Diseño y Programación de Software Multiplataforma DPS G05L

Desafio Practico 1

Nombre: Jorge Nahum Mira Flores
Carnet: MF232232

Link del video:

https://youtu.be/pjmXk_lQoLM

Estructura del Proyecto:

Este proyecto de React es una aplicación de comercio electrónico llamada "Mira Flores Shop". Aquí hay una explicación del código y la estructura del proyecto:

Archivo `page.jsx`:

1. Importaciones:
- Se importan las librerías necesarias de React, React-PDF, y otros componentes y estilos.

2. Clase `App`:
- La clase principal que extiende `React.Component`.
- Contiene el estado de la aplicación, que incluye la lista de productos, el carrito de compras y las facturas.
- Métodos:
  - `agregarAlCarrito`: Agrega un producto al carrito, reduce el stock del producto.
  - `eliminarDelCarrito`: Elimina un producto del carrito, incrementa el stock del producto.
  - `vaciarCarrito`: Vacía el carrito.
  - `calcularTotal`: Calcula el total de los productos en el carrito.
  - `handleCompra`: Maneja la acción de realizar una compra, generando una nueva factura.

3. Renderización:
- Renderiza la interfaz de usuario con la lista de productos, el carrito, el total del carrito y un botón para realizar la compra.

Archivo `carrito.jsx`:

1. Componente `Carrito`:
- Recibe las propiedades `carrito` y `eliminarDelCarrito`.
- Renderiza la lista de productos en el carrito con la opción de eliminar cada uno.

Archivo `producto.jsx`:

1. Componente `Producto`:
- Recibe las propiedades de un producto y la función `agregarAlCarrito`.
- Renderiza la información del producto con un botón para agregar al carrito.

Archivo `calcularTotal.jsx`:

1. Componente `CalcularTotal`:
- Recibe las propiedades `carrito`.
- Calcula el total del carrito y lo muestra en la interfaz.

Estructura general:

1. Componentes React:
- La aplicación está dividida en componentes React, lo que facilita el mantenimiento y la reutilización del código. Tenemos componentes como `Producto`, `Carrito`, y `CalcularTotal`.

2. Estilos CSS Modulares:
- Se utiliza un módulo de estilos CSS (`page.module.css`) para cada componente. Esto mejora la modularidad y evita conflictos de nombres de clases en la aplicación.

3. Manejo del Estado:
- La aplicación utiliza el estado de React para gestionar dinámicamente la lista de productos, el carrito de compras y las facturas generadas.

Programación y Lógica:

1. Acciones de Compra:
- La lógica de agregar, eliminar y vaciar el carrito se maneja mediante métodos como `agregarAlCarrito`, `eliminarDelCarrito` y `vaciarCarrito`. Estas acciones actualizan el estado de la aplicación de manera eficiente.

2. Generación de Facturas:
- La función `handleCompra` se encarga de generar facturas aleatorias cuando se realiza una compra. Los elementos de React representan la información de la factura, y se actualiza el estado con la nueva factura y se vacía el carrito.

3. Cálculo del Total:
- El componente `CalcularTotal` realiza el cálculo del total del carrito utilizando el método `calcularTotal`. Esta lógica es encapsulada en un componente para facilitar la reutilización.

Paradigma de Programación:

1. Programación Orientada a Objetos (OOP):
- La aplicación sigue los principios de OOP al usar clases y componentes. La clase `App` representa el estado global de la aplicación y encapsula la lógica relacionada con la gestión de productos y carrito.

2. Programación Declarativa:
- Se emplea un enfoque declarativo en la creación de componentes y en la representación visual. La interfaz de usuario se actualiza automáticamente en respuesta a los cambios de estado sin la necesidad de manipulación directa del DOM.

3. React como Biblioteca Declarativa:
- React se utiliza como una biblioteca declarativa para construir la interfaz de usuario. Los componentes se actualizan automáticamente en respuesta a cambios en el estado, simplificando la lógica y reduciendo efectivamente los posibles errores.