Aprendizaje Basado en Proyectos: Seguridad en la Web del Hospital

Contexto:

En este proyecto, los estudiantes deberán aplicar medidas de seguridad avanzadas en la **web del hospital** desarrollada en ReactJS. Integrarán protección contra ataques comunes, asegurarán las rutas de la aplicación y protegerán el consumo de APIs utilizando **API Key** y **JWT**. Además, se implementará seguridad por roles y autenticación de usuarios para restringir el acceso a áreas específicas de la aplicación.

Duración: **2 horas**

Requisitos:

1. **Protección de Rutas con React Router DOM (1.5 puntos)**

- Implementa seguridad en las rutas del sistema del hospital, asegurando que solo los usuarios autenticados puedan acceder a ciertas secciones (como la gestión del equipo médico o los registros de pacientes).
    - Utiliza **React Router DOM** para gestionar las rutas protegidas.
    - Asegúrate de que las rutas públicas (como la página principal) sean accesibles sin autenticación.

2. **Implementación de Autenticación de Usuarios y Roles (1.5 puntos)**

- Integra un sistema básico de **autenticación de usuarios** que permita el login en la aplicación del hospital.
    - Los usuarios deben autenticarse para acceder a secciones protegidas.
    - Implementa roles (por ejemplo, **doctor** y **administrador**) para que ciertos usuarios solo tengan acceso a áreas específicas según su rol.

3. **Consumo de APIs Protegido con API Key y JWT (1.5 puntos)**

- Asegura el consumo de APIs utilizando una **API Key** y **JWT**. Los datos sensibles (como la información de pacientes o citas) deben ser accesibles solo si el usuario ha iniciado sesión y tiene un JWT válido.

- Implementa la verificación del **token JWT** en las solicitudes a la API.
- Muestra un mensaje de error si el token no es válido o ha expirado, y redirige al usuario a la página de inicio de sesión.

4. **Prevención de Vulnerabilidades Comunes (1.5 puntos)**

- Implementa medidas de seguridad en la web del hospital para prevenir ataques comunes como:
  - **Clickjacking**: Protege la aplicación para que no pueda ser incrustada en iframes no autorizados.
  - **XSS (Cross-Site Scripting)**: Escapa o limpia cualquier entrada del usuario que pueda inyectar código malicioso.
  - **SQL Injection**: Asegúrate de que las solicitudes a la API estén protegidas contra inyecciones de SQL.
  - **Ataque DoS**: Implementa mecanismos para mitigar posibles ataques de denegación de servicio.

5. **Encriptación de Datos en el Front-End (1 punto)**

- Utiliza técnicas de **encriptación de datos** para proteger la información sensible en el front-end, como las contraseñas de los usuarios o los datos personales de los pacientes.
  - Asegúrate de que los datos se encripten antes de ser enviados a la API.

Herramientas a Utilizar:

- **React Router DOM** para la gestión de rutas protegidas.
- **JWT (JSON Web Token)** para la autenticación y protección de APIs.
- **ReactJS** para la implementación de la lógica de seguridad.
- **Bibliotecas de seguridad** (como `bcrypt` para encriptar contraseñas o `Helmet` para proteger contra ataques XSS).

Entrega:

- **Formato de entrega**:
  - Opción 1: Enviar un **enlace al repositorio de GitHub** con el proyecto que incluya las medidas de seguridad implementadas.

- Opción 2: Entregar un archivo **ZIP comprimido** con el proyecto ReactJS completo, que incluya la autenticación, seguridad de las rutas, consumo de APIs protegido, y la prevención de vulnerabilidades.