**Proyecto: API REST - Plataforma Académica Interna**

**Objetivo**

Crear una **API RESTful segura** que permita a instituciones educativas (colegios, institutos, academias) gestionar de forma centralizada la información académica de estudiantes, profesores, asignaturas y calificaciones, incluyendo subida de materiales y reportes.

Tecnologías utilizadas

| **Componente** | **Tecnología** |
| --- | --- |
| Framework | Spring Boot |
| Seguridad | Spring Security + JWT |
| ORM / Persistencia | Spring Data JPA (Hibernate) |
| Base de Datos | MySQL |
| Utilidades | Lombok |

**Módulo de seguridad**

**Funcionalidades:**

* Registro e inicio de sesión con JWT
* Control de acceso basado en roles
* Expiración y renovación de tokens
* Middleware para proteger rutas

**Roles:**

* ADMIN: Control total
  + Gestiona usuarios (alumnos/profesores)
  + Crea cursos y asignaturas
  + Genera reportes
* PROFESOR: Cargar notas y materiales
  + Administra sus asignaturas
  + Registra notas
  + Sube materiales para sus clases
* ESTUDIANTE: Visualizar calificaciones y materiales
  + Consulta sus asignaturas
  + Visualiza notas y materiales de clase

**Entidades principales y relaciones**

Usuario: id, nombre, email, password, rol

Estudiante: (FK Usuario), código de matrícula, curso actual

Profesor: (FK Usuario), especialidad, asignaturas

Curso: id, nombre, año académico

Asignatura: id, nombre, profesor\_id (FK), curso\_id (FK)

Matricula: estudiante\_id (FK), curso\_id (FK), periodo\_id (FK)

Nota: estudiante\_id (FK), asignatura\_id (FK), periodo\_id (FK), valor, observaciones

Material: titulo, descripción, archivo\_url, asignatura\_id (FK), profesor\_id (FK)

PeriodoLectivo (Año o semestre académico): id, nombre, fecha\_inicio, fecha\_fin

**Algunas relaciones:**

* Un Usuario puede ser Estudiante o Profesor
* Un Curso tiene muchas Asignaturas
* Una Asignatura tiene un Profesor asignado

**Endpoints RESTful (resumen por módulo)**

**Autenticación (/api/auth)**

* POST /login: Autenticación con JWT
* POST /register: Registro de usuarios (admin/profesor/estudiante)
* GET /me: Ver datos del usuario autenticado

**Usuarios (/api/usuarios)**

* GET /: Listado de todos los usuarios
* GET /{id}: Detalle de un usuario
* PUT /{id}: Editar usuario
* DELETE /{id}: Eliminar usuario

**Profesores (/api/profesores)**

* POST /: Crear profesor
* GET /: Listado de profesores
* GET /{id}: Ver detalles
* GET /{id}/asignaturas: Ver asignaturas asignadas

**Estudiantes (/api/estudiantes)**

* POST /: Crear estudiante
* GET /: Listado
* GET /{id}: Ver detalles
* GET /{id}/notas: Ver notas por estudiante

**Asignaturas (/api/asignaturas)**

* POST /: Crear asignatura
* GET /: Listado
* GET /{id}: Detalle
* PUT /{id}: Editar
* DELETE /{id}: Eliminar

**Cursos (/api/cursos)**

* POST /: Crear curso
* GET /: Listar cursos
* GET /{id}: Ver detalles

**Periodos lectivos (/api/periodos)**

* POST /: Crear periodo
* GET /: Listar
* GET /{id}: Ver detalles

**Notas (/api/notas)**

* POST /: Registrar nota (por asignatura y estudiante)
* GET /asignatura/{id}: Ver notas por asignatura
* GET /estudiante/{id}: Ver notas por estudiante
* PUT /{id}: Editar nota
* DELETE /{id}: Eliminar nota

**Materiales (/api/materiales)**

* POST /: Subir material (requiere archivo y metadatos)
* GET /asignatura/{id}: Ver materiales por asignatura
* DELETE /{id}: Eliminar material

**Reportes (/api/reportes)**

* GET /notas-promedio: Promedio por curso y asignatura
* GET /historial-estudiante/{id}: Historial completo de notas
* GET /reporte-final/{cursoId}: Consolidado por curso

**Buenas prácticas**

* DTOs para entrada y salida de datos
* Validaciones con @Valid y BindingResult