

# Arquitectura Orientada a Servicios

Trabajo Práctico 2

Daniel Montes Peris

Eric Olmo Sobrino

Curso 2018-2019

Grado Informática y Servicio

## Introducción

En este proyecto vamos a explicar la solución que hemos planteado a nuestro caso a resolver, el cual se basa en diseñar y hacer un sistema de incidencias.

En este proyecto hemos pensado que necesitábamos 3 entidades base, que son los trabajadores, las aplicaciones, y las incidencias.

Los trabajadores son las personas encargadas de poder crear incidencias sobre aplicaciones que no funcionen correctamente o que tengan algún problema con ellas y esto deba ser atendido por alguna persona del departamento de informática

Las aplicaciones son básicamente un listado de todas las aplicaciones que usan los trabajadores de la empresa.

Y las incidencias son el foco principal de nuestra propuesta ya que es donde hemos hecho hincapié en hacer un sistema robusto y bien diseñado. Estas no son mas que quejas o peticiones de los trabajadores sobre el mal funcionamiento de alguna aplicación que usen en la empresa a la hora de trabajar, y estas tienen que ser atendidas por algún trabajador del departamento de informática, es digamos el medio de comunicación de problemas entre trabajadores e informáticos.

## Modelo de operaciones

Descripción	Operación/URL	Parámetros de entrada	Salida OK
Alta de incidencia	POST /incidencia	Nombre de Usuario, Aplicación, Titulo, Descripción	Id_Incidencia, Nombre, Aplicación, Titulo
Mostrar incidencia por ID	GET /comprobar	Id_Incidencia	Id_Incidencia, Estado, Aplicación, Titulo, Nombre usuario, Descripcion, Fechas
Cambiar Estado Incidencia	PATCH /wip GET	Id_Incidencia + número	Id_Incidencia, Estado con un numero
Listar incidencias	GET /listar	Id_Incidencia	Id_Incidencia, Estado

## Alta Incidencia

Este WS lo utilizaremos para que el trabajador pueda crear una incidencia sobre una aplicación para que el equipo de incidencias pueda proceder a atender la petición y poder resolverla.

Los datos que pedimos para crear la incidencia en la base de datos es el nombre del usuario, el nombre de la aplicación, el título que le pondremos a la incidencia para poder tener una idea de la incidencia, y una descripción en la cual se explica detalladamente cual es el problema para poder solucionarla.

Como retorno al trabajador se le devolverá los datos que ha ingresado y la ID que se ha generado de esta incidencia

## Mostrar incidencia por ID

Esta operación lo que hace es mostrar todos los datos de la incidencia solicitada por el usuario a través de la ID de la incidencia.

Para así poder leer todos los campos de esta y poder sacar información de ella tanto para poder solucionarla como para poder ver su estado etc.

## Cambiar Estado Incidencia

Este WS lo que hace es pedir la Id de la incidencia y un número que indicaran 1= incidencia no revisada, 2= Trabajando en la incidencia y 3= incidencia finalizada.

## Listar incidencias

Este WS pide que incidencias se quieren listar (Las opciones de incidencias a mostrar son: Todas, Sin revisar, Trabajando en ellas o Finalizada.

## Clases Java utilizadas

### Incidencia:

#### Clase incidencia

Crearemos la clase incidencia la cual va a contener los campos de la tabla incidencia.

Añadiremos los métodos Get y Set, para poder acceder a los atributos.

Esta clase contiene la especificación ManyToOne ya que un trabajador puede crear muchas incidencias de una o diversas aplicaciones.

### Repositorio Incidencia

Crea un repositorio RestResource el cual usaremos para poder trabajar con más de una incidencia.

### Incidencia Controller

Este controlador lo que hace es controlar los métodos que se pueden hacer con la clase incidencia.

## Trabajador:

### Clase trabajador

Esta clase nos permite hacer objetos de tipo trabajador para poder trabajar con la tabla Trabajador.

### Trabajador repository

En este repositorio lo usaremos directamente en la base de datos, utilizando la clase trabajador.

Directamente haremos un Post a la base de datos del objeto trabajador, que se crea al llamar a la operación.

### Alta trabajador

Tenemos que crear esta operación porque para que exista la incidencia debe existir un trabajador, que indique la incidencia sobre alguna aplicación, y así otro o el mismo trabajador la resuelva y se encargue de todo el procedimiento para resolverla.

A la hora de hacer la tabla tenemos que introducir los siguientes datos: el Id de cada uno de los trabajadores que se den de alta y crean una incidencia, el nombre de cada trabajador indicado para poder saber quién realiza cada acción y por último también necesitamos la ID de la aplicación que no funciona bien.

## Aplicación

### Clase Aplicación

Esta clase nos permite hacer objetos de tipo aplicación para poder trabajar con la tabla Aplicación.

### Aplicacion repository

En este repositorio lo usaremos directamente en la base de datos, utilizando la clase trabajador.

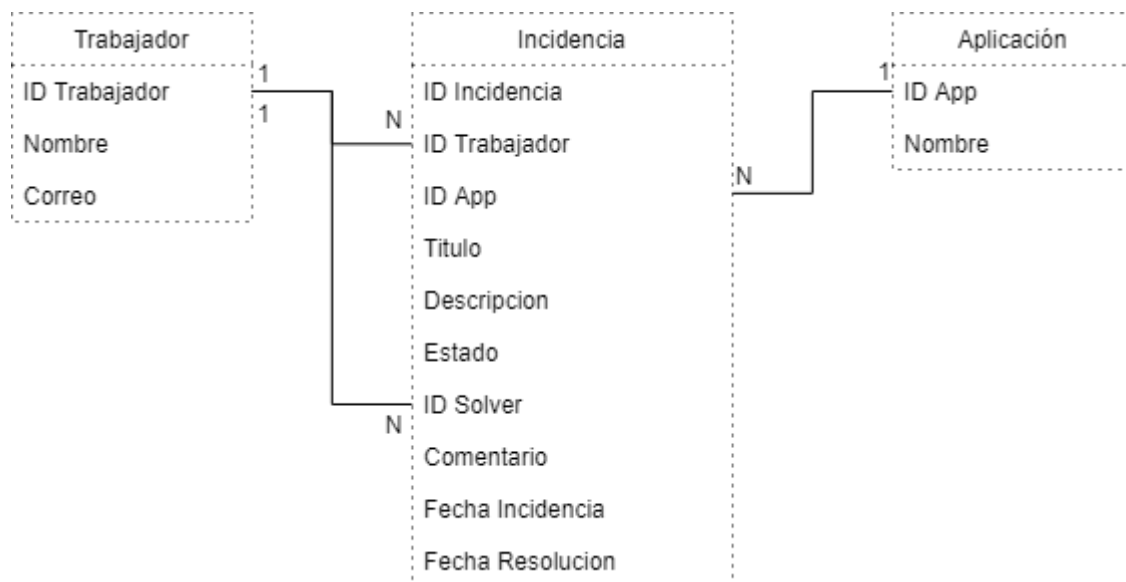
Directamente haremos un Post a la base de datos del objeto trabajador, que se crea al llamar a la operación.

### Alta Aplicación

Esta tabla se crea ya que servirá para poner todas las aplicaciones con la ID de cada una y el nombre de dicha aplicación, así pudiendo con ManyToOne enlazar la tabla de incidencias con la de aplicación para saber que incidencia/s con que aplicación.

## Modelo de entidades

En el modelo de entidades hemos detectado que el sistema se basará en tres tablas, una de trabajadores, otra de aplicaciones de la empresa y la de incidencias, en la cual se guardan las incidencias con las id del trabajador que la ha hecho junto la id de la aplicación de la cual se queja, mientras que de la tabla trabajadores también sale el id de la persona encargada de solventar el problema.





# Probar Servicio de Incidencia

## Crear Aplicación

Method

POST

Request URL

http://localhost:8080/aplicacion

SEND

Parameters ^

Headers

Body

Variables

Body content type

application/json

Editor view

Raw input

FORMAT JSON   MINIFY JSON

{ "nombre": "Excel" }

201 Created

48.50 ms

DETAILS v

	id	nombre
▶	1	Excel
✱	NULL	NULL

# Crear Trabajador

Method

POST

Request URL

http://localhost:8080/trabajador

SEND

Parameters ^

Headers

Body

Variables

Body content type

application/json

Editor view

Raw input

FORMAT JSON

MINIFY JSON

{ "nombre": "Dani", "correo": "dani@montes.com" }

201 Created

787.50 ms

DETAILS ^

	id	correo	nombre
▶	2	dani@montes.com	Dani
•	NULL	NULL	NULL

## Crear Incidencia

```
@PostMapping("/incidencias/{idapp}/{idtrabajador}")
public Incidencia createIncidencia(@PathVariable int idapp, @PathVariable int idtrabajador,
    @Valid @RequestBody Incidencia incidencia) {

    return aplicacionRepository.findById(idapp).map(app -> { // busca la app
        incidencia.setApp(app);
        return trabajadorRepository.findById(idtrabajador).map(trabajador -> { // busca el trabajador
            incidencia.setTrabajador(trabajador);
            return incidenciaRepository.save(incidencia);
        }).orElseThrow(() -> new ResourceNotFoundException("Trabajador " + idtrabajador + " not found"));
    }).orElseThrow(() -> new ResourceNotFoundException("App " + idapp + " not found"));
}
```

Method POST Request URL http://localhost:8080/incidencias/1/2

**Parameters**

Headers	Body	Variables
Body content type application/json	Editor view JSON visual editor	
titulo	Incidencia 1	
descripcion	Esta es la incidencia 1	
estado	1	
idSolver	2	
comentario	Incidencia no comentada	
fechaIncidencia	04/01/2019	
fechaResolucion	09/01/2019	

+ ADD PROPERTY

200 OK 37.80 ms

DETAILS

	id	comentario	descripcion	estado	fecha_incidencia	fecha_resolucion	id_solver	titulo	app_id	trabajador_id
*	3	Incidencia no comentada	Esta es la incidencia 1	1	04/01/2019	09/01/2019	2	Incidencia 1	1	2

## Tests:

- Probar a hacer insert de una incidencia de un trabajador inexistente

Method POST Request URL http://localhost:8080/incidencias/1/8 SEND





Parameters Show panel

**404 Not Found** 130.30 ms DETAILS ^

POST http://localhost:8080/incidencias/1/8

Response headers 3 Request headers 1 Redirects 0 Timings

content-type: application/json;charset=UTF-8  
transfer-encoding: chunked  
date: Fri, 04 Jan 2019 11:40:13 GMT

```
{
  "timestamp": "2019-01-04T11:40:13.301+0000",
  "status": 404,
  "error": "Not Found",
  "message": "Trabajador 8 not found",
  "path": "/incidencias/1/8"
}
```

- Probar a hacer insert de una incidencia de una aplicación inexistente

Method POST Request URL http://localhost:8080/incidencias/6/2 SEND





Parameters ▼

**404 Not Found** 18.00 ms DETAILS ^

POST http://localhost:8080/incidencias/6/2

Response headers 3 Request headers 1 Redirects 0 Timings

content-type: application/json;charset=UTF-8  
transfer-encoding: chunked  
date: Fri, 04 Jan 2019 11:41:29 GMT

```
{
  "timestamp": "2019-01-04T11:41:29.946+0000",
  "status": 404,
  "error": "Not Found",
  "message": "App 6 not found",
  "path": "/incidencias/6/2"
}
```

## Operaciones de Incidencias

### Listar todas las Incidencias

```
@GetMapping("/incidencias")
public Page<Incidencia> getAllIncidencia(Pageable pageable) { return incidenciaRepository.findAll(pageable); }
```

Method GET Request URL http://localhost:8080/incidencias SEND

Parameters Show panel

200 OK 851.00 ms DETAILS

```
{
  "content": [Array(1)]
  -0: {
    "id": 3,
    "titulo": "Incidencia 1",
    "descripcion": "Esta es la incidencia 1",
    "estado": 1,
    "idSolver": 2,
    "comentario": "Incidencia no comentada",
    "fechaIncidencia": "04/01/2019",
    "fechaResolucion": "09/01/2019",
    "app": {
      "id": 1,
      "nombre": "Excel"
    },
    "trabajador": {
      "id": 2,
      "nombre": "Dani",
      "correo": "dani@montes.com"
    }
  }
}
```

### Tests:

- Probar la operación sin incidencias en la base de datos: No retorna nada

## Cambiar Estado Incidencia

```
@PutMapping("/{incidencias}/{incidenciaID}/{estado}")
public Incidencia updateIncidenciaEstado(@PathVariable Integer incidenciaID, @PathVariable Integer estado) {
    return incidenciaRepository.findById(incidenciaID).map(post -> {
        post.setEstado(estado);
        return incidenciaRepository.save(post);
    }).orElseThrow(() -> new ResourceNotFoundException("Incidencia " + incidenciaID + " not found"));
}
```

Method Request URL  
PUT http://localhost:8080/incidencias/3/2

SEND

Parameters

200 OK 594.10 ms

DETAILS

```
{
  "id": 3,
  "titulo": "Incidencia 1",
  "descripcion": "Esta es la incidencia 1",
  "estado": 2,
  "idSolver": 2,
  "comentario": "Incidencia no comentada",
  "fechaIncidencia": "04/01/2019",
  "fechaResolucion": "09/01/2019",
  "trabajador": {
    "id": 2,
    "nombre": "Dani",
    "correo": "dani@montes.com"
  },
  "app": {
    "id": 1,
    "nombre": "Excel"
  }
}
```

## Test

- Poner una incidencia inexistente en la base de datos

Method Request URL  
PUT http://localhost:8080/incidencias/6/2

SEND

Parameters

404 Not Found 44.00 ms

DETAILS

```
{
  "timestamp": "2019-01-04T13:59:17.448+0000",
  "status": 404,
  "error": "Not Found",
  "message": "Incidencia 6 not found",
  "path": "/incidencias/6/2"
}
```





## Mostrar incidencia por ID

```
@GetMapping("/incidencias/{idincidencia}")
public String getIncidencia(@PathVariable int idincidencia) {
    return incidenciaRepository.findById(idincidencia).map(incidencia -> { // busca la incidencia
        | return incidencia.toString();
    }).orElseThrow(() -> new ResourceNotFoundException("Incidencia " + idincidencia + " not found"));
}
```

Method GET Request URL http://localhost:8080/incidencias/3 SEND

Parameters

200 OK 14.90 ms DETAILS

Incidencia{id=3, Aplicacion=1, Trabajador=2, titulo='Incidencia 1', descripcion='Esta es la incidencia 1', estado='2', idSolver=2, comentario='Incidencia no comenta da', fechaIncidencia='04/01/2019', fechaResolucion='09/01/2019'}





## Test:

- Buscar una incidencia inexistente: No encuentra nada

Method GET Request URL http://localhost:8080/incidencias/4 SEND

Parameters

404 Not Found 497.60 ms DETAILS

{
 "timestamp": "2019-01-04T14:06:07.459+0000",
 "status": 404,
 "error": "Not Found",
 "message": "Incidencia 4 not found",
 "path": "/incidencias/4"
}