



# ¿Cómo trabajas con la IA?

**GONZALO CANO SEBASTIÁN**

**2ºDAM**

**Acceso a datos**

**29/01/2026**

**Índice:**

<b>1. Problema / Necesidad de Gestión que se ha desarrollado</b>	<b>3</b>
<b>2. Estructura del Proyecto</b>	<b>3</b>
<b>3. Primer Prompt que le diste a la IA y por qué</b>	<b>5</b>
<b>4. Qué le pediste a la IA, qué no funcionaba y cómo lo resolviste</b>	<b>6</b>
<b>5. Preguntas para entender el proyecto</b>	<b>6</b>

## 1. Problema / Necesidad de Gestión que se ha desarrollado

El proyecto consiste en desarrollar un sistema de gestión del inventario de productos de una tienda de dispositivos informáticos. La necesidad surge de la dificultad para controlar de forma manual el stock y la información de los productos, lo que puede generar problemas de abastecimiento o control.

Este sistema permite:

- Registrar productos con sus datos básicos: nombre, descripción, precio, stock actual, etc.
- Visualizar todos los productos en stock y filtrarlos por categorías.
- Eliminar y modificar productos.
- Controlar que los datos introducidos sean válidos.
- Ver de forma clara los productos que se encuentren por debajo del mínimo en existencias.

## 2. Estructura del Proyecto

El proyecto está desarrollado en Java con Spring Boot para la capa de servicios y acceso a base de datos, y Swing para la interfaz gráfica.

- **Paquete modelo:** Clases que representan entidades (Producto, Categoria, Proveedor).
- **Paquete servicios:** Lógica de negocio, acceso a datos (repositorios) y validaciones.
- **Paquete UI:** Ventanas de Swing que interactúan con el usuario y consumen los servicios.
- **Configuración Spring:** Permite la inyección de dependencias para desacoplar componentes.

Las ventanas principales son:

- VentanaPrincipal: menú inicial con botones para stock, nuevo producto y categoría.
- VentanaStock: muestra listado de productos, permite modificar o eliminar.
- VentanaNuevoProducto: formulario para crear productos nuevos con validaciones.
- VentanaModificarProducto: formulario para alterar la información de un producto.

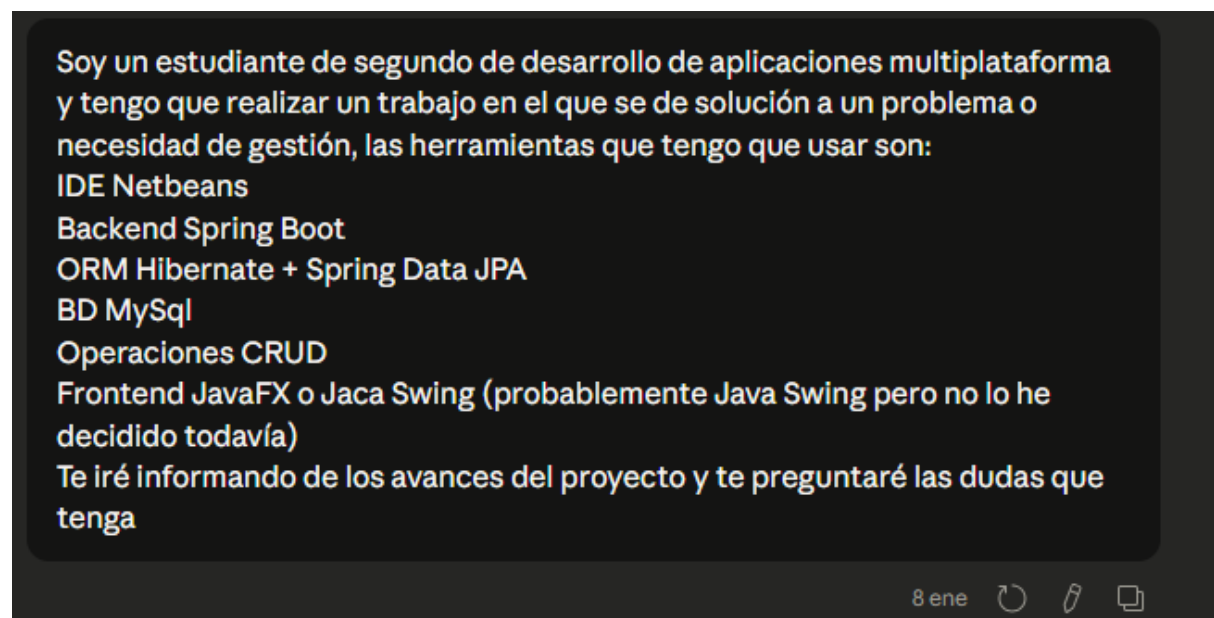
```
src
├── main
│   └── java
│       ├── com.gestion_inventario
│       │   ├── modelo
│       │   │   ├── Categoria
│       │   │   ├── Producto
│       │   │   └── Proveedor
│       │   ├── repositorios
│       │   │   ├── CategoriaRepositorio
│       │   │   ├── ProductoRepositorio
│       │   │   └── ProveedorRepositorio
│       │   ├── servicios
│       │   │   ├── CategoriaServicio
│       │   │   ├── ProductoServicio
│       │   │   └── ProveedorServicio
│       │   └── ui
│       │       ├── renderers
│       │       │   ├── VentanaModificarProducto
│       │       │   │   └── VentanaModificarProducto.form
│       │       │   ├── VentanaNuevoProducto
│       │       │   │   └── VentanaNuevoProducto.form
│       │       │   ├── VentanaPrincipal
│       │       │   │   └── VentanaPrincipal.form
│       │       │   ├── VentanaStock
│       │       │   │   └── VentanaStock.form
│       │       └── GestionInventarioApplication
```

### 3. Primer Prompt que le diste a la IA y por qué

Prompt:

"Soy un estudiante de segundo de desarrollo de aplicaciones multiplataforma y tengo que realizar un trabajo en el que se dé solución a un problema o necesidad de gestión, las herramientas que tengo que usar son: IDE Netbeans Backend Spring Boot ORM Hibernate + Spring Data JPA BD MySql Operaciones CRUD Frontend JavaFX o Jaca Swing (probablemente Java Swing pero no lo he decidido todavía) Te iré informando de los avances del proyecto y te preguntaré las dudas que tenga"

Razón: Normalmente cuando comienzo un proyecto con tecnologías que no he usado me informo antes de empezar, por eso le dí el contexto de lo que iba a hacer y que iba a usar para después preguntarle las dudas que tenga sobre esas tecnologías.



Su respuesta no es muy larga solo dice que le parece un proyecto muy completo y que le pregunte cuando tenga cualquier duda, esta IA es la de Claude pero luego pasé a usar ChatGPT porque claude tiene limite de mensajes diarios.

## 4. Qué le pediste a la IA, qué no funcionaba y cómo lo resolviste

- **Problema:** Uno de los primeros problemas que tuve fue que en la ventana de stock al principio no cargaban los datos y después de unos cambios me dió un error que no sabía solucionar.
- **Prompt:** "error al abrir la ventana de stock que antes al tener la tabla vacía no salía el error: Exception in thread "AWT-EventQueue-0"  
org.hibernate.LazyInitializationException: failed to lazily initialize a collection of role: com.gestion\_inventario.modelo.Producto.proveedores: could not initialize proxy - no Session"
- **Respuesta IA:** Este error **LazyInitializationException** significa que estás intentando acceder a una colección marcada como **lazy** (perezosa) — en este caso, la colección **proveedores** dentro de **Producto** — **fuera del contexto de una sesión activa de Hibernate**.
- **Solución:** Me dió varias soluciones que sí que arreglaron el error pero la tabla seguía sin mostrar nada, al final terminé pasándole el código entero de la ventana y el problema era que usaba el modelo de la tabla mal, por algún motivo cree un modelo de la tabla extra del que tenía la tabla y estaba usando el otro.

## 5. Preguntas para entender el proyecto

Algunas preguntas que hice a la IA para entender mejor y estructurar el proyecto:

- "¿Cómo estructurar un proyecto con Spring Boot y Swing para gestionar inventario?" esta pregunta la hice para poder estructurar bien el proyecto.

```
Estructura general recomendada para tu proyecto gestion-inventario Copiar código

sql

gestion-inventario/
├─ src/
│  ├─ main/
│  │  ├─ java/
│  │  │  └─ com/
│  │  │     └─ inventario/
│  │  │        ├── GestionInventarioApplication.java <-- Clase main Spring Boot
│  │  │        ├── controller/ <-- Controladores REST
│  │  │        ├── model/ <-- Clases de dominio (Producto, Categ
│  │  │        ├── repository/ <-- Repositorios JPA
│  │  │        ├── service/ <-- Servicios con lógica de negocio
│  │  │        └─ util/ <-- Clases utilitarias
│  │  └─ resources/
│  │     ├── application.properties <-- Configuración Spring Boot
│  │     ├── static/ <-- Recursos estáticos (si usas web)
│  │     └─ templates/ <-- Plantillas (si usas Thymeleaf, etc
│  └─ swing-frontend/ <-- (Opcional) carpeta para código Swi
│     └─ com/inventario/frontend/ <-- Código Java Swing para interfaz gr
└─ test/
   └─ java/
      └─ com/
         └─ inventario/
            └─ (tests unitarios y de integración)
└─ pom.xml <-- Archivo Maven de configuración
```

Al final he hecho algunos cambios en la estructura pero es muy parecida a la que me dió.

- "¿Cómo funcionan Hibernate + Spring Data JPA?"

Excelente pregunta. Te lo explico de forma clara:

## Hibernate vs Spring Data JPA

Son dos capas que trabajan juntas:

### Hibernate (la capa baja)

Es el ORM real que hace el trabajo sucio:

- Traduce objetos Java ↔ tablas SQL
- Genera las consultas SQL automáticamente
- Gestiona las conexiones a la base de datos
- Crea las tablas según tus clases Java

### Spring Data JPA (la capa alta)

Es una capa de abstracción sobre Hibernate que te facilita aún más la vida:

- Te da interfaces mágicas donde no escribes casi nada de código
- Genera los métodos CRUD automáticamente

- "¿Qué tengo que configurar en las propiedades?"

En el archivo `application.properties` necesitas configurar la conexión a MySQL. Te explico paso a paso:

### 1. Ubicación del archivo

Ve a: Other Sources → src/main/resources → application.properties

### 2. Configuración básica para MySQL:

Copia y pega esto en tu `application.properties`:

```
properties

# Configuración de la base de datos MySQL
spring.datasource.url=jdbc:mysql://localhost:3306/gestion_inventario
spring.datasource.username=root
spring.datasource.password=tu_contraseña_mysql

# Configuración de Hibernate
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
spring.jpa.properties.hibernate.format_
```