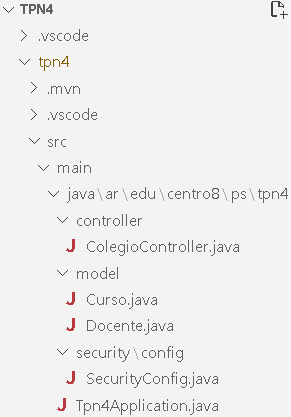
**Detalles del TP Nro. 4**



Curso.java

package ar.edu.centro8.ps.tpn4.model;

import lombok.AllArgsConstructor;

import lombok.Getter;

import lombok.NoArgsConstructor;

import lombok.Setter;

@NoArgsConstructor

@AllArgsConstructor

@Getter @Setter

public class Curso {

    private Long id;

    private String nombre;

    private String descripcion;

}

Docente.java

package ar.edu.centro8.ps.tpn4.model;

import lombok.AllArgsConstructor;

import lombok.Getter;

import lombok.NoArgsConstructor;

import lombok.Setter;

@NoArgsConstructor

@AllArgsConstructor

@Getter @Setter

public class Docente {

    private Long id;

    private String nombre;

    private String especialidad;

}

ColegioController.java

package ar.edu.centro8.ps.tpn4.controller;

import ar.edu.centro8.ps.tpn4.model.Curso;

import ar.edu.centro8.ps.tpn4.model.Docente;

import org.springframework.security.access.prepost.PreAuthorize;

import org.springframework.web.bind.annotation.\*;

import java.util.ArrayList;

import java.util.List;

@RestController

@PreAuthorize("denyAll()") // Niega acceso por defecto a todos los métodos

public class ColegioController {

    // Simulamos almacenamiento en memoria

    private List<Curso> cursos = new ArrayList<>();

    private List<Docente> docentes = new ArrayList<>();

    private Long cursoId = 1L;

    private Long docenteId = 1L;

    // Endpoint público, no requiere autenticación

    @GetMapping("/public")

    @PreAuthorize("permitAll()")

    public String bienvenida() {

        return "Bienvenido al Sistema de Gestión de Cursos del Colegio";

    }

    // Lista todos los cursos, requiere permiso READ\_CURSOS

    @GetMapping("/cursos")

    @PreAuthorize("hasAuthority('READ\_CURSOS')")

    public List<Curso> listarCursos() {

        return cursos;

    }

    // Crea un nuevo curso, requiere permiso WRITE\_CURSOS

    @PostMapping("/cursos")

    @PreAuthorize("hasAuthority('WRITE\_CURSOS')")

    public Curso crearCurso(@RequestBody Curso curso) {

        curso.setId(cursoId++);

        cursos.add(curso);

        return curso;

    }

//Aquí va el resto de los endpoints

}

SecurityConfig.java

@Bean

public UserDetailsService userDetailsService() {

    List<UserDetails> users = new ArrayList<>();

    // Administrador: acceso completo

    users.add(User.withUsername("admin")

           .password(passwordEncoder().encode("admin123"))

           .roles("ADMIN")

           .authorities("READ\_CURSOS", "WRITE\_CURSOS", "READ\_DOCENTES", "WRITE\_DOCENTES")

           .build());

//Aquí va el resto de los usuarios

return new InMemoryUserDetailsManager(users);

}