Ejercicio Práctico: Desarrollo de Componentes Básicos con ReactJS para el Proyecto del Hospital

Contexto:

En este ejercicio práctico, los estudiantes comenzarán a desarrollar el **sistema del hospital** utilizando los **elementos fundamentales de ReactJS**. Implementarán componentes reutilizables para distintas secciones de la web del hospital, usando **JSX** para renderizar datos y manejar el flujo de información con **props**. También deberán utilizar **Hooks** y formularios para manejar la interacción del usuario.

Duración: **2 horas**

Requisitos:

1. **Creación de Componentes en ReactJS (1.5 puntos)**

- Crea los siguientes componentes reutilizables para la aplicación del hospital:
  - `DoctorCard`: Muestra la información de un doctor (nombre, especialidad, años de experiencia).
  - `ServiceList`: Lista los servicios médicos disponibles en el hospital.
  - `AppointmentForm`: Un formulario para que los usuarios agenden una cita con un doctor.
- Asegúrate de que los componentes sean modulares y puedan ser reutilizados en otras secciones.

2. **Uso de JSX para Renderización de Datos (1 punto)**

- Utiliza **JSX** para crear la estructura visual de los componentes. Asegúrate de:
  - Insertar expresiones JSX para mostrar los datos dinámicos de los doctores, servicios y citas.
  - Utilizar correctamente atributos en JSX y evitar errores comunes como el uso incorrecto de `className` en lugar de `class`.

3. **Flujo de Datos con Props (1 punto)**

- Implementa **props** para pasar datos entre los componentes:

- Pasa los datos de doctores al componente `DoctorCard` desde el componente principal.
- Pasa los datos de servicios al componente `ServiceList` desde el componente principal.
- Usa **props** para manejar los datos de entrada en el formulario de citas (`AppointmentForm`), como el nombre del paciente y el doctor seleccionado.

4. **Listas y Keys en React (1 punto)**

- Utiliza correctamente **listas y keys** en React para renderizar dinámicamente la lista de doctores y servicios.
    - Asegúrate de que cada ítem de la lista tenga una **key** única para optimizar el rendimiento de la aplicación.

5. **Formulario con Manejo de Estado (1 punto)**

- Crea el **formulario de citas** (`AppointmentForm`) y utiliza **Hooks** como `useState` para manejar los datos del formulario (nombre del paciente, especialidad del doctor, fecha de la cita).
    - Al enviar el formulario, muestra los datos ingresados en la consola del navegador o en la interfaz.

6. **Introducción a Hooks y Ciclo de Vida (1.5 puntos)**

- Utiliza **Hooks** para manejar el ciclo de vida de los componentes:
    - Usa `useEffect` para cargar la lista de doctores y servicios cuando el componente principal se monta en el DOM.
    - Usa `useState` para manejar el estado de las citas y los servicios seleccionados por el usuario.

Herramientas a Utilizar:

- **ReactJS** para la creación de los componentes reutilizables y la gestión del estado.
- **JSX** para la renderización dinámica de datos en los componentes.
- **React Developer Tools** para depurar y verificar la correcta creación de componentes y el paso de props.

Entrega:

- **Formato de entrega**:
    - Opción 1: Enviar un **enlace al repositorio de GitHub** con el proyecto ReactJS implementado, incluyendo los componentes creados y el archivo README.
    - Opción 2: Entregar un archivo **ZIP comprimido** con el proyecto React, incluyendo el código de los componentes y el archivo README.