Proyecto Integrador Módulo de Fundamentos de Desarrollo Web Frontend

Se requiere el desarrollo de una aplicación web market place para la compra de accesorios de joyería utilizando HTML, CSS, JavaScript, JSON server para crear una API REST falsa (o mock API), sistema de control de versiones Git, GitHub como plataforma de alojamiento de código, así como opciones de implementación en la nube como Render y GitHub pages, entre otras herramientas relevantes para la construcción de la aplicación.

La aplicación debe permitir a los usuarios:

1. Observar el catálogo de productos,
2. Realizar filtrado de productos por categorías (o tipo de accesorio),
3. Búqueda por nombre del producto,
4. Ordenar productos por precio (de menor a mayor y mayor a menor precio),
5. Ver información de detalle de un producto,
6. Seleccionar productos, color, talla y cantidad,
7. Enviar productos al carrito de compras.
8. Ver el listado de productos en carrito de compras, y
9. Completar el flujo de una compra

El enfoque principal del proyecto es desarrollar un interfaz utilizando HTML para la estructura del contenido de las páginas, JavaScript para darle funcionalidad e interacción y CSS para la aplicación de estilos y diseño responsivo (Responsive and Adaptive Design) que cumpla con el diseño proporcionado a continuación:

https://www.figma.com/file/DArwEkQCrrbZUUUwzSPWZO/Darling?type=design&node-id=1%3A1123&mode=design&t=yUhX5T7BimmkIsVT-1

El objetivo final del proyecto es adquirir experiencia en el desarrollo de aplicaciones frontend responsivas utilizando JavaScript Vanilla, así como la implementación de despliegue y buenas prácticas de desarrollo y calidad del código.

A continuación, se detallan los requerimientos del proyecto:

Diseño UI/UX:

- Cumplir con el diseño proporcionado por el área de diseño.
- Maquetar todas las vistas presentes en el prototipo.
- Implementar diseño responsivo (Responsive Design y Adaptive Design) .
- Realizar propuesta diseño responsive adaptativo para visualizar la aplicación en dispositivos tablets y mobile.

Sistema de control de versiones:

- Utilizar Git y GitHub para gestionar el control de versiones del proyecto.
- Realizar commits frecuentes para asegurar el código. Se recomienda realizar commits por cada feature completamente funcional y sin errores o bugs.

- Para trabajar de manera colaborativa, se recomienda el uso de la extensión live share o la creación de ramas por cada integrante del equipo y posterior fusión del código presente en las ramas a la rama principal.

Implementación de JSON Server:

- Simular REST API con los endpoints necesarios que proporcionen la información de los productos, historial de compras, entre otros datos que considere de relevancia para la aplicación.
- Proponer una estructura de datos óptima en el JSON Server que facilite las operaciones de lectura, filtrado, búsqueda, ordenamiento y creación o envío de datos a los diferentes endpoints.
- Tener disponible en la API al menos 20 productos.
- Si desea generar datos de prueba en formato JSON, puede utilizar JSON Data AI.

Mostrar todos los productos del catálogo:

- Realizar la petición GET con Fetch API que permita obtener todos los productos de la API
- El resultado de esta consulta se debe mostrar mediante cards en la página donde se listan los productos.

Filtrado, búsqueda, ordenamiento y paginación de productos:

- Es posible realizar la operación de filtrado, búsqueda y ordenamiento de dos maneras: la primera forma es implementando las operaciones de Filter, Sort y Paginate disponibles en JSON Server o haciendo uso de los métodos de arrays como .filter() o .sort().
- Mostrar los resultados en la interfaz de usuario donde se listan los productos.
- La paginación de los productos en la UI es una funcionalidad opcional.

Ver información de detalle y agregar producto al carrito de compras:

- Cuando un usuario dé click sobre cualquier card de productos en la página de listado de productos, debe ser redireccionado a la página de detalles correspondiente al producto seleccionado.
- En la página de detalles, el usuario debe observar la galería de imágenes del producto, junto con toda la información relacionada con el artículo de joyería seleccionado, así como debe tener la posibilidad de seleccionar un color, talla y cantidad de artículos a comprar, para finalmente enviar el producto (o productos) al carrito de compras.
- En la vista de detalles, se deben aplicar las validaciones necesarias para garantizar que se envíe toda la información requerida para poder completar una compra (como, por ejemplo, la selección del color y la talla del artículo si aplican e incluso cantidad de producto válida y que no exceda el stock).

Ver carrito de compras:

- Para acceder al carrito de compra es mediante el enlace que se encuentra disponible en la barra de navegación del diseño.
- En la vista del carrito de compra se deben listar los productos que el usuario desea comprar y el total de la compra en un modal o drawer.

- En caso de que el usuario aún no haya enviado o agregado algún producto al carrito, en esta vista debe mostrarse el total de la compra en $0, el botón **Continue to checkout** deshabilitado y siguiente texto: “Carrito de compra vacío”.
- No es requerido la funcionalidad de edición o modificación de un artículo en el cart.

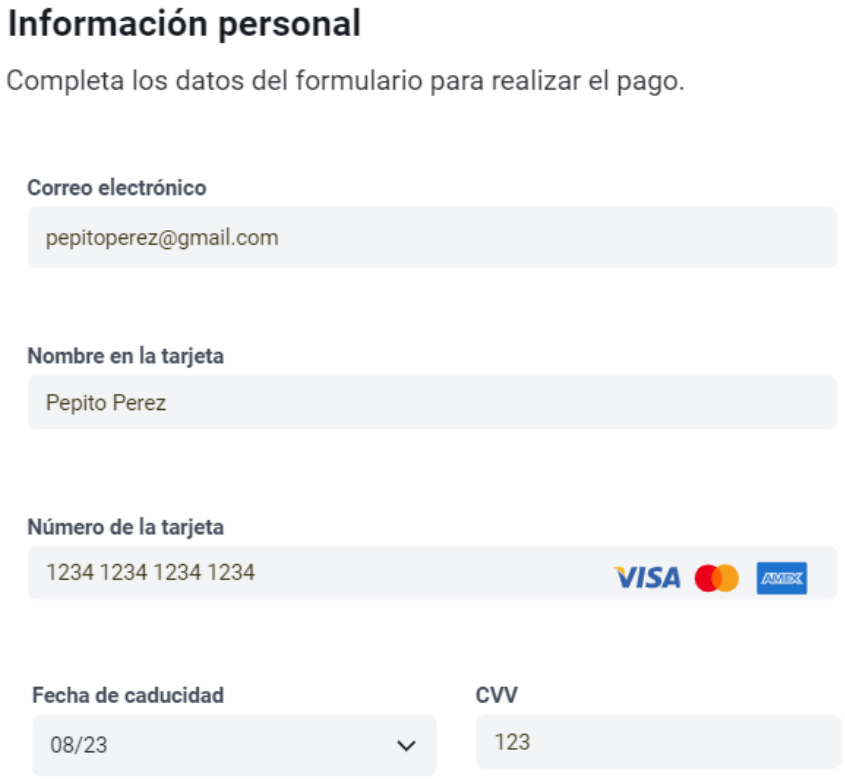
Completar flujo de compra:

- Para completar la compra es necesario redireccionar al usuario a la página de Checkout. Para que el usuario sea direccionado a esta página, debe dar click al botón **Buy now** presente en la página de detalles del producto o al botón **Continue to checkout** presente en la vista del carrito de compras.
- En la vista Checkout, se deben listar los productos que estaban guardados en el carrito de compras, asimismo el usuario debe tener la posibilidad de eliminar o descartar productos, agregar método de pago e información de contacto y, por último, confirmar la compra al dar click en el botón **Proceed to payment**.
- Cuando la compra haya sido finalizada, se debe guardar la información relacionada al usuario y los productos al endpoint disponible para guardar estos datos.
- Debe realizar propuesta de diseño para los formularios donde se le solicitará al usuario su información de contacto y método de pago.

Sugerencias de diseño:

Figura 1. Sugerencia para formulario de datos de contactor del comprador.

Figura 2. Formulario para agregar método de pago.



Despliegue de aplicaciones en la nube:

- Desplegar la aplicación en un servicio de alojamiento en la nube como GitHub pages, vercel, Netlify, entre otros.
- Desplegar API REST simulada en servicios de Hosting como Render, fly.oi, fl0.com, entre otros

En la aplicación se debe evidenciar el uso e implementación de las siguientes herramientas:

1. Responsive Design usando Flex box y media query, opcional Grid Layout
2. Funciones
3. Peticiones HTTP (GET, POST, PUT, PATCH y DELETE) con Fetch API
4. Estructuras de control condicionales y cíclicas
5. JSON Server
6. Eventos
7. Métodos del DOM
8. LocalStorage o SessionStorage
9. Async/Await o .then().catch
10. Realización de README professional

Proceso de entrega:

Para el desarrollo del proyecto integrador tendrá un total de siete (7) semanas contadas a partir de la entrega de los requerimientos, asimismo, las entregas serán porcionadas en cinco (5) partes distribuidas de la siguiente manera:

Primera entrega (enero 9 del 2024):

- Estructura de carpetas y archivo inicial del proyecto alojada en GitHub
- Construcción de la interfaz de usuario Homepage
- La primera entrega se debe realizar diligenciando el siguiente formulario: https://forms.office.com/r/k4GcM0ZXsZ

Segunda entrega (enero 18 del 2024):

- Construcción de las vistas: Product listing, Product details, Checkout – add payment method y Checkout – Payment success.
- Funcionalidad de redirección de las páginas.

Tercera entrega (enero 29 del 2024):

En un archivo script.js, realizar lo siguiente:

- Declarar una lista de los productos con los siguientes datos: id, nombre, código, precio unitario, tipo de accesorio (anillo, brazalete, collar, aretes, etc.), imágenes, descripción, cantidad en stock por color y/o talla.
- Escribir una función que reciba como parámetros un array de productos y el nombre de un tipo de producto, que utilice la función de array que permita filtrar la lista por la categoría o tipo y devuelva el array resultante. Luego, llamar la función pasándole como argumentos la lista de productos declarado en el ítem anterior y cualquier tipo de accesorio que exista en la lista y, por último, mostrar el resultado en la consola del navegador.
- Escribir una función que realice la búsqueda de productos por nombre, reciba como parámetro un array de productos y un término de búsqueda (es decir, una cadena de caracteres) y retorne un array con todos los productos cuyos nombres contengan los caracteres del segundo parámetro. Luego, llamar la función pasándole como argumentos datos de prueba y mostrar el resultado en la consola del navegador.
- Crear una función que ordene un array de productos por precios de manera ascendente y descendente y retorne el array resultante. Ejecutar la función y mostrar el resultado en consola.
- Crear una función que calcule el total a pagar de una compra, reciba como parámetros un array de productos donde cada producto, tenga como propiedades la cantidad y precio unitario del producto y devuelva el valor que corresponda a la sumatoria de la cantidad por el precio de cada producto. Ejecutar la función con datos de prueba y mostrar el resultado en la consola del navegador.

Cuarta entrega (febrero 5 del 2024):

- Construcción de la vista Cart
- Vista de product listing completamente funcional. Se debe hacer uso de las funciones construidas en la entrega anterior.
- Formularios de la vista Checkout – add payment method funcionales con las validaciones requeridas y que se muestren los datos en la consola del navegador al ejecutarse el evento submit.

Quinta entrega (febrero 13 del 2024):

- API REST falsa con JSON Server
- Vista de product details, cart, checkout – add payment method y checkout – payment success completamente funcionales.
- Entregar la URL del despliegue del JSON Server y de la Market place.
- La última entrega del proyecto se debe realizar diligenciando el siguiente formulario: https://forms.office.com/r/V5WDgDDqBe