



## PROBLEMÁTICA

La farmacia Econofarma presenta diversos problemas:

- Gestión de inventario y ventas
- Procesos administrativos
- Procedimientos manuales
- Errores humanos en los registros

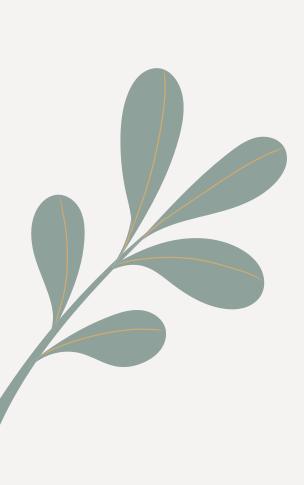
Todo esto afecta la eficiencia interna, así como impacta la experiencia del cliente y la rentabilidad del negocio.

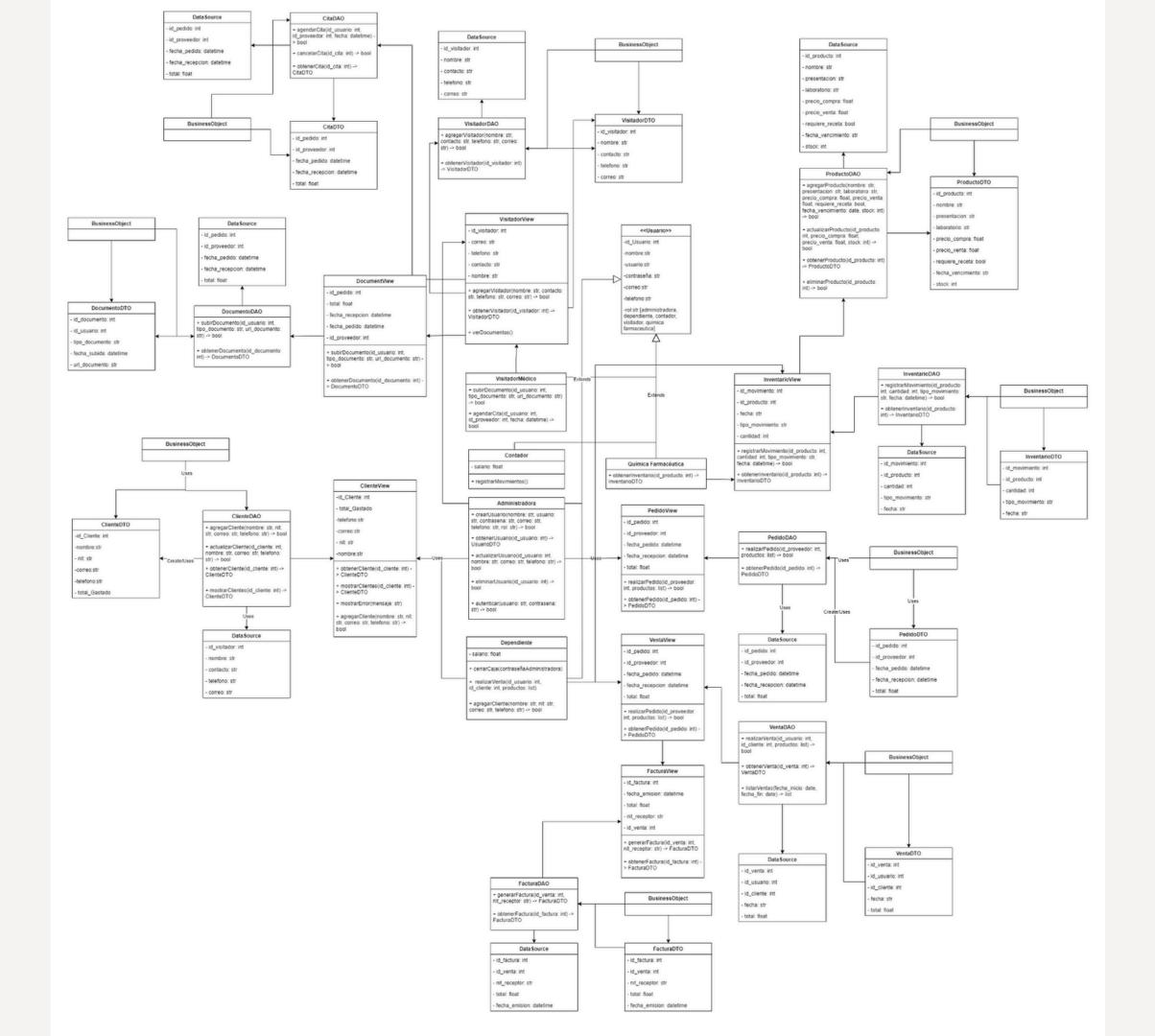


## DIAGRAMA DE CLASES



Se tomaron en cuenta las respectivas clases. Además, se considera la implementación del patrón DAO y DTO.







## DIAGRAMA DE CLASES PERSISTENTES

#### <<ORM Persistable>> Inventario

id int [pk, not null]

producto\_id int [not null]

cantidad int [not null]

tipo\_movimiento\_id int [not null]

venta\_id int

compra\_id int

precio\_venta numeric [not null]

precio\_costo numeric [not null]

local\_id int [not null]

encargado\_id int [not null]

fecha datetime [not null]

- RegistrarEntradaInventario(producto\_id, cantidad, precio\_costo, local\_id, encargado\_id, fecha)
- RegistrarSalidaInventario(producto\_id, cantidad, precio\_venta, local\_id, encargado\_id, fecha, venta\_id)
- AjustarInventario(id, cantidad, motivo)
  - ObtenerInventarioPorProducto(producto\_id)
    - ObtenerInventarioPorLocal(local\_id)
      - ListarInventario()

#### <<ORM Persistable>> Usuarios

id int [pk, not null]
nombre varchar [not null]
apellidos varchar [not null]
rol\_id int [not null]
email varchar [not null]
status enum('activo', 'inactivo') [not null]
local int [not null]
contraseña varchar [not null] //encriptada supongo
fechaNacimiento date [not null]
creacion datetime [not null]

CrearUsuario(nombre, apellidos, rol\_id, email, status, local, contraseña, fechaNacimiento)

ActualizarUsuario(id, nombre, apellidos, rol\_id, email, status, local, contraseña, fechaNacimiento)

echanacimiento)

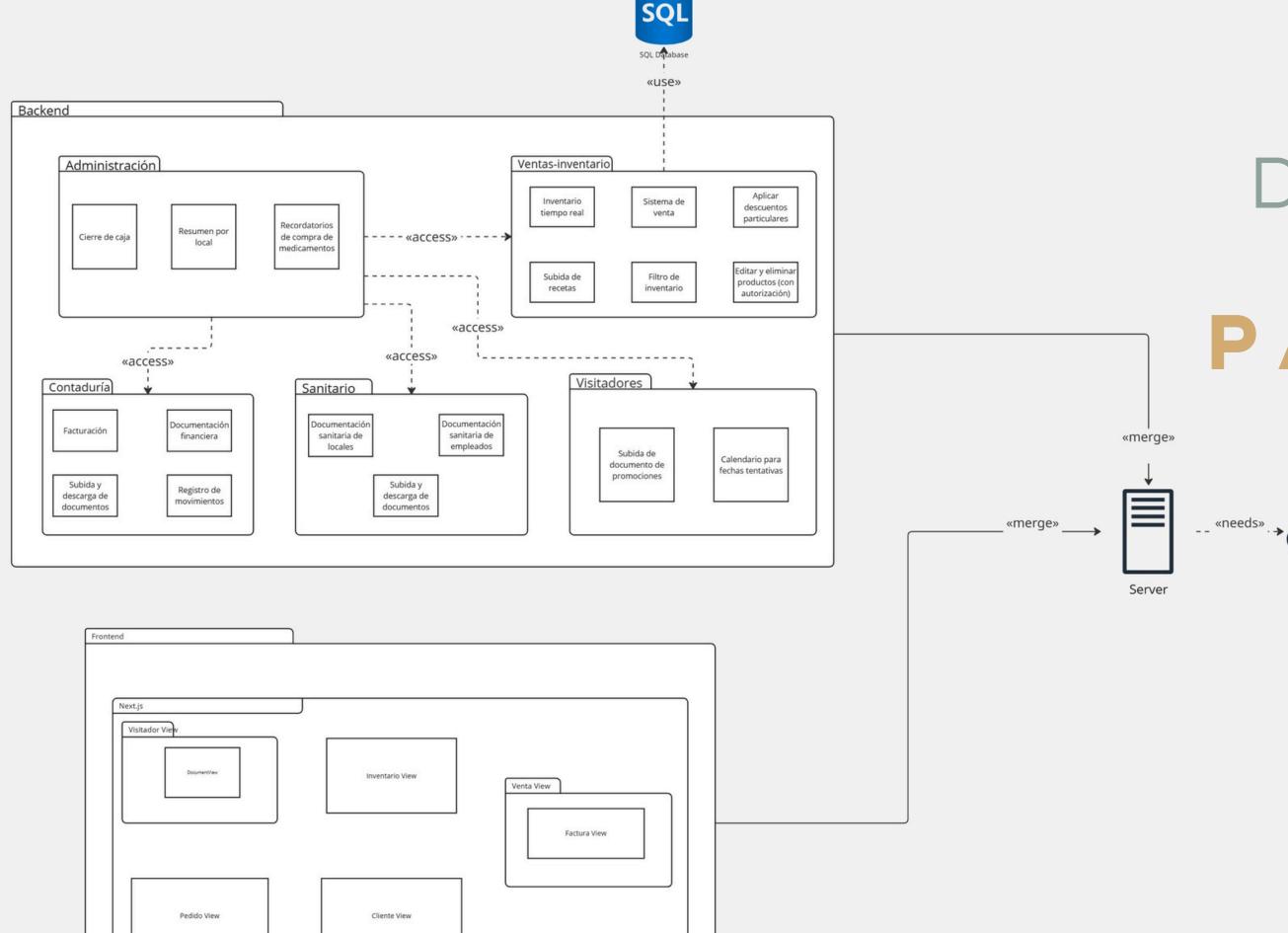
DesactivarUsuario(id)

ActivarUsuario(id)

ObtenerUsuario(id)

ListarUsuarios()

CambiarContraseña(id, nuevaContraseña)



## DIAGRAMA DE PAQUETES



## OPCIONES DE

## BACKEND



NestJs Javascript



.Net C#



**Express**Javascript



Node.js Javascript/TypeScript



Deno TypeScript



Fresh TypeScript

#### CUADRO COMPARATIVO

## BACKEND

| Deno  |  |
|-------|--|
| Alta  |  |
| Alta  |  |
| Alta  |  |
| Media |  |

|                   |               | Node.js | .Net | Nest  | Express | Fresh | Deno  |
|-------------------|---------------|---------|------|-------|---------|-------|-------|
| Curva Aprendizaje |               | Media   | Alta | Media | Baja    | Media | Alta  |
|                   | Escalabilidad | Alta    | Alta | Alta  | Media   | Media | Alta  |
|                   | Seguridad     | Media   | Alta | Alta  | Media   | Alta  | Alta  |
|                   | Uso Recursos  | Baja    | Alta | Baja  | Baja    | Baja  | Media |
|                   | Migración     | Alta    | Baja | Media | Alta    | Baja  | Baja  |
|                   | Modularidad   | Media   | Alta | Alta  | Media   | Alta  | Media |
|                   | Costo         | Baja    | Alta | Baja  | Bajo    | Baja  | Baja  |



# TECNOLOGÍA BACKEND SELECCIONADA







#### Ventajas

- Código abierto
- Gran comunidad
- Escalabilidad
- Ideal para apps en tiempo real
- Eficiente en recursos

#### Desventajas

- No tipado por defecto (a menos que se use TypeScript),
- Menor seguridad que .NET
- Dificultad en debugging en proyectos grandes.

## OPCIONES DE FRONTEND



Vue.js Javascript



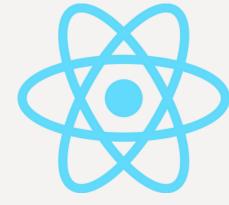
Angular TypeScript



Next.js Javascript TypeScript



Vite Javascript TypeScript



React Javascript

#### CUADRO COMPARATIVO

## FRONTEND

| React |
|-------|

|                   |               | Next.js | Vite  | Vue.js | Angular | React |
|-------------------|---------------|---------|-------|--------|---------|-------|
| Curva Aprendizaje |               | Media   | Baja  | Baja   | Alta    | Media |
|                   | Escalabilidad | Alta    | Media | Media  | Alta    | Alta  |
|                   | Seguridad     | Alta    | Media | Media  | Alta    | Media |
|                   | Uso Recursos  | Media   | Baja  | Baja   | Alta    | Media |
|                   | Migración     | Alta    | Alta  | Media  | Baja    | Alta  |
|                   | Modularidad   | Alta    | Media | Alta   | Alta    | Alta  |
|                   | Costo         | Baja    | Baja  | Baja   | Alta    | Baja  |



## TECNOLOGÍA FRONTEND SELECCIONADA



#### Ventajas

- Soporte para SSR y SSG
- Rendimiento optimizado
- Escalabilidad alta
- Comunidad grande.

### Desventajas

- Mayor complejidad en configuración
- Requiere más backend.

### OPCIONES DE

## PERSISTENCIA



SQLite Relacional

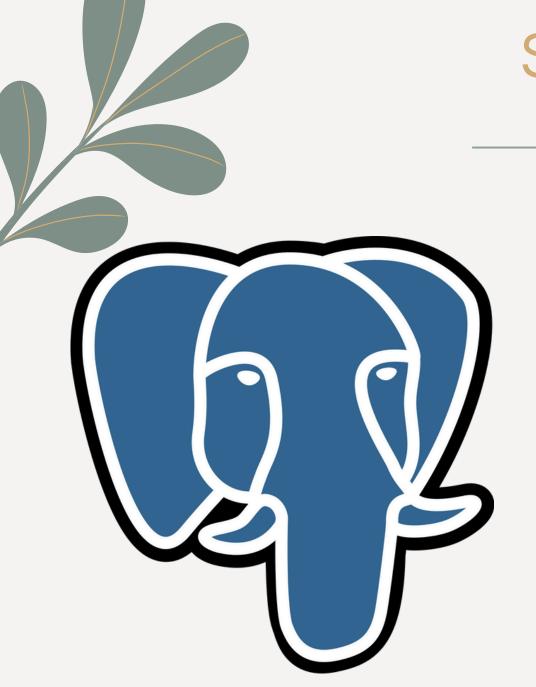






## CUADRO COMPARATIVO PERSISTENCIA

|               | MySQL      | PostgreSQL | MongoDB    | SQLite    |
|---------------|------------|------------|------------|-----------|
|               | Horizontal | Horizontal |            |           |
| Escalabilidad | Vertical   | Vertical   | Horizontal | Limitada  |
| Rendimiento   | Alto       | Alto       | Alto       | Baja      |
| Integridad de | Llaves     | Llaves     | No llaves  | No llaves |
| datos         | foráneas   | foráneas   | foráneas   | foráneas  |
| Costo         | Medio      | Medio      | Bajo       | Bajo      |



## TECNOLOGÍA PERSISTENCIA SELECCIONADA



### Ventajas

- Altamente extensible y flexible.
- Soporte completo de ACID.
- Ideal para consultas complejas.

### Desventajas

- Rendimiento más lento en operaciones simples que MySQL.
- Requiere más recursos de hardware.

