Evidencia 01: Configuración inicial de entorno React + Vite + Tailwind + Axios

Objetivo

Evaluar la capacidad del estudiante para:

- Configurar un entorno de desarrollo moderno con React utilizando Vite.
- Instalar y configurar Tailwind CSS en un proyecto de React.
- Realizar fetching de datos desde una API externa usando Axios.
- Mostrar datos dinámicamente en pantalla a partir de la información recibida.

Actividades

1. Crear un nuevo proyecto con Vite

- Inicializar un nuevo proyecto con Vite utilizando React como plantilla.
- Ejecutar correctamente el proyecto de forma local con `npm run dev`.

2. Configurar Tailwind CSS

- Instalar Tailwind CSS siguiendo la documentación oficial de Tailwind con Vite.
- Asegurarse de que la configuración esté correcta agregando estilos personalizados a algún elemento (por ejemplo, un título con `text-3xl text-blue-600 font-bold`).

3. Instalar y usar Axios

- Instalar Axios en el proyecto (`npm install axios`).

- Crear un componente principal (`App.jsx` o `Main.jsx`) que:
  - Utilizar `useEffect` y `useState` para realizar una solicitud GET a la API de DummyJSON.
  - Almacenar la respuesta en el estado.
  - Renderizar al menos el título y el precio de los productos en pantalla.

4. Estilos básicos con Tailwind

- Aplicar clases de Tailwind para mejorar la visualización de los productos.
- Por ejemplo: Mostrar cada producto en una tarjeta con borde, padding y separación (`border p-4 m-2`), y distribuir los productos en una grilla si es posible (`grid grid-cols-2 md:grid-cols-4`).

Entregables

- Proyecto completo en carpeta o repositorio.
- Archivo `README.md` con:
  - Instrucciones para ejecutar el proyecto.
  - Captura de pantalla del resultado en funcionamiento (si se entrega en repositorio).

Criterios de evaluación

Criterio	Puntaje
Vite instalado correctamente y proyecto ejecuta	2 pts
Tailwind configurado correctamente (se verifica en el estilo aplicado)	2 pts
Axios instalado y se realiza fetching desde `https://dummyjson.com`	3 pts
Uso de `useState` y `useEffect` para manejar el fetching	2 pts
Renderizado de datos recibido en componentes	2 pts
Uso básico de clases de Tailwind para diseño	2 pts

Total: 13 puntos posibles

Objetivo

Evaluar la capacidad del estudiante para:

- Obtener datos de una API utilizando Axios.
- Aplicar filtros dinámicos usando `useState` y `useEffect`.
- Mostrar estadísticas basadas en los datos obtenidos.
- Crear y utilizar componentes reutilizables en React.
- Implementar renderizado condicional y manejo de listas.

Actividades

A partir del código base proporcionado, realizar las siguientes modificaciones y agregados:

1. Agregar estadísticas adicionales

Incorporar **estadísticas nuevas** a la sección de estadísticas:

- El producto más caro (mostrar el nombre y el precio).
- El producto más barato (mostrar el nombre y el precio).
- La cantidad de productos cuyo título contiene más de 20 caracteres.
- El precio total de todos los productos filtrados.
- El promedio de descuento (`discountPercentage`) de los productos filtrados.

Proponer otras estadísticas, siempre que se basen en los productos filtrados y se calculen dinámicamente.

2. Crear dos componentes adicionales

Dividir el código en al menos **dos nuevos componentes funcionales**:

- **ProductList**: se encargará de mostrar la lista de productos.
- **StatsPanel**: se encargará de mostrar todas las estadísticas.

Ambos componentes deben recibir las props necesarias para funcionar correctamente. No deben contener lógica de estado interno.

3. Validar búsqueda vacía

Cuando el campo de búsqueda esté vacío, mostrar todos los productos sin errores. Asegurarse de que el código no falla ni en las estadísticas ni en el renderizado de la lista.

4. Diseño básico

Aplicar estilos básicos con CSS o Tailwind para mejorar la visualización. Ejemplos:

- Separar la sección de estadísticas con un fondo distinto.
- Mostrar los productos en tarjetas.
- Agregar una animación leve cuando se actualiza la lista o las estadísticas.

Entregables

- Código fuente completo del proyecto en una carpeta o repositorio.
- Breve explicación escrita (puede ser en un archivo README.md) de:
  - Qué estadísticas se agregaron.
  - Cómo se dividió el código en componentes.

Criterios de evaluación

Criterio	Puntaje
Axios se utiliza correctamente	2 pts
Filtro dinámico funciona correctamente	2 pts
Se agregaron al menos 3 estadísticas nuevas	3 pts
Uso correcto de props y componentes	3 pts
Código organizado y legible	2 pts
(Bonificación) Estilos o animaciones útiles	+1 pt

Total: 12 puntos posibles