Trabajo Práctico Integrador (FASE 2) - E-Commerce con React

📋 Información General

- **Modalidad**: Individual
- **Fecha de entrega**: viernes 31 de octubre
- **Formato de entrega**: Repositorio GitHub con acceso público

Objetivo

Desarrollar una aplicación web de e-commerce completa utilizando React para el frontend y JSON Server para simular el backend. El proyecto debe demostrar el dominio de conceptos fundamentales de React, gestión de estado, comunicación con APIs y buenas prácticas de desarrollo.

Descripción del Proyecto

Crear una tienda online temática (a elección del alumno) que permita a los usuarios navegar por productos, agregarlos a un carrito de compras y simular el proceso de compra. La temática puede ser: tecnología, ropa, libros, alimentos, instrumentos musicales, etc.

Stack Tecnológico

Frontend

- **React** (versión 18 o superior)
- **Vite** como bundler
- **React Router DOM** para navegación SPA
- **Zustand** para manejo de estado global
- **CSS, SCSS, Tailwind, Bootstrap** para estilos (a elección)

Backend

- **JSON Server** para simular API REST
- Base de datos JSON con al menos 10 productos

📌 Requerimientos Funcionales

1. Navegación y Estructura (15 puntos)

- ☐ **Header/Navbar** con:
  - Logo o nombre de la tienda
  - Menú de navegación (Home, Productos, Carrito, Acerca de..)
  - Diseño responsive
- ☐ **Footer** con información de la tienda
- ☐ **Sistema de rutas** implementado con React Router:
  - `/` - Página de inicio

- `/products` - Catálogo de productos
- `/about` - Información sobre la tienda
- Página de respuesta para rutas no encontradas

2. Catálogo de Productos (20 puntos)

- ☐ **Lista de productos** con diseño de grilla responsive
- ☐ **Tarjeta de producto** mostrando:
  - Imagen del producto
  - Nombre
  - Descripción (puede ser truncada)
  - Precio
  - Botón “Agregar al carrito”
- ☐ **Carga de productos** desde el backend (JSON Server)
- ☐ **Manejo de estados de carga** (loading, error, success)

3. Carrito de Compras (25 puntos)

- ☐ **Drawer/Modal lateral** para el carrito
- ☐ **Funcionalidades del carrito**:
  - Agregar productos
  - Incrementar/decrementar cantidad
  - Eliminar productos
  - Mostrar subtotal por producto
  - Mostrar total general
  - Botón para vaciar carrito
- ☐ **Persistencia del carrito** en localStorage
- ☐ **Actualización en tiempo real** del contador en el header

4. Proceso de Compra (20 puntos)

- ☐ **Botón de checkout** en el carrito
- ☐ **Confirmación de compra** con:
  - Resumen de productos
  - Total a pagar
  - Formulario básico (nombre, email, teléfono)
- ☐ **Envío de orden** al backend (POST a /orders)
- ☐ **Mensaje de confirmación** tras compra exitosa
- ☐ **Limpieza del carrito** después de compra

5. Estado Global con Zustand (10 puntos)

- ☐ **Store de carrito** con:
  - Estado de items del carrito
  - Acciones (add, remove, clear, etc.)
  - Getters (total, cantidad de items)
  - Persistencia con middleware

6. Comunicación con API (10 puntos)

- ☐ **Servicio API centralizado** con funciones para:
  - Obtener productos
  - Crear órdenes de compra (deben ser almacenadas en el backend)
  - Manejo de errores

Criterios de Evaluación

Funcionalidad (40%)

- Todas las características funcionan correctamente
- Sin errores importantes en consola
- Manejo adecuado de casos

Código (30%)

- Organización y estructura clara
- Componentes reutilizables
- Uso correcto de hooks y estado
- Buenas prácticas de React

Diseño y UX (20%)

- Interfaz atractiva y profesional
- Responsive design
- Feedback visual al usuario
- Navegación intuitiva

Documentación (10%)

- README completo con instrucciones
- Comentarios en código donde sea necesario
- Commits descriptivos en Git

Instrucciones de Entrega

1. **Crear repositorio en GitHub** con nombre: `tp-ecommerce-[apellido]`
2. **README principal** debe incluir:
   - Nombre del alumno
   - Temática elegida
   - Tecnologías utilizadas
   - Instrucciones de instalación y ejecución

Tips y Recomendaciones

1. **Comenzar por la estructura base**: Configurar el proyecto, rutas y componentes básicos
2. **Desarrollar incrementalmente**: Primero funcionalidad, luego diseño
3. **Usar datos mock inicialmente**: Antes de conectar con JSON Server
4. **Commits frecuentes**: Documentar el progreso del desarrollo
5. **Probar en diferentes dispositivos**: Asegurar responsive design
6. **Reutilizar componentes**: Evitar código duplicado
7. **Manejar estados de carga y error**: Mejorar la experiencia del usuario

Consideraciones Importantes

- **NO copiar código directamente** del proyecto de ejemplo
- **Personalizar la temática** y diseño según elección
- **Documentar decisiones técnicas** importantes
- **Asegurar que el proyecto funcione** antes de entregar
- **Respetar la fecha de entrega** establecida

Recursos Útiles

- Documentación de React
- React Router
- Zustand
- TailwindCSS
- JSON Server
- Vite
- Repo ejemplo