**API de autenticación de usuarios**

**Informe Técnico: API de Autenticación para Sistema de Registro de Alumnos y Vigilantes**

**1. Introducción** El presente informe describe el desarrollo de una API de autenticación basada en Flask y JWT (JSON Web Tokens), orientada al control de acceso para un sistema que gestiona la información de alumnos y vigilantes.

**2. Tecnologías Utilizadas**

* Python 3
* Flask
* Flask-JWT-Extended
* Flask-SQLAlchemy
* Flask-Migrate
* MySQL (base de datos)
* Dotenv (para variables de entorno)
* Werkzeug (para encriptación de contraseñas)

**3. Estructura del Proyecto** La aplicación se estructura en un único archivo principal que contiene:

* Configuración de la conexión a la base de datos.
* Definición de modelos (Vigilante y Alumno).
* Rutas para registro e inicio de sesión.
* Rutas protegidas que requieren autenticación JWT.

**4. Modelos de Base de Datos**

* **Vigilante**: incluye campos como id, dni, nombre y password (encriptado).
* **Alumno**: incluye id, dni, nombre, programa\_estudios, estado, observaciones y password (encriptado).

**5. Rutas Implementadas**

* **/register\_vigilante**: Registra un nuevo vigilante.
* **/register\_alumnos**: Registra un nuevo alumno.
* **/login\_vigilante**: Inicio de sesión para vigilantes.
* **/login\_alumnos**: Inicio de sesión para alumnos.
* **/login**: Inicio de sesión general que identifica el tipo de usuario.
* **/protected**: Ruta protegida con JWT.
* **/alumnos**: Devuelve una lista de todos los alumnos registrados (requiere JWT).
* **/vigilantes**: Devuelve una lista de todos los vigilantes registrados (requiere JWT).

**6. Seguridad**

* Las contraseñas se almacenan en la base de datos usando generate\_password\_hash() para mayor seguridad.
* El inicio de sesión valida con check\_password\_hash().
* Se utiliza JWT para proteger rutas y garantizar el acceso únicamente a usuarios autenticados.

**7. Observaciones**

* Se implementa un único endpoint /login que determina si el usuario es un alumno o vigilante y genera un token JWT con su rol.
* El código está preparado para un entorno de producción, pero se recomienda cambiar la clave JWT a una más segura y almacenarla en el archivo .env.

**8. Conclusión** Esta API permite registrar y autenticar usuarios de forma segura, con una estructura escalable. Se puede integrar fácilmente con una aplicación frontend para control de acceso en tiempo real.

**Autor:** [Gandy William Humiri Qusipe] **Fecha:** 04 de abril de 2025