ESCUELA SUPERIOR FRANCISCANA ESPECIALIZADA / AGAPE

CARRERA
Técnico en Ingeniería de Desarrollo de Software

MÓDULO
Creación de Aplicaciones en Java Enterprise Edition.

TÍTULO
Construcción de los Controladores del Proyecto.

ESTUDIANTE(S)
Daniel Alberto Hernández Masin
Metzelder Norberto Marroquín Ruiz
Armando José Jiménez Ochoa
Crisia Valeria Reyes Romero
Alejandro Ernesto Juárez Argumedo

DOCENTE
Luis Roberto García Zaña

FECHA DE PRESENTACIÓN
22/08/2025

CICLO
02

Capturas

1. Modelos

- Categoría

Categoria.java

```java
package org.esfe.modelos;

import jakarta.persistence.*;
import jakarta.validation.constraints.NotBlank;

@Entity  14 usages  AlejandroJuarez002
@Table(name="categorias")
public class Categoria {
    @Id  2 usages
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;

    @NotBlank(message="El nombre del Categoria es requerido")  2 usages
    private String nombre;

    public String getNombre() { return nombre; }

    public void setNombre(String nombre) { this.nombre = nombre; }

    public Integer getId() { return id; }

    public void setId(Integer id) { this.id = id; }
}
```

- Departamento

Departamento.java

```java
package org.esfe.modelos;

import jakarta.persistence.*;
import jakarta.validation.constraints.NotBlank;

@Entity 14 usages AlejandroJuarez002
@Table(name="departamentos")
public class Departamento {
    @Id 2 usages
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;

    @NotBlank(message="El nombre del Departamento es requerido") 2 usages
    private String nombre;

    public Integer getId() { return id; }

    public void setId(Integer id) { this.id = id; }

    public String getNombre() { return nombre; }

    public void setNombre(String nombre) { this.nombre = nombre; }
}
```

- DestinoTuristico

```java
@Entity  25 usages  DanielMasin
@Table(name = "destinoTuristicos")
public class DestinoTuristico {
    @Id  2 usages
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;

    @NotBlank(message = "El nombre es requerido")  2 usages
    private String nombre;

    @NotBlank (message = "La descripcion es requerida")  2 usages
    private String descripcion;

    @NotBlank (message = "La ubicacion es requerida")  2 usages
    private String ubicacion;

    @NotBlank (message = "El horario es requerido")  2 usages
    private String horario;

    // Relación con Imagen (foránea en Imagen)
    @OneToMany(mappedBy = "destinoTuristico", cascade = CascadeType.ALL, orphanRemoval = true)  2 usages
    private List<Imagen> imagenes;

    public Integer getId() { return id; }

    public void setId(Integer id) { this.id = id; }

    public String getNombre() { return nombre; }

    public void setNombre(String nombre) { this.nombre = nombre; }

    public String getDescripcion() { return descripcion; }

    public void setDescripcion(String descripcion) { this.descripcion = descripcion; }

    public String getUbicacion() { return ubicacion; }

    public void setUbicacion(String ubicacion) { this.ubicacion = ubicacion; }

    public String getHorario() { return horario; }

    public void setHorario(String horario) { this.horario = horario; }
    public List<Imagen> getImagenes() { return imagenes; }

    public void setImagenes(List<Imagen> imagenes) { this.imagenes = imagenes; }
}
```

- Rol

```java
@Entity  19 usages  AlejandroJuarez002
@Table(name="roles")
public class Rol {
    @Id  2 usages
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;

    @NotBlank(message = "El nombre del rol es requerido")  2 usages
    private String nombre;

    @OneToMany(mappedBy = "rol", fetch = FetchType.LAZY)  2 usages
    private List<Usuario> usuarios; // Muchos usuarios pueden tener este rol

    public Integer getId() { return id; }

    public void setId(Integer id) { this.id = id; }

    public String getNombre() { return nombre; }

    public void setNombre(String nombre) { this.nombre = nombre; }

    public List<Usuario> getUsuarios() { return usuarios; }

    public void setUsuarios(List<Usuario> usuarios) { this.usuarios = usuarios; }
}
```

- Usuario

```java
@Entity  31 usages  AlejandroJuarez002
@Table(name="usuarios")
public class Usuario {
    @Id  2 usages
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;

    @NotBlank(message = "Su nombre es requerido")  2 usages
    private String nombre;

    @NotBlank(message = "Su apellido es requerido")  2 usages
    private String apellido;

    @NotBlank(message = "El nombre de usuario de requerido")  2 usages
    private String nombreUsuario;

    @NotBlank(message = "La contraseña es requerida")  2 usages
    private String clave;


    private int status;  2 usages

    @ManyToOne(fetch = FetchType.EAGER)  2 usages
    @JoinColumn(name = "rol_id")
    private Rol rol;
```

```java
    //private int status;

    public Integer getId() { return id; }

    public void setId(Integer id) { this.id = id; }

    public String getNombre() { return nombre; }

    public void setNombre(String nombre) { this.nombre = nombre; }

    public String getApellido() { return apellido; }

    public void setApellido(String apellido) { this.apellido = apellido; }

    public String getNombreUsuario() { return nombreUsuario; }

    public void setNombreUsuario(String nombreUsuario) { this.nombreUsuario = nombreUsuario; }

    public String getClave() { return clave; }

    public void setClave(String clave) { this.clave = clave; }
}
```

2. Repositorios

- CategoriaRepository

ICategoriaRepository.java

```java
package org.esfe.repositorios;

import org.esfe.modelos.Categoria;
import org.springframework.data.jpa.repository.JpaRepository;

public interface ICategoriaRepository extends JpaRepository<Categoria, Integer> {  2 usages  AlejandroJuarez002
}
```

- DepartamentoRepository

IDepartamentoRepository.java

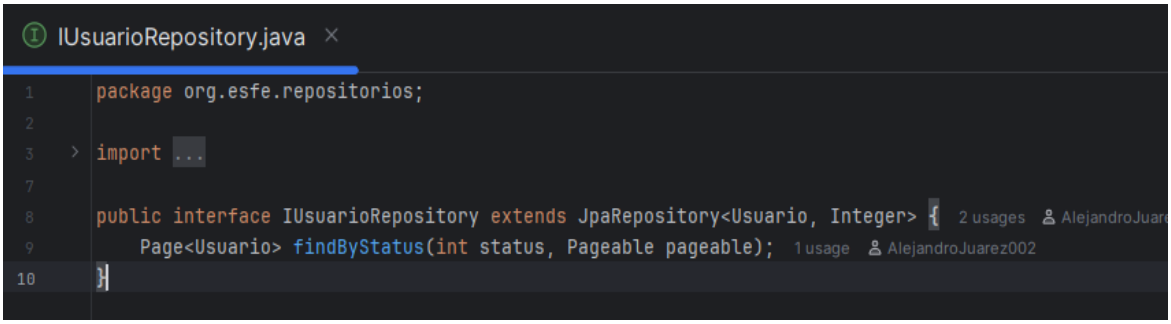
```java
package org.esfe.repositorios;

import ...

public interface IDepartamentoRepository extends JpaRepository<Departamento, Integer> {  2 usages  AlejandroJuarez002
}
```

- DestinoTuristicoRepository

```java
public interface IDestinoTuristicoRepository extends JpaRepository<DestinoTuristico, Integer> {  4 usages  DanielMasin

    // Buscar por coincidencia parcial en el nombre (ignora mayúsculas/minúsculas)
    List<DestinoTuristico> findByNombreContainingIgnoreCase(String nombre);  2 usages  DanielMasin

    // Opcional: si también quieres buscar en descripción o ubicación
    List<DestinoTuristico> findByNombreContainingIgnoreCaseOrDescripcionContainingIgnoreCaseOrUbicacionContainingIgnoreCase(  no usag
            String nombre, String descripcion, String ubicacion
    );
}
```

- RolRepository

- UsuarioRepository



3. Servicios

- Implementaciones

- CategoriaService

CategoriaService.java

```java
package org.esfe.servicios.implementaciones;

import org.esfe.modelos.Categoria;
import org.esfe.repositorios.ICategoriaRepository;
import org.esfe.servicios.interfaces.ICategoriaService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import org.springframework.stereotype.Service;

import java.util.List;
import java.util.Optional;

/**
 * Se implementan los metodos de ICategoriaService
 */
@Service  no usages  AlejandroJuarez002
public class CategoriaService implements ICategoriaService {
    @Autowired  5 usages
    private ICategoriaRepository categoriaRepository;

    /**
     * Devuelve registros de manera paginada
     * @param pageable
     * @return
     */
    @Override  no usages  AlejandroJuarez002
    public Page<Categoria> buscarTodosPaginados(Pageable pageable) { return categoriaRepository.findAll(pageable); }

    /**
     * Devuelve una lista
     * @return
     */
    @Override  no usages  AlejandroJuarez002
    public List<Categoria> obtenerTodos() { return categoriaRepository.findAll(); }

    @Override  no usages  AlejandroJuarez002
    public Optional<Categoria> buscarPorId(Integer id) { return categoriaRepository.findById(id); }

    /**
     * Devuelve metodo save
     * @param categoria
     * @return
     */
    @Override  no usages  AlejandroJuarez002
    public Categoria crearOEditar(Categoria categoria) { return categoriaRepository.save(categoria); }

    @Override  no usages  AlejandroJuarez002
    public void eliminarPorId(Integer id) { categoriaRepository.deleteById(id); }
}
```

- DepartamentoService

DepartamentoService.java

```java
package org.esfe.servicios.implementaciones;

import org.esfe.modelos.Departamento;
import org.esfe.repositorios.IDepartamentoRepository;
import org.esfe.servicios.interfaces.IDepartamentoService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import org.springframework.stereotype.Service;

import java.util.List;
import java.util.Optional;

/**
 * Se implementan los metodos de IDepartamentoService
 */
@Service  no usages  AlejandroJuarez002
public class DepartamentoService implements IDepartamentoService {
    @Autowired  5 usages
    private IDepartamentoRepository departamentoRepository;

    /**
     * Devuelve registros de manera paginada
     * @param pageable
     * @return
     */
    @Override  no usages  AlejandroJuarez002
    public Page<Departamento> buscarTodosPaginados(Pageable pageable) {
        return departamentoRepository.findAll(pageable);
    }

    /**
     * Devuelve una lista
     * @return
     */
    @Override  no usages  AlejandroJuarez002
    public List<Departamento> obtenerTodos() { return departamentoRepository.findAll(); }

    @Override  no usages  AlejandroJuarez002
    public Optional<Departamento> buscarPorId(Integer id) { return departamentoRepository.findById(id); }

    /**
     * Devuelve metodo save
     * @param departamento
     * @return
     */
    @Override  no usages  AlejandroJuarez002
    public Departamento crearOEditar(Departamento departamento) { return departamentoRepository.save(departamento); }

    @Override  no usages  AlejandroJuarez002
    public void eliminarPorId(Integer id) { departamentoRepository.deleteById(id); }
}
```

- DestinoTuristicoService

DestinoTuristicoService.java

```java
package org.esfe.servicios.implementaciones;

import ...

@Service  no usages  DanielMasin
public class DestinoTuristicoService implements IDestinoTuristicoService {
    @Autowired  4 usages
    private IDestinoTuristicoRepository destinoTuristicoRepository;

    @Override  no usages  DanielMasin
    public Page<DestinoTuristico> buscarTodosPaginados(Pageable pageable) {
        return destinoTuristicoRepository.findAll(pageable);
    }

    @Override  no usages  DanielMasin
    public List<DestinoTuristico> ObtenerTodo() { return destinoTuristicoRepository.findAll (); }

    @Override  no usages  DanielMasin
    public DestinoTuristico createOEdit(DestinoTuristico destinoTuristico) {
        return destinoTuristicoRepository.save (destinoTuristico);
    }

    @Override  DanielMasin
    public void eliminarPorId(Integer id) { destinoTuristicoRepository.deleteById (id); }
}
```

- RolService

RolService.java

```java
package org.esfe.servicios.implementaciones;

import ...

@Service  no usages  AlejandroJuarez002
public class RolService implements IRolService {
    @Autowired  4 usages
    private IRolRepository rolRepository;

    @Override  no usages  AlejandroJuarez002
    public Page<Rol> buscarTodosPaginados(Pageable pageable) { return rolRepository.findAll(pageable); }

    @Override  no usages  AlejandroJuarez002
    public List<Rol> obtenerTodos() { return rolRepository.findAll(); }

    @Override  no usages  AlejandroJuarez002
    public Rol crearOEditar(Rol rol) { return rolRepository.save(rol); }

    @Override  AlejandroJuarez002
    public void eliminarPorId(Integer id) { rolRepository.deleteById(id); }
}
```

- UsuarioService

UsuarioService.java

```java
package org.esfe.servicios.implementaciones;  2

import ...

@Service  no usages  AlejandroJuarez002
public class UsuarioService implements IUsuarioService {
    @Autowired  5 usages
    private IUsuarioRepository usuarioRepository;

    @Override  no usages  AlejandroJuarez002
    public Page<Usuario> buscarTodosPaginados(Pageable pageable) { return usuarioRepository.findAll(pageable); }

    @Override  no usages  AlejandroJuarez002
    public List<Usuario> obtenerTodos() { return usuarioRepository.findAll(); }

    @Override  no usages  AlejandroJuarez002
    public Optional<Usuario> buscarPorId(Integer id) { return usuarioRepository.findById(id); }

    @Override  no usages  AlejandroJuarez002
    public Usuario crearOEditar(Usuario usuario) { return usuarioRepository.save(usuario); }

    @Override  AlejandroJuarez002
    public void eliminarPorId(Integer id) { usuarioRepository.deleteById(id); }
}
```

- Interfaces

- ICategoriaService

ICategoriaService.java

```java
package org.esfe.servicios.interfaces;

import org.esfe.modelos.Categoria;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;

import java.util.List;
import java.util.Optional;

public interface ICategoriaService {  2 usages  1 implementation  AlejandroJuarez002
    Page<Categoria> buscarTodosPaginados(Pageable pageable);  no usages  1 implementation  AlejandroJuarez002

    List<Categoria> obtenerTodos();  no usages  1 implementation  AlejandroJuarez002

    Optional<Categoria> buscarPorId(Integer id);  no usages  1 implementation  AlejandroJuarez002

    Categoria crearOEditar(Categoria categoria);  no usages  1 implementation  AlejandroJuarez002

    void eliminarPorId(Integer id);  no usages  1 implementation  AlejandroJuarez002
}
```

- IDepartamentoService

IDepartamentoService.java

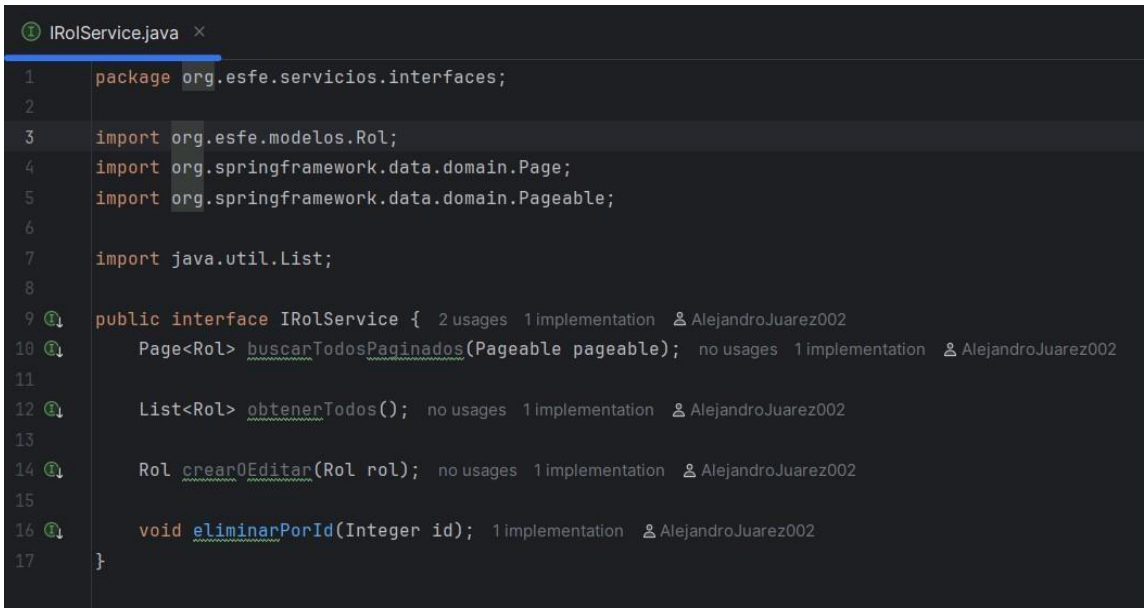
```java
package org.esfe.servicios.interfaces;

import org.esfe.modelos.Departamento;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;

import java.util.List;
import java.util.Optional;

public interface IDepartamentoService {  2 usages  1 implementation  AlejandroJuarez002
    Page<Departamento> buscarTodosPaginados(Pageable pageable);  no usages  1 implementation  AlejandroJuarez002

    List<Departamento> obtenerTodos();  no usages  1 implementation  AlejandroJuarez002

    Optional<Departamento> buscarPorId(Integer id);  no usages  1 implementation  AlejandroJuarez002

    Departamento crearOEditar(Departamento departamento);  no usages  1 implementation  AlejandroJuarez002

    void eliminarPorId(Integer id);  no usages  1 implementation  AlejandroJuarez002
}
```

- IDestinoTuristicoService

IDestinoTuristicoService.java

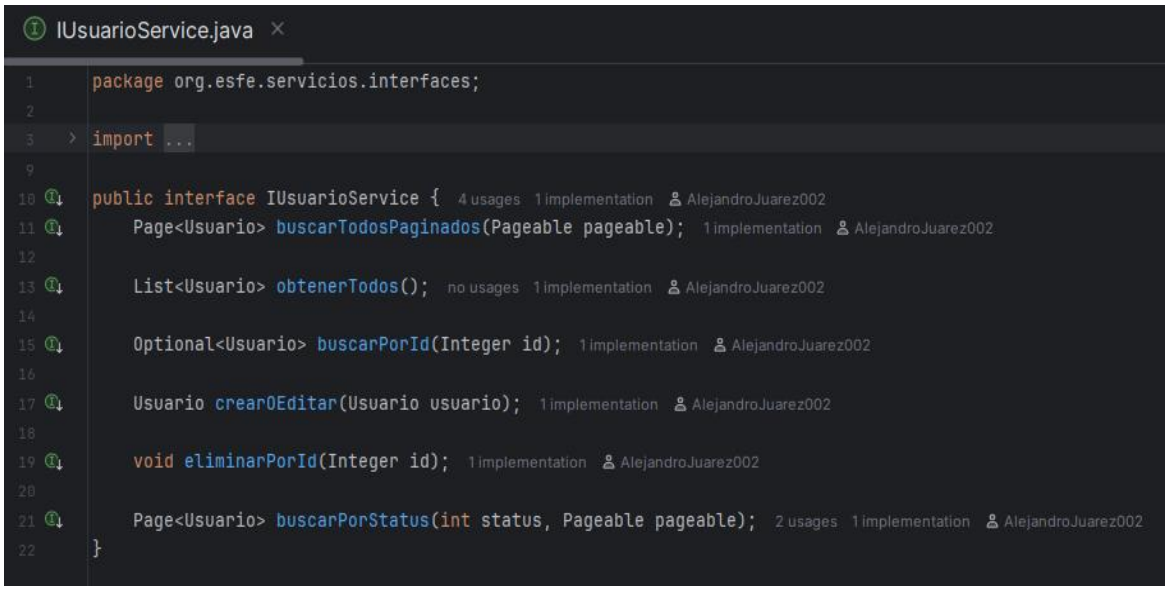
```java
package org.esfe.servicios.interfaces;


import org.esfe.modelos.DestinoTuristico;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;

import java.util.List;

public interface IDestinoTuristicoService {  2 usages  1 implementation  DanielMasin

    Page<DestinoTuristico> buscarTodosPaginados(Pageable pageable);  no usages  1 implementation  DanielMasin

    List<DestinoTuristico> ObtenerTodo();  no usages  1 implementation  DanielMasin

    DestinoTuristico createOEdit (DestinoTuristico destinoTuristico);  no usages  1 implementation  DanielMasin

    void eliminarPorId(Integer destinoTuristico);  1 implementation  DanielMasin
}
```

- IRolService



```java
package org.esfe.servicios.interfaces;

import org.esfe.modelos.Rol;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;

import java.util.List;

public interface IRolService {  2 usages  1 implementation  AlejandroJuarez002
    Page<Rol> buscarTodosPaginados(Pageable pageable);  no usages  1 implementation  AlejandroJuarez002

    List<Rol> obtenerTodos();  no usages  1 implementation  AlejandroJuarez002

    Rol crearOEditar(Rol rol);  no usages  1 implementation  AlejandroJuarez002

    void eliminarPorId(Integer id);  1 implementation  AlejandroJuarez002
}
```

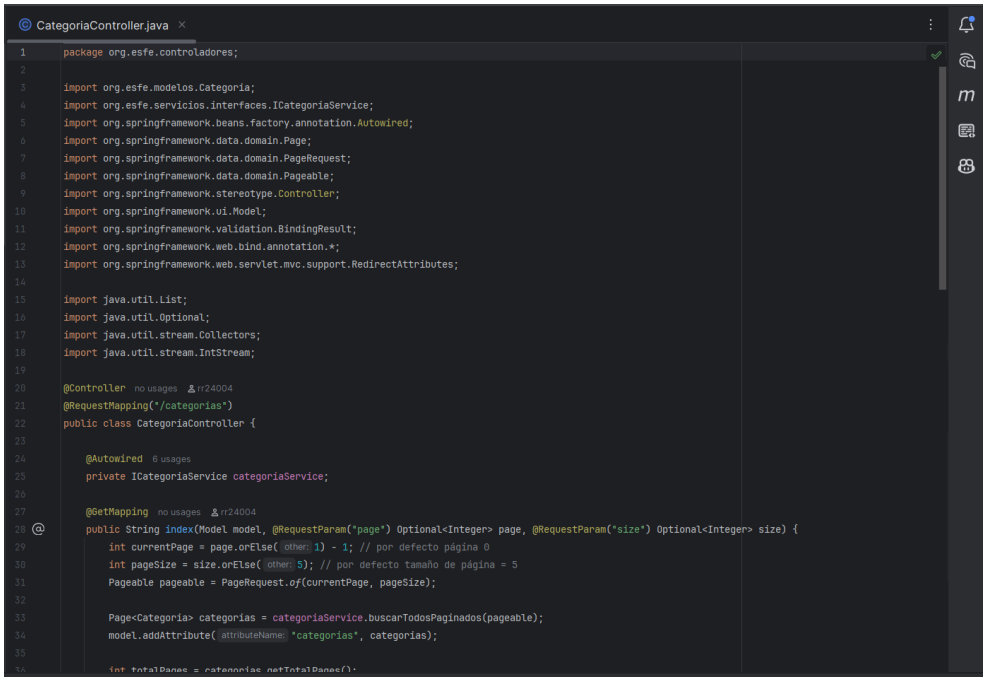
- IUsuarioService

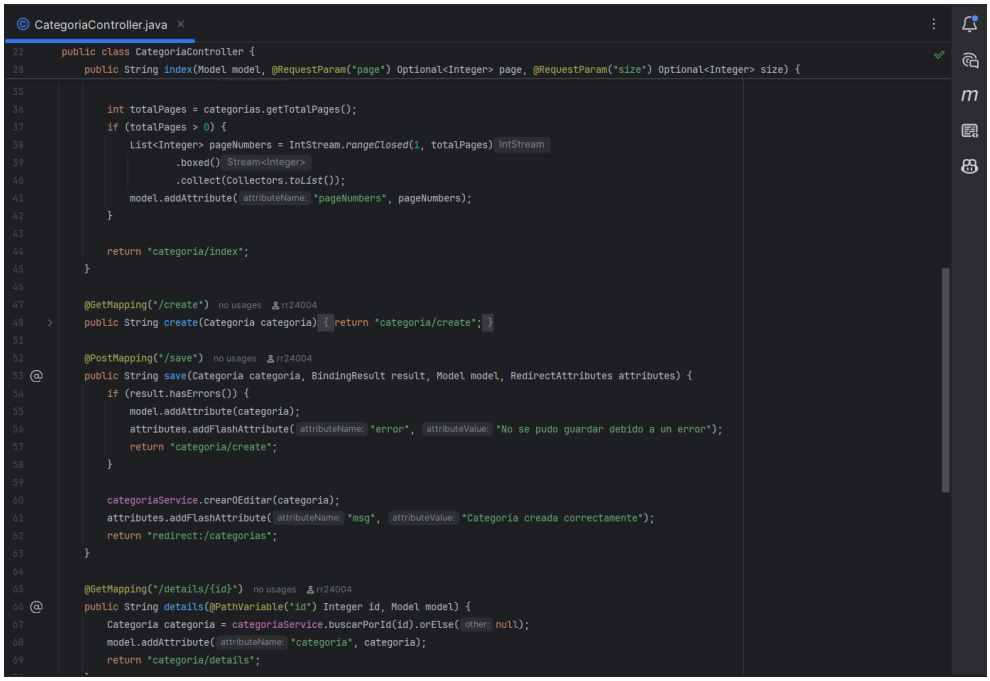


```java
package org.esfe.servicios.interfaces;

import ...

public interface IUsuarioService {  4 usages  1 implementation  AlejandroJuarez002
    Page<Usuario> buscarTodosPaginados(Pageable pageable);  1 implementation  AlejandroJuarez002

    List<Usuario> obtenerTodos();  no usages  1 implementation  AlejandroJuarez002

    Optional<Usuario> buscarPorId(Integer id);  1 implementation  AlejandroJuarez002

    Usuario crearOEditar(Usuario usuario);  1 implementation  AlejandroJuarez002

    void eliminarPorId(Integer id);  1 implementation  AlejandroJuarez002

    Page<Usuario> buscarPorStatus(int status, Pageable pageable);  2 usages  1 implementation  AlejandroJuarez002
}
```

➢ Controladores

- CategoriaController





```java
public class CategoriaController {

    @GetMapping("/details/{id}") no usages rr24004
    public String details(@PathVariable("id") Integer id, Model model) {
        Categoria categoria = categoriaService.buscarPorId(id).orElse( other: null);
        model.addAttribute( attributeName: "categoria", categoria);
        return "categoria/details";
    }

    @GetMapping("/edit/{id}") no usages rr24004
    public String edit(@PathVariable("id") Integer id, Model model) {
        Categoria categoria = categoriaService.buscarPorId(id).orElse( other: null);
        model.addAttribute( attributeName: "categoria", categoria);
        return "categoria/edit";
    }

    @GetMapping("/remove/{id}") no usages rr24004
    public String remove(@PathVariable("id") Integer id, Model model) {
        Categoria categoria = categoriaService.buscarPorId(id).orElse( other: null);
        model.addAttribute( attributeName: "categoria", categoria);
        return "categoria/delete";
    }

    @PostMapping("/delete") no usages rr24004
    public String delete(Categoria categoria, RedirectAttributes attributes) {
        categoriaService.eliminarPorId(categoria.getId());
        attributes.addFlashAttribute( attributeName: "msg", attributeValue: "Categoría eliminada correctamente");
        return "redirect:/categorias";
    }
}
```

- DepartamentoController

```java
package org.esfe.controladores;

import org.esfe.modelos.Departamento;
import org.esfe.servicios.interfaces.IDepartamentoService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Pageable;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;


import java.util.List;
import java.util.Optional;
import java.util.stream.Collectors;
import java.util.stream.IntStream;

@Controller no usages rr24004
@RequestMapping("/departamentos")
public class DepartamentoController {
    @Autowired 6 usages
    private IDepartamentoService departamentoService;

    @GetMapping no usages rr24004
    public String index(Model model, @RequestParam("page") Optional<Integer> page, @RequestParam("size") Optional<Integer> si
        int currentPage = page.orElse( other: 1) - 1; //si no esta seteado se asigna 0
        int pageSize = size.orElse( other: 5); //tamano de la pagina se asigna 5
        Pageable pageable = PageRequest.of(currentPage, pageSize);
```

CategoriaController.java | DepartamentoController.java

```java
public class DepartamentoController {
    public String index(Model model, @RequestParam("page") Optional<Integer> page, @RequestParam("size") Optional<Integer> si

        Page<Departamento> departamentos = departamentoService.buscarTodosPaginados(pageable);
        model.addAttribute( attributeName: "departamentos", departamentos);

        int totalPages = departamentos.getTotalPages();
        if (totalPages > 0) {
            List<Integer> pageNumbers = IntStream.rangeClosed(1, totalPages) IntStream
                    .boxed() Stream<Integer>
                    .collect(Collectors.toList());
            model.addAttribute( attributeName: "pageNumbers", pageNumbers);
        }

        return "departamento/index";
    }

    @GetMapping("/create") no usages rr24004
    public String create(Departamento departamento) { return "departamento/create"; }

    @PostMapping("/save") no usages rr24004
    public String save(Departamento departamento, BindingResult result, Model model, RedirectAttributes attributes){
        if (result.hasErrors()){
            model.addAttribute(departamento);
            attributes.addFlashAttribute( attributeName: "error", attributeValue: "No se pudo guardar debido a un error");
            return "departamento/create";
        }

        departamentoService.crearOEditar(departamento);
        attributes.addFlashAttribute( attributeName: "msg", attributeValue: "Grupo creado correctamente");
        return "redirect:/departamentos";
    }
```

CategoriaController.java | DepartamentoController.java

```java
public class DepartamentoController {
    }

    @GetMapping("/details/{id}") no usages rr24004
    public String details(@PathVariable("id") Integer id, Model model){
        Departamento departamento = departamentoService.buscarPorId(id).get();
        model.addAttribute( attributeName: "departamento", departamento);
        return "departamento/details";
    }

    @GetMapping("/edit/{id}") no usages rr24004
    public String edit(@PathVariable("id") Integer id, Model model){
        Departamento departamento = departamentoService.buscarPorId(id).get();
        model.addAttribute( attributeName: "departamento", departamento);
        return "departamento/edit";
    }


    @GetMapping("/remove/{id}") no usages rr24004
    public String remove(@PathVariable("id") Integer id, Model model) {
        Departamento departamento = departamentoService.buscarPorId(id).get();
        model.addAttribute( attributeName: "departamento", departamento);
        return "departamento/delete";
    }

    @PostMapping("/delete") no usages rr24004
    public String delete(Departamento departamento, RedirectAttributes attributes) {
        departamentoService.eliminarPorId(departamento.getId());
        attributes.addFlashAttribute( attributeName: "msg", attributeValue: "Departamento eliminado correctamente");
        return "redirect:/departamentos";
    }
}
```

- DestinoTuristicoController

DestinoTuristicoController.java

```java
package org.esfe.controladores;


import org.esfe.modelos.DestinoTuristico;
import org.esfe.modelos.Imagen;
import org.esfe.modelos.ImagenDTO;
import org.esfe.repositorios.IDestinoTuristicoRepository;
import org.esfe.servicios.interfaces.IDestinoTuristicoService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.Banner;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageImpl;
import org.springframework.data.domain.PageRequest;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.*;
import org.springframework.data.domain.Pageable;
import org.springframework.web.multipart.MultipartFile;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import java.io.IOException;
import java.util.*;
import java.util.stream.Collectors;
import java.util.stream.IntStream;

@Controller  no usages  DanielMasin
@RequestMapping("/destinoTuristicos")
public class DestinoTuristicoController {

    @Autowired  6 usages
    private IDestinoTuristicoService destinoTuristicoService;

    @Autowired  2 usages
    private IDestinoTuristicoRepository destinoTuristicoRepository;
```

DestinoTuristicoController.java

```java
public class DestinoTuristicoController {
    private IDestinoTuristicoRepository destinoTuristicoRepository;

    @GetMapping  no usages  DanielMasin
    public String index(Model model,
                        @RequestParam("page") Optional<Integer> page,
                        @RequestParam("size") Optional<Integer> size,
                        @RequestParam(value = "q", required = false) String q) {

        int currentPage = page.orElse( other: 1) - 1;
        int pageSize = size.orElse( other: 5);
        Pageable pageable = PageRequest.of(currentPage, pageSize);

        Page<DestinoTuristico> destinos;

        if (q != null && !q.isEmpty()) {
            // Si hay búsqueda, no paginar (opcional: se puede implementar paginación también)
            List<DestinoTuristico> resultados = destinoTuristicoRepository.findByNombreContainingIgnoreCase(q);
            destinos = new PageImpl<> (resultados, pageable, resultados.size());
        } else {
            destinos = destinoTuristicoService.buscarTodosPaginados(pageable);
        }

        // Convertir imágenes a Baseó4
        Page<Map<String, Object>> destinosDTO = destinos.map( DestinoTuristico dest -> {
            List<ImagenDTO> imagenesDTO = dest.getImagenes().stream() Stream<Imagen>
                    .map( Imagen img -> new ImagenDTO(
                            img.getId(),
                            img.getBytesArrayImage() != null ?
                                    Baseó4.getEncoder().encodeToString(img.getBytesArrayImage()) : null)) Stream<ImagenDTO>
                    .collect(Collectors.toList());

            Map<String, Object> dto = new HashMap<>();
            dto.put("destino", dest);
            dto.put("imagenes", imagenesDTO);
            return dto;
```

DestinoTuristicoController.java

```java
public class DestinoTuristicoController {
    private IDestinoTuristicoRepository destinoTuristicoRepository;

    @GetMapping  no usages  DanielMasin
    public String index(Model model,
                        @RequestParam("page") Optional<Integer> page,
                        @RequestParam("size") Optional<Integer> size,
                        @RequestParam(value = "q", required = false) String q) {

        int currentPage = page.orElse( other: 1) - 1;
        int pageSize = size.orElse( other: 5);
        Pageable pageable = PageRequest.of(currentPage, pageSize);

        Page<DestinoTuristico> destinos;

        if (q != null && !q.isEmpty()) {
            // Si hay búsqueda, no paginar (opcional: se puede implementar paginación también)
            List<DestinoTuristico> resultados = destinoTuristicoRepository.findByNombreContainingIgnoreCase(q);
            destinos = new PageImpl<> (resultados, pageable, resultados.size());
        } else {
            destinos = destinoTuristicoService.buscarTodosPaginados(pageable);
        }

        // Convertir imágenes a Base64
        Page<Map<String, Object>> destinosDTO = destinos.map( DestinoTuristico dest -> {
            List<ImagenDTO> imagenesDTO = dest.getImagenes().stream() Stream<Imagen>
                    .map( Imagen img -> new ImagenDTO(
                            img.getId(),
                            img.getBytesArrayImage() != null ?
                                    Base64.getEncoder().encodeToString(img.getBytesArrayImage()) : null)) Stream<ImagenDTO>
                    .collect(Collectors.toList());

            Map<String, Object> dto = new HashMap<>();
            dto.put("destino", dest);
            dto.put("imagenes", imagenesDTO);
            return dto;
```

DestinoTuristicoController.java

```java
public class DestinoTuristicoController {
    public String save(@ModelAttribute DestinoTuristico destinoTuristico,

        try {
            List<Imagen> listaImagenes = new ArrayList<>();
            for (MultipartFile file : imagenFiles) {
                if (!file.isEmpty()) {
                    Imagen img = new Imagen();
                    img.setBytesArrayImage(file.getBytes());
                    img.setDestinoTuristico(destinoTuristico);
                    listaImagenes.add(img);
                }
            }
            destinoTuristico.setImagenes(listaImagenes);

        } catch (IOException e) {
            attributes.addFlashAttribute( attributeName: "error",  attributeValue: "Error al procesar las imágenes.");
            return "destinoTuristico/create";
        }

        destinoTuristicoService.createOEdit(destinoTuristico);
        attributes.addFlashAttribute( attributeName: "msg",  attributeValue: "Destino turístico creado correctamente.");
        return "redirect:/destinoTuristicos";
    }

    @GetMapping("/details/{Id}")  no usages  DanielMasin
    public String details(@PathVariable("Id") Integer Id, Model model){
        DestinoTuristico destinoTuristico = destinoTuristicoService.buscarPorId (Id).get();
        model.addAttribute ( attributeName: "destinoTuristico", destinoTuristico);
        return "destinoTuristico/details";
    }

    @GetMapping("/edit/{Id}")  no usages  DanielMasin
    public String edit(@PathVariable("Id") Integer Id, Model model){
        DestinoTuristico destinoTuristico = destinoTuristicoService.buscarPorId (Id).get();
```

DestinoTuristicoController.java

```java
public class DestinoTuristicoController {

    @GetMapping("/edit/{Id}") no usages DanielMasin
    public String edit(@PathVariable("Id") Integer Id, Model model){
        DestinoTuristico destinoTuristico = destinoTuristicoService.buscarPorId (Id).get();
        model.addAttribute ( attributeName: "destinoTuristico", destinoTuristico);
        return "destinoTuristico/edit";
    }

    @GetMapping("/remove/{Id}") no usages DanielMasin
    public  String remove(@PathVariable("Id") Integer Id, Model model){
        DestinoTuristico destino = destinoTuristicoService.buscarPorId(Id).get();
        model.addAttribute( attributeName: "destino", destino);
        return "destinoTuristico/delete";
    }

    @PostMapping("/delete") no usages DanielMasin
    public String delete(DestinoTuristico destino, RedirectAttributes attributes){
        destinoTuristicoService.eliminarPorId(destino.getId());
        attributes.addFlashAttribute( attributeName: "msg", attributeValue: "Destino eliminado correctamente");
        return "redirect:/destinoTuristicos";
    }

    @GetMapping("/search") no usages DanielMasin
    public String search(@RequestParam("q") String query, Model model) {
        List<DestinoTuristico> resultados = destinoTuristicoRepository.findByNombreContainingIgnoreCase(query);
        model.addAttribute( attributeName: "destinos", resultados);
        model.addAttribute( attributeName: "q", query);
        return "destinoTuristicos/index"; // usa la misma vista que el listado normal
    }

}
```

- HomeController

HomeController.java

```java
package org.esfe.controladores;

import org.esfe.modelos.DestinoTuristico;
import org.esfe.modelos.ImagenDTO;
import org.esfe.servicios.interfaces.IDestinoTuristicoService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;

import java.util.Base64;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.stream.Collectors;

@Controller no usages DanielMasin +1
public class HomeController {

    @Autowired 1 usage
    private IDestinoTuristicoService destinoTuristicoService;

    @GetMapping({"/", "/home"}) no usages DanielMasin +1
    public String index(Model model) {
            List<DestinoTuristico> destinos = destinoTuristicoService.ObtenerTodo();
            List<Map<String, Object>> destinosDTO = destinos.stream().map( DestinoTuristico dest -> {
                List<ImagenDTO> imagenesDTO = dest.getImagenes().stream() Stream<Imagen>
                        .map( Imagen img -> new ImagenDTO(
                                img.getId(),
                                img.getBytesArrayImage() != null ?
                                        Base64.getEncoder().encodeToString(img.getBytesArrayImage()) : null)) Stream<ImagenDTO>
                        .collect(Collectors.toList());
                Map<String, Object> dto = new HashMap<>();
                dto.put("destino", dest);
                dto.put("imagenes", imagenesDTO);
                return dto;
            }).collect(Collectors.toList());
            model.addAttribute( attributeName: "destinoTuristicos", destinosDTO);
            return "home/index";
        }

    }
```

- UsuarioController

UsuarioController.java

```java
package org.esfe.controladores;

import org.esfe.modelos.Rol;
import org.esfe.modelos.Usuario;
import org.esfe.servicios.interfaces.IRolService;
import org.esfe.servicios.interfaces.IUsuarioService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Pageable;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import java.util.List;
import java.util.Optional;
import java.util.stream.Collectors;
import java.util.stream.IntStream;

@Controller no usages AlejandroJuarez002
@RequestMapping("/usuarios")
public class UsuarioController {
    @Autowired 13 usages
    private IUsuarioService usuarioService;

    @Autowired 1 usage
    private IRolService rolService;

    @Autowired 1 usage
    private PasswordEncoder passwordEncoder;
```

UsuarioController.java

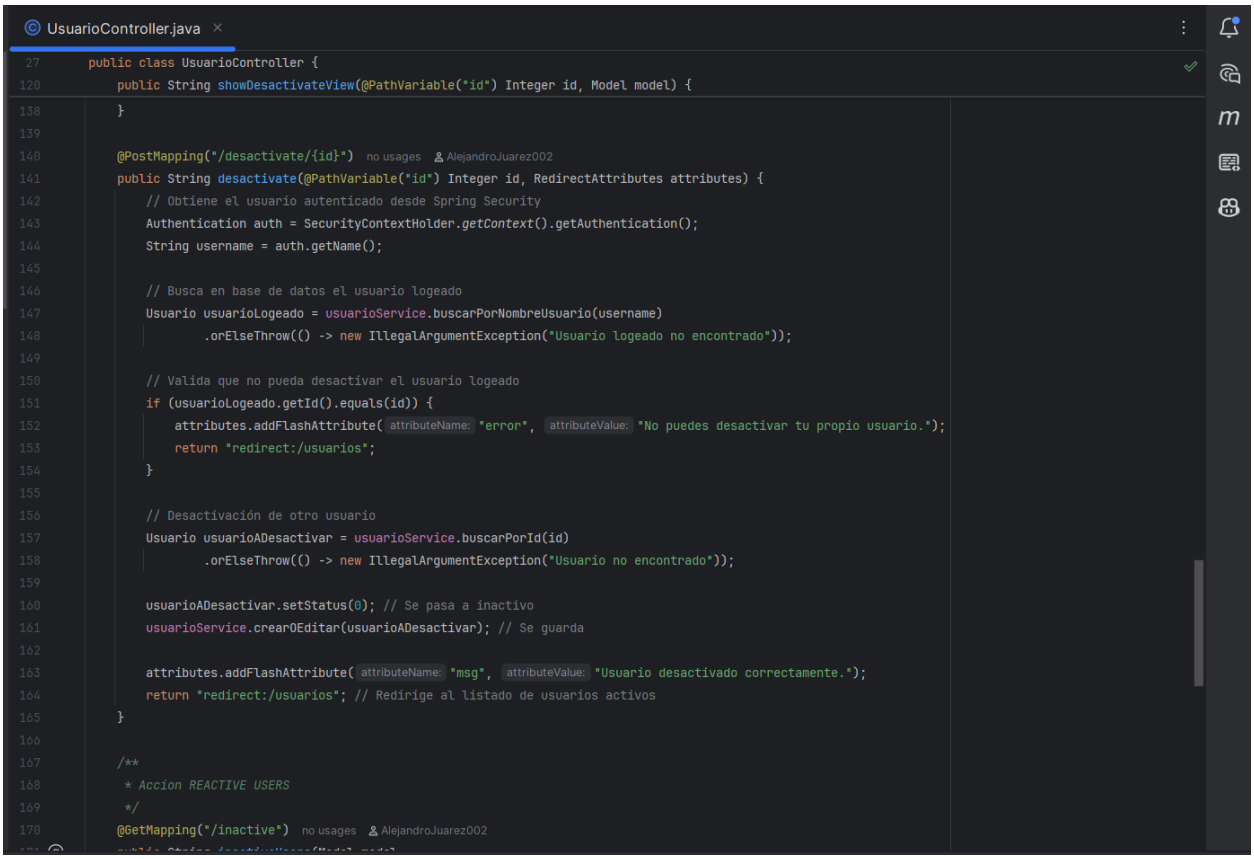
```java
public class UsuarioController {

    @GetMapping no usages AlejandroJuarez002
    private String index(Model model,
                         @RequestParam("page") Optional<Integer> page,
                         @RequestParam("size") Optional<Integer> size) {
        int currentPage = page.orElse( other: 1) - 1; //Si no esta seteado se asigna 0
        int pageSize = size.orElse( other: 5); //Tamaño de la pagina, se asigna 5
        Pageable pageable = PageRequest.of(currentPage, pageSize);

        Page<Usuario> usuarios = usuarioService.buscarPorStatus( status: 1, pageable); //Solo usuarios activos
        model.addAttribute( attributeName: "usuarios", usuarios);

        int totalPages = usuarios.getTotalPages();
        if (totalPages > 0) {
            List<Integer> pageNumbers = IntStream.rangeClosed(1, totalPages) IntStream
                    .boxed() Stream<Integer>
                    .collect(Collectors.toList());
            model.addAttribute( attributeName: "pageNumbers", pageNumbers);
        }
        return "usuario/index";
    }

    /**
     * Accion CREATE
     */
    @GetMapping("/create") no usages AlejandroJuarez002
    public String create(Usuario usuario) { return "usuario/create"; }

    @PostMapping("/save") no usages AlejandroJuarez002
    public String save(Usuario usuario, BindingResult result, Model model, RedirectAttributes attributes) {

        if (result.hasErrors()) {
            model.addAttribute(usuario);
            attributes.addFlashAttribute( attributeName: "error", attributeValue: "No se pudo guardar debido a un error.");
            return "usuario/create";
```

UsuarioController.java

```
27      public class UsuarioController {
67          public String save(Usuario usuario, BindingResult result, Model model, RedirectAttributes attributes) {
71                  attributes.addFlashAttribute( attributeName: "error",  attributeValue: "No se pudo guardar debido a un error.");
72                  return "usuario/create";
73              }
74
75              /**
76               * Validar que el rol exista si se asignó uno
77               */
78              if (usuario.getRol() == null) {
79                  Rol rol = rolService.buscarPorId(1)
80                          .orElseThrow(() -> new IllegalArgumentException("Rol no encontrado"));
81                  usuario.setRol(rol);
82              } else {
83                  attributes.addFlashAttribute( attributeName: "error",  attributeValue: "Debe seleccionar un rol válido.");
84                  return "usuario/create";
85              }
86
87              String password = passwordEncoder.encode(usuario.getClave());
88
89              usuario.setStatus(1);
90              usuario.setClave(password);
91              usuarioService.crearOEditar(usuario);
92              attributes.addFlashAttribute( attributeName: "msg",  attributeValue: "Usuario creado correctamente.");
93              return "redirect:/usuarios";
94          }
95
96          /**
97           * Accion DETAILS
98           */
99          @GetMapping("/details/{id}*")  no usages  AlejandroJuarez002
100 @        public String details(@PathVariable("id") Integer id, Model model) {
101             Usuario usuario = usuarioService.buscarPorId(id).get();
102             model.addAttribute( attributeName: "usuario", usuario);
103             return "usuario/details";
```

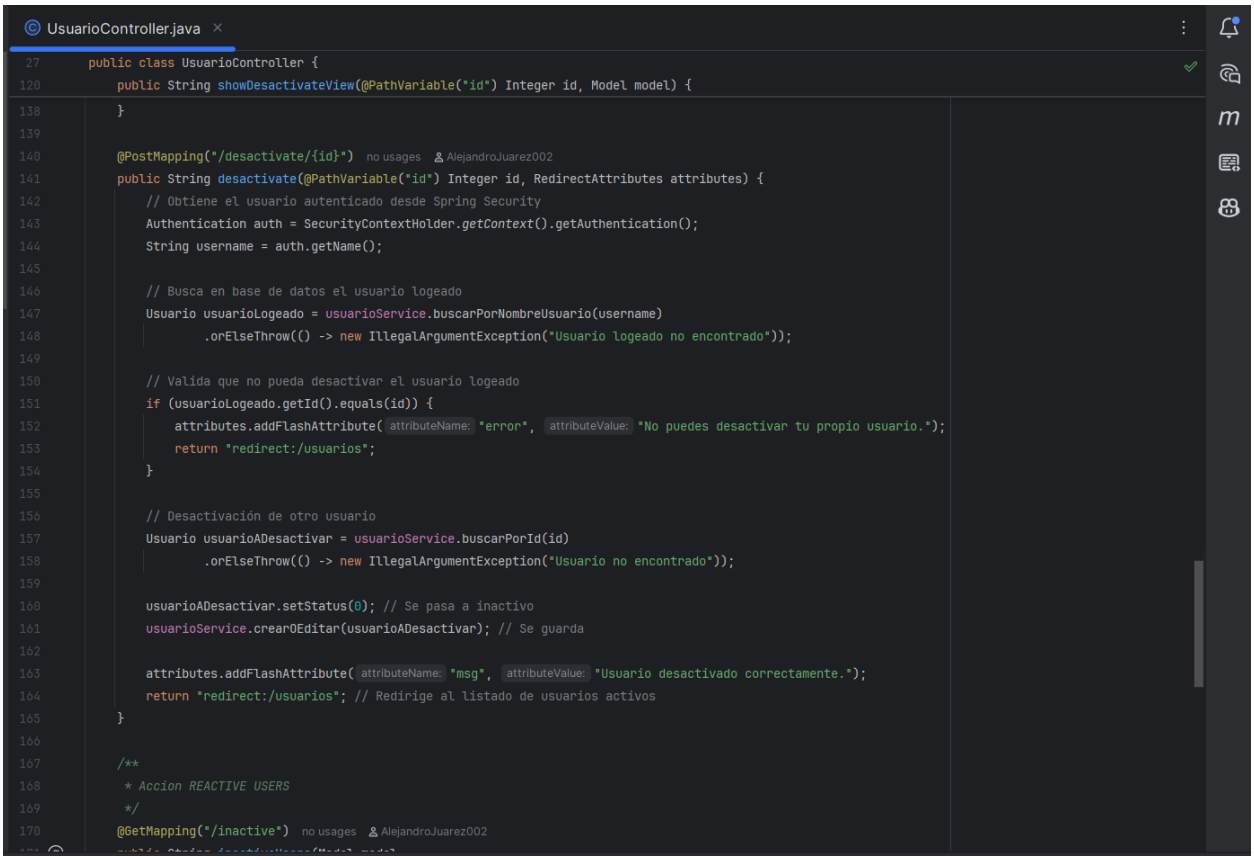
UsuarioController.java

```
27      public class UsuarioController {
100         public String details(@PathVariable("id") Integer id, Model model) {
104         }
105
106         /**
107          * Accion EDIT
108          */
109         @GetMapping("/edit/{id}*")  no usages  AlejandroJuarez002
110 @       public String edit(@PathVariable("id") Integer id, Model model) {
111             Usuario usuario = usuarioService.buscarPorId(id).get();
112             model.addAttribute( attributeName: "usuario", usuario);
113             return "usuario/edit";
114         }
115
116         /**
117          * Accion DESACTIVATE
118          */
119         @GetMapping("/desactivate/{id}*")  no usages  AlejandroJuarez002
120 @       public String showDesactivateView(@PathVariable("id") Integer id, Model model) {
121             Usuario usuario = usuarioService.buscarPorId(id)
122                     .orElseThrow(() -> new IllegalArgumentException("Usuario no encontrado"));
123
124             // Usuario logeado
125             Authentication auth = SecurityContextHolder.getContext().getAuthentication();
126             String username = auth.getName();
127
128             Usuario usuarioLogeado = usuarioService.buscarPorNombreUsuario(username)
129                     .orElseThrow(() -> new IllegalArgumentException("Usuario logeado no encontrado"));
130
131             // Se agrega bandera si es el mismo
132             boolean esMismoUsuario = usuarioLogeado.getId().equals(usuario.getId());
133
134             model.addAttribute( attributeName: "usuario", usuario);
135             model.addAttribute( attributeName: "esMismoUsuario", esMismoUsuario);
136
137             return "usuario/desactivate";
```



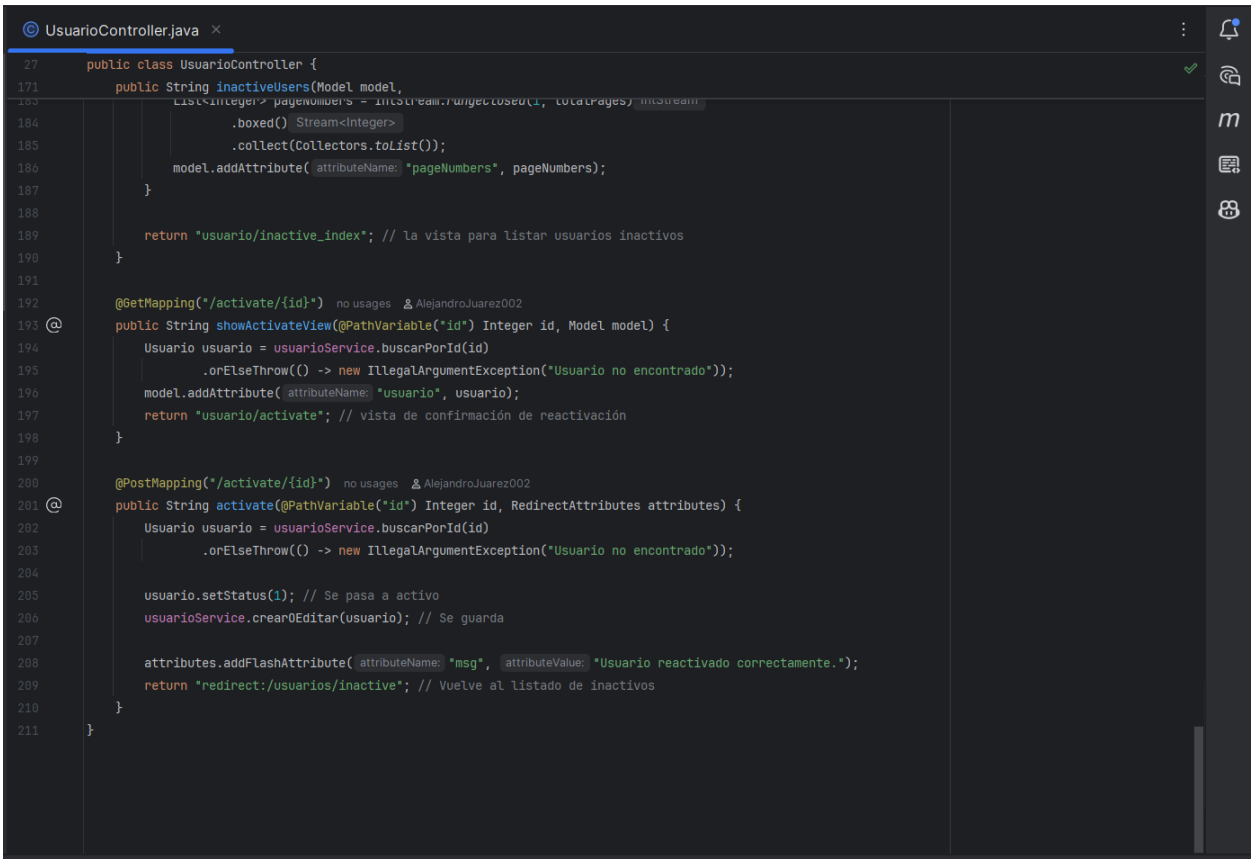
UsuarioController.java

```
27    public class UsuarioController {
120       public String showDesactivateView(@PathVariable("id") Integer id, Model model) {
138       }
139
140       @PostMapping("/desactivate/{id}")  no usages  AlejandroJuarez002
141       public String desactivate(@PathVariable("id") Integer id, RedirectAttributes attributes) {
142           // Obtiene el usuario autenticado desde Spring Security
143           Authentication auth = SecurityContextHolder.getContext().getAuthentication();
144           String username = auth.getName();
145
146           // Busca en base de datos el usuario logeado
147           Usuario usuarioLogeado = usuarioService.buscarPorNombreUsuario(username)
148                   .orElseThrow(() -> new IllegalArgumentException("Usuario logeado no encontrado"));
149
150           // Valida que no pueda desactivar el usuario logeado
151           if (usuarioLogeado.getId().equals(id)) {
152               attributes.addFlashAttribute( attributeName: "error",  attributeValue: "No puedes desactivar tu propio usuario.");
153               return "redirect:/usuarios";
154           }
155
156           // Desactivación de otro usuario
157           Usuario usuarioADesactivar = usuarioService.buscarPorId(id)
158                   .orElseThrow(() -> new IllegalArgumentException("Usuario no encontrado"));
159
160           usuarioADesactivar.setStatus(0); // Se pasa a inactivo
161           usuarioService.crearOEditar(usuarioADesactivar); // Se guarda
162
163           attributes.addFlashAttribute( attributeName: "msg",  attributeValue: "Usuario desactivado correctamente.");
164           return "redirect:/usuarios"; // Redirige al listado de usuarios activos
165       }
166
167       /**
168        * Accion REACTIVE USERS
169        */
170       @GetMapping("/inactive")  no usages  AlejandroJuarez002
```

UsuarioController.java

```
27    public class UsuarioController {
120       public String showDesactivateView(@PathVariable("id") Integer id, Model model) {
138       }
139
140       @PostMapping("/desactivate/{id}")  no usages  AlejandroJuarez002
141       public String desactivate(@PathVariable("id") Integer id, RedirectAttributes attributes) {
142           // Obtiene el usuario autenticado desde Spring Security
143           Authentication auth = SecurityContextHolder.getContext().getAuthentication();
144           String username = auth.getName();
145
146           // Busca en base de datos el usuario logeado
147           Usuario usuarioLogeado = usuarioService.buscarPorNombreUsuario(username)
148                   .orElseThrow(() -> new IllegalArgumentException("Usuario logeado no encontrado"));
149
150           // Valida que no pueda desactivar el usuario logeado
151           if (usuarioLogeado.getId().equals(id)) {
152               attributes.addFlashAttribute( attributeName: "error",  attributeValue: "No puedes desactivar tu propio usuario.");
153               return "redirect:/usuarios";
154           }
155
156           // Desactivación de otro usuario
157           Usuario usuarioADesactivar = usuarioService.buscarPorId(id)
158                   .orElseThrow(() -> new IllegalArgumentException("Usuario no encontrado"));
159
160           usuarioADesactivar.setStatus(0); // Se pasa a inactivo
161           usuarioService.crearOEditar(usuarioADesactivar); // Se guarda
162
163           attributes.addFlashAttribute( attributeName: "msg",  attributeValue: "Usuario desactivado correctamente.");
164           return "redirect:/usuarios"; // Redirige al listado de usuarios activos
165       }
166
167       /**
168        * Accion REACTIVE USERS
169        */
170       @GetMapping("/inactive")  no usages  AlejandroJuarez002
```


UsuarioController.java
public class UsuarioController {
public String inactiveUsers(Model model,
.boxed() Stream<Integer>
.collect(Collectors.toList());
model.addAttribute(attributeName: "pageNumbers", pageNumbers);
}
return "usuario/inactive_index"; // la vista para listar usuarios inactivos
}
@GetMapping("/activate/{id}") no usages AlejandroJuarez002
public String showActivateView(@PathVariable("id") Integer id, Model model) {
Usuario usuario = usuarioService.buscarPorId(id)
.orElseThrow(() -> new IllegalArgumentException("Usuario no encontrado"));
model.addAttribute(attributeName: "usuario", usuario);
return "usuario/activate"; // vista de confirmación de reactivación
}
@PostMapping("/activate/{id}") no usages AlejandroJuarez002
public String activate(@PathVariable("id") Integer id, RedirectAttributes attributes) {
Usuario usuario = usuarioService.buscarPorId(id)
.orElseThrow(() -> new IllegalArgumentException("Usuario no encontrado"));
usuario.setStatus(1); // Se pasa a activo
usuarioService.crearOEditar(usuario); // Se guarda
attributes.addFlashAttribute(attributeName: "msg", attributeValue: "Usuario reactivado correctamente.");
return "redirect:/usuarios/inactive"; // Vuelve al listado de inactivos
}
}