Trabajo Práctico: Aplicación de React con Chat y Consumo de API

Objetivo:
El objetivo de este trabajo práctico es crear una aplicación sencilla en React que consuma información de una API, tenga un sistema de navegación con **React Router**, y un **chat comunitario** que almacene y recupere los mensajes en **Firebase**.

Consigna 1: Estructura de la Aplicación (Valor: 20 puntos)

Objetivo:
Crear una estructura básica de la aplicación con un **Home** y un **Chat**, utilizando **React Router** para la navegación.

Instrucción:
Crea una aplicación en React con las siguientes características:

1. **Página de inicio (Home)**:
    - La página debe mostrar un título que diga **"Bienvenidos a la App"** y un pequeño texto introductorio sobre la aplicación.
    - En esta página, debes consumir información de alguna API pública y mostrarla en la interfaz. Por ejemplo JSONPlaceHolder.
    - **React Router** debe ser utilizado para navegar desde el **Home** a la página de **Chat**.
2. **Menú de navegación**:
    - En la parte superior, crea un **menú de navegación** que permita a los usuarios ir del **Home** al **Chat** y viceversa.
    - Usa **React Router** para manejar las rutas.

Consigna 2: Implementación del Chat (Valor: 40 puntos)

Objetivo:
Crear un sistema de **chat comunitario** donde los mensajes se guardan en **Firebase**.

Instrucción:

1. **Componente Chat**:
    - Crea un **componente Chat** al que los usuarios puedan acceder desde el menú de navegación.
    - El chat debe permitir que los usuarios envíen mensajes de texto.
    - Los mensajes deben mostrarse en la pantalla en tiempo real a medida que se envían.
2. **Guardar los mensajes en Firebase**:
    - Los mensajes deben ser almacenados en Firebase Firestore.
    - Cada mensaje debe tener al menos dos campos: **texto** (el contenido del mensaje) y **nombre.**
3. **Mostrar los mensajes en el Chat**:

- Los mensajes deben aparecer en el orden en que se envían, y deben mostrarse en la interfaz con el nombre y el texto.

Consigna 3: Organización y Buenas Prácticas (Valor: 20 puntos)

Objetivo:
Asegurarse de que la aplicación esté bien organizada y estructurada, utilizando buenas prácticas.

Instrucción:

1. **Organización de Componentes**:
    - Organiza bien los componentes, separando el código en archivos y carpetas (por ejemplo, un archivo para el componente `Home`, otro para el `Chat`, y otro para la conexión a Firebase).

Requisitos Adicionales:

- Utiliza **Vite** para crear y correr la aplicación.
- Utiliza **React Router** para las rutas.
- Utiliza **Firebase Firestore** para almacenar y recuperar los mensajes en tiempo real.
- Los mensajes deben aparecer en tiempo real sin necesidad de recargar la página.

Criterios de Evaluación:

1. **Funcionalidad** (50 puntos):
    - ¿La aplicación cumple con los requisitos establecidos?
    - ¿El consumo de la API en la página de inicio es correcto?
    - ¿El chat funciona correctamente, con mensajes guardados en Firebase y actualizados en tiempo real?
2. **Organización del Código** (20 puntos):
    - ¿Está el código bien organizado y dividido en componentes reutilizables?
3. **Uso de Firebase** (20 puntos):
    - ¿La integración con Firebase es correcta?
    - ¿Los mensajes se guardan en Firestore y se actualizan en tiempo real?
4. **Interfaz y Diseño** (10 puntos):
    - ¿La aplicación tiene una interfaz clara y fácil de usar?
    - ¿El diseño es responsivo y funciona bien en dispositivos móviles?

Fecha de entrega: Lunes 11 de Noviembre.