  };

// productos.js
document.addEventListener('DOMContentLoaded', () => {
    // Solo la primera vez, agrega productos (id, nombre, precio, características)

// Variable que guarda un Arreglo (JSON)
    const productosEjemplo = [
      {id: '1', nombre: 'Laptop X', precio: 999, caracteristicas: ['Intel i7', '16GB RAM', '512GB
SSD', 'Pantalla 15"', 'Windows 11', 'Batería 8h', 'Peso 1.5kg', 'Color Gris']},
      {id: '2', nombre: 'Smartphone Y', precio: 599, caracteristicas: ['Android 13', '128GB', '6GB
RAM', 'Cámara 48MP', 'Pantalla 6.5"', 'Batería 4000mAh', '5G', 'Color Negro']},
      // ...agrega hasta 20 productos
    ];
    // Solo descomenta la siguiente línea la primera vez para poblar la BD
    // guardarProductosIniciales(productosEjemplo);

    mostrarProductos();
  });

// Funcion para mostrar productos filtrados

  function mostrarProductos(filtrados) {
    obtenerProductos().then(productos => { // Obtener todos los productos
      const lista = filtrados || productos; // Conseguir una lista de filtrados o todos los
productos
      const container = document.getElementById('productos-container');
```

```
    if (!container) return;
    container.innerHTML = lista.map(p => ` //por cada producto se añadirá los siguientes elementos html con su respectivo dato correspondido
      <div class="producto">
        <h3>${p.nombre}</h3>
        <p>Precio: $${p.precio}</p>
        <button onclick="verDetalles('${p.id}')">Ver detalles</button>
        <button onclick="agregarAlCarrito('${p.id}')">Agregar al carrito</button>
      </div>
    `).join('');
  });
 }

 // Búsqueda AJAX
 window.buscarProductos = function(termino) {
  obtenerProductos().then(productos => {
    const filtrados = productos.filter(p =>
      p.nombre.toLowerCase().includes(termino.toLowerCase())
    );
    mostrarProductos(filtrados);
  });
 };

 // Mostrar detalles
 window.verDetalles = function(id) {
  obtenerProductos().then(productos => { // Obtener todos los productos
    const producto = productos.find(p => p.id === id); // Devuelve un unico producto mediante su id
    if (producto) { // Si producto existe

        // variable para obtener el elemento por mediante el id
      const form = document.getElementById('form-contacto');

        // cambiar el texto por lo siguiente.
      form.querySelector('textarea').value = `Me interesa el producto: ${producto.nombre}`;
      window.location.hash = '#contacto';
    }
  });
 };
```