# Desarrollo de un Carrito de Compras Profesional

### Índice

Introducción	1
Requisitos Generales	1
Componentes y Requisitos Específicos	
1. Store (Pinia)	
2. Composables	
3. Componente de Producto	
4. Componente de Carrito	
5. Vista de Productos	

### Introducción

Crear un sistema de carrito de compras reutilizable, modular y profesional utilizando Vue 3 con TypeScript, la Composition API, Pinia para la gestión del estado global, y un diseño que permita la reutilización de componentes mediante el uso de **slots** y **composables**.

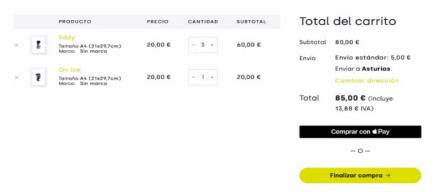
Crear la aplicación a partir de npm create <u>vue@latest</u> y crea una barra de navegación que contenga como mínimo los items Tienda y Carrito.

Puebla la Tienda con 5 productos como mínimo con los atributos: nombre, cantidad, precio, imagen, color, talla.

## **Requisitos Generales**

- 1. **Carrito de Compras**: El sistema debe permitir agregar, eliminar y actualizar productos en el carrito de compras.
- 2. **Pinia**: Utiliza Pinia para manejar el estado global del carrito.
- 3. **Composables**: Utiliza composables para la lógica reutilizable (por ejemplo, gestión de productos en el carrito, cálculo de totales, etc.).
- 4. **Componentes reutilizables**: Todos los componentes deben ser modulares y reutilizables.
- 5. **Interfaz de usuario**: Debe incluir una interfaz de usuario mínima para agregar productos, ver el carrito, modificar la cantidad y proceder al pago.
- 6. **Tipado con TypeScript**: Todo el código debe estar tipado correctamente para garantizar la integridad de los datos.
- 7. **Estilo:** usa cualquier framework CSS para proporcionar un formato agradable.

### Podría ser algo como:



# **Componentes y Requisitos Específicos**

## 1. Store (Pinia)

Crea una tienda Pinia para manejar el estado global del carrito.

Archivo: src/stores/cart.ts

#### **Requisitos:**

- Estado:
  - Un array de productos que estarán en el carrito, cada producto debe tener al menos id, nombre, cantidad, precio, imagen.
  - Un getter para calcular el total del carrito (precio total).
- Acciones:
  - agregarProducto(producto): Añadir un producto al carrito.
  - eliminarProducto(id): Eliminar un producto del carrito por su id.
  - actualizarCantidad(id, cantidad): Actualizar la cantidad de un producto en el carrito.
  - vaciarCarrito(): Vaciar todo el carrito.

#### Tipado:

• Usa interfaces para tipar los productos y el estado global.

## 2. Composables

Crea composables reutilizables para la gestión de productos en el carrito y cualquier lógica que se pueda compartir entre componentes.

Archivo: src/composables/useCart.ts

#### **Requisitos:**

 useCart: Este composable debe manejar la lógica relacionada con la interacción del carrito de compras, como agregar productos, eliminar productos, calcular el IGIC (7%) y obtener el total. • Los productos deben ser proporcionados desde el store Pinia, pero todo el manejo lógico debe estar en el composable.

## 3. Componente de Producto

Archivo: src/components/ProductItem.vue

#### **Requisitos:**

- Mostrar un producto con su imagen, nombre y precio.
- Debe permitir al usuario agregar la cantidad deseada al carrito.
- Añadir validación de stock y mostrar un mensaje de error si el usuario intenta agregar más cantidad de la disponible.
- Usa **props** para recibir la información del producto.

# 4. Componente de Carrito

Archivo: src/components/Cart.vue

#### **Requisitos:**

- Mostrar todos los productos en el carrito.
- Para cada producto, debe mostrar su nombre, cantidad, precio total (precio \* cantidad) y un botón para eliminarlo.
- Incluir un botón para vaciar el carrito y un resumen con el total del carrito.
- Permitir modificar la cantidad de los productos.
- Implementar un sistema de persistencia (por ejemplo, usando localStorage o API para almacenar el carrito).

## 5. Vista de Productos

Archivo: src/views/ProductList.vue

#### **Requisitos:**

- Muestra una lista de productos utilizando el componente ProductItem.
- Usa **slots** para permitir que la vista sea extensible (por ejemplo, para mostrar una lista filtrada de productos).