

Documentación del proyecto Shopi

ALBERT ALARCÓN MARTÍNEZ



Uso de Tailwind CSS en el Proyecto



Configuración en `tailwind.config.cjs`

El archivo de configuración de Tailwind `tailwind.config.cjs` define las reglas básicas del framework en el proyecto:



Modo Oscuro

```
darkMode: 'class',
```

- **Modo oscuro activado manualmente** mediante la clase `dark`.
- Permite alternar entre `light` y `dark` aplicando `document.documentElement.classList.add('dark')`.



Rutas de los archivos CSS

```
content: [  
  "./index.html",  
  "./src/**/*.{js,ts,jsx,tsx}",  
],
```

- **Tailwind escanea los archivos** dentro de `src/` y `index.html` para optimizar los estilos generados.



Personalización del tema

Fuente global personalizada

```
fontFamily: {  
  sans: ['Poppins', 'sans-serif'],  
},
```

- **Fuente primaria:** `Poppins`, garantizando un diseño limpio y moderno.

Colores personalizados

```
colors: {  
  primary: '#3ea987', // Verde principal  
  secondary: '#49caa1', // Verde más claro  
  accent: '#60a5fa', // Azul para detalles  
},
```

- **Colores principales para la interfaz** (`primary`, `secondary`, `accent`).
- **Facilita la coherencia en toda la UI.**

Bordes redondeados personalizados

```
borderRadius: {  
  'x1': '1rem',  
  '2x1': '1.5rem',  
},
```

- Define radios de borde más grandes (`x1` y `2x1`) para un diseño más **moderno y suave**.

Estilos Globales en `Pages/App/App.css`

```
@tailwind base;          /* Estilos base de Tailwind */  
@tailwind components;    /* Componentes de Tailwind */  
@tailwind utilities;     /* Utilidades de Tailwind */
```

- **Organización estándar de Tailwind** para manejar la carga de estilos.

Variables CSS personalizadas

```
:root {  
  --primary-color: #3ea987;  
  --secondary-color: #000000;  
}
```

- Permite **usar los colores personalizados en todo el CSS**.
- **Ejemplo de uso:**

```
background-color: var(--primary-color);
```

Comportamiento del scroll

```
html {  
  scroll-behavior: smooth;  
}
```

- **Habilita el scroll suave** al navegar en la página.

Estilos base del **body**

```
body {  
  font-family: 'Poppins', sans-serif;  
  @apply bg-white dark:bg-gray-900 text-gray-900 dark:text-white transition-  
  colors duration-200;  
}
```

- **Usa Tailwind para:**
 - Establecer el fondo blanco (**bg-white**) en modo claro y **dark:bg-gray-900** en modo oscuro.
 - **Colores del texto:** **text-gray-900** en claro y **dark:text-white** en oscuro.
 - **Transiciones suaves** al cambiar de tema.

Uso de **flex** y **grid** en el Diseño

Sistema de Layout con **flex**

Tailwind se usa ampliamente para **estructurar los componentes con flex**:

```
<div className="flex flex-col min-h-screen">
```

- **flex**: Distribuye los elementos en fila/columna.
- **flex-col**: Organiza los hijos en **columna**.
- **min-h-screen**: Hace que ocupe **toda la pantalla**.

Ejemplo de uso en un contenedor de productos:

```
<div className="flex flex-wrap gap-4">
```

- **flex-wrap**: Permite que los elementos pasen a la siguiente línea si no caben.
- **gap-4**: Añade un espacio uniforme entre los elementos.

Sistema de Layout con **grid**

Para el **listado de productos y otras secciones** se usa **grid**:

```
<div className="grid gap-4 grid-cols-1 sm:grid-cols-2 lg:grid-cols-4 p-4">
```

- **grid**: Activa CSS Grid.
- **gap-4**: Espaciado uniforme entre elementos.
- **grid-cols-1 sm:grid-cols-2 lg:grid-cols-4**:
 - **Móvil**: 1 columna (**grid-cols-1**).
 - **Pantallas medianas (sm)**: 2 columnas (**grid-cols-2**).
 - **Pantallas grandes (lg)**: 4 columnas (**grid-cols-4**).
- **Esto hace el diseño 100% responsive automáticamente.**

🌙 Modo Oscuro con Tailwind

El modo oscuro está habilitado con la configuración:

```
darkMode: 'class',
```

y se usa en los estilos:

```
@apply bg-white dark:bg-gray-900 text-gray-900 dark:text-white;
```

- **Ejemplo aplicado al <body>**:
 - En modo claro → **bg-white text-gray-900**
 - En modo oscuro → **bg-gray-900 text-white**
 - **El cambio es automático según la clase dark en <html>**.

📱 Estilos para Inputs (**range**) en modo oscuro

```
.dark input[type="range"]::-webkit-slider-runable-track {  
  background: #374151;  
}
```

- **Se cambia el color de la barra en modo oscuro** para adaptarse mejor a la UI.

🌀 Estilización de **Swiper.js**

🎮 Botones de navegación personalizados

```
.swiper-button-next,  
.swiper-button-prev {
```

```
    color: white !important;
}
```

- Cambia el color de los botones de navegación (`next` y `prev`) a blanco.

🔴 Paginación personalizada

```
.swiper-pagination-bullet {
  background: white !important;
}

.swiper-pagination-bullet-active {
  background: var(--primary-color) !important;
}
```

- Bullets (• • •) en blanco por defecto.
- Bullet activo usa el color primario (`var(--primary-color)` → `#3ea987`).

📦 Componentes Reutilizables con `@apply`

El uso de `@apply` en Tailwind permite **evitar repetición de estilos**:

```
body {
  @apply bg-white dark:bg-gray-900 text-gray-900 dark:text-white transition-
  colors duration-200;
}
```

Esto evita escribir manualmente:

```
body {
  background-color: white;
  color: #111;
}
body.dark {
  background-color: #1a1a1a;
  color: white;
}
```


- Más limpio y fácil de mantener.

📖 Framer Motion y Swiper

En **Shopi**, usamos dos librerías principales para mejorar la experiencia de usuario con animaciones y carruseles:

1. **Framer Motion** - Para animaciones suaves y transiciones dinámicas.
2. **Swiper** - Para implementar un carrusel interactivo en la página de inicio.

1. Framer Motion - Animaciones en la App

 **Framer Motion** es una librería para React que permite añadir **animaciones fluidas** a los componentes.

Importación

```
import { motion } from 'framer-motion'
```

- **motion** es un componente especial que reemplaza elementos HTML estándar (**div**, **button**, etc.) para agregar **animaciones declarativas**.

Uso Básico

```
<motion.div
  initial={{ opacity: 0, y: -20 }}
  animate={{ opacity: 1, y: 0 }}
  transition={{ duration: 0.5 }}
>
  ¡Hola, Framer Motion!
</motion.div>
```

Explicación:

- **initial={{ opacity: 0, y: -20 }}** → Estado inicial (invisible y desplazado hacia arriba).
- **animate={{ opacity: 1, y: 0 }}** → Estado final (visible y en su posición original).
- **transition={{ duration: 0.5 }}** → Duración de la animación (0.5 segundos).

Uso en Shopi

Animación en Navbar

 La **Navbar** aparece con un **desplazamiento desde arriba**.

```
<motion.nav
  initial={{ y: -100 }}
  animate={{ y: 0 }}
  transition={{ duration: 0.3 }}
  className="fixed top-0 w-full bg-white shadow-md"
```

```
>  
  <h1>Shopi</h1>  
</motion.nav>
```

☑ Explicación:

- Se **desplaza desde y: -100 hasta y: 0** al cargarse la página.

💡 Botón con animación de escala

📍 Al hacer **hover o clic**, cambia su tamaño.

```
<motion.button  
  whileHover={{ scale: 1.1 }}  
  whileTap={{ scale: 0.9 }}  
  className="px-4 py-2 bg-primary text-white rounded-lg"  
>  
  Comprar Ahora  
</motion.button>
```

☑ Explicación:

- `whileHover={{ scale: 1.1 }}` → Aumenta el tamaño en hover.
- `whileTap={{ scale: 0.9 }}` → Se reduce ligeramente al hacer clic.

🐼 Paginación Animada en ProductFilters

📍 En la paginación de productos, los botones tienen un **efecto de rebote**.

```
<motion.button  
  whileHover={{ scale: 1.05 }}  
  whileTap={{ scale: 0.95 }}  
  className="bg-primary text-white px-4 py-2 rounded-lg"  
>  
  Siguiente  
</motion.button>
```


☑ Explicación:

- **Aumenta** al pasar el ratón (`scale: 1.05`).
- **Se reduce** al hacer clic (`scale: 0.95`).


☑ Resumen de Framer Motion

Uso

🐼 Ejemplo

Uso	 Ejemplo
Animación de entrada	<code>initial={{ opacity: 0 }} y animate={{ opacity: 1 }}</code>
Hover & Tap	<code>whileHover={{ scale: 1.1 }} y whileTap={{ scale: 0.9 }}</code>
Transiciones suaves	<code>transition={{ duration: 0.5 }}</code>
Animaciones en listas	<code>layout</code> en <code>motion.div</code>

2. Swiper - Carrusel de Productos

 **Swiper** es una librería para implementar **carruseles y sliders responsivos**.

Instalación

```
npm install swiper
```

Importación

```
import { Swiper, SwiperSlide } from 'swiper/react'
import { Autoplay, Pagination, Navigation } from 'swiper/modules'
import 'swiper/css'
import 'swiper/css/pagination'
import 'swiper/css/navigation'
```

- **Swiper**: Componente principal del carrusel.
- **SwiperSlide**: Representa cada diapositiva.
- **Autoplay**: Habilita la reproducción automática.
- **Pagination**: Muestra puntos de navegación.
- **Navigation**: Agrega botones de siguiente y anterior.

Uso en Shopi - Hero Section

 En **Hero/index.jsx**, se usa **Swiper** para mostrar los productos destacados.

```
<Swiper
  spaceBetween={0}
  centeredSlides={true}
  effect="fade"
  autoplay={{
    delay: 5000,
    disableOnInteraction: false,
  }}
  pagination={{
    clickable: true,
```



```
      dynamicBullets: true,
    }}
    navigation={true}
    modules={[Autoplay, Pagination, Navigation]}
    className="w-full h-full"
  >
  {featuredProducts.map((product) => (
    <SwiperSlide key={product.id}>
      <div className="flex flex-col md:flex-row items-center justify-between">
        <div className="text-center md:text-left">
          <h2>{product.title}</h2>
          <p>{product.description}</p>
          <button className="bg-primary text-white px-4 py-2 rounded-lg">
            Ver Producto
          </button>
        </div>
        <img src={product.image} className="w-1/2 h-auto" />
      </div>
    </SwiperSlide>
  ))}
</Swiper>
```

☑ Explicación:

- `effect="fade"` → Aplica un **efecto de transición suave**.
- `autoplay={{ delay: 5000 }}` → Cambia cada **5 segundos**.
- `pagination={{ clickable: true }}` → Permite cambiar de slide con **puntos interactivos**.
- `navigation={true}` → Agrega **botones de siguiente/anterior**.

⚙ Configuración de Swiper en `tailwind.config.js`

🔗 Swiper se integra bien con Tailwind CSS sin necesidad de configuraciones adicionales.

☑ Resumen de Swiper

Funcionalidad	🐼 Ejemplo
Carrusel básico	<code><Swiper> <SwiperSlide></code>
Paginación	<code>pagination={{ clickable: true }}</code>
Autoplay	<code>autoplay={{ delay: 5000 }}</code>
Botones de navegación	<code>navigation={true}</code>
Efecto Fade	<code>effect="fade"</code>

🔗 Comparación: Framer Motion vs. Swiper

Librería	Uso Principal	Ejemplo en Shopi
----------	---------------	------------------

Librería	Uso Principal	Ejemplo en Shopi
Framer Motion	Animaciones & Transiciones	Navbar, botones, carrito
Swiper	Carruseles & Sliders	Hero Section