|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | **ACTIVIDADES**  **FACULTAD DE INGENIERÍA**  **PROGRAMA:** INGENIERÍA INFORMÁTICA | | | |
| **ASIGNATURA** | **CÓDIGO:** ING01206 | | **NOMBRE:** PROGRAMACION DISTRIBUIDA Y PARALELA | | |
| **PROFESOR:** | HERNANDO RECAMAN CHAUX | | | **FECHA:** 16/09/2025 |  |
| **Estudiante** | **NOMBRES Y APELLIDOS** | | | **CÉDULA** | **ACTIVIDAD #** |
| DIEGO ALEJANDRO MEJIA GIRALDO | | | 1007418004 | **2** |
| RAFAEL ESTIVEN URIBE ÁLVAREZ | | | 1039465990 |

**Tabla de contenidos**

[1. ELEMENTOS DE LA SOLUCIÓN. 1](#_Toc208770316)

[1.1. Descripción del problema. 1](#_Toc208770317)

[1.2. Objetivos del sistema 2](#_Toc208770318)

[1.3. Requerimientos del sistema 2](#_Toc208770319)

[1.4. Arquitectura en capas 2](#_Toc208770320)

[2. DIAGRAMAS DE LA SOLUCIÓN. 3](#_Toc208770321)

[3. RESULTADO E IMPLEMENTACIÓN DE LA SOLUCIÓN 4](#_Toc208770322)

[4. REFERENCIAS 5](#_Toc208770323)

# ELEMENTOS DE LA SOLUCIÓN.

## Descripción del problema.

La Granja S.A maneja los procesos de registro de información sobre porcinos, clientes y alimentación de manera manual a través de hojas de Excel, lo que genera dificultades en:

* Control de información.
* generación de informes
* Acceso simultaneo a los datos
* Fiabilidad y seguridad de los registros

El administrador quiere una aplicación web que implemente un modelo en capas que centralice la información.

## Objetivos del sistema

### Objetivo general:

Desarrollar una solución web que permita la gestión de la información relacionada con porcinos, clientes y alimentación, optimizando la consulta, almacenamiento y generación de reportes

### Objetivos específicos:

1. Remplazar los registros en Excel por una base de datos estructurada
2. Permitir el registro, actualización, eliminación y consulta de información
3. Implementar un modelo de capas para garantizar escalabilidad, mantenibilidad y modularidad

## Requerimientos del sistema

Gestión de porcinos: Registrar, actualizar, consultar y eliminar

* identificación
* Raza
* Edad
* Peso
* Alimentación
* Cliente asociado

Gestión de clientes: Registrar, actualizar, consultar y eliminar

* *Cedula*
* *Nombres*
* *Apellidos*
* *Dirección*
* *Teléfono*

Alimentación: Registrar, actualizar, consultar y eliminar

* Tipo
* Descripción
* Dosis

Generar informe de clientes y porcinos

## Arquitectura en capas

### Capa de presentación

Angular: Formularios de registro, tablas y generación de reportes

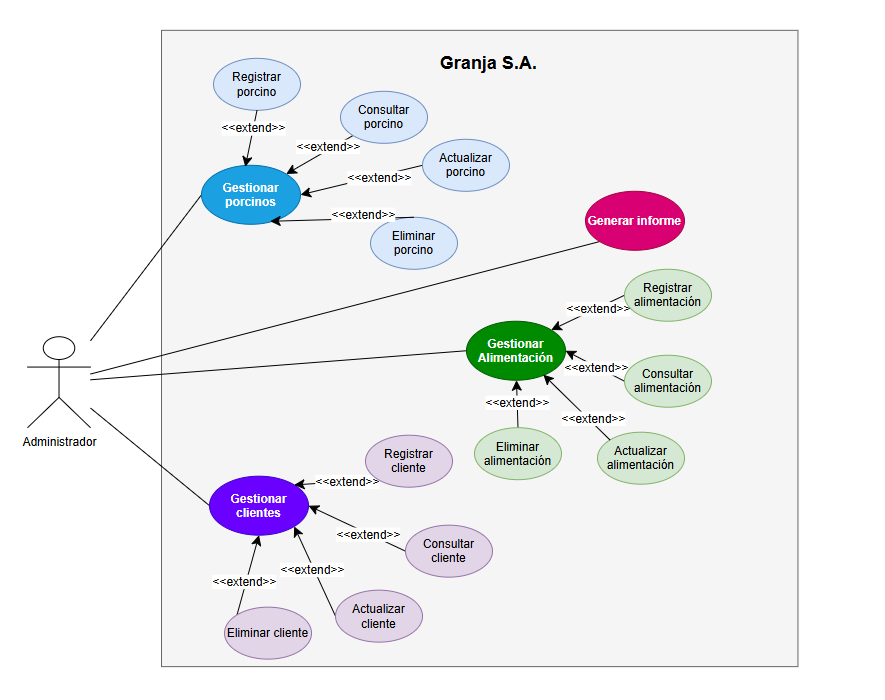
### Capa de negocio

SpringBoot: Validaciones, reglas de negocio y procesamiento de datos antes de enviarlo a la capa de datos

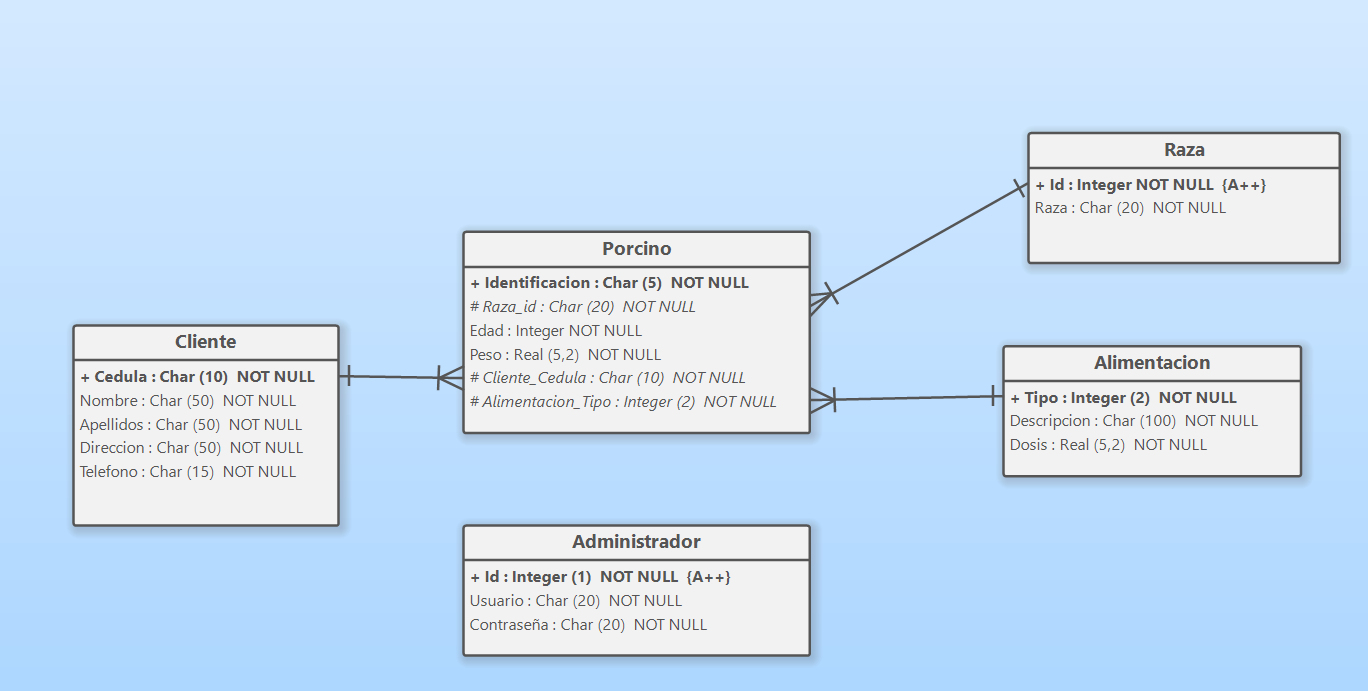
### Capa de datos

PostgreSQL: para la persistencia de datos

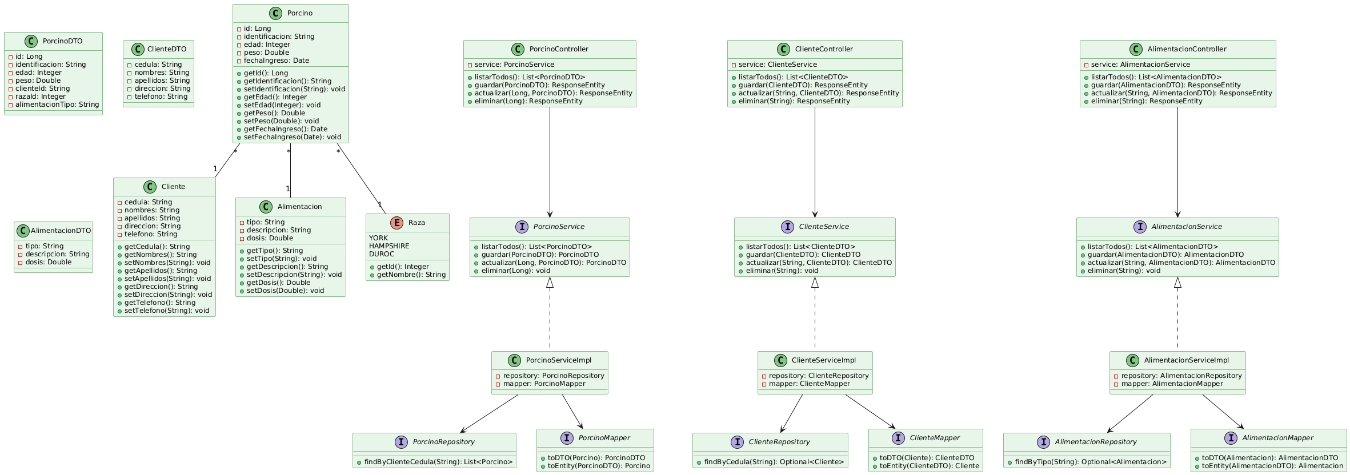
# DIAGRAMAS DE LA SOLUCIÓN.



*Fig 1.* Caso de usos de la solución [Fuente: elaboración propia]



*Fig 2.* Diagrama relacional de la solución [Fuente: elaboración propia]



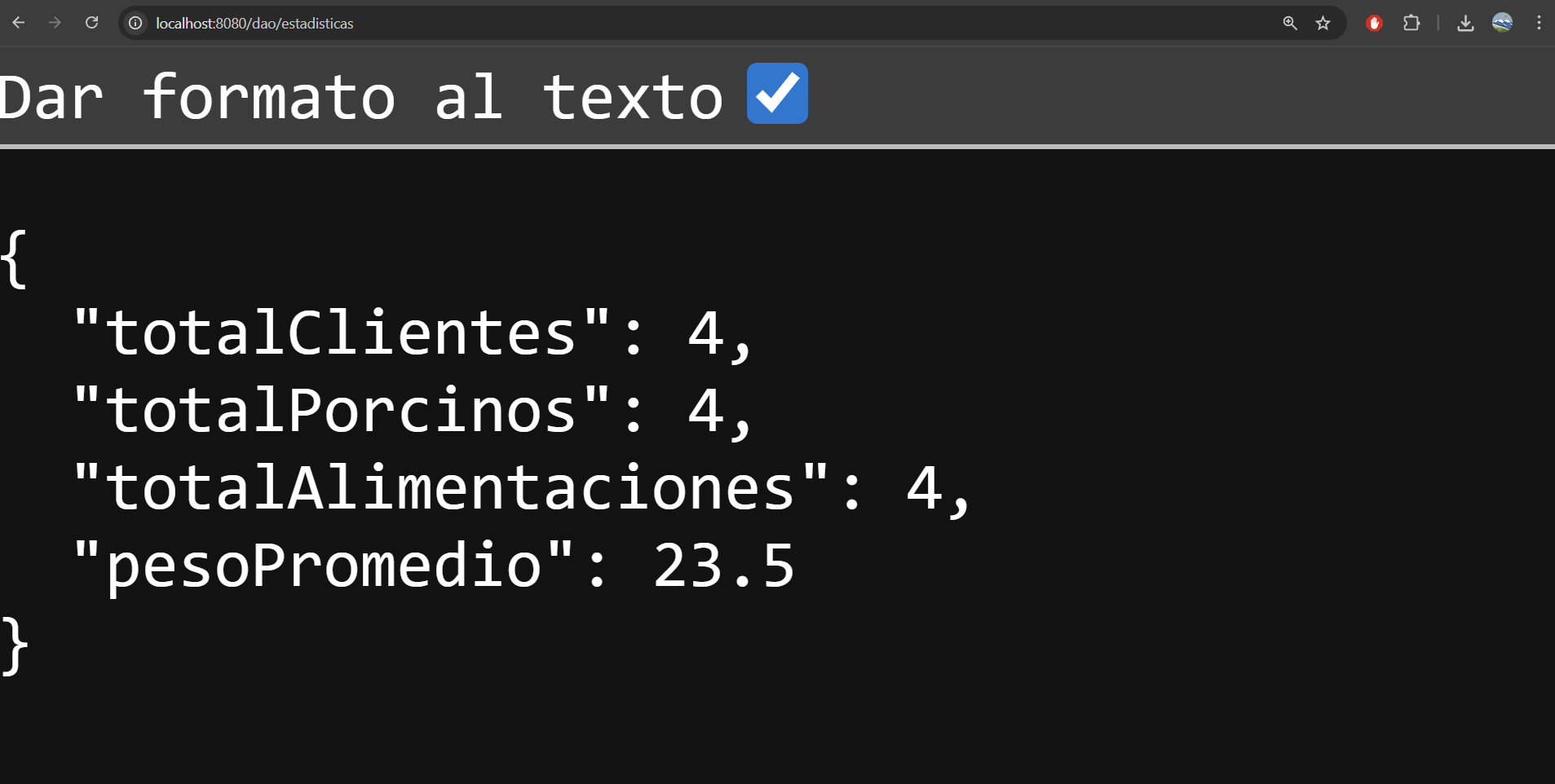
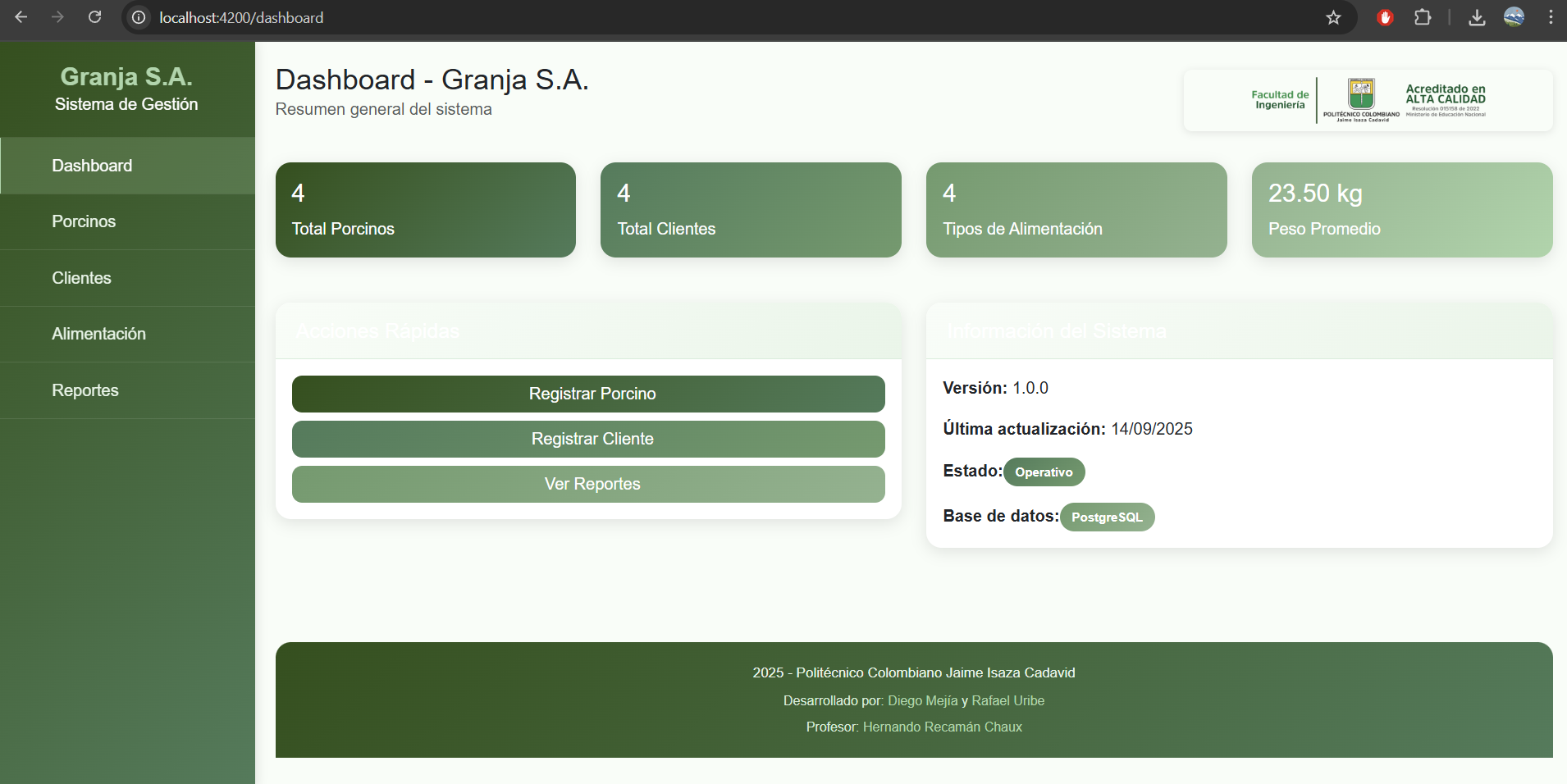
*Fig 3.* Diagrama de clases completo [Fuente: elaboración propia]

# RESULTADO E IMPLEMENTACIÓN DE LA SOLUCIÓN

La solución presenta una arquitectura en capas para la gestión porcina, implementando un sistema distribuido con separación clara de responsabilidades. La estructura del código sigue, también, el patrón de microservicios, con un backend en Spring Boot expuesto en el puerto 8080 y un frontend en Angular 17 en el puerto 4200. La arquitectura implementa:

* Clean Architecture con separación por capas
* Domain-Driven Design (DDD) en la organización de entidades
* SOLID principles en la implementación de servicios
* RESTful API con documentación Postman integrada
* Gestión de estados mediante DTOs y mappers
* Sistemas de reportes con capacidades de exportación.

[**Ver el código aquí**](https://github.com/DiegoAM13/Granja-S.A.).

*Fig 4.* Springboot (Izquierda) y Angular (derecha) de la solución [Fuente: elaboración propia]

Hibernate representa una implementación de la especificación JPA (Java Persistence API) que revoluciona la forma en que las aplicaciones Java interactúan con bases de datos relacionales. Como framework ORM (Object-Relational Mapping) líder en la industria, Hibernate abstrae la complejidad de la persistencia de datos mediante un sofisticado sistema de mapeo objeto-relacional.

**Características Innovadoras**

**1. Persistence Context**

@PersistenceContext

private EntityManager entityManager;

El contexto de persistencia actúa como una "caché de primer nivel" inteligente, optimizando las operaciones de base de datos y garantizando la consistencia de los objetos.

**2. Ciclo de Vida Sofisticado**

1. **Transient**: Objetos recién creados
2. **Managed**: Objetos gestionados por el EntityManager
3. **Detached**: Objetos desvinculados de la sesión
4. **Removed**: Objetos marcados para eliminación

# REFERENCIAS

* Young, G. (2021). Event Centric: Finding Simplicity in Complex Systems. Addison-Wesley Professional.
* Axon Framework. (2024). Event Sourcing and CQRS with Axon Framework. https://docs.axoniq.io/reference-guide/
* Microsoft. (2023). Event Sourcing pattern - Cloud Design Patterns. https://learn.microsoft.com/en-us/azure/architecture/patterns/event-sourcing