INGENIERÍA EN SISTEMAS DE INFORMACIÓN

ASIGNATURA:
Seminario Integrador

Proyecto tienda Magallanes

Docente:

- Ing. Martin Cordoba.

Integrantes del grupo:

- Gerbaudo Mateo (mgerbaudo02@gmail.com)

1. Descripción **4**
2. Objetivo del Proyecto **4**
3. Alcance del Sistema **4**
3.1 Incluye 4
3.2 No incluye 4
3.3 Restricciones 5
4. Tecnologías utilizadas **5**
4.1 Backend **5**
4.2 Frontend **5**
5. Requisitos Funcionales **6**
6. Requisitos No Funcionales (RNF) **8**
7. Diagrama de Clases **10**
7. Diagrama de Caso de Uso **11**
8. Descripción CU N°11 - Crear Órdenes 13
8.1 Diagrama de Secuencia CU N°07 **14**
9. Vista Arquitectónica de Subsistemas/Interfaces **15**
10. API REST - Endpoints del Sistema **16**
11. Diagrama de Entidad y Relación (DER) **20**
12. Backend 21
13.1 Estructura 21
13. Frontend 22
13.1 Estructura 22
13.2 Pantallas 22
14. Reglas de Negocio 26
RN-001 – Gestión de Empleados 26
RN-002 – Control de Stock por Talle 26
RN-003 – Cálculo del Precio Total de la Orden 26
RN-004 – Validaciones de Datos 26
RN-005 – Formas de Pago 27
RN-006 – Movimientos de Stock 27
RN-007 – Consultas y Reportes 27
RN-008 – Gestión de Órdenes 28
RN-009 – Seguridad y Autenticación 28
15. Casos de Prueba 28
Documentación de Testing 28
16. Documentación Adicional 29
Instrucciones para ejecutar el proyecto localmente 29
1. Preparar la base de datos **29**
2. Configurar el proyecto 29
3. Construir el proyecto 29
4. Ejecutar la aplicación 30
5. Probar endpoints 30

6. Notas adicionales 30
17. Repositorio 30

Magallanes – Sistema de Gestión de Tienda

1. Descripción

Magallanes es una aplicación web para la **gestión de productos, stock y órdenes de compra**.
Permite registrar productos con talles, controlar el inventario automáticamente al generar órdenes, y facilitar la administración de ventas de manera simple y organizada.

2. Objetivo del Proyecto

El objetivo del proyecto **Magallanes** es realizar el **análisis, diseño y desarrollo** de una aplicación destinada al **registro de ventas**, permitiendo administrar los productos disponibles, controlar el stock por talle y generar órdenes de venta de manera organizada y eficiente.

3. Alcance del Sistema

El sistema **Magallanes** tiene como propósito principal registrar y gestionar las ventas realizadas en la tienda, manteniendo actualizado el stock de productos diferenciados por talles.

3.1 Incluye

- Registro de productos con nombre, descripción, precio y stock por talle.
- Registro de ventas (órdenes) asociadas a productos.
- Actualización automática del stock al confirmarse una venta.
- Consulta de órdenes registradas y detalle de cada una.
- Reporte básico de ventas realizadas.
- Gestión de usuarios con **roles de Administrador y Empleado**:
    - El **Administrador** puede crear y actualizar empleados.
    - El **Empleado** solo puede registrar ventas.

3.2 No incluye

- Gestión de compras a proveedores.
- Control de cuentas corrientes de clientes.
- Integración con sistemas de pago electrónicos o facturación oficial.

- Gestión avanzada de permisos más allá de los roles básicos de Administrador y Empleado.

3.3 Restricciones

- El sistema está limitado a un único punto de venta (no multi-sucursal).
- El acceso se realiza únicamente mediante navegador web desde la red local.
- No contempla escalabilidad a gran volumen de transacciones (uso orientado a pequeña o mediana tienda).

4. Tecnologías utilizadas

4.1 Backend

- **Java 17**: Lenguaje de programación utilizado para la implementación de la aplicación.
- **Spring Boot 3.3.13**: Framework principal para el desarrollo de aplicaciones Java.
- **Spring Data JPA**: Para la gestión de la persistencia de datos en la base de datos.
- **Spring Security**: Para la autenticación de usuarios y control de roles (Empleado / Administrador).
- **JSON Web Token (JWT)**: Para la implementación de tokens de autenticación y seguridad en la comunicación.
- **Hibernate Validator**: Para la validación de datos de entrada.
- **MySQL**: Sistema de gestión de bases de datos relacional.
- **Spring Boot Actuator**: Para monitoreo y métricas de la aplicación.
- **OWASP Encoder**: Para codificación segura de datos y protección contra vulnerabilidades.
- **Lombok**: Para generar automáticamente getters, setters, constructores y otros métodos comunes.
- **JUnit & Mockito**: Para pruebas unitarias y creación de mocks en tests.
- **DTOs y mapeo manual**: Uso de Data Transfer Objects (DTO) para separar la capa de persistencia de la lógica de negocio y del frontend.

4.2 Frontend

- **HTML**: Estructura de las páginas web.
- **CSS**: Estilo y presentación visual.

- **JavaScript**: Comportamiento e interactividad de la interfaz.
- **Bootstrap**: Framework CSS para diseño responsivo y componentes visuales.
- **AJAX / Fetch API**: Para la comunicación con el backend y manejo dinámico de datos.

5. Requisitos Funcionales

1. **Gestión de Usuarios (Empleados y Administradores)**
    - RF1.1: El sistema debe permitir a los administradores **crear nuevos empleados** con datos personales y rol asignado.
    - RF1.2: El sistema debe permitir a los administradores **actualizar los datos de un empleado** existente, incluyendo su rol.
    - RF1.3: El sistema debe permitir a los administradores **eliminar empleados** del sistema.
    - RF1.4: El sistema debe permitir a cualquier usuario autenticado **consultar la lista de empleados** y **buscar por nombre, DNI o ID**.
    - RF1.5: Los empleados con rol estándar no pueden crear ni actualizar empleados; solo los administradores tienen ese permiso.
2. **Autenticación y Seguridad**
    - RF2.1: El sistema debe permitir que los usuarios se **autentiquen mediante nombre de usuario y contraseña**.
    - RF2.2: El sistema debe generar un **token JWT** para mantener la sesión del usuario.
    - RF2.3: El sistema debe **proteger los endpoints sensibles** (crear, actualizar, eliminar) para que solo usuarios autorizados puedan ejecutarlos.
3. **Gestión de Clientes**
    - RF3.1: El sistema debe permitir **crear nuevos clientes** con datos personales.
    - RF3.2: El sistema debe permitir **actualizar los datos de un cliente existente**.
    - RF3.3: El sistema debe permitir **eliminar clientes**.
    - RF3.4: El sistema debe permitir **buscar clientes por nombre, DNI o ID**.
    - RF3.5: El sistema debe mostrar la **lista completa de clientes**.
4. **Gestión de Productos y Tipos de Prenda**

- RF4.1: El sistema debe permitir **crear productos** con nombre, precio, color, marca, tipo de prenda y stock por talle.
- RF4.2: El sistema debe permitir **actualizar productos existentes**, incluyendo stock y movimientos asociados.
- RF4.3: El sistema debe permitir **eliminar productos**.
- RF4.4: El sistema debe permitir **consultar productos** por nombre, marca, color o ID.
- RF4.5: El sistema debe permitir **gestionar tipos de prendas** (crear, actualizar, eliminar, listar y buscar por nombre).

5. **Gestión de Órdenes**
   - RF5.1: El sistema debe permitir **crear órdenes de venta**, asociando cliente, empleado, forma de pago y productos con talles y cantidades.
   - RF5.2: El sistema debe **verificar stock disponible** antes de confirmar la orden.
   - RF5.3: El sistema debe **reducir automáticamente el stock** de los productos vendidos.
   - RF5.4: El sistema debe **registrar los movimientos de stock** (entrada y salida).
   - RF5.5: El sistema debe permitir **actualizar órdenes existentes**, incluyendo productos, cantidades y talles.
   - RF5.6: El sistema debe permitir **eliminar órdenes**.
   - RF5.7: El sistema debe permitir **consultar órdenes** por cliente, empleado, rango de fechas y rango de precio.
   - RF5.8: El sistema debe calcular automáticamente el **precio total de la orden**.
   - RF5.9: El sistema debe generar estadísticas de ventas, incluyendo **productos más vendidos** y **formas de pago utilizadas**.

6. **Gestión de Formas de Pago**
   - RF6.1: El sistema debe **cargar automáticamente las formas de pago iniciales**, que incluyen: **efectivo, tarjeta de crédito y tarjeta de débito**.
   - RF6.2: Al crear o actualizar una orden, el usuario **solo puede elegir una de las formas de pago existentes** en el sistema.
   - RF6.3: El sistema debe permitir **consultar las formas de pago disponibles**, pero **no permite crear, modificar ni eliminar formas de pago**.

6. Requisitos No Funcionales (RNF)

RNF-01 – Rendimiento

El sistema debe responder consultas de lectura y escritura a la base de datos en menos de 2 segundos bajo carga normal (hasta 50 usuarios concurrentes). Las operaciones críticas como creación de órdenes o actualización de stock deben completarse en menos de 1 segundo.

RNF-02 – Seguridad

Todas las contraseñas deben almacenarse en la base de datos MySQL de forma cifrada usando BCrypt.
Los endpoints de creación, actualización y eliminación sólo deben ser accesibles por usuarios autenticados con el rol correspondiente.
La comunicación entre cliente y servidor debe usar HTTPS para proteger datos sensibles.

RNF-03 – Disponibilidad

El sistema debe estar disponible al menos un 99% del tiempo durante los horarios de operación (8:00 a 22:00). Los fallos de MySQL o del servidor web deben registrarse y enviar alertas al administrador.

RNF-04 – Usabilidad

La interfaz web debe ser responsiva y compatible con dispositivos de escritorio, tablet y móvil.
Las pantallas principales (productos, órdenes, clientes) deben cargarse en menos de 1 segundo.
Los formularios deben tener validaciones inmediatas y mensajes claros para errores de datos (DNI duplicado, stock negativo, etc.).

RNF-05 – Mantenibilidad

El código debe estar modular y documentado para facilitar futuras mejoras.
Se debe usar control de versiones (Git) y buenas prácticas de commits descriptivos.
La base de datos MySQL debe incluir scripts de creación y actualización de tablas, y las migraciones deben ser controladas mediante herramientas como Flyway o Liquibase.

RNF-06 – Escalabilidad

La arquitectura del sistema debe permitir agregar más tablas y funcionalidades sin afectar la estabilidad del sistema.
Se debe poder migrar la base de datos MySQL a un servidor remoto sin modificar la lógica del backend.

RNF-07 – Integridad de Datos

Todas las transacciones que afectan stock, órdenes o clientes deben realizarse de manera atómica en MySQL para evitar inconsistencias.
Se deben registrar logs de cambios en la base de datos para auditoría.

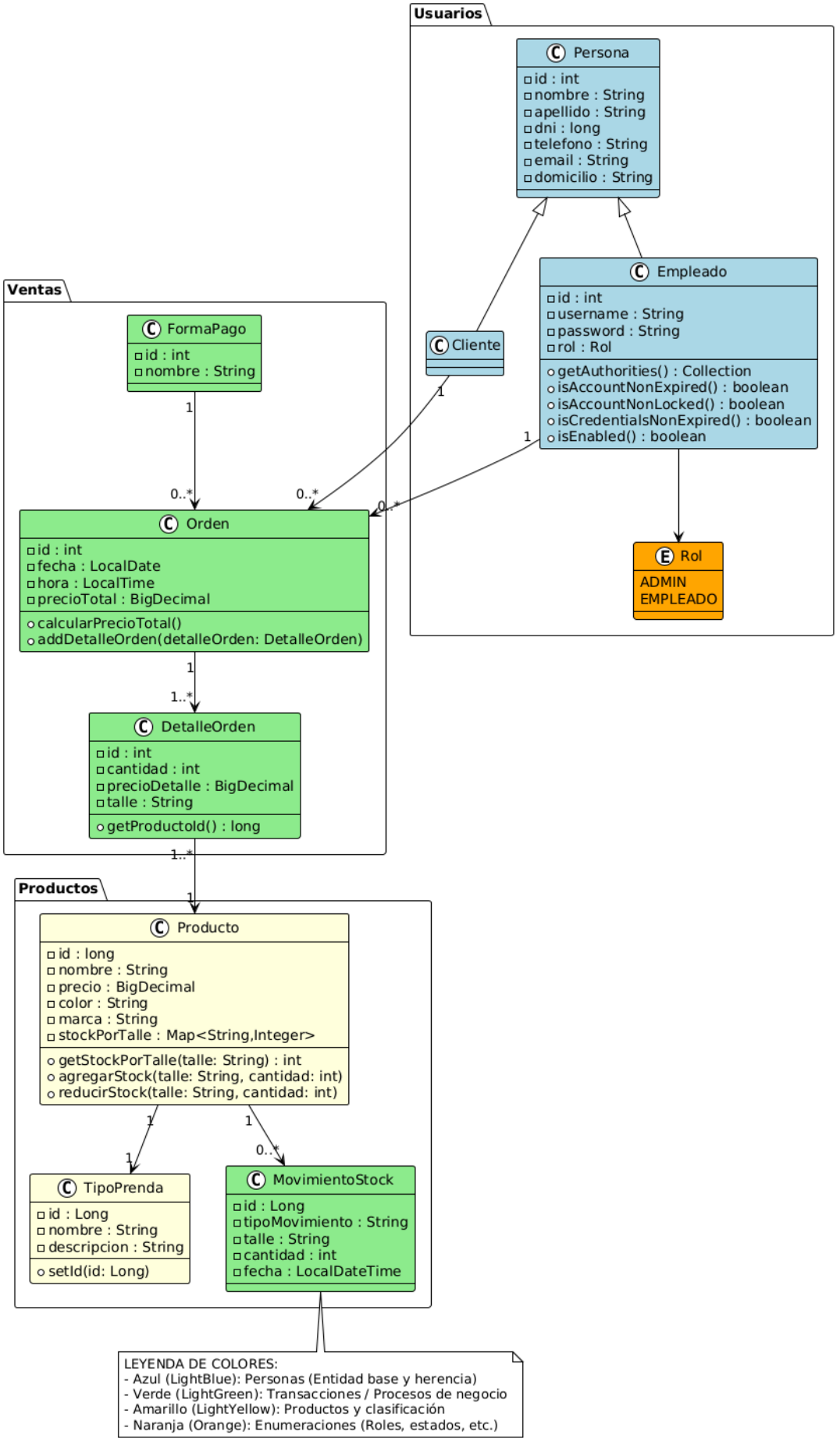
RNF-08 – Portabilidad

El sistema debe poder ejecutarse en cualquier máquina con Java 17, MySQL y un navegador moderno.
La configuración del application.properties debe permitir cambiar la conexión a otra base de datos MySQL con mínima modificación.

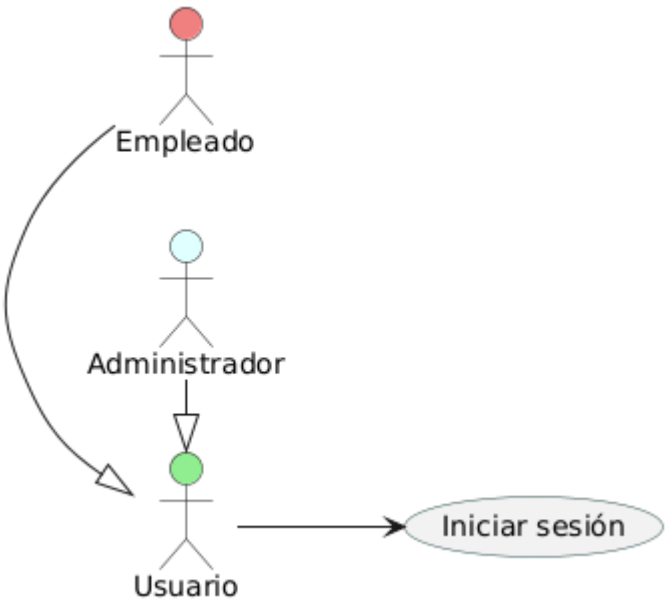
RNF-09 – Monitoreo y Logs

El sistema debe registrar logs de errores, accesos y operaciones críticas en archivos o consola, y permitir su consulta para auditoría y debugging.
Se debe implementar Spring Boot Actuator para métricas de rendimiento y salud del sistema.

7. Diagrama de Clases



7. Diagrama de Caso de Uso

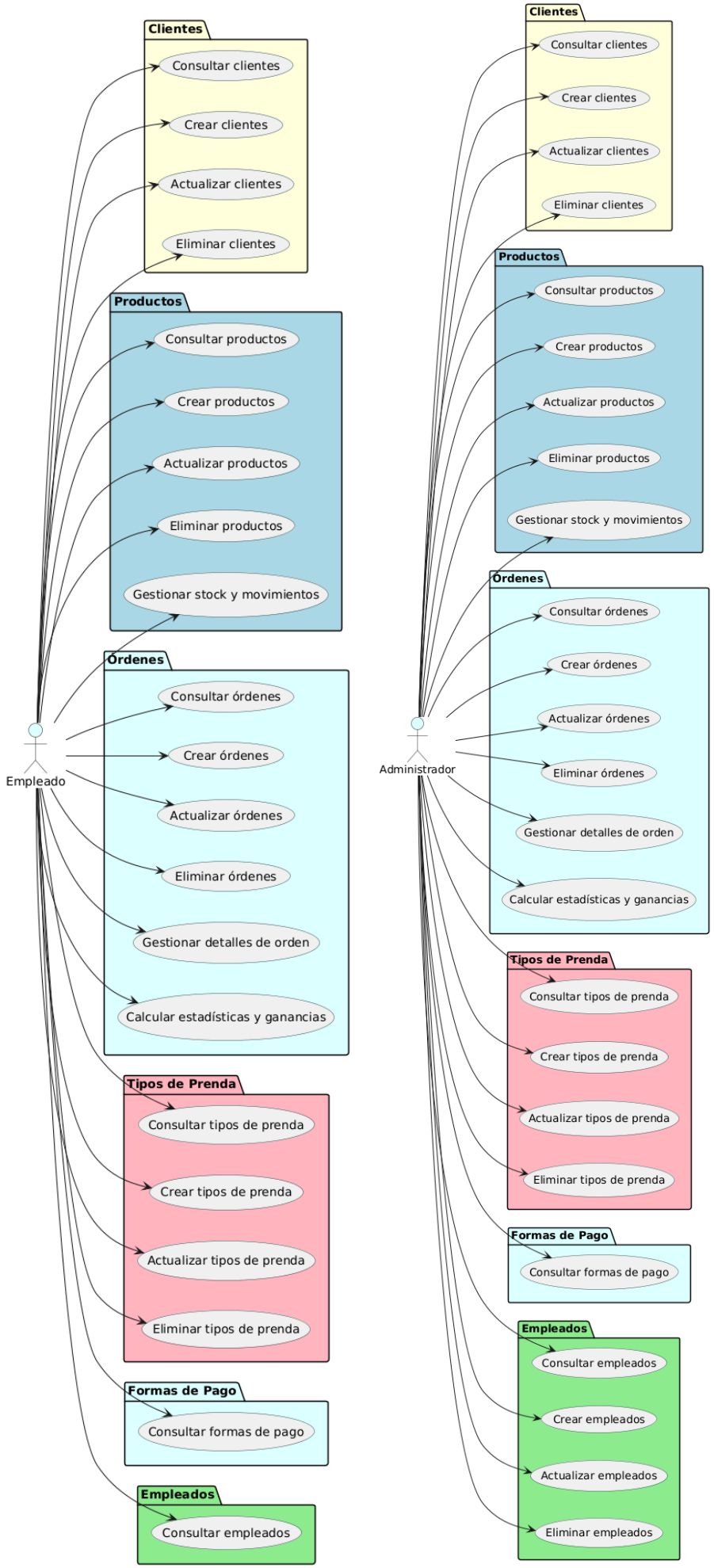


En el sistema, **todos los usuarios deben registrarse primero** antes de poder acceder a las funcionalidades de la aplicación. El registro implica proporcionar información básica y obligatoria, como nombre, apellido, DNI, teléfono, email y domicilio. Una vez completado el registro, el usuario recibe acceso al sistema mediante el **inicio de sesión**, donde debe autenticarse utilizando su nombre de usuario y contraseña. Este proceso asegura que solo personas autorizadas puedan interactuar con la aplicación, protegiendo la información de clientes, productos y órdenes.

Una vez autenticados, los usuarios pueden operar dentro del sistema, pero las funcionalidades disponibles dependen del **rol asignado**:

- **Empleado:**
  Este rol está diseñado para el personal operativo de la tienda. Los empleados pueden gestionar clientes, productos, órdenes, detalles de órdenes, tipos de prendas y consultar formas de pago. Sin embargo, no tienen autorización para **crear, actualizar o eliminar otros empleados**. Esto garantiza que las tareas de administración y control de personal queden restringidas a los roles superiores.

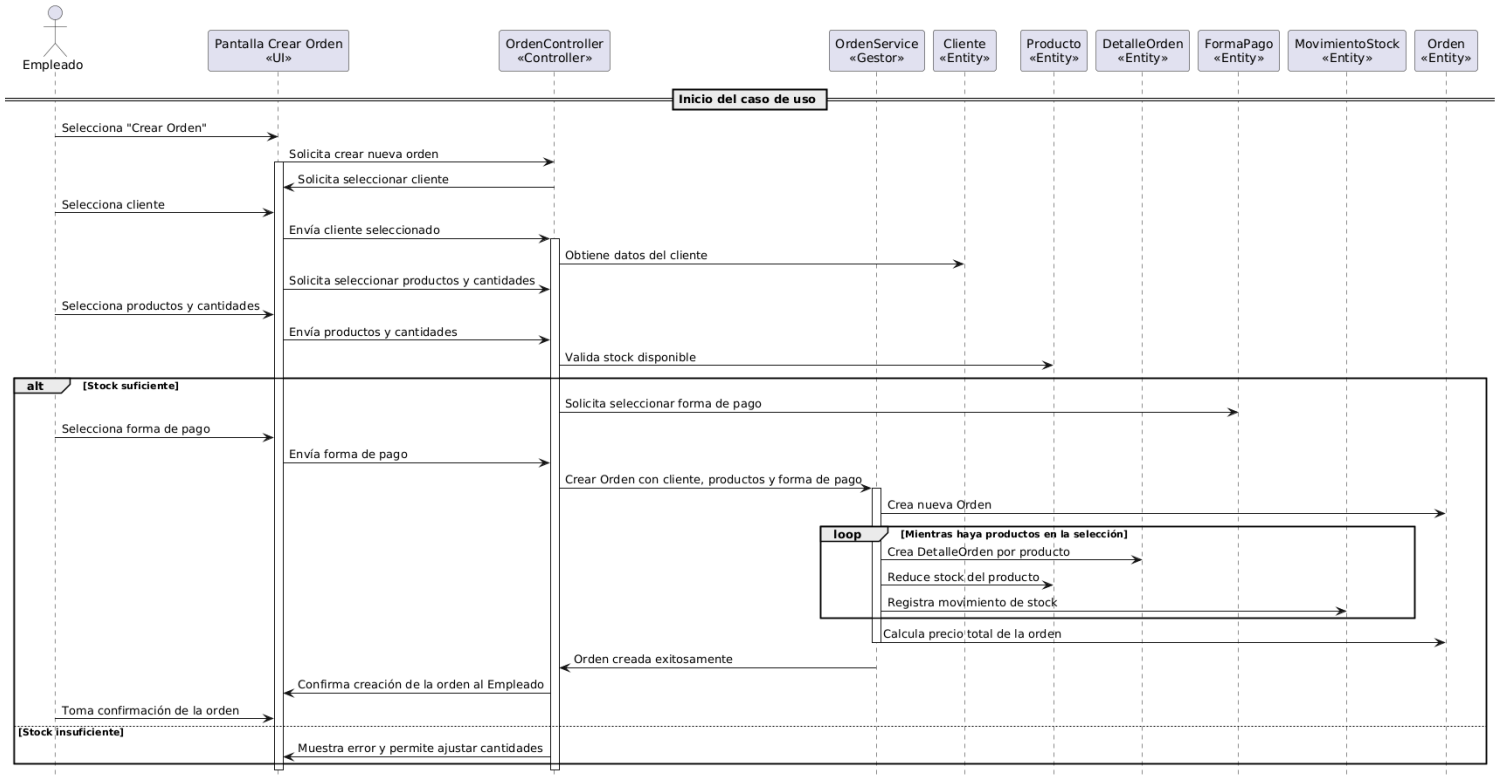
- **Administrador:**
  El administrador tiene acceso completo a todas las funcionalidades del sistema. Además de poder gestionar clientes, productos, órdenes y demás recursos, puede **crear, actualizar y eliminar empleados**, asegurando el control total sobre el personal y la administración de la tienda. Este rol es responsable de mantener la integridad y seguridad de la información, así como de supervisar las operaciones realizadas por los empleados.


Clientes
Consultar clientes
Crear clientes
Actualizar clientes
Eliminar clientes
Productos
Consultar productos
Crear productos
Actualizar productos
Eliminar productos
Gestionar stock y movimientos
Órdenes
Consultar órdenes
Crear órdenes
Actualizar órdenes
Eliminar órdenes
Gestionar detalles de orden
Calcular estadísticas y ganancias
Tipos de Prenda
Consultar tipos de prenda
Crear tipos de prenda
Actualizar tipos de prenda
Eliminar tipos de prenda
Formas de Pago
Consultar formas de pago
Empleados
Consultar empleados
Empleado
Clientes
Consultar clientes
Crear clientes
Actualizar clientes
Eliminar clientes
Productos
Consultar productos
Crear productos
Actualizar productos
Eliminar productos
Gestionar stock y movimientos
Órdenes
Consultar órdenes
Crear órdenes
Actualizar órdenes
Eliminar órdenes
Gestionar detalles de orden
Calcular estadísticas y ganancias
Tipos de Prenda
Consultar tipos de prenda
Crear tipos de prenda
Actualizar tipos de prenda
Eliminar tipos de prenda
Formas de Pago
Consultar formas de pago
Empleados
Consultar empleados
Crear empleados
Actualizar empleados
Eliminar empleados
Administrador

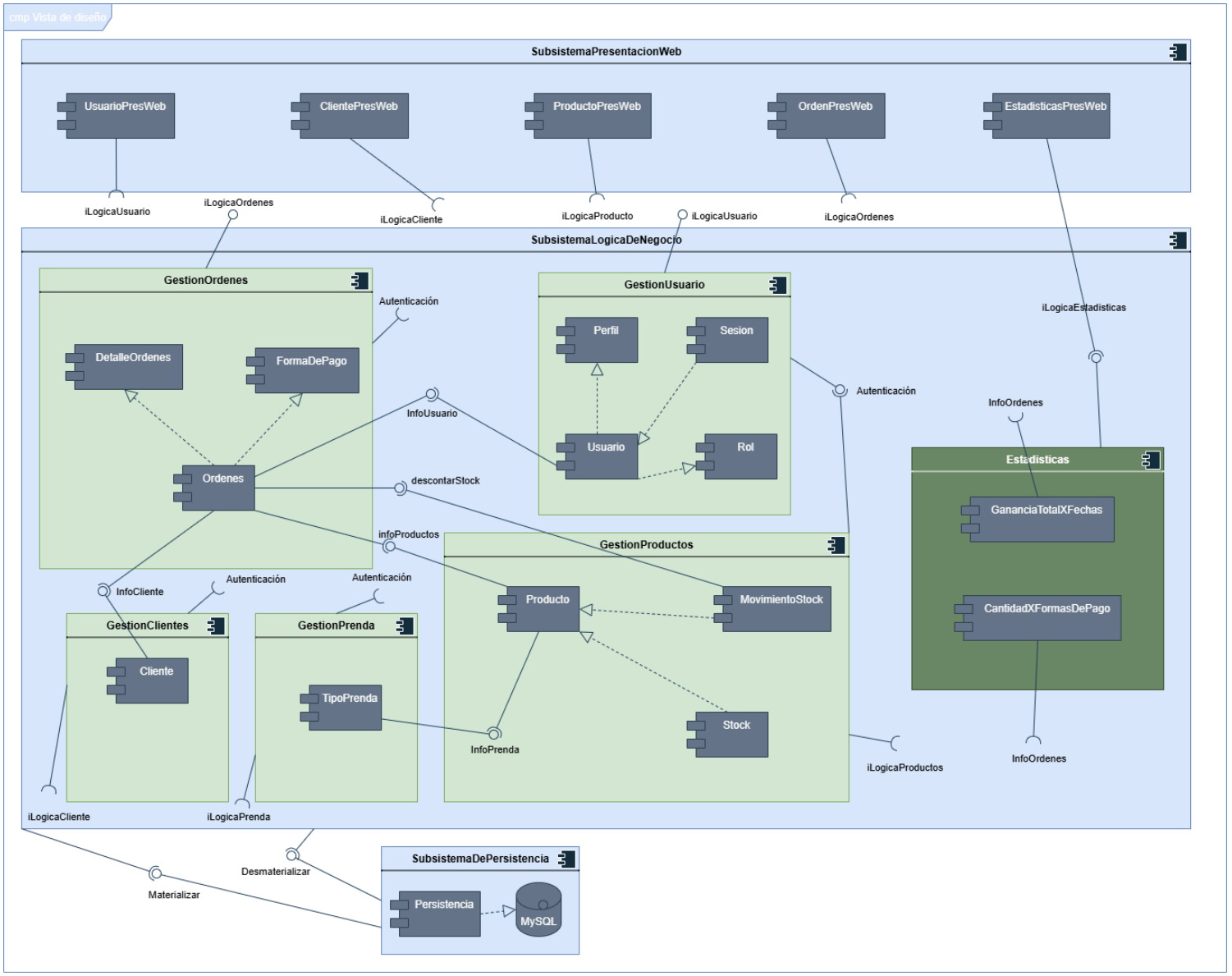
8. Descripción CU N°11 - Crear Órdenes

Nombre del Caso de uso: Crear Ordenes				**Nro. de Orden:** 11
Prioridad: ☒ Alta		Media		Baja
Complejidad: Simple	Mediano ☒	Complejo	Muy Complejo	Extremadamente Complejo
Actor Principal: Empleado / Administrador			**Actor Secundario:** no aplica	
Tipo de Caso de uso: ☒ Concreto		Abstracto		
Objetivo: Permitir que el usuario cree una nueva orden de venta asignando un cliente, productos y forma de pago, actualizando stock automáticamente.				
Flujo Básico				
1. **AC:** El usuario inicia el caso de uso seleccionando "Crear Orden" en el menú principal del sistema.				
2. **Sistema:** solicita al usuario seleccionar un cliente existente de la lista de clientes registrados.				
3. **AC:** elige los productos que desea agregar a la orden y especifica la cantidad y talle correspondiente de cada uno.				
4. **Sistema:** valida que haya stock disponible de cada producto y talle seleccionado; si no hay suficiente stock, informa al usuario.				
5. **AC:** selecciona la forma de pago de la orden, eligiendo entre las opciones disponibles (Efectivo, Tarjeta de Crédito o Débito).				
6. **Sistema:** calcula automáticamente el precio total de la orden sumando los precios unitarios por la cantidad de cada producto.				
7. **Sistema:** registra la orden en la base de datos, guardando todos los datos de cliente, productos, cantidades, talle y forma de pago.				
8. **Sistema:** actualiza el stock de cada producto según la cantidad seleccionada y genera los registros de movimientos de stock correspondientes (tipo de movimiento, producto, cantidad, fecha y talle).				
9. **Sistema:** confirma al usuario que la orden fue creada correctamente, mostrando un resumen con los productos, cantidades, precio total y forma de pago.				
Flujos Alternativos				
A1: Si algún producto no tiene stock suficiente, el sistema informa al usuario y permite ajustar la cantidad o eliminar el producto de la orden. **A2:** Si ocurre un error en el cálculo del total o actualización de stock, se cancela la operación y se informa al usuario.				
Observaciones: 1. Solo un **Empleado** o **Administrador** autenticado puede crear la orden. 2. La selección de productos incluye el talle correspondiente y se valida automáticamente contra el stock disponible. 3. La forma de pago debe seleccionarse entre las opciones configuradas (Efectivo, Tarjeta de Crédito o Débito). 4. El sistema registra de manera automática los movimientos de stock (entrada/salida) asociados a la orden. 5. El cálculo del precio total considera la cantidad de cada producto y sus respectivos precios unitarios. 6. En caso de error de validación o actualización de stock, la orden **no se guarda**, garantizando consistencia en la base de datos. 7. Este caso de uso puede ser la base para otros flujos como **Modificar Orden** o **Cancelar Orden**, aunque no se incluyen en este caso de uso.				

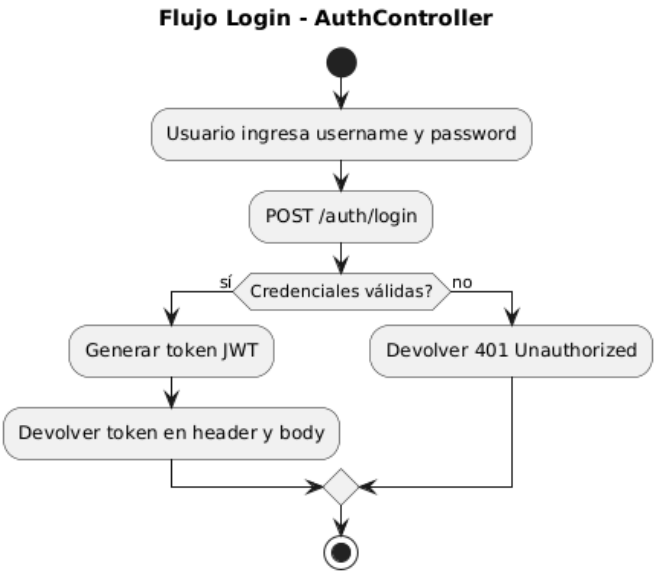
8.1 Diagrama de Secuencia CU N°07

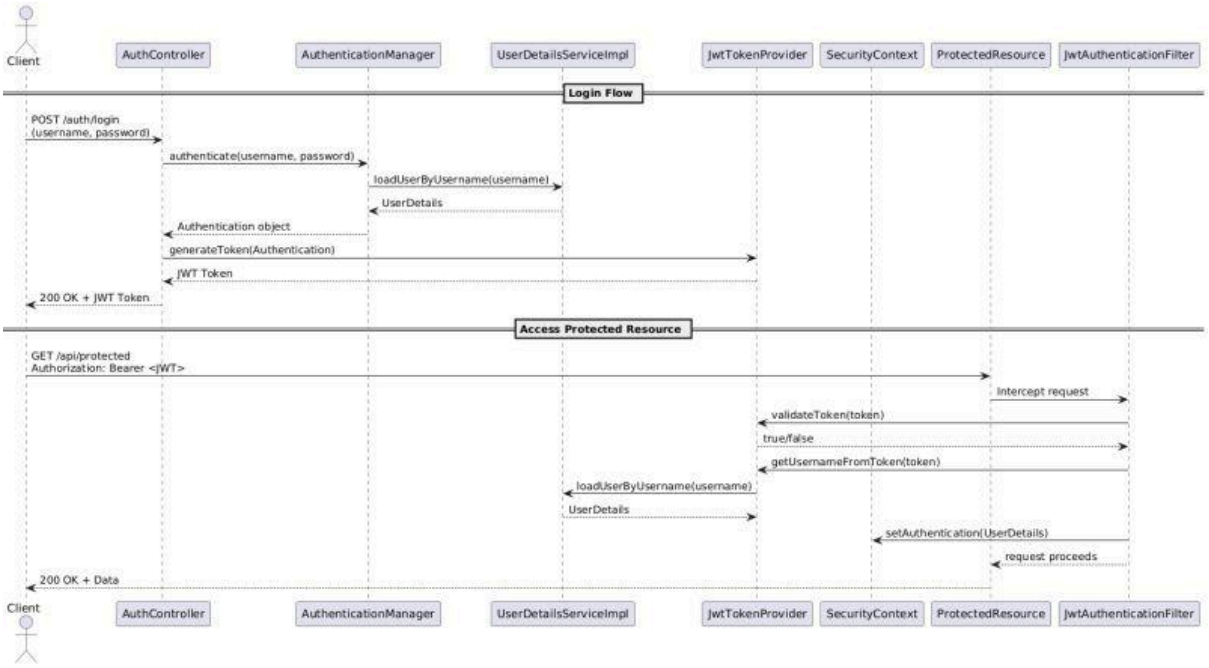


9. Vista Arquitectónica de Subsistemas/Interfaces

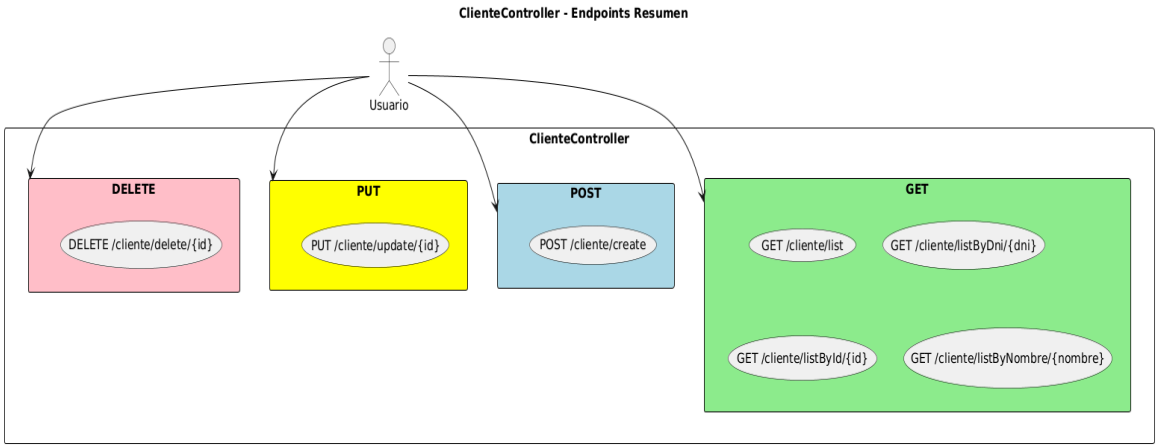


10. API REST - Endpoints del Sistema

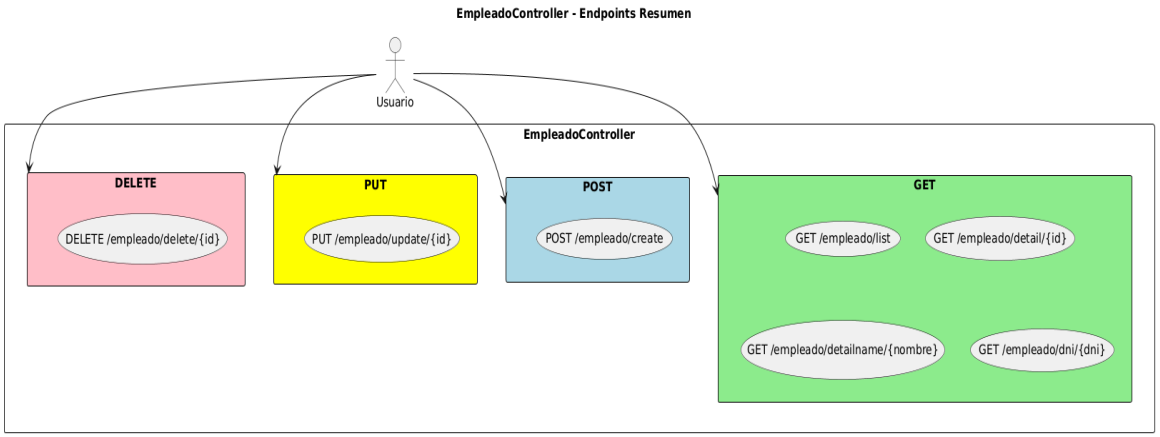




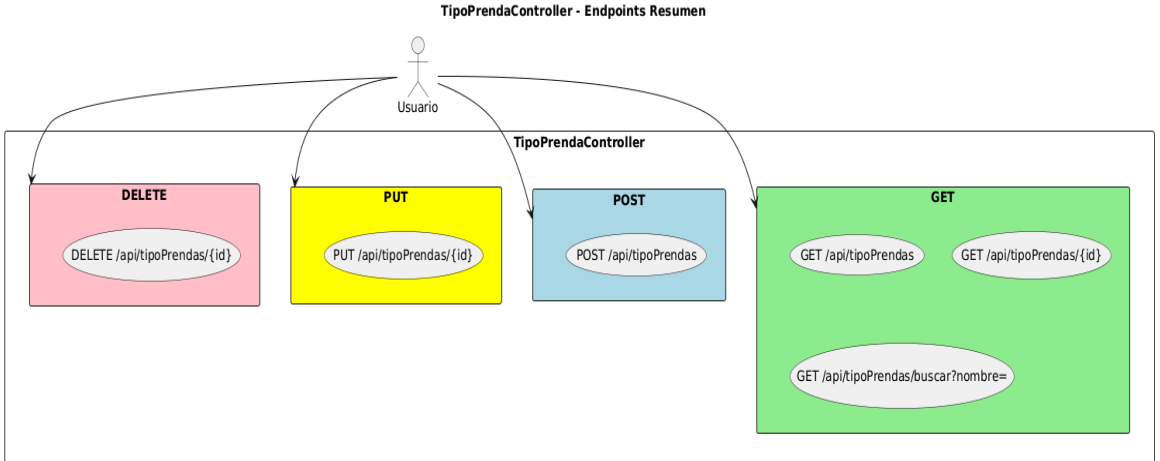
ClienteController:
Este controlador gestiona toda la información de los clientes de la tienda. Permite listar todos los clientes o buscar por DNI, ID o nombre, crear nuevos clientes con datos completos, actualizar información de clientes existentes y eliminar clientes por su ID.



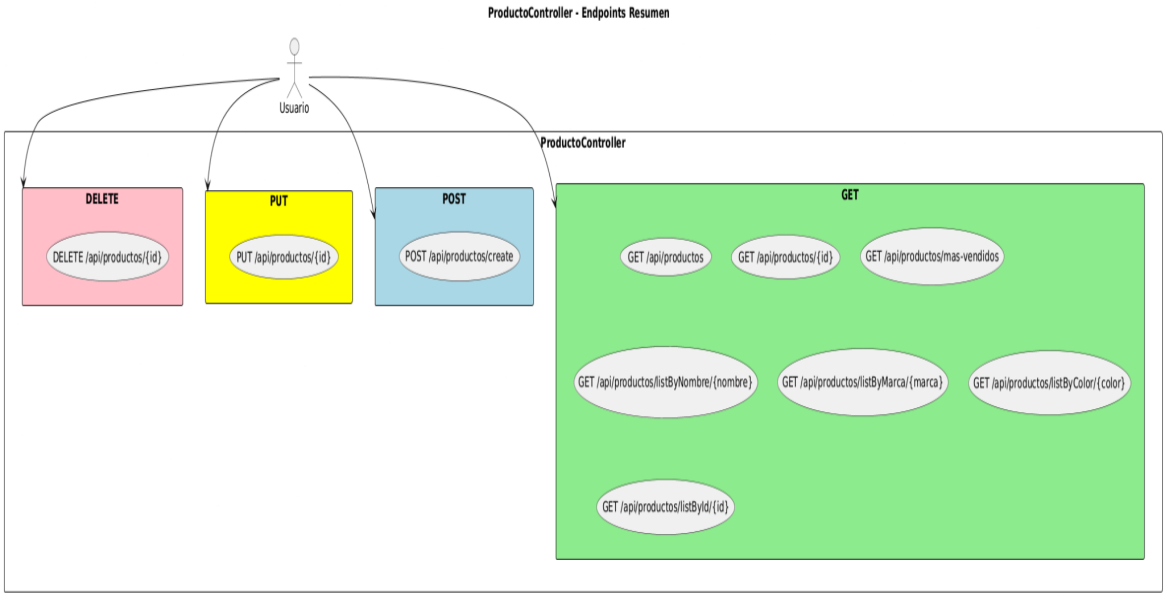
EmpleadoController:
Administra los empleados de la tienda, incluyendo la creación de nuevos empleados con validación de datos y asignación de rol, la actualización de su información, la eliminación por ID, y la posibilidad de listar todos los empleados o buscarlos por nombre o DNI.



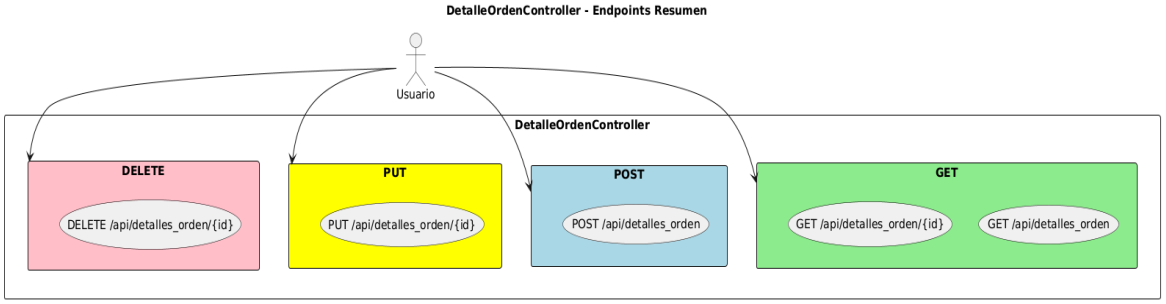
TipoPrendaController:
Se encarga de la gestión de los tipos de prendas disponibles en la tienda. Permite listar todos los tipos de prendas o buscar por ID, crear y actualizar tipos de prendas, eliminar por ID y filtrar tipos de prendas por nombre.



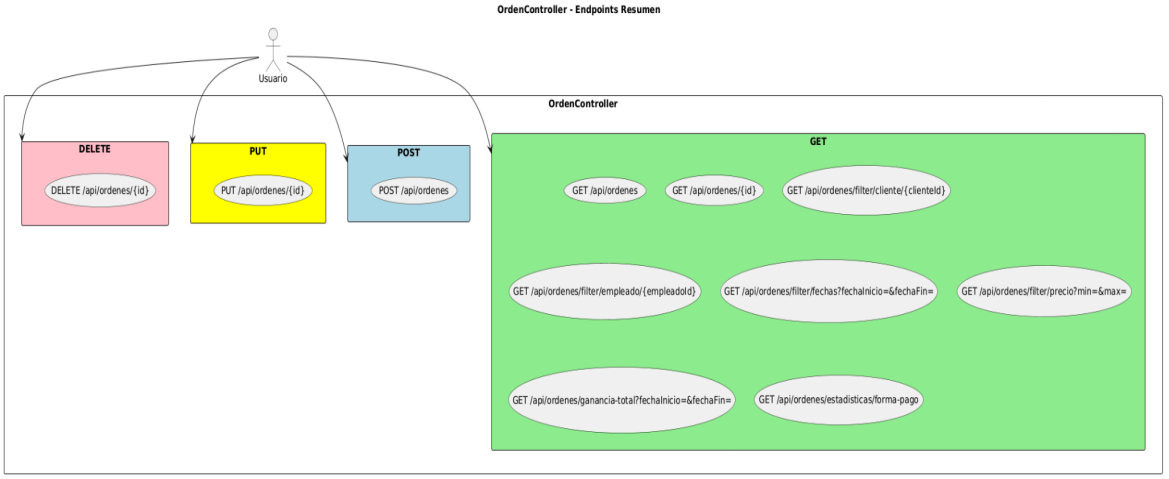
ProductoController:
Gestiona los productos de la tienda y su stock. Permite listar productos, buscar por ID, nombre, marca o color, crear productos asignando tipo de prenda y stock por talle, actualizar productos y su stock registrando movimientos de inventario, eliminar productos por ID y consultar los productos más vendidos.



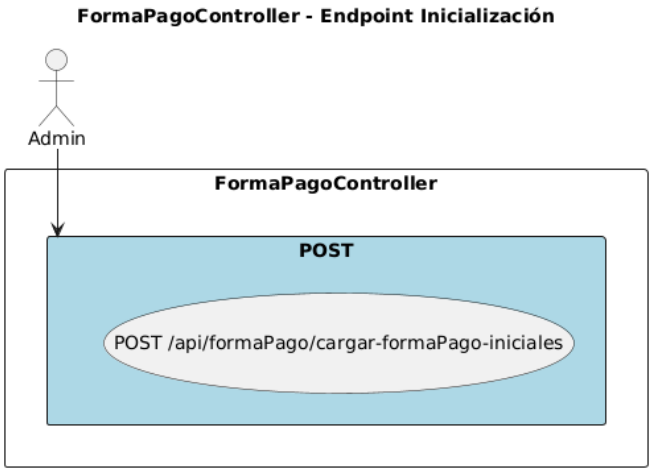
DetalleOrdenController:
Administra los detalles de cada orden de venta. Permite listar todos los detalles, consultar un detalle por ID, crear nuevos detalles de orden, actualizar detalles existentes y eliminar por ID.



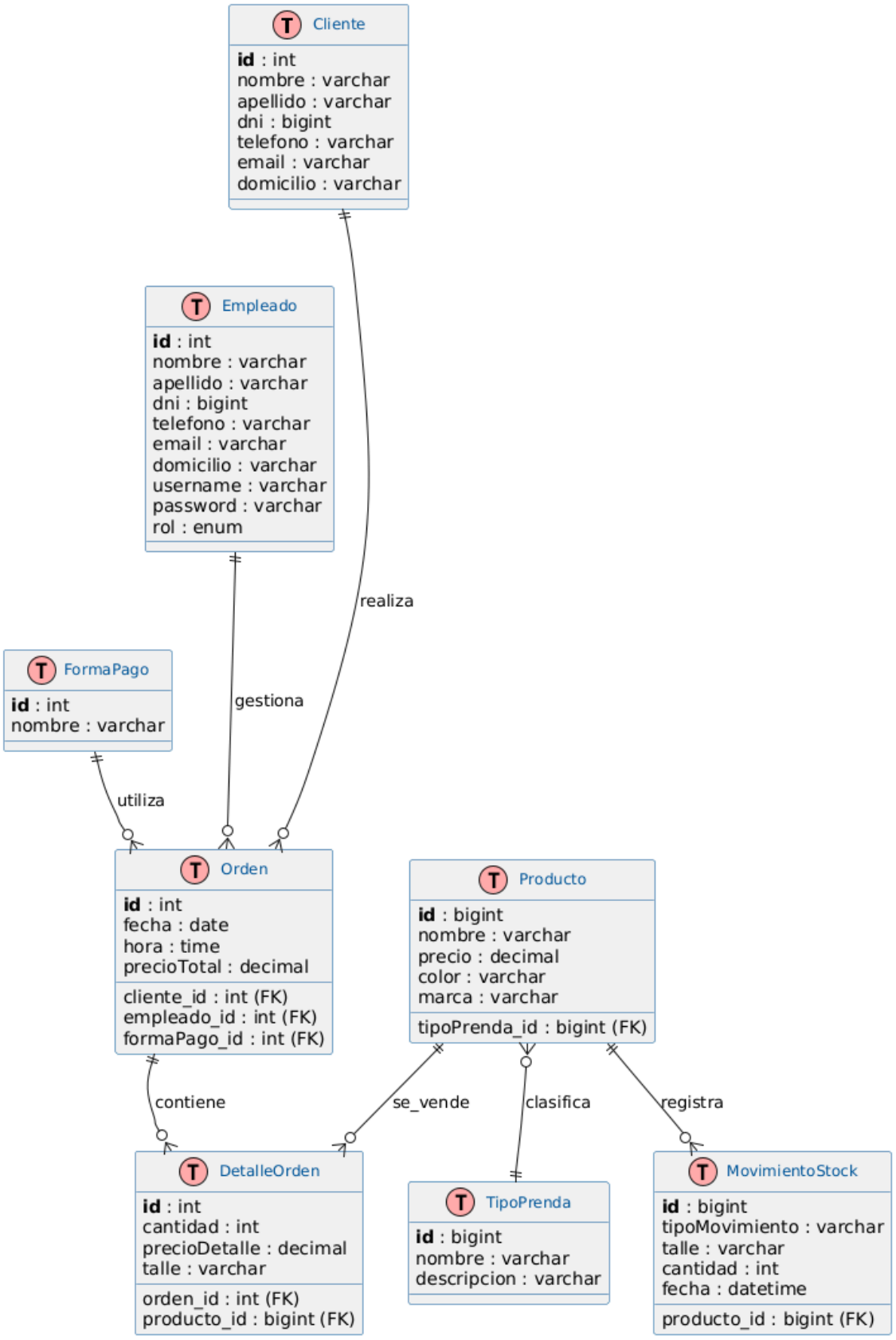
OrdenController:
Gestiona las órdenes completas de venta, incluyendo la asignación de cliente, empleado, forma de pago y detalles de productos. Permite listar órdenes, consultar por ID, crear nuevas órdenes, actualizar órdenes existentes controlando los detalles y el stock, eliminar órdenes por ID y filtrar por cliente, empleado, rango de fechas o precio total. Además, permite calcular las ganancias totales por rango de fechas y obtener estadísticas por forma de pago.



FormaPagoController:
Se encarga de inicializar las formas de pago disponibles en la tienda. No tiene endpoints para CRUD estándar, solo permite cargar las formas de pago iniciales desde el backend para que estén disponibles en el sistema.

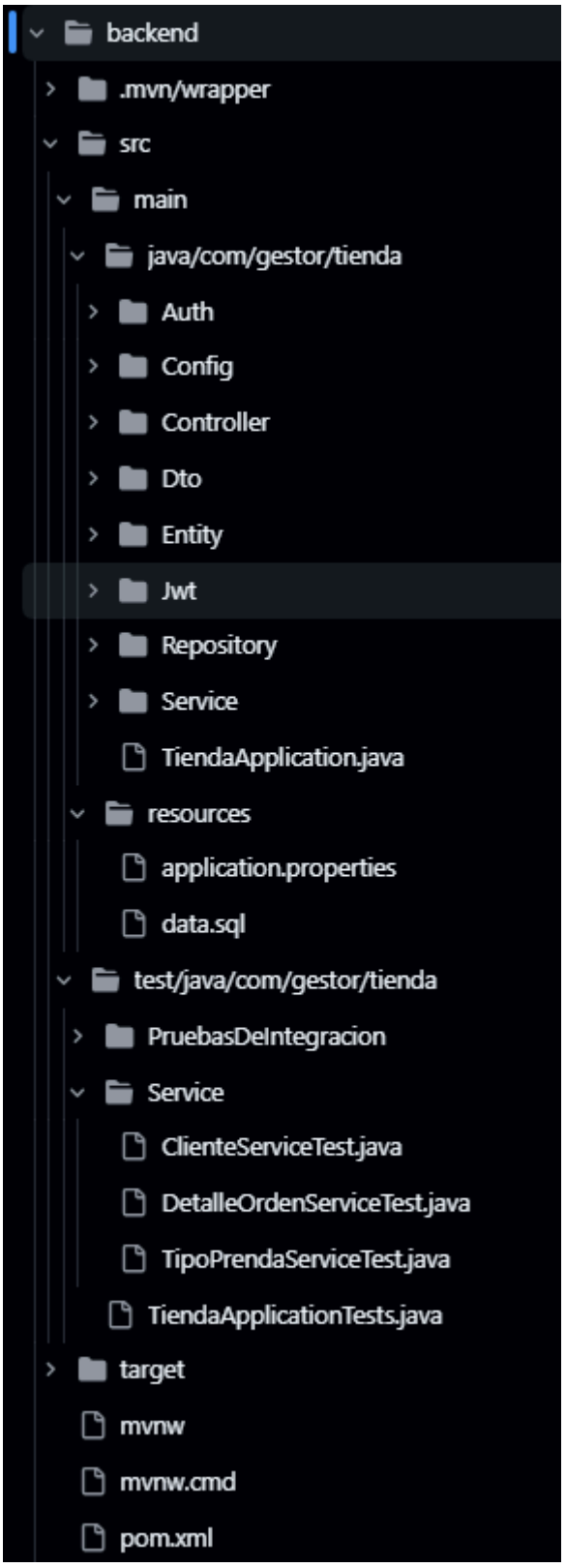


11. Diagrama de Entidad y Relación (DER)



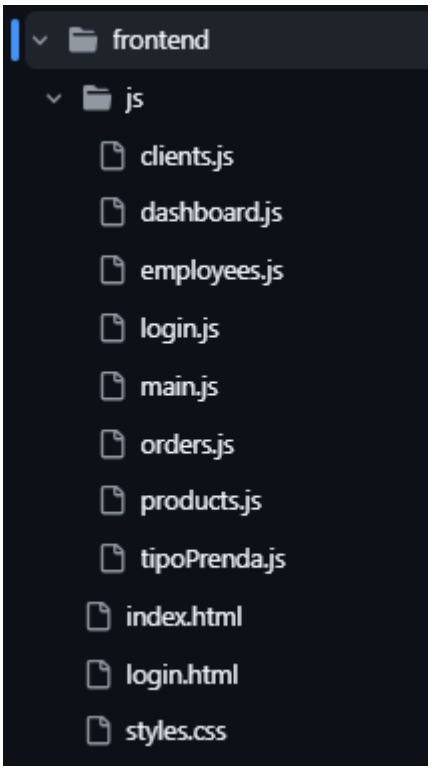
12. Backend

13.1 Estructura

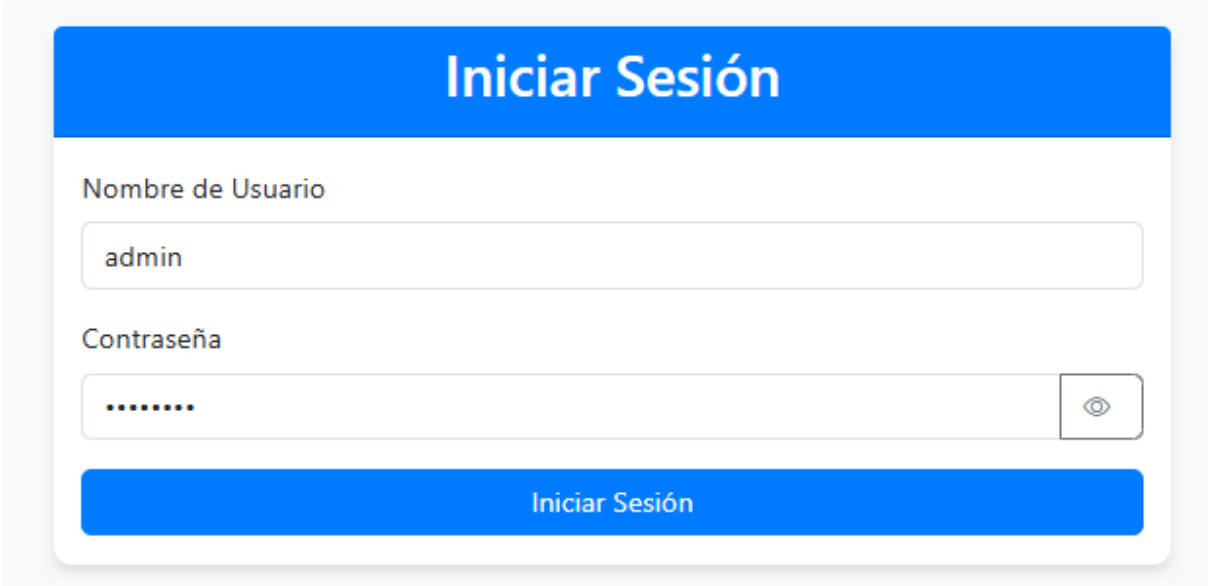


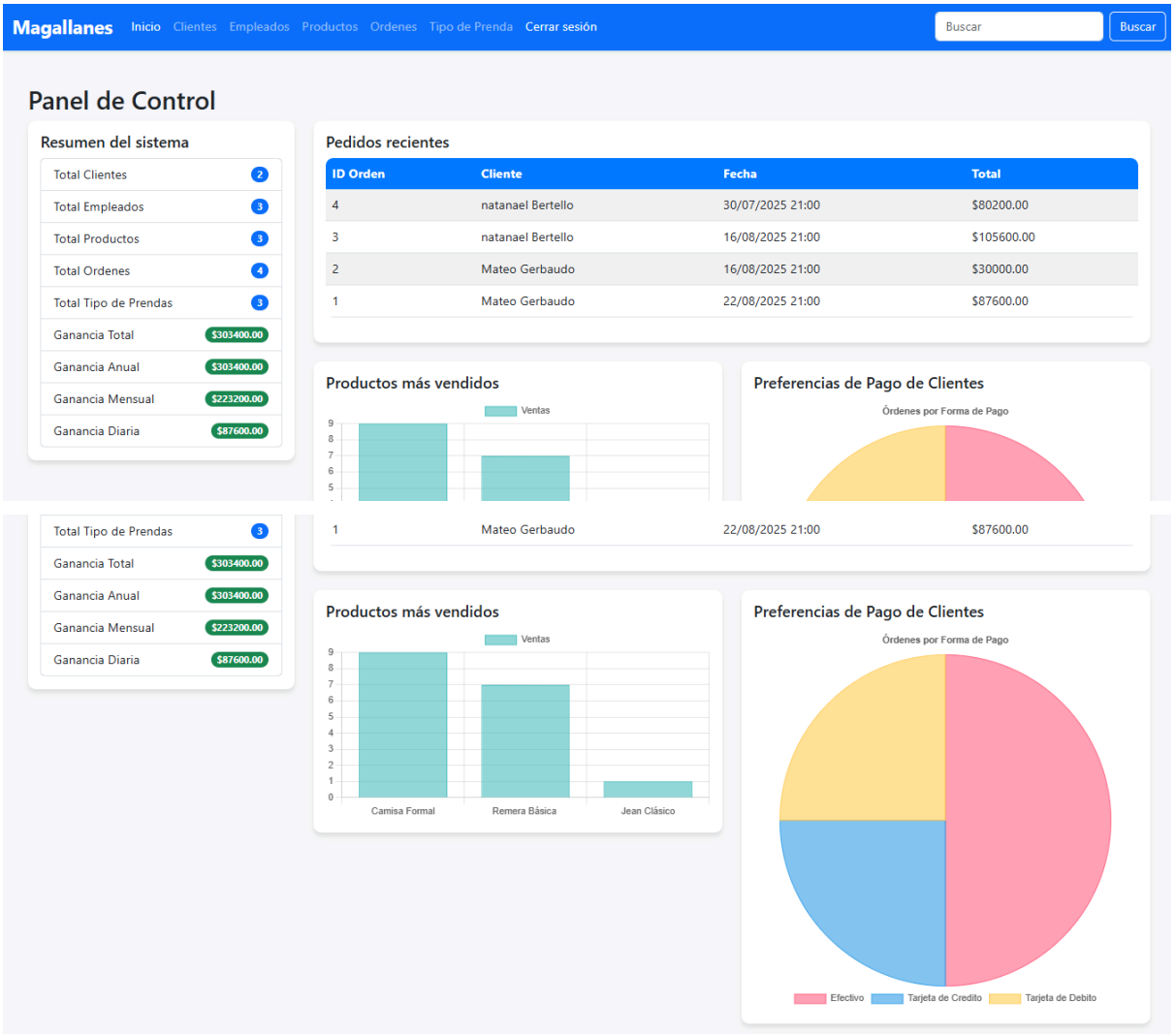
13. Frontend

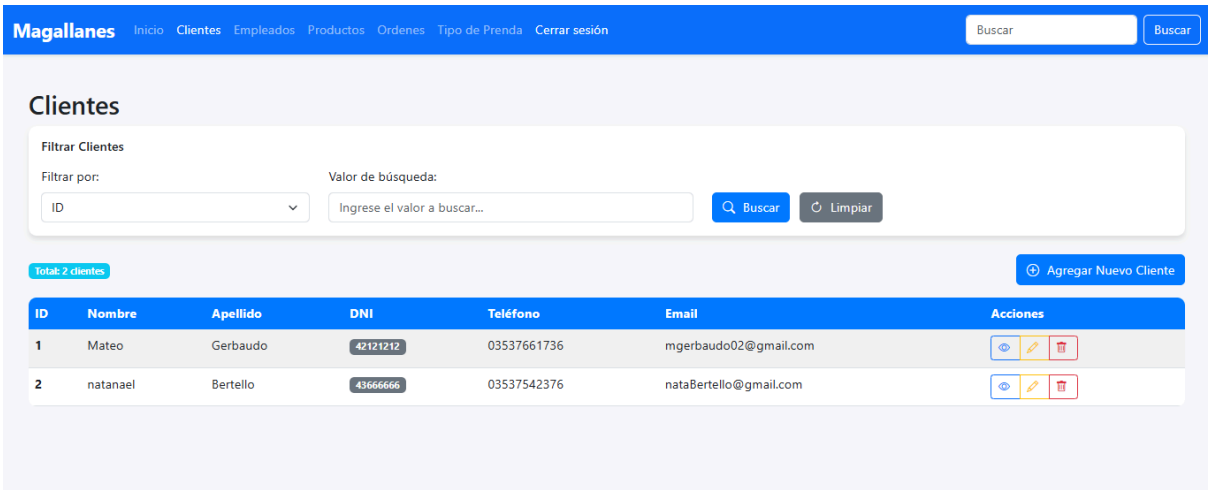
13.1 Estructura

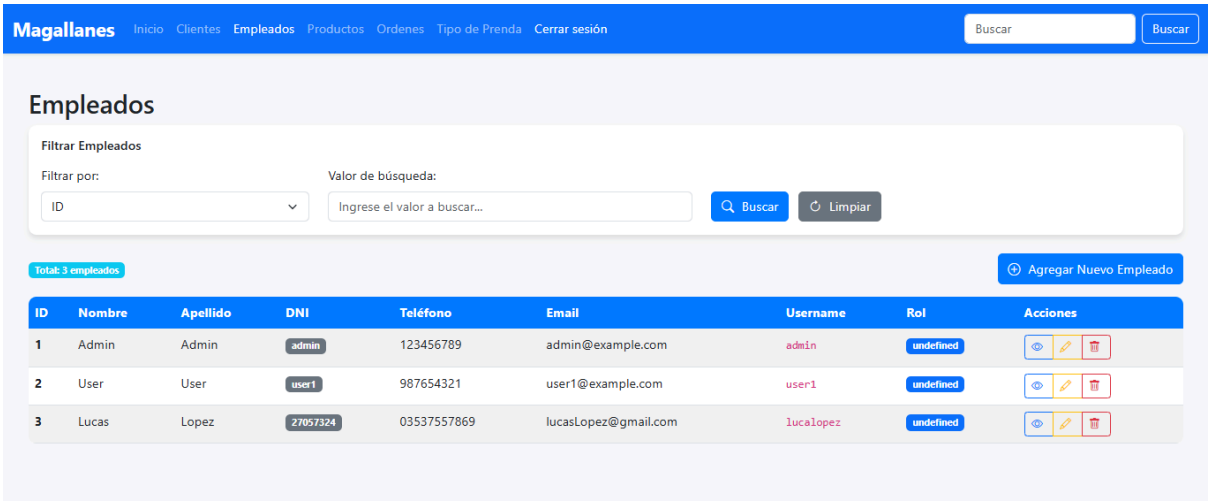


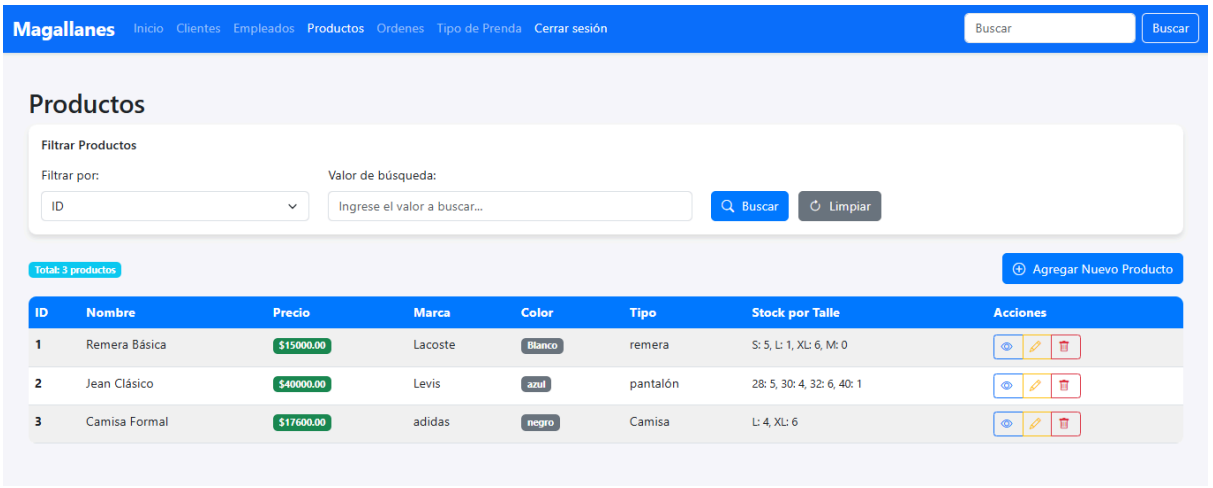
13.2 Pantallas

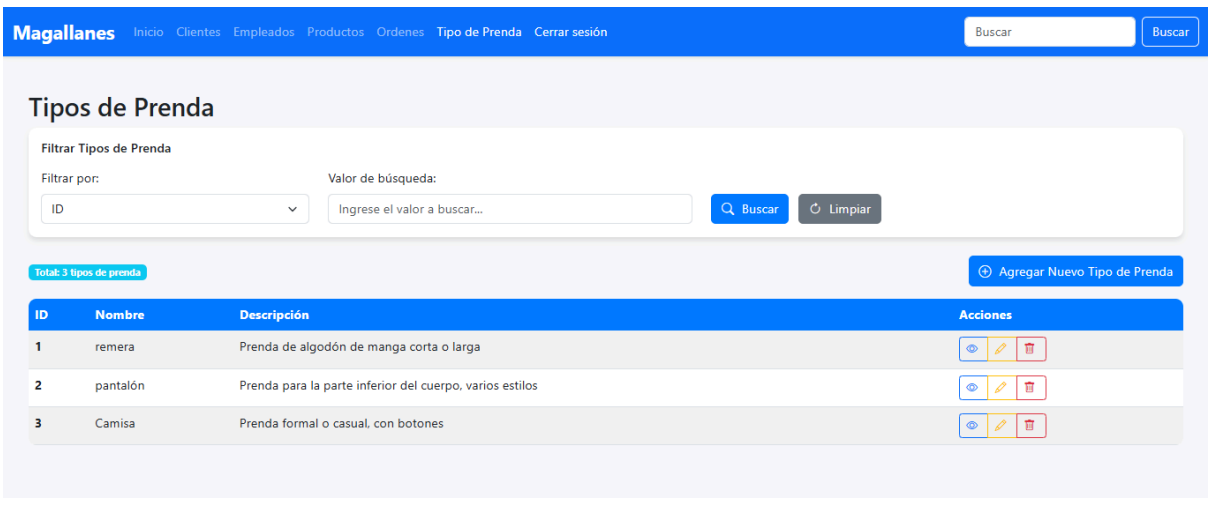


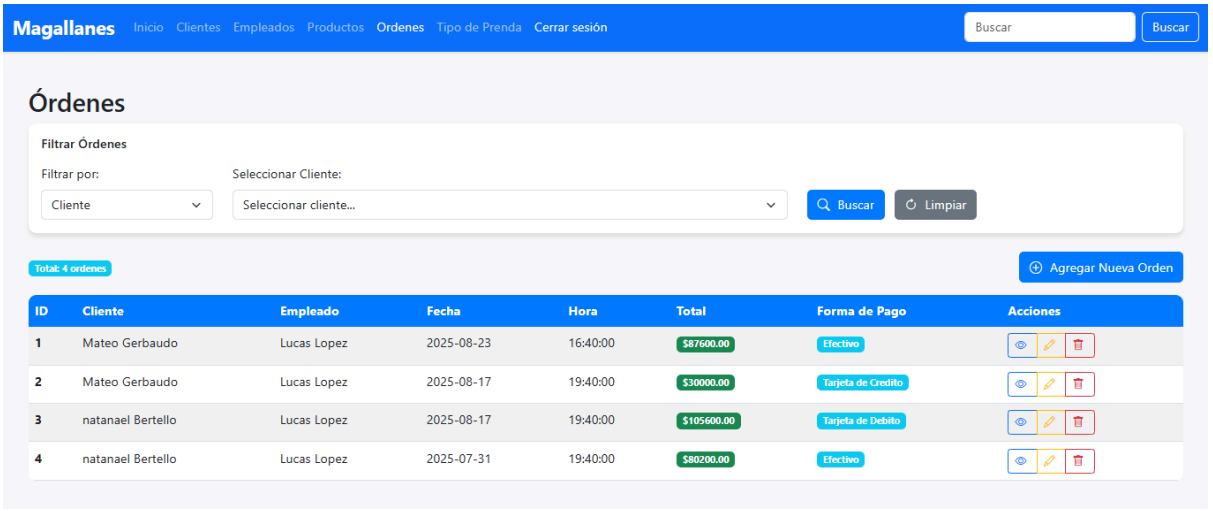

Magallanes
Inicio
Clientes
Empleados
Productos
Ordenes
Tipo de Prenda
Cerrar sesión
Buscar
Buscar
Panel de Control
Resumen del sistema
Total Clientes 2
Total Empleados 3
Total Productos 3
Total Ordenes 4
Total Tipo de Prendas 3
Ganancia Total $303400.00
Ganancia Anual $303400.00
Ganancia Mensual $223200.00
Ganancia Diaria $87600.00
Pedidos recientes
ID Orden
Cliente
Fecha
Total
4 natanael Bertello 30/07/2025 21:00 $80200.00
3 natanael Bertello 16/08/2025 21:00 $105600.00
2 Mateo Gerbaudo 16/08/2025 21:00 $30000.00
1 Mateo Gerbaudo 22/08/2025 21:00 $87600.00
Productos más vendidos
Ventas
Preferencias de Pago de Clientes
Órdenes por Forma de Pago
Camisa Formal
Remera Básica
Jean Clásico
Efectivo
Tarjeta de Credito
Tarjeta de Debito


Magallanes
Inicio
Clientes
Empleados
Productos
Ordenes
Tipo de Prenda
Cerrar sesión
Buscar
Buscar
Clientes
Filtrar Clientes
Filtrar por:
ID
Valor de búsqueda:
Ingrese el valor a buscar...
Buscar
Limpiar
Total: 2 clientes
Agregar Nuevo Cliente
ID
Nombre
Apellido
DNI
Teléfono
Email
Acciones
1 Mateo Gerbaudo 42121212 03537661736 mgerbaudo02@gmail.com
2 natanael Bertello 43666666 03537542376 nataBertello@gmail.com


Magallanes
Inicio
Clientes
Empleados
Productos
Ordenes
Tipo de Prenda
Cerrar sesión
Buscar
Buscar
Empleados
Filtrar Empleados
Filtrar por:
ID
Valor de búsqueda:
Ingrese el valor a buscar...
Buscar
Limpiar
Total: 3 empleados
Agregar Nuevo Empleado
ID
Nombre
Apellido
DNI
Teléfono
Email
Username
Rol
Acciones
1
Admin
Admin
admin
123456789
admin@example.com
admin
undefined
2
User
User
user1
987654321
user1@example.com
user1
undefined
3
Lucas
Lopez
27057324
03537557869
lucasLopez@gmail.com
lucalopez
undefined


Magallanes
Inicio
Clientes
Empleados
Productos
Ordenes
Tipo de Prenda
Cerrar sesión
Buscar
Buscar
Productos
Filtrar Productos
Filtrar por:
ID
Valor de búsqueda:
Ingrese el valor a buscar...
Buscar
Limpiar
Total: 3 productos
Agregar Nuevo Producto
ID
Nombre
Precio
Marca
Color
Tipo
Stock por Talle
Acciones
1
Remera Básica
$15000.00
Lacoste
Blanco
remera
S: 5, L: 1, XL: 6, M: 0
2
Jean Clásico
$40000.00
Levis
azul
pantalón
28: 5, 30: 4, 32: 6, 40: 1
3
Camisa Formal
$17600.00
adidas
negro
Camisa
L: 4, XL: 6


Magallanes
Inicio
Clientes
Empleados
Productos
Ordenes
Tipo de Prenda
Cerrar sesión
Buscar
Buscar
Tipos de Prenda
Filtrar Tipos de Prenda
Filtrar por:
ID
Valor de búsqueda:
Ingrese el valor a buscar...
Buscar
Limpiar
Total: 3 tipos de prenda
Agregar Nuevo Tipo de Prenda
ID
Nombre
Descripción
Acciones
1
remera
Prenda de algodón de manga corta o larga
2
pantalón
Prenda para la parte inferior del cuerpo, varios estilos
3
Camisa
Prenda formal o casual, con botones


Magallanes
Inicio
Clientes
Empleados
Productos
Ordenes
Tipo de Prenda
Cerrar sesión
Buscar
Buscar
Órdenes
Filtrar Órdenes
Filtrar por:
Cliente
Seleccionar Cliente:
Seleccionar cliente...
Buscar
Limpiar
Total: 4 ordenes
Agregar Nueva Orden
ID
Cliente
Empleado
Fecha
Hora
Total
Forma de Pago
Acciones
1
Mateo Gerbaudo
Lucas Lopez
2025-08-23
16:40:00
$87600.00
Efectivo
2
Mateo Gerbaudo
Lucas Lopez
2025-08-17
19:40:00
$30000.00
Tarjeta de Credito
3
natanael Bertello
Lucas Lopez
2025-08-17
19:40:00
$105600.00
Tarjeta de Debito
4
natanael Bertello
Lucas Lopez
2025-07-31
19:40:00
$80200.00
Efectivo

14. Reglas de Negocio

RN-001 – Gestión de Empleados

Nombre de la regla: Administración de empleados
Responsable: Equipo de desarrollo Magallanes
Descripción: Solo los administradores pueden crear, actualizar y eliminar empleados. Los empleados no tienen permisos sobre la gestión de personal, pero ambos roles pueden registrar y actualizar órdenes, consultar clientes, productos y stock, y acceder a reportes.
Condiciones: El usuario debe estar autenticado y tener rol asignado.
Acciones: Permitir o denegar la operación de creación, actualización o eliminación de empleados según el rol.
Excepciones: Ninguna; la regla se aplica en todo momento.

RN-002 – Control de Stock por Talle

Nombre de la regla: Validación y actualización de stock por talle
Responsable: Equipo de desarrollo Magallanes
Descripción: Cada producto tiene un stock diferenciado por talle. Al registrar o actualizar una orden, se debe verificar que exista stock suficiente para cada talle solicitado. El stock se ajusta automáticamente al confirmar, actualizar o eliminar órdenes.
Condiciones: La orden debe incluir la cantidad de productos por talle.
Acciones: Reducir el stock al confirmar la venta, ajustar el stock al actualizar la orden, y recuperar el stock al eliminar la orden.
Excepciones: Si un producto no tiene stock por talle definido, se aplica la regla de stock total.

RN-003 – Cálculo del Precio Total de la Orden

Nombre de la regla: Precio total de orden
Responsable: Equipo de desarrollo Magallanes
Descripción: El precio total de la orden se calcula sumando los subtotales de cada detalle, donde el subtotal corresponde al precio unitario del producto multiplicado por la cantidad. El cálculo se actualiza automáticamente al modificar la orden.
Condiciones: La orden debe contener al menos un producto con cantidad y precio definido.
Acciones: Calcular y almacenar el precio total de la orden al agregar, modificar o eliminar detalles.
Excepciones: Ninguna; la regla se aplica en todo momento.

RN-004 – Validaciones de Datos

Nombre de la regla: Integridad y consistencia de datos
Responsable: Equipo de desarrollo Magallanes
Descripción: Todos los datos ingresados deben cumplir con validaciones obligatorias. DNI de clientes y empleados debe ser único. Cantidades de stock y precios deben ser positivos. Campos obligatorios como nombre, apellido, tipo de prenda, stock por talle, cliente, empleado y forma de pago deben completarse.
Condiciones: La operación debe ser de creación o actualización de cliente, empleado, producto u orden.
Acciones: Validar los datos y bloquear la operación si alguna condición no se cumple, mostrando un mensaje de error.
Excepciones: Ninguna; todas las operaciones deben cumplir estas validaciones.

RN-005 – Formas de Pago

Nombre de la regla: Formas de pago disponibles
Responsable: Equipo de desarrollo Magallanes
Descripción: Las formas de pago disponibles son fijas: efectivo, tarjeta de crédito y tarjeta de débito. El usuario debe seleccionar una de las formas de pago existentes al registrar o actualizar una orden.
Condiciones: Al crear o actualizar una orden, se debe indicar la forma de pago.
Acciones: Permitir solo las formas de pago predefinidas.
Excepciones: Ninguna; no se permite crear nuevas formas de pago desde el sistema.

RN-006 – Movimientos de Stock

Nombre de la regla: Registro de movimientos de stock
Responsable: Equipo de desarrollo Magallanes
Descripción: Cada entrada o salida de productos en el sistema debe registrar un movimiento de stock indicando producto, cantidad, talle, tipo de movimiento y fecha. Esto asegura trazabilidad y control del inventario.
Condiciones: La operación debe ser de creación, actualización o eliminación de órdenes, o ajustes manuales de stock por un administrador.
Acciones: Registrar automáticamente un movimiento de stock cada vez que se realiza una operación que afecta el inventario.
Excepciones: Ninguna; todos los cambios en stock generan movimientos registrados.

RN-007 – Consultas y Reportes

Nombre de la regla: Acceso a consultas y reportes
Responsable: Equipo de desarrollo Magallanes
Descripción: Todos los usuarios autenticados pueden consultar información de clientes, productos, órdenes y stock. Los reportes consolidados y estadísticas detalladas solo son accesibles para administradores.

Condiciones: El usuario debe estar autenticado y tener el rol correspondiente para el tipo de reporte.
Acciones: Permitir consultas y generación de reportes según rol.
Excepciones: Los empleados no pueden acceder a reportes consolidados ni estadísticas avanzadas.

RN-008 – Gestión de Órdenes

Nombre de la regla: Registro y actualización de órdenes
Responsable: Equipo de desarrollo Magallanes
Descripción: Todas las órdenes deben asociarse a un cliente, un empleado, productos con talles y cantidades, y una forma de pago. Antes de confirmar la orden, el sistema verifica stock suficiente y calcula el precio total automáticamente.
Condiciones: La orden debe contener al menos un producto con cantidad válida y forma de pago seleccionada.
Acciones: Registrar la orden, actualizar stock por talle, calcular precio total y generar movimientos de stock.
Excepciones: La orden no se confirma si no hay stock suficiente o si faltan campos obligatorios.

RN-009 – Seguridad y Autenticación

Nombre de la regla: Control de acceso
Responsable: Equipo de desarrollo Magallanes
Descripción: Todo usuario debe autenticarse mediante nombre de usuario y contraseña. Los roles definen qué acciones pueden ejecutar. Las operaciones críticas están protegidas por Spring Security y tokens JWT.
Condiciones: El usuario debe estar registrado y activo.
Acciones: Permitir o denegar el acceso a funcionalidades según el rol del usuario.
Excepciones: Ninguna; todas las operaciones requieren autenticación.

15. Casos de Prueba

Documentación de Testing

Los casos de prueba, resultados esperados y procedimientos de testing del sistema fueron elaborados durante la cátedra de Testing. Para acceder al documento completo, utilizar el siguiente enlace:

Documento de Testing del Sistema

Este documento contiene todas las pruebas realizadas sobre los módulos de la aplicación, incluyendo validaciones de endpoints, manejo de datos y flujo de usuarios. Se recomienda revisarlo antes de cualquier despliegue o modificación del sistema.

16. Documentación Adicional

Instrucciones para ejecutar el proyecto localmente

1. Preparar la base de datos

1. Abrir **XAMPP** y asegurarte de que **Apache** y **MySQL** estén activos.
2. Acceder a **phpMyAdmin** (http://localhost/phpmyadmin).
3. Crear una nueva base de datos con el nombre que usa el proyecto. Por ejemplo:

   Nombre de la base de datos: **tiendaDB**
4. Importar si tenés un **script SQL inicial** para crear las tablas y registros necesarios.

---

2. Configurar el proyecto

1. Abrir el proyecto en tu **IDE** (IntelliJ, Eclipse, VSCode).
2. Abrir el archivo application.properties o application.yml y asegurarse de que los datos de conexión a la base de datos coincidan con XAMPP:

   ```
   spring.datasource.url=jdbc:mysql://localhost:3306/tiendaDB
   spring.datasource.username=root
   spring.datasource.password=
   spring.jpa.hibernate.ddl-auto=update
   spring.jpa.show-sql=true
   ```

   Ajustar username y password según tu configuración de MySQL.
3. Verificar que las demás propiedades de Spring Boot estén correctas, como el puerto de la aplicación:

   ```
   server.port=8080
   ```

---

3. Construir el proyecto

1. Abrir la terminal en la carpeta raíz del proyecto.
2. Ejecutar Maven o Gradle según corresponda:

Maven:

```
mvn clean install
```

---

4. Ejecutar la aplicación

1. Desde el IDE, ejecutar la clase principal con @SpringBootApplication (por ejemplo GestorTiendaApplication).
2. Desde la terminal, podés ejecutar:

   ```
   mvn spring-boot:run
   ```

3. La aplicación debería estar corriendo en **http://localhost:8080**.

---

5. Probar endpoints

1. Usar **Postman** o cualquier cliente HTTP para probar los endpoints de la API.
2. Ejemplo de endpoint para clientes:

   GET http://localhost:8080/cliente/list

3. Para autenticación, usar el endpoint de login:

   POST http://localhost:8080/auth/login

Cuerpo JSON:

```
{
   "username": "admin",
   "password": "admin123"
}
```

---

6. Notas adicionales

- Asegurarse de que **XAMPP** no tenga conflictos de puerto (MySQL y Apache).
- La primera vez, Spring Boot puede crear automáticamente las tablas si spring.jpa.hibernate.ddl-auto=update.
- Revisar los logs en la consola del IDE para verificar que la conexión a la base de datos se haya realizado correctamente.

17. Repositorio

https://github.com/gerbaudo19/magallanes.github.io