Trabajo Práctico grupal de presentación obligatoria.

En esta parte 5 del trabajo práctico n° 3, se introducirá el concepto de estado global en React mediante el uso de **Context API**, esto permitirá centralizar la información que necesita ser compartida por múltiples componentes independientes de la aplicación, eliminando la necesidad de pasar propiedades de manera vertical (*Prop Drilling*).

1. Creación del Contexto de Usuario (UsuarioContext.jsx)

Deberán crear un archivo de contexto dentro de una nueva carpeta en src/**context**/UsuarioContext.jsx.

- **Estado Centralizado:** Este contexto debe almacenar un objeto usuario con los datos del perfil (nombre, dni, rol: "Docente" o "Alumno", e institución) y una función para actualizar dichos datos (por ejemplo, actualizarPerfil).

  La aplicación inicia con datos por defecto en el estado (es una simulación de un usuario que ya está logueado) y se les permite editarlos desde la vista de perfil para comprobar que el estado global se actualiza.

- **Proveedor (Provider):** Asegurar la envoltura de la aplicación en el archivo App.jsx utilizando el <UsuarioProvider> para que todo el árbol de componentes (Rutas, Header, Vistas) tenga acceso a esta información de manera directa.

2. Consumo del Estado Global en el Encabezado (<Header />)

El componente de la barra superior o encabezado de la aplicación deberá mutar para volverse dinámico.

- Utilizando el hook **useContext**, el <Header /> debe conectarse al estado global para extraer el nombre del usuario “logueado” y su rol, mostrándolos en una esquina de la barra superior.

3. Refactorización Dinámica de la Vista de Perfil (<PerfilUsuario />)

La pantalla de perfil dejará de contener datos fijos.

- El componente <PerfilUsuario /> debe conectarse al mismo contexto global para renderizar la información del usuario en pantalla a través de los componentes visuales del framework (Material UI/React Bootstrap).

- **Interactividad:** Se debe añadir un botón de "Editar Perfil" que habilite campos de texto en un formulario. Al presionar "Guardar Cambios", el componente llamará a la función del contexto global (**actualizarPerfil**), lo que impactará los datos inmediatamente. Comprobar que el nombre en el <Header /> cambia de forma sincronizada y automática en el mismo instante.

Apartado de Complejidad: Persistencia del Estado Global (LocalStorage)

El principal problema de los estados globales en memoria de React es que, al presionar **F5** (recargar página), los estados se reinician a sus valores por defecto.

Deberán lograr que las modificaciones del perfil del usuario sean persistentes frente a las recargas del navegador combinando el ciclo de vida y el almacenamiento local.

- Dentro del archivo del proveedor del contexto (**UsuarioProvider**), utilizar un hook useEffect que "escuche" los cambios del objeto usuario. Cada vez que el usuario se actualice, guarden una copia de este objeto en el localStorage del navegador convertido a cadena de texto (JSON.stringify). Al inicializar el estado del contexto, verificar primero si existe algo guardado en el almacenamiento local para usarlo como valor inicial.