Prueba técnica Moontech (Full stack)

La prueba técnica consiste en desarrollar una pequeña aplicación de autenticación que contenga un inicio de sesión (Login) y un formulario CRUD de usuarios.

REQUISITOS BACKEND

Parte backend, realizarla con tecnología Node (Express / Nestjs), esta debe contener los siguientes puntos:

- Conexión a gestor de bases de datos (MongoDB)

- Colección de documentos (Tabla) de *Usuarios*, un usuario contiene las siguientes propiedades:
    - Nombre → Texto alfanumérico
    - Contraseña → Texto alfanumérico
    - Email → Texto alfanumérico
    - Activo → Booleano

- Colección de documentos (Tabla) de *Log de conexiones*, un log de conexión contiene las siguientes propiedades:
    - Fecha
    - Usuario
    - Login → Booleano que indica si es conexión o desconexión

Funcionalidad:
- Crear/ Actualizar/ Eliminar *usuarios*
- Autenticación de usuarios existentes en la aplicación
- Creación de *log de conexión* cuando el usuario se identifica o se desconecta

Valorable:
- Conexión con websockets
- Uso de estrategías de seguridad (JWT)
- Populado de datos inicicales en la base de datos
- Tratamiento de variables de entorno con Node
- Encriptación de contraseñas
- El proyecto debe ser una API Rest consumida por el Frontend

REQUISITOS FRONTEND

Realizarla tecnología Angular / Javascript. Debe contener los siguientes puntos:

- Pantalla de inicio de sesión que permita a los usuarios autenticarse
- Pantalla de mantenimiento de usuarios (Formulario CRUD) donde se puedan visualizar, añadir, editar y eliminar usuarios
- Opcional → Pantalla de visualización de las conexiones de usuarios.

Funcionalidad:

- Permitir a un usuario acceder a la gestión de usuarios de la aplicación, mediante una pantalla de autenticación (LogIn)
- Una vez autentificado, el usuario podrá ver, añadir, actualizar y eliminar usuarios
- El usuario podrá visualizar el registro de conexiones de los usuarios.

Puntos valorables:

- Conexión vía websockets con la API
- Gestión y tratamiento de tokens (JWT)
- Uso de estados para gestionar las llamadas a la API
- Uso de interceptors
- Uso de route guards
- Formularios reactivos de Angular y tratamiento de errores
- Diseño responsive