INSTITUTO SUPERIOR SANTA ROSA DE CALAMUCHITA

Hackaton 2025

Introducción General para los Estudiantes

¡Bienvenidos al Desafío de la Semana: Hackaton Full-Stack JS!

Durante los próximos 7 días, trabajarán en equipos para llevar sus habilidades de desarrollo al siguiente nivel. Cada equipo recibirá por sorteo un desafío único que pondrá a prueba sus conocimientos en React, Node.js, Express y su capacidad para integrar tecnologías modernas.

El objetivo no es solo completar la tarea, sino también colaborar, aprender a resolver problemas bajo presión y crear un producto funcional y bien estructurado.

Además el desafío incluye la jornada final el día viernes en donde tienen que hacer la presentación del resultado y explicación del mismo, sin superar los 30 minutos.

Reglas Generales:

- **Stack tecnológico:** React.js para el Frontend, Node.js y Express.js para el Backend.
- **Entrega:** Al final de la semana, cada grupo deberá presentar un repositorio de GitHub con el código fuente para compartir con el resto de compañeros. Realizar una breve presentación de 30 minutos como máximo.
- **Colaboración:** Usen Git y GitHub para gestionar el trabajo en equipo. ¡La comunicación es clave!

¡Mucha suerte y a programar!

GRUPO 4: Sistema de Autenticación Híbrido: Clásico y Social

Contexto:
Para aumentar la tasa de registro y facilitar el acceso, una tienda de comercio electrónico quiere modernizar su sistema de autenticación. Desean permitir que los usuarios se registren y accedan no solo con el clásico correo y contraseña, sino también usando sus cuentas de redes sociales populares como Google o Facebook.

Objetivo Principal:
Implementar un sistema de autenticación completo que incluya registro y login local (email/contraseña) y login social (OAuth 2.0) con al menos un proveedor (Google, Facebook o Instagram).

Requisitos Funcionales:

1. **Autenticación Local:**
    - Crear páginas de Registro y Login en React.
    - El formulario de registro debe capturar nombre, email y contraseña.
    - La contraseña debe ser hasheada en el backend antes de guardarla en la base de datos.
    - El login debe validar las credenciales contra los datos guardados.
2. **Autenticación Social (OAuth 2.0):**
    - En la página de login/registro, añadir botones como "Iniciar sesión con Google".
    - Implementar el flujo de OAuth 2.0:
        - Al hacer clic, redirigir al usuario a la página de autorización del proveedor (ej: Google).
        - Tras autorizar, el proveedor redirigirá de vuelta a una URL de callback en tu backend.
        - El backend recibirá los datos del perfil del usuario, lo buscará en la base de datos y, si no existe, lo creará.
3. **Gestión de Sesión:**
    - Una vez que el usuario se autentica (por cualquier método), el backend debe generar un token (JWT - JSON Web Token).
    - Este token se envía al frontend, que debe almacenarlo de forma segura (ej: localStorage o cookies).
    - El token debe ser enviado en las cabeceras de las peticiones a rutas protegidas.
4. **Rutas Protegidas:**
    - Crear al menos una ruta en el frontend (ej: /perfil) que solo sea accesible para usuarios autenticados.
    - Crear el middleware correspondiente en el backend para verificar el token JWT en las peticiones a endpoints protegidos.

Requisitos Técnicos:

- Frontend: React, React Router, Axios/Fetch.
- Backend: Node.js, Express, bcryptjs para hashear contraseñas, jsonwebtoken para los tokens, passport con sus estrategias (passport-local, passport-google-oauth20, passport-facebook).

Criterios de Evaluación:

- Funcionalidad completa de ambos flujos de autenticación (local y social).
- Seguridad: correcto hasheo de contraseñas y gestión de tokens.
- Correcta implementación de rutas públicas y protegidas.
- Experiencia de usuario fluida durante el login/registro.

Puntos Extra (Bonus):

- Implementar una funcionalidad de "Cerrar Sesión" que invalide el token o lo elimine del cliente.
- Añadir un segundo proveedor de login social.