Sistema CRUD de Soporte Técnico

Aplicación Web RESTful con Spring Boot y Swagger

Proyecto Final de **Desarrollo de los Componentes del Negocio** – IDAT 2025

Integrantes :

Velarde Robles Francisco Xavier Leon
Roman Huaman Josled Luis Antonio

Peña Chavez Gissel Melani
Osorio Guzman Jose Luis

Colina Martin Jesus Gabriel.

1. Introducción al Desarrollo CRUD

El acrónimo **CRUD** (Create, Read, Update, Delete) representa las cuatro operaciones básicas y fundamentales en la persistencia de datos. Dominar el desarrollo de sistemas CRUD es esencial para cualquier aplicación empresarial, ya que constituye la base de la gestión de información.

Crear (Create)

Registro de nuevos datos en el sistema.

Leer (Read)

Recuperación y visualización de la información.

Actualizar (Update)

Modificación de registros existentes.

Eliminar (Delete)

Borrado permanente o lógico de los datos.

> Este proyecto tiene como objetivo desarrollar un **Sistema de Soporte Técnico** para la gestión completa de solicitudes, implementando una API RESTful robusta y documentada con Spring Boot.

2. Tecnologías Clave del Stack

Hemos seleccionado un conjunto de herramientas modernas y estándares de la industria para asegurar la eficiencia, la escalabilidad y la documentación de nuestra aplicación web.

Spring Boot

Framework Java que simplifica la configuración y despliegue, ideal para construir **APIs REST** de alto rendimiento.

Java 21

El lenguaje de programación base, utilizado en su versión **LTS** para garantizar la estabilidad y el soporte a largo plazo.

Maven

Herramienta esencial para la gestión de dependencias y el ciclo de vida de la construcción del proyecto.

Herramientas de Documentación y Prueba

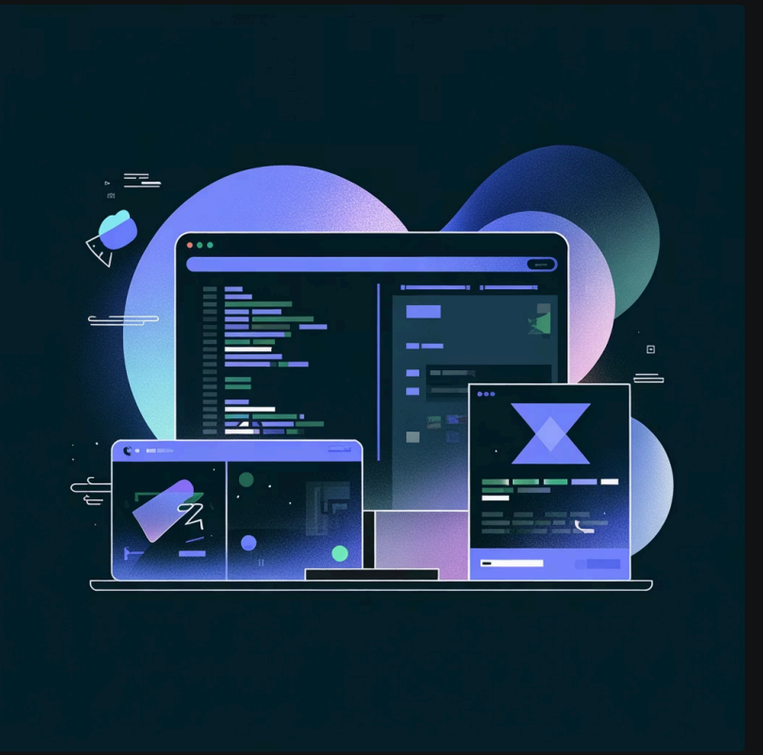
La profesionalidad de una API no reside solo en su código, sino en cómo se interactúa y se prueba.

Swagger (Springdoc OpenAPI)

Facilita la documentación automática y la **prueba en vivo** de todos los endpoints REST directamente desde el navegador, mejorando la colaboración.

Postman

Aplicación de escritorio fundamental para el desarrollo y el **testing exhaustivo** de las peticiones HTTP (GET, POST, PUT, DELETE).



Git & GitHub

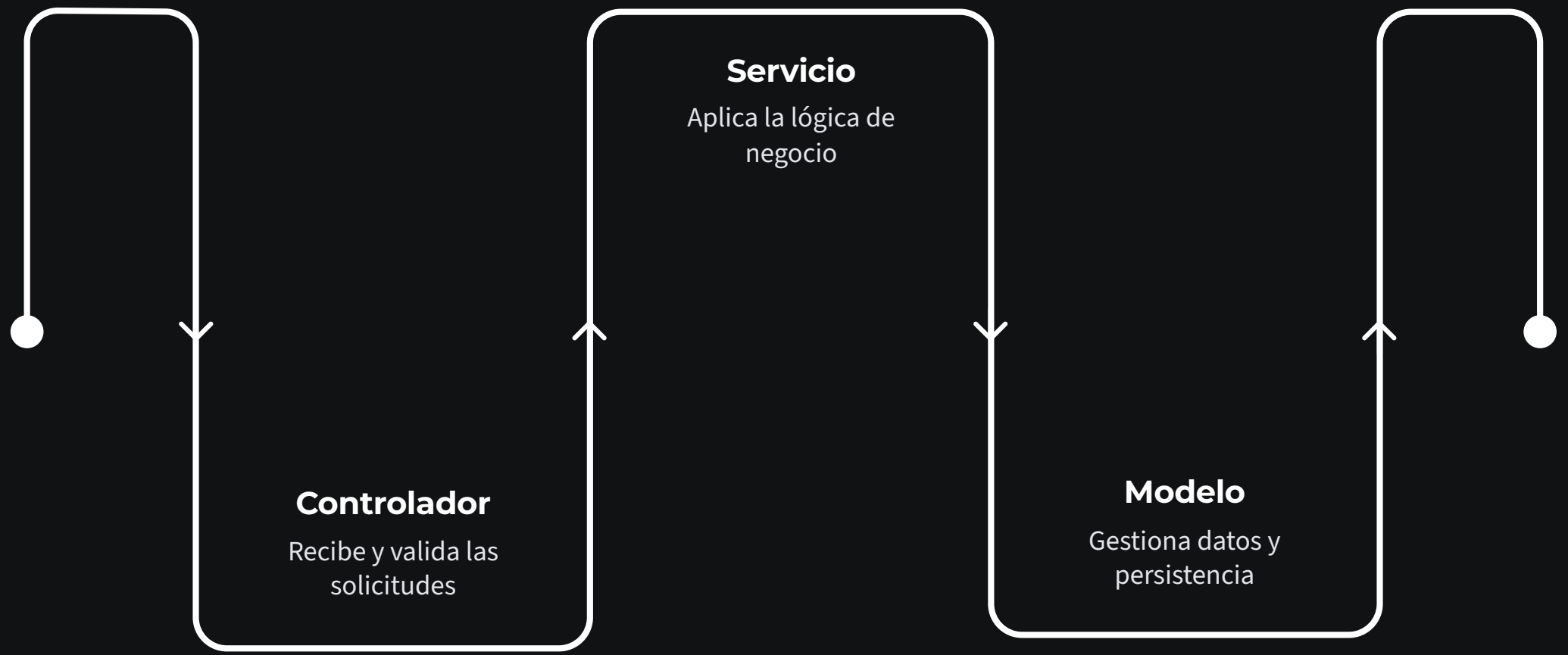
Control de versiones distribuido, indispensable para el trabajo en equipo, trazabilidad de cambios y despliegue continuo.

Lombok

para generar getters, setters, constructores y builders automáticamente.

3. Estructura de Proyecto en Capas

Nuestro proyecto adopta la arquitectura de tres capas (Model-View-Controller, simplificado a M-C-S en el backend) para una clara separación de responsabilidades, facilitando el mantenimiento y la escalabilidad del código.



Controller

`com.empresa.soporte_tecnico.controller/` — Recibe y gestiona las peticiones HTTP (endpoints REST).

Service

`com.empresa.soporte_tecnico.service/` — Contiene la lógica de negocio pura (la implementación del CRUD).

Model

`com.empresa.soporte_tecnico.model/` — Define la estructura de los datos (la entidad `Solicitud.java`).

4. Clases Esenciales del Sistema

La base del sistema reside en tres componentes clave que interactúan para procesar cada solicitud de soporte técnico.

Solicitud.java (Modelo)

Clase POJO (Plain Old Java Object) que define la estructura de datos que se intercambia en el sistema. Representa una única solicitud de soporte.

- Atributos: ID único, Título del problema, Descripción detallada.
- Implementa: Métodos **Getters y Setters** para manipulación segura de datos.

SolicitudController.java (Controlador)

Componente que mapea las URLs a las funciones de negocio. Es la **puerta de entrada** a nuestra API REST.

- Utiliza anotaciones como `@RestController` y `@RequestMapping`.
- Actúa como intermediario entre el cliente (navegador/Postman) y la capa de servicio.

Mapeo de Endpoints CRUD

El controlador expone cinco endpoints RESTful que permiten la gestión completa del recurso "Solicitud", siguiendo los verbos HTTP estándar.

`POST`	`/solicitudes`	Crear una nueva solicitud de soporte técnico (Create)
`GET`	`/solicitudes`	Listar todas las solicitudes existentes (Read - All)
`GET`	`/solicitudes/{id}`	Mostrar detalles de una solicitud específica por ID (Read - Single)
`PUT`	`/solicitudes/{id}`	Actualizar completamente una solicitud existente (Update)
`DELETE`	`/solicitudes/{id}`	Eliminar la solicitud del sistema (Delete)

Lógica de Servicio (SolicitudServiceImpl.java)

Este componente implementa los métodos de CRUD utilizando una **colección en memoria** para simular la persistencia de datos (sin necesidad de configurar una base de datos real en esta etapa del proyecto).

5. Ejecución y Pruebas del Servicio

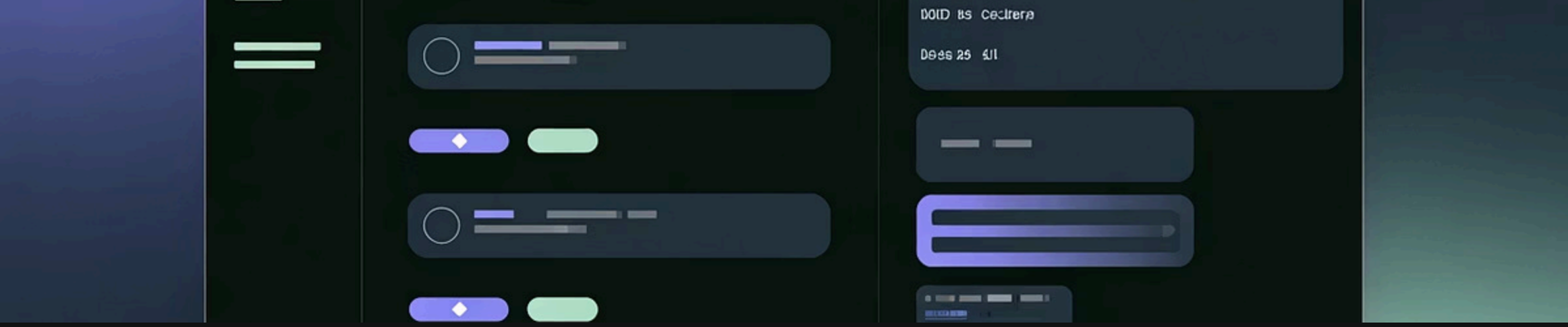
El proyecto está diseñado para una puesta en marcha rápida, tanto desde un entorno de desarrollo integrado (IDE) como desde la línea de comandos, gracias a la gestión de Maven y Spring Boot.

Ejecución con IntelliJ IDEA

1. Abrir el proyecto seleccionando el archivo `pom.xml`.
2. Esperar la descarga automática de dependencias por parte de Maven.
3. Ejecutar la clase principal `@SpringBootApplication`.
4. Acceder a la API base en **http://localhost:8080/solicitudes**.

Ejecución desde Terminal

```
mvn clean install
mvn spring-boot:run
```



6. Documentación Automática con Swagger UI

Una de las mayores ventajas de usar **Springdoc OpenAPI** es la generación automática de una interfaz interactiva de documentación inmediatamente después de la ejecución.

> Acceda a la interfaz de prueba en:
>
> **http://localhost:8080/swagger-ui/index.html**

Esta herramienta permite a los desarrolladores y testers **ejecutar y validar** cada endpoint (POST, GET, PUT, DELETE) sin necesidad de herramientas externas, sirviendo como una interfaz gráfica directa para la API.

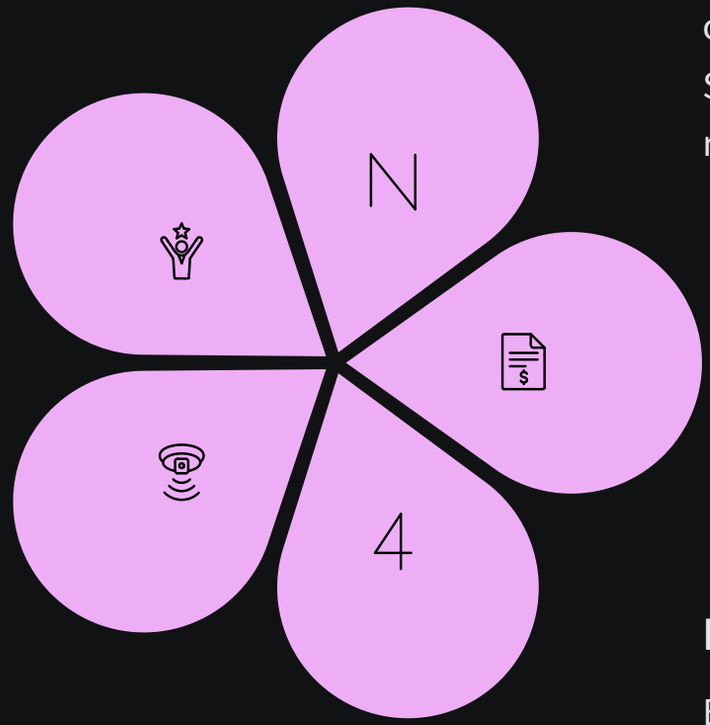
Resultados y Conclusiones del Proyecto

API Funcional

El proyecto demuestra la implementación exitosa de las operaciones CRUD bajo el patrón RESTful.

Herramientas

Dominio del ecosistema de desarrollo Java (Maven, Git, Spring Boot) para despliegue rápido.



Arquitectura Clara

Se verificó la importancia de la organización en capas (Controller, Service, Model) para el desarrollo robusto.

Documentación

Integración fluida de Swagger, validando la documentación automática como estándar profesional.

Escalabilidad

Base sólida para proyectos empresariales escalables, listos para integrarse con una base de datos real.

Proyecto desarrollado como parte del curso de **Desarrollo de los Componentes del Negocio** — IDAT 2025.