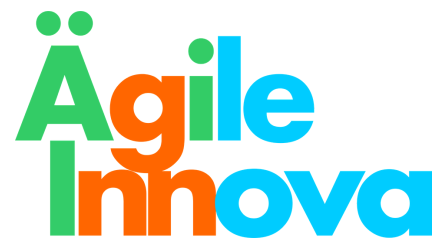


# Academia Geek



RETO 2: DESARROLLO ECOMMERCE

1. Index: Esta es la primera página donde en cada uno de las páginas tendrá un menú con las opciones: Home, Productos (1), y un ícono que refleja las cantidad de ítems agregados al carrito con un enlace al detalle del carrito de compras (2) (Imagen 2). Agregamos una imagen de Bienvenida (3) que se cargará desde una URL referente al tipo de productos que se venderán en la tienda. Agregamos dos Banners (4,5) referentes a las categorías de productos de la tienda, en este ejemplo, tiene categorías: Hombre y Mujer, lo cuál podría ser aplicable a una tienda de ropa, por ejemplo, cada banner debe tener un enlace a la página de productos (Imagen 3), pero cargando los productos pertenecientes a esta categoría. Por último, creamos un Footer (6) que al igual que la sección de menú, se compartirá en todas las vistas del sitio.



Imagen 1. Index

2. Carrito de Compras: En esta página se listan los productos que se han agregado al carrito, donde se muestra el nombre del producto, imagen, cantidad, precio unitario y subtotal equivalente a el costo de ese ítem multiplicado por la cantidad. Al dar clic en el botón pagar nos lleva al formulario de datos de "Información de Pago" (Imagen 5).





1 Menú				2 
Producto 1	Cantidad	Precio	Sub TOTAL	
	2	\$ 35.000	\$70.000	
Producto 2	Cantidad	Precio	Sub TOTAL	
	1	\$ 10.000	\$ 10.000	
TOTAL			\$ 80.0000	
			3 	
Footer Información de Contacto y Redes sociales				6

Imagen 2. Detalle carrito de compras

3. Página de Productos: Acá se listan los productos según la categoría seleccionada, esta recibe el id de la misma por parámetro y carga la categoría deseada, y por ende los productos que pertenecen a la categoría seleccionada, cada producto tiene enlace a una página de detalle (Imagen 4)..

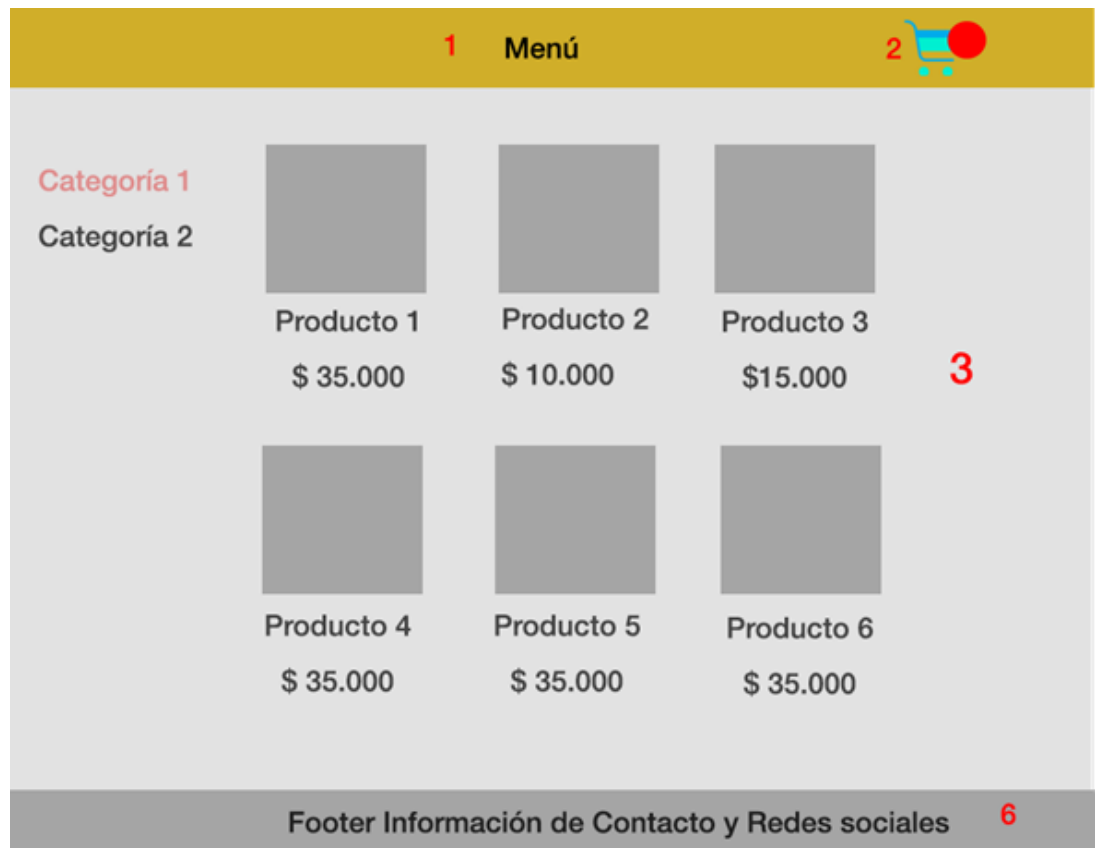


Imagen 3: Mosaico de productos, donde cada producto tiene un enlace al detalle de este. (Imagen 4).

4. Detalle del producto: Acá se muestra el producto deseado, según el id que viene por parámetro, mostrando la imagen, el título, el precio, la descripción, un input para escribir la cantidad y un botón para agregarlo al carrito de compras de la sesión actual.



Imagen 4: Detalle del producto.

5. Información de Pago: Formulario que captura los datos personales y de pago del usuario, estos datos se deben guardar junto con la relación de los productos agregados al carrito, teniendo en cuenta: productos, cantidad, precio, y subtotal.

The image shows a web interface for a payment form. At the top, there is a yellow header bar containing a red number '1' next to the text 'Menú' and a red number '2' next to a shopping cart icon. The main content area has a light gray background and is titled 'Información de Pago' with a red number '3' to its right. Below the title are four input fields: 'Nombre completo', 'Correo electrónico', 'No. de Tarjeta', and 'F. Vencimiento MM/AA' (with a 'CVV' field to its right). A large blue button with the text 'PAGAR' is centered below the fields. At the bottom, there is a dark gray footer bar with the text 'Footer Información de Contacto y Redes sociales' and a red number '6' to its right.

Imagen 5: Formulario de pago



# TALLER

## Creación del frontend

Teniendo en cuenta la estructura y mockups planteados, creamos las 5 interfaces que requerimos para el ecommerce, es necesario conservar la distribución planteada en los mockups pero se puede utilizar como referencia el diseño deseado.

## Creación del Carrito de Compras

Creamos una clase JavaScript llamada Carrito, la cuál instanciaremos en el frontend y al amcenaremos su estado de manera local, que tendrá los métodos necesarios para:

- Adicionar producto al carrito
- Sumar el valor de todos los productos
- Listar los productos en el carrito
- Eliminar un producto, y vaciar el carrito.

Para lograr almacenar la información del carrito en la sesión, podemos usar localStorage, donde se pueden guardar los objetos JSON referentes a la información de la compra y la información del formulario de pago.

Una vez el usuario haga clic en PAGAR se debe almacenar los datos del carrito y la compra en un archivo JSON y remover los valores almacenados en el localStorage para que el carrito quede nuevamente vacío.

<https://dribbble.com/search/%20ecommerce>

<https://www.npmjs.com/package/lowdb>

<https://pugjs.org/api/getting-started.html>