Guía de Estudio de Arquitectura Frontend

Proyecto de Referencia: Tienda Online con React

Juan Felipe Orozco Cortés

Instructor - SENA

Septiembre de 2025

Índice

1. Introducción y Configuración Inicial **3**

1.1. Propósito de esta Guía 3

1.2. Paso 1: Descargar y Ejecutar el Proyecto 3

2. Análisis de la Arquitectura Feature-Based **3**

2.1. El Plano del Proyecto 3

2.2. El Rol de cada Carpeta 4

2.3. El Patrón Clave: Layout Component 4

3. Flujo de Datos y Enrutamiento **4**

3.1. El Director de Orquesta: `App.jsx` 4

3.2. El Punto de Entrada: `main.jsx` 5

4. Próximos Pasos: Hacia una Tienda Funcional **5**

1. Introducción y Configuración Inicial

1.1. Propósito de esta Guía

Esta guía de estudio tiene como objetivo deconstruir el proyecto de referencia `react-tienda-arquitectu`. Analizaremos las decisiones de arquitectura, la estructura de carpetas y los patrones de diseño implementados para que puedan entender los fundamentos antes de aplicarlos a su propio proyecto formativo.

1.2. Paso 1: Descargar y Ejecutar el Proyecto

Para estudiar el código, primero deben tenerlo funcionando en su entorno local.

1. **Clonar el repositorio desde GitHub:**

```
git clone https://github.com/topassky3/react-tienda-arquitectura.
    git
```

2. **Navegar a la carpeta e instalar dependencias:**

```
cd react-tienda-arquitectura
npm install
```

3. **Ejecutar el proyecto:**

```
npm run dev
```

4. **Verificar:** Abran su navegador en http://localhost:5173/. Deberían ver la página del Dashboard.

2. Análisis de la Arquitectura Feature-Based

El proyecto no está organizado por tipo de archivo, sino por **funcionalidad** o **çaracterística"(feature)**. Esta es una práctica profesional para proyectos que buscan ser escalables y fáciles de mantener.

2.1. El Plano del Proyecto

```
/src
|-- /features/
|   |-- /dashboard/
|   |   `-- /pages/
|   |       `-- DashboardPage.jsx
|   `-- /shared/
|       |-- /components/
```

```
|       |   |-- /Header/ ...
|       |   `-- /Footer/ ...
|       `-- /layouts/
|           `-- MainLayout.jsx
|
|-- App.jsx
`-- main.jsx
```

2.2. El Rol de cada Carpeta

- `/features`: Es el corazón de la aplicación. Cada subcarpeta aquí representa una gran funcionalidad de nuestra tienda (dashboard, productos, carrito, etc.).
- `/shared`: Es una çaracterística.[es]pecial que contiene código reutilizable por **todas** las demás características. Aquí viven los componentes de layout como el `Header` y el `Footer`.

2.3. El Patrón Clave: Layout Component

La decisión de arquitectura más importante en este proyecto es el uso del componente `MainLayout.jsx`.

- **El Problema:** Sin un layout, tendríamos que repetir el código del `<Header />` y `<Footer />` en cada una de nuestras páginas, lo cual viola el principio de "No Repetirse"(DRY).
- **La Solución:** `MainLayout.jsx` actúa como un "marco de fotos". Define la estructura visual común (Header y Footer) y deja un "hueco.[en] el medio.
- **La Magia:** El componente `<Outlet />` de `react-router-dom`, que usamos dentro de `MainLayout`, es el encargado de renderizar la página correspondiente a la URL activa en ese "hueco".

3. Flujo de Datos y Enrutamiento

Ahora que entendemos la estructura, veamos cómo se conectan las piezas.

3.1. El Director de Orquesta: `App.jsx`

El componente `App.jsx` tiene una única y muy importante responsabilidad: definir las **rutas** de la aplicación. Observa cómo se estructura:

```
function App() {
  return (
    <Routes>
```

```
      {/* Ruta Padre: aplica el MainLayout a todas sus hijas */}
      <Route path="/" element={<MainLayout />}>

        {/* Ruta Hija: se renderiza DENTRO del Outlet del MainLayout
           */}
        <Route index element={<DashboardPage />} />

        {/* Futuras rutas que tambien usaran el mismo layout */}
        {/* <Route path="productos" element={<ProductsPage />} /> */}
      </Route>
    </Routes>
  );
}
```

Esta estructura de rutas anidadas es la que permite que el patrón Layout funcione.

3.2. El Punto de Entrada: `main.jsx`

Este archivo inicia todo el proceso. Su trabajo es .^{en}chufar"nuestra aplicación de React al HTML. La única pieza clave aquí es el componente `<BrowserRouter>`, que activa la magia del enrutamiento en toda la aplicación.

4. Próximos Pasos: Hacia una Tienda Funcional

La arquitectura que hemos construido es el esqueleto profesional sobre el que crecerá nuestra tienda. Los siguientes pasos en nuestro proyecto serán:

1. **Construir la Característica de Productos:**
   - Crear los componentes `ProductCard` y `ProductList`.
   - Crear la página `ProductsPage`.
   - Conectar la página a una API real usando los hooks `useState` y `useEffect`.
2. **Implementar la Gestión de Estado Global:**
   - Crear la característica del Carrito de Compras.
   - Utilizar un patrón como **"Lifting State Up"** o **Context API** para que el `Header` pueda saber cuántos productos hay en el carrito.

Esta base sólida nos permitirá añadir nuevas funcionalidades de forma ordenada y profesional.