

# “Plataforma de gestión de Incidencias Empresariales con administración de Usuarios y Empresas con Spring Boot”

Memoria del Proyecto del Ciclo Formativo de GS IFC302

**Desarrollo de Aplicaciones Multiplataforma**

Curso: 2022/2023

Academia Marco Formación

**Autor: Pablo Buey Martín**

## **ÍNDICE**

- 1. Descripción**
- 2. Objetivos**
- 3. Recursos Empleados**
- 4. Desarrollo o Explicación**
- 5. Conclusión y Posibles Mejoras**
- 6. Bibliografía y Referencias**

# 1. Descripción.

“Plataforma de gestión de Incidencias Empresariales con administración de Usuarios y Empresas con Spring Boot” es una aplicación web diseñada para la gestión de incidencias por parte de empresas y usuarios.

En la aplicación se pueden registrar Usuarios, Empresas e Incidencias.

Las Empresas pueden crear, listar, actualizar y eliminar sus incidencias.

Los Usuarios pueden listar Usuarios, Empresas e Incidencias, así como actualizarlos y eliminarlos.

Para poder utilizar la aplicación tanto como usuario como por empresa será necesario registrarse y logearse.

(Esta parte no se si es necesaria: )

En el índice de la web se puede encontrar un menú muy simple situado a la izquierda que recuerda al interfaz de una aplicación móvil, en el que se encuentran dos botones: “soy usuario” y “soy empresa”.

En el cuerpo de la página principal se encuentra la bienvenida y la descripción de la página, así como las instrucciones a seguir dependiendo de si eres usuario o empresa.

En la página podrás navegar hacia el apartado de usuarios o hacia el de empresas.

Para hacer uso de las funcionalidades de la web tendrás que registrarte como Usuario o como Empresa para después logearte.

En cuanto al diseño:

La interfaz gráfica es muy simple, buscando la funcionalidad y la eficiencia, centrándose en la usabilidad.

Los colores empleados son agradables y simples, los cuales pueden recordar a plataformas que resultan familiares como Facebook o Twitter.

La fuente de letras es redondeada y muy legible para facilitar su lectura.

He evitado cargar la web con fotografías y texto que no aporte nada ya que el objetivo principal es la eficiencia y el uso fácil y rápido del servicio web.

## 2. Objetivos.

El objetivo principal de este proyecto ha sido implementar los conocimientos adquiridos en el Grado Superior de Desarrollo de Aplicaciones Multiplataforma en un Framework Java específico como Spring Boot y Maven junto con el IDE IntelliJ, puesto que nunca los había utilizado y esto iba a permitirme aprender y familiarizarme con estas herramientas, las cuales tienen una amplia demanda en el mercado laboral.

Al ser una aplicación construida con Java, HTML, CSS, JavaScript, XML, JSON y SQL, ha sido una buena oportunidad para fortalecer mis habilidades en los lenguajes que ya conocía, así como aprender JavaScript, el cual no había utilizado anteriormente. Esto me ha permitido desarrollar un enfoque Full-Stack, trabajando tanto en la capa back-end como de la capa front-end, aunque con un enfoque mayor en el back-end ya que es la especialización del grado que he estudiado.

Además de poner a prueba mis conocimientos, mi objetivo final ha sido realizarlo con éxito estableciendo pequeños objetivos a lo largo del camino, desde la conexión del IDE a la base de datos, pasando por la creación de una tabla y un modelo para la creación, lectura, actualización y eliminación de usuarios (CRUD), ampliándolo a un login para la creación de una sesión, para después desarrollar más tablas y modelos para distintos tipos de funcionalidades según el login utilizando JWT como librería para la autenticación y autorización de usuarios.

Uno de los aspectos más interesantes del proyecto ha sido comprender el funcionamiento de un API REST para la definición de endpoints a través de solicitudes HTTP, de manera que la capa cliente se comunica a través del servidor mediante scripts de JavaScript con solicitudes a dichas APIs, utilizando controladores, DAOs y modelos, para acabar persistiendo los datos en la base de datos mediante la librería Hibernate.

La utilización de Lombok para la reducción del uso de código repetitivo también ha sido un valioso descubrimiento.

Todo esto ha resultado un divertido rompecabezas que he disfrutado y completado con éxito.

### 3. Recursos empleados.

Durante el desarrollo de este proyecto he utilizado los siguientes recursos:

#### Lenguajes de programación:

**Java:** Como lenguaje principal para el desarrollo de la capa de negocio.

**HTML:** Para crear la estructura y el contenido de la capa cliente.

**CSS:** Se empleó para definir el estilo y la apariencia visual de la página web.

**JavaScript:** Para agregar interactividad y funcionalidades dinámicas en el lado del cliente.

**SQL:** Para la creación y el manejo de la base de datos.

**XML:** Para la configuración del proyecto.

**JSON:** Para el manejo de datos entre JavaScript y las apis de Java.

#### Frameworks y bibliotecas:

**Spring Boot:** Se empleó como el framework principal para el desarrollo de la aplicación web, aprovechando su capacidad de configuración automática y su soporte para la creación de servicios web RESTful.

**Hibernate:** Se utilizó como el framework de mapeo objeto-relacional para interactuar con la base de datos, simplificando las operaciones de persistencia mediante anotaciones.

**Lombok:** Se utilizó como una biblioteca para reducir la cantidad de código repetitivo, mediante la generación automática de getters, setters y constructores, entre otros.

**JSON Web Token (JWT):** Se empleó como una biblioteca para la autenticación y autorización de usuarios, generando y validando tokens de seguridad.

#### Herramientas y tecnologías:

**Maven:** Se utilizó como la herramienta de gestión de dependencias y construcción del proyecto, facilitando la configuración y descarga automática de las bibliotecas necesarias.

**IntelliJ IDEA:** Se empleó como el Entorno de Desarrollo Integrado (IDE) principal, proporcionando un entorno de desarrollo potente y funcionalidades avanzadas para la codificación y depuración.

**XAMPP:** Se utilizó una base de datos relacional MySQL, para almacenar y recuperar los datos de usuarios, empresas e incidencias.

**GitHub:** Se empleó como sistema de control de versiones.

**YouTube:** Para crear la base del proyecto seguí un curso de YouTube de Spring Boot.

**ChatGPT:** Esta herramienta me ha resultado clave para comprender la base del proyecto y así poder ampliarlo. Poder preguntar en cualquier momento a una IA partes de código para comprenderlo, o plantearle problemas para buscar distintas soluciones y, en definitiva, la interacción con un modelo de lenguaje como es ChatGPT ha sido todo un aprendizaje en sí mismo que me ha ayudado a ampliar mucho mis conocimientos.

## 4. Desarrollo o Explicación.

Teniendo en cuenta que tenía claro que quería desarrollar una aplicación utilizando Spring Boot para la gestión de usuarios como mínimo, el primer problema que se me presenta es que nunca lo he utilizado.

El primer paso es buscar información sobre Spring Boot, ya que todo va a girar en torno a este Framework.

### ¿Qué es Spring Boot?



Spring Boot es un Framework de desarrollo de aplicaciones en Java que simplifica y acelera la creación de aplicaciones empresariales. Se basa en el framework **Spring** y proporciona una configuración automática y predeterminada, lo que significa que requiere menos configuración manual por parte del desarrollador.

Una de las principales ventajas de Spring Boot es que proporciona convenciones predefinidas y toma decisiones predeterminadas para simplificar el desarrollo. Esto permite a los desarrolladores comenzar rápidamente con sus proyectos sin tener que realizar configuraciones complicadas.

Spring Boot se centra en la creación de aplicaciones basadas en microservicios y arquitecturas de servicios web RESTful. Ofrece una amplia gama de características y funcionalidades que facilitan la creación de servicios web, la administración de dependencias, la configuración de bases de datos, la seguridad, la implementación y el monitoreo.

Además, Spring Boot se integra bien con otros proyectos de Spring, como Spring Data, Spring Security y Spring Cloud, lo que permite una mayor flexibilidad y potencia en el desarrollo de aplicaciones empresariales completas.

En resumen, **Spring Boot** es un framework que agiliza el desarrollo de aplicaciones Java al proporcionar una configuración automática y predefinida, convenciones y características para crear servicios web **RESTful** y aplicaciones empresariales de manera rápida y eficiente.

## Pero, ¿Qué es RESTful?

Primero habría que definir qué es una API.

Una API (Application Programming Interface) es un conjunto de reglas y protocolos que permiten la comunicación y la interacción entre diferentes software y aplicaciones. **Actúa como un intermediario** que permite a los sistemas solicitar y compartir información o realizar operaciones específicas de forma estandarizada y programática.

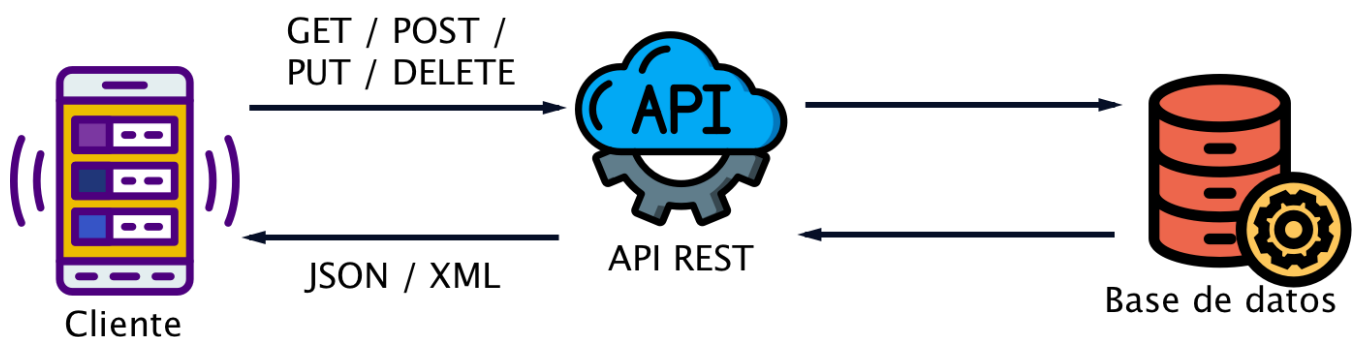
Una **API** proporciona una interfaz para que los desarrolladores puedan acceder y utilizar funcionalidades o datos de un sistema o servicio de manera controlada y automatizada.

Una **API REST** es una interfaz de programación de aplicaciones que permite la comunicación entre diferentes sistemas de software. Se basa en el **protocolo HTTP** (Hypertext Transfer Protocol) y utiliza **URLs** para identificar los recursos que se pueden solicitar o manipular. Estos recursos se envían y reciben en formato JSON, que es un formato de intercambio de datos legible por humanos.

La principal **ventaja** de una API REST es que simplifica el trabajo de los programadores al proporcionar una estructura predefinida para las solicitudes y respuestas. **Los desarrolladores no necesitan empezar desde cero**, solo necesitan realizar modificaciones específicas según sus necesidades y el contexto en el que se va a utilizar. Una API REST puede ser utilizada para comunicarse con un sistema operativo que tenga bases de datos o protocolos de comunicación.

La arquitectura REST se basa en el principio **cliente-servidor**, donde el servidor contiene los recursos y los clientes acceden mediante URLs específicas. Además, las APIs REST son escalables, lo que significa que pueden manejar grandes volúmenes de solicitudes al distribuir la carga en múltiples servidores.

Una característica importante de las APIs REST es su independencia. Cada solicitud que realiza el cliente debe incluir toda la información necesaria en la cabecera y el cuerpo HTTP, lo que permite que el servidor procese la solicitud de manera independiente sin necesidad de almacenar información de contexto o sesiones. Esto mejora el rendimiento y facilita el diseño e implementación del servidor.



## Aprender Spring Boot

Para aprender Spring Boot lo primero que hago es ver distintos videos en YouTube para tener una idea general.

Después busco cursos específicos en Udemy y YouTube para terminar decidiéndome por uno que podría servirme tanto para comprender el funcionamiento del framework como para comenzar a desarrollar las bases de mi proyecto.

### Primeros pasos

Ya que el curso que encuentro lo desarrolla en el **IDE IntelliJ** y me resulta interesante, lo descargo desde su página web oficial, instalo y configuro para la creación del proyecto con Maven.



Búsqueda de un curso de Spring Boot.

Descarga de las herramientas: IntelliJ, XAMPP, GitHub Desktop.

Comienzo del desarrollo, creación del proyecto y Maven, explicación de las distintas partes del proyecto (modelos, dao.. etc). Mirar en el video de youtube los pasos que se han ido siguiendo, así como la configuración.

(A PARTIR DE AQUÍ VOY A EXPLICAR TODA LA PARTE DE CÓDIGO, SERVIDOR, BASE DE DATOS, ETC).



## **5. Conclusión y Posibles Mejoras**

## **6. Bibliografía y Referencias**