PROYECTO JAVA BACKEND – ARKA

NTEGRANTES

RAÚL ANTONIO MAYORGA MARROQUÍN

DAVID JOSUÉ LÓPEZ PORTILLO

CESAR WILFREDO SANCHEZ

Definicion del proyecto

Arka es una empresa distribuidora de accesorios de PC, que necesita pedir frecuentemente productos en grandes cantidades.

Aparte de gestionar los pedidos de sus clientes al momento de comprar y modificar pedidos, debido a la cantidad de clientes y los productos se ha vuelto difícil administrar el inventario, los procesos de despachar pedidos y los abastecimientos de accesorios.

La compañía necesita tener un sistema que le permita automatizar los procesos de abastecimientos, venta, actualización de stock, generación de reportes de ventas, notificaciones de cambios de estado del pedido y recordatorios de carrito abandonado con el fin de atender un mayor número de clientes de manera simultánea y en un menor tiempo

Historias de Usuario

Módulo 1: Gestión de Inventario y Abastecimiento.

HU1 - Registrar productos en el sistema

Descripción: Como administrador, quiero registrar nuevos productos con sus características para que los clientes puedan comprarlos.

Criterios de aceptación:

- Se debe permitir la carga de nombre, descripción, precio, stock y categoría.
- Validaciones de datos requeridos.
- Mensaje de confirmación tras el registro exitoso.

HU2 - Actualizar stock de productos

Descripción: Como administrador, quiero actualizar la cantidad de productos en stock para evitar sobreventas.

Criterios de aceptación:

- El sistema debe permitir modificar el stock de un producto.
- No se deben permitir valores negativos.
- Historial de cambios en el stock.

HU3 - Generar reportes de productos por abastecer

Descripción: Como administrador, quiero recibir reportes de productos con bajo stock para tomar decisiones de abastecimiento.

Criterios de aceptación:

- El reporte debe generarse automáticamente cada semana.
- Debe incluir productos con stock menor a un umbral configurable.
- Exportación en formato CSV o PDF.

Módulo 2: Gestión de órdenes de compra (David)

HU4 - Registrar una orden de compra

Descripción: Como cliente, quiero poder registrar una orden de compra con múltiples productos para realizar mi pedido.

Criterios de aceptación:

- Se debe validar la disponibilidad del stock.
- Registro de fecha y detalles del pedido.
- Mensaje de confirmación con resumen del pedido.

HU5 - Modificar una orden de compra

Descripción: Como cliente, quiero modificar mi pedido antes de su confirmación para corregir errores o agregar productos.

Criterios de aceptación:

- Solo se pueden modificar pedidos en estado 'pendiente'.
- Se debe actualizar el stock en caso de eliminación de productos.

HU6 - Notificación de cambio de estado de pedido

Descripción: Como cliente, quiero recibir notificaciones sobre el estado de mi pedido para estar informado de su progreso.

Criterios de aceptación:

- Notificación por correo o en la plataforma.
- Estados: pendiente, confirmado, en despacho, entregado.

Módulo 3: Reportes y análisis de ventas

HU7 - Generar reportes de ventas semanales

Descripción: Como administrador, quiero generar reportes semanales de ventas para analizar el rendimiento del negocio.

Criterios de aceptación:

- El reporte debe incluir total de ventas, productos más vendidos y clientes más frecuentes.
- Exportación en formato CSV o PDF.

HU8 - Identificar carritos abandonados (no tenemos carrito)

Descripción: Como administrador, quiero visualizar los carritos abandonados para contactar a los clientes y recuperar ventas.

Criterios de aceptación:

- Listado de carritos abandonados con fecha y productos.
- Opción de enviar recordatorio por correo al cliente.

Priorización

Historia de Usuario	Prioridad
HU1 - Registrar productos	Alta
HU2 - Actualizar stock	Alta
HU4 - Registrar orden de compra	Alta
HU6 - Notificación de cambio de estado	Alta
HU7 - Reporte de ventas semanales	Media
HU8 - Carritos abandonados	Media
HU5 - Modificar orden de compra	Baja
HU3 - Reporte de productos por abastecer	Baja

Solucion de la problematica

Analisis DDD (Domain Driven Design)

1) Dominio Principal

Gestión de Inventario y Órdenes (e-commerce B2C/B2B ligero). Motivación: permitir registrar productos, gestionar stock, crear y modificar órdenes, generar reportes y manejar carritos/notifications

2) Event Storming

- Eventos de dominio (Domain Events)
- ProductoRegistrado
- StockActualizado
- StockBajoDetectado
- OrdenCreada
- OrdenModificada
- EstadoPedidoCambiado (pendiente → confirmado → en_despacho → entregado)
- OrdenAbandonado
- ReporteSemanalGenerado
- NotificacionEnviada

3) Modelo de Dominio (entidades y relaciones conceptuales)

Se determino que para solventar la necesidad de ARKA se debe basar en las siguientes entidades dentro del modelo de dominio:

Entidades principales y atributos:

Inventario

id, nombre, descripcion, precio, categoria

Inventario

id, productoId, stockActual, umbralMinimo, fechaActualizacion

HistoricoInventario

id, productoId, cantidaAnterior, cantidadNueva, motivo, fecha

Orden

id, correo, nombreCliente, direccionEntrega, subTotal, estado, fechaCreacion, fechaModificacion

DetalleOrden

- id, ordenId, productoId, cantidad, precioUnitario

4) Bounded Contexts y Subdominios

- **Inventario —** gestión de productos, stock y historial.
- **Ventas / Pedidos —** creación/modificación de órdenes, carritos, estados de pedido.
- **Reportes —** generación y exportación de reportes**.**
- **Notificaciones —** correo, notificaciones en plataforma. (En desarrollo)
- **Gateway / API —** orquestación y contratos HTTP.

Subdominios

- **Principales:** Inventario, Pedidos.
- **Soporte:** Reportes.
- **Genéricos:** Autenticación.

5) Lenguaje ubicuo (Ubiquitous Language)

Como parte de la definición de términos importantes dentro dominio del sistema, tenemos:

- **Inventario**: Artículo que se compran y se venden, estos artículos pertenecen a una categoría.
- **Orden**: Petición de compra en un estado (pendiente, confirmado, en_despacho, entregado).
- **Reporte Semanal**: Reportes a demanda según las necesidades del negocio
    - Reportes de ventas por fechas, reportes de productos en stock, descarga de reportes en archivo csv.
- **Stock Disponible**: Se maneja las cantidades de artículos disponibles, máximos y mínimos en existencias.
- **Historial de stock**: registro de todos los movimientos de inventario.

Diagramas

Diagrama de Base de Datos.

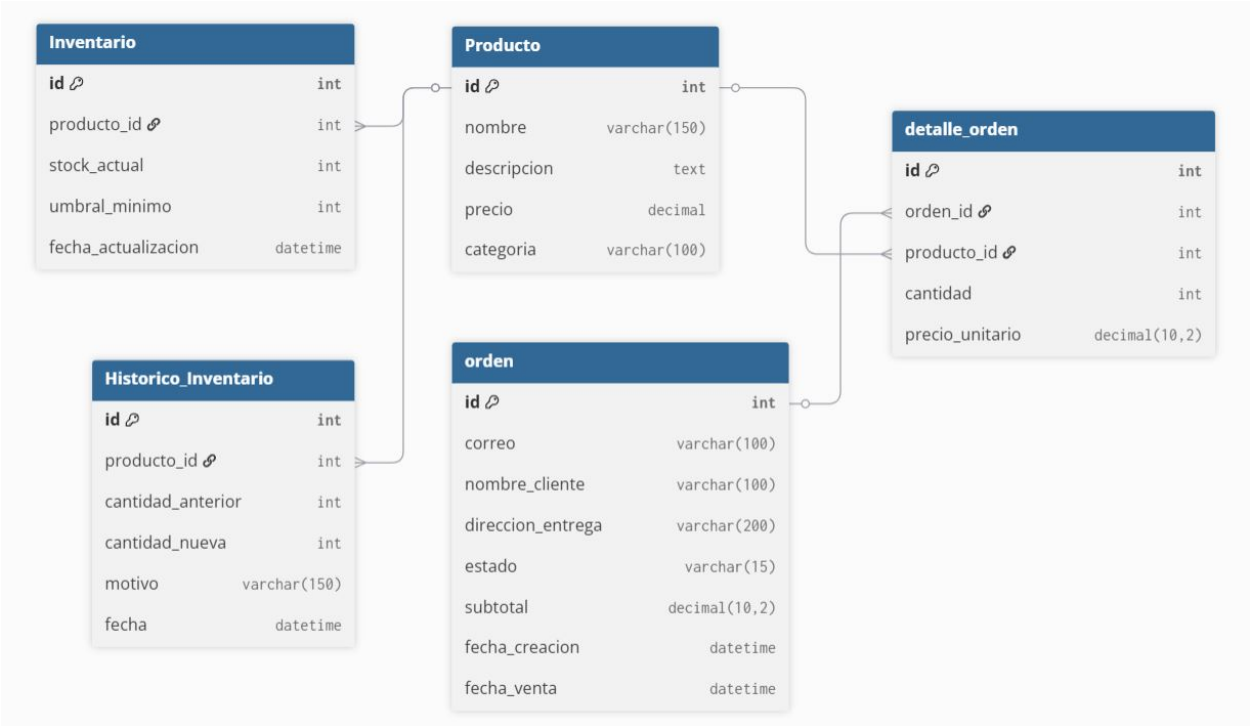


Diagrama de Infraestructura

Diagrama de componentes

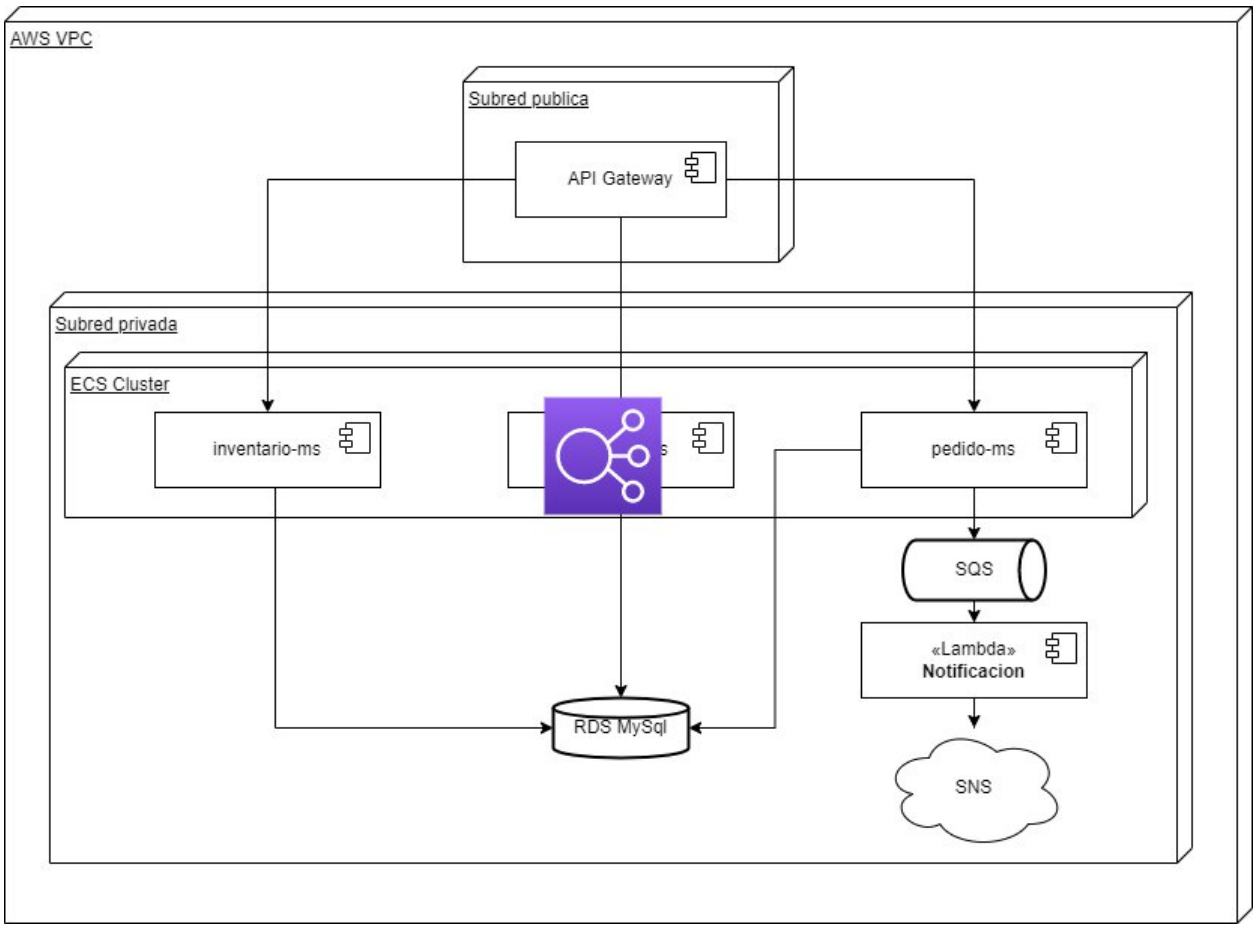


Diagrama de Arquitectura

Modelo en capas

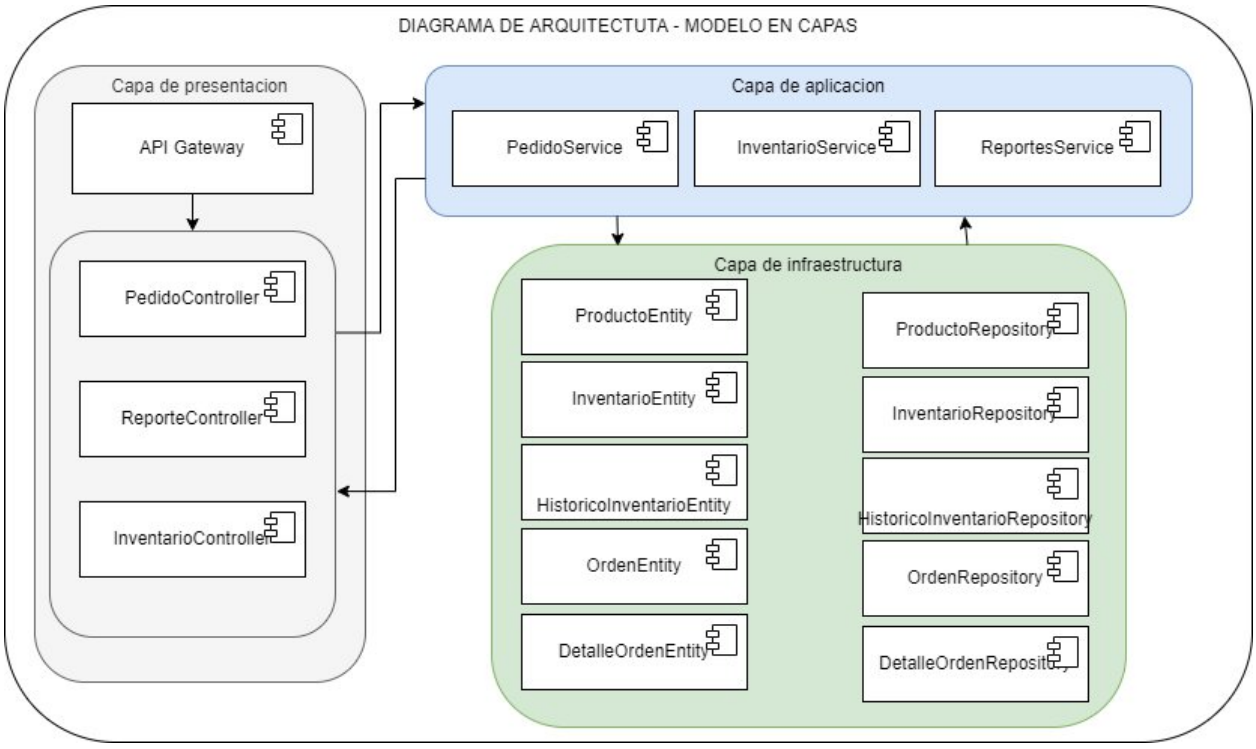
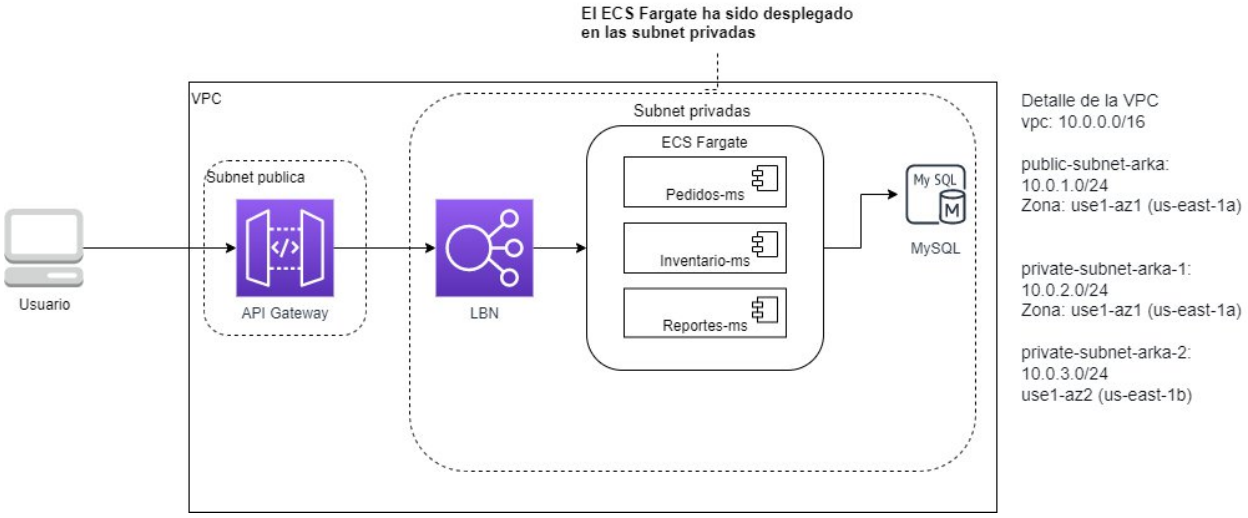


Diagrama de servicios



Plan de acción

Script de la base de datos

```
-- MySQL dump 10.13  Distrib 8.0.19, for Win64 (x86_64)
--
-- Host: arka.c4pg2u66azll.us-east-1.rds.amazonaws.com    Database: arka
-- ------------------------------------------------------
-- Server version    8.0.42

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!50503 SET NAMES utf8mb4 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;
SET @MYSQLDUMP_TEMP_LOG_BIN = @@SESSION.SQL_LOG_BIN;
SET @@SESSION.SQL_LOG_BIN= 0;

--
-- GTID state at the beginning of the backup
--

SET @@GLOBAL.GTID_PURGED=/*!80000 '+'*/ '';

--
-- Table structure for table `detalleorden`
--

/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `detalleorden` (
  `id` int NOT NULL AUTO_INCREMENT,
  `orden_id` int DEFAULT NULL,
  `producto_id` int DEFAULT NULL,
  `cantidad` int DEFAULT NULL,
  `precio_unitario` decimal(10,2) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=25 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `detalleorden`
```

```
--

LOCK TABLES `detalleorden` WRITE;
/*!40000 ALTER TABLE `detalleorden` DISABLE KEYS */;
INSERT INTO `detalleorden` VALUES
(1,1,1,1,850.00),(2,1,2,2,12.50),(3,2,3,2,55.00),(4,2,10,4,15.00),(5,3,4,1,160.00),(6,4,6,1,70.00),(7,5,7,1,299.00),(8,6,5,1,130.00),
(9,6,9,2,50.00),(10,7,8,1,145.00),(11,8,9,1,60.00),(12,9,7,1,299.00),(13,10,3,2,55.00),(14,10,2,4,12.50),(15,10,10,1,15.00),(16,12
,1,2,850.00),(17,12,4,1,160.00),(18,13,5,2,130.00),(19,13,2,6,12.50),(20,14,6,1,70.00),(21,14,22,1,65.50),(22,15,6,1,70.00),(23,1
5,22,1,65.50),(24,14,1,1,850.00);
/*!40000 ALTER TABLE `detalleorden` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `historico_inventario`
--

/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `historico_inventario` (
  `id` int NOT NULL AUTO_INCREMENT,
  `producto_id` int NOT NULL,
  `cantidad_anterior` int NOT NULL,
  `cantidad_nueva` int NOT NULL,
  `motivo` varchar(150) DEFAULT NULL,
  `fecha_cambio` datetime NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=30 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `historico_inventario`
--

LOCK TABLES `historico_inventario` WRITE;
/*!40000 ALTER TABLE `historico_inventario` DISABLE KEYS */;
INSERT INTO `historico_inventario` VALUES (1,1,10,8,'Venta de 2 unidades','2025-11-01 09:30:00'),(2,2,50,40,'Alta rotación -
venta masiva','2025-11-02 10:15:00'),(3,3,2,4,'Reposición de stock','2025-11-03 11:00:00'),(4,4,15,12,'Venta de monitores','2025-
11-03 15:45:00'),(5,5,8,6,'Venta de sillas ergonómicas','2025-11-04 09:10:00'),(6,6,12,20,'Ingreso de nuevo lote','2025-11-04
14:25:00'),(7,7,12,10,'Venta de auriculares premium','2025-11-04 17:00:00'),(8,8,6,2,'Salida para demostración en tienda','2025-
11-05 09:00:00'),(9,9,20,15,'Venta a cliente corporativo','2025-11-05 13:40:00'),(10,10,45,50,'Reposición de memorias
USB','2025-11-05 16:30:00'),(11,1,8,48,'Compra a proveedor principal (COMPRA)','2025-11-09 17:51:41'),(12,1,48,3,'Venta
online (VENTA)','2025-11-09 17:52:27'),(13,2,40,39,'PRUEBA (VENTA)','2025-11-10 20:09:47'),(14,2,39,36,'COMPRA
(VENTA)','2025-11-10 23:11:08'),(15,1,3,0,'COMPRA (VENTA)','2025-11-10 23:11:08'),(16,12,0,40,'Compra a proveedor
principal (COMPRA)','2025-11-11 21:13:36'),(17,1,0,40,'Compra a proveedor principal (COMPRA)','2025-11-11
16:57:33'),(18,1,40,37,'COMPRA (VENTA)','2025-11-11 22:30:57'),(19,2,36,33,'COMPRA (VENTA)','2025-11-11
22:30:57'),(20,1,37,34,'COMPRA (VENTA)','2025-11-11 22:43:03'),(21,2,33,30,'COMPRA (VENTA)','2025-11-11
22:43:03'),(22,1,34,31,'COMPRA (VENTA)','2025-11-11 22:43:39'),(23,2,30,27,'COMPRA (VENTA)','2025-11-11
22:43:39'),(24,1,31,28,'COMPRA (VENTA)','2025-11-12 00:06:59'),(25,2,27,24,'COMPRA (VENTA)','2025-11-12
00:06:59'),(26,12,40,0,'Compra a proveedor principal (VENTA)','2025-11-12 20:55:37'),(27,5,6,4,'VENTA (VENTA)','2025-11-12
22:10:38'),(28,2,24,18,'VENTA (VENTA)','2025-11-12 22:10:43'),(29,12,0,40,'Compra a proveedor principal (COMPRA)','2025-
11-13 00:37:44');
/*!40000 ALTER TABLE `historico_inventario` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `inventario`
--

/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `inventario` (
  `id` int NOT NULL AUTO_INCREMENT,
  `producto_id` int DEFAULT NULL,
  `stock_actual` int DEFAULT NULL,
  `umbral_minimo` int DEFAULT NULL,
  `fecha_actualizacion` datetime DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `id` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=24 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `inventario`
--

LOCK TABLES `inventario` WRITE;
/*!40000 ALTER TABLE `inventario` DISABLE KEYS */;
INSERT INTO `inventario` VALUES (1,1,28,20,'2025-11-12 00:06:59'),(2,2,18,10,'2025-11-12 22:10:43'),(3,3,4,10,'2025-11-05
```

```
09:20:00'),(4,4,12,10,'2025-11-05 09:30:00'),(5,5,4,20,'2025-11-12 22:10:38'),(6,6,20,15,'2025-11-05 09:50:00'),(7,7,15,10,'2025-11-05 10:00:00'),(8,8,2,15,'2025-11-05 10:10:00'),(9,9,15,15,'2025-11-05 10:20:00'),(10,10,50,25,'2025-11-05 10:30:00'),(11,11,0,5,'2025-11-09 16:45:42'),(12,12,40,10,'2025-11-13 00:37:44'),(13,13,35,10,'2025-11-11 21:23:40'),(19,19,0,10,'2025-11-11 16:46:17'),(22,22,20,5,'2025-11-12 17:54:25'),(23,23,15,5,'2025-11-13 00:32:30');
/*!40000 ALTER TABLE `inventario` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `orden`
--

/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `orden` (
  `id` int NOT NULL AUTO_INCREMENT,
  `correo` varchar(100) DEFAULT NULL,
  `nombre_cliente` varchar(100) DEFAULT NULL,
  `direccion_entrega` varchar(200) DEFAULT NULL,
  `estado` varchar(15) DEFAULT NULL,
  `subtotal` decimal(10,2) DEFAULT NULL,
  `fecha_creacion` datetime DEFAULT NULL,
  `fecha_venta` datetime DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=16 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `orden`
--

LOCK TABLES `orden` WRITE;
/*!40000 ALTER TABLE `orden` DISABLE KEYS */;
INSERT INTO `orden` VALUES (1,'carlos@gmail.com','Carlos Pérez','San Salvador','COMPLETADA',875.00,'2025-11-01 08:00:00','2025-11-01 09:00:00'),(2,'maria@yahoo.com','María López','Santa Tecla','COMPLETADA',320.00,'2025-11-02 10:00:00','2025-11-02 11:30:00'),(3,'juan@outlook.com','Juan Torres','Soyapango','ABANDONADA',160.00,'2025-11-02 15:00:00',NULL),(4,'ana@gmail.com','Ana Hernández','San Miguel','COMPLETADA',70.00,'2025-11-03 09:00:00','2025-11-03 10:00:00'),(5,'roberto@empresa.com','Roberto Díaz','Santa Ana','COMPLETADA',299.00,'2025-11-03 12:00:00','2025-11-03 13:00:00'),(6,'laura@gmail.com','Laura Ramos','Sonsonate','COMPLETADA',230.00,'2025-11-03 14:00:00','2025-11-03 15:00:00'),(7,'jose@correo.com','José Martínez','San Vicente','COMPLETADA',145.00,'2025-11-04 09:00:00','2025-11-04 09:30:00'),(8,'carmen@correo.com','Carmen Salazar','Usulután','COMPLETADA',60.00,'2025-11-04 10:00:00','2025-11-04 10:30:00'),(9,'ricardo@work.com','Ricardo Mejía','Chalatenango','ABANDONADA',299.00,'2025-11-04 11:00:00',NULL),(10,'sandra@gmail.com','Sandra Gómez','La Unión','COMPLETADA',170.00,'2025-11-05 08:00:00','2025-11-05 09:00:00'),(12,'cliente@correo.com','Juan Pérez','Calle Los Pinos #123, San Salvador','PENDIENTE',1860.00,'2025-11-12 00:00:00',NULL),(13,'cliente@correo.com','Juan Perez','Calle Los Pinos #123, San Salvador','VENDIDO',335.00,'2025-11-12 00:00:00','2025-11-12 00:00:00'),(14,'cliente@correo.com','Juan Pérez','Calle Los Pinos #123, San Salvador','PENDIENTE',985.50,'2025-11-12 00:00:00',NULL),(15,'cliente@correo.com','Juan Pérez','Calle Los Pinos #123, San Salvador','PENDIENTE',135.50,'2025-11-12 00:00:00',NULL);
/*!40000 ALTER TABLE `orden` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `producto`
--

/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `producto` (
  `id` int NOT NULL AUTO_INCREMENT,
  `nombre` varchar(100) DEFAULT NULL,
  `descripcion` varchar(300) DEFAULT NULL,
  `precio` decimal(10,2) DEFAULT NULL,
  `categoria` varchar(300) DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `id` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=24 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `producto`
--

LOCK TABLES `producto` WRITE;
/*!40000 ALTER TABLE `producto` DISABLE KEYS */;
INSERT INTO `producto` VALUES (1,'Laptop Lenovo ThinkPad','Laptop 14\" Core i5, 8GB RAM, SSD 512GB',850.00,'Tecnología'),(2,'Mouse Logitech M90','Mouse óptico USB de alta precisión',12.50,'Accesorios'),(3,'Teclado Mecánico Redragon','Teclado mecánico retroiluminado RGB',55.00,'Accesorios'),(4,'Monitor Samsung 24\"','Monitor LED 24\" Full HD',160.00,'Pantallas'),(5,'Silla ergonómica','Silla de oficina ergonómica con soporte lumbar',130.00,'Mobiliario'),(6,'Disco Duro Externo 1TB','Disco duro portátil USB 3.0',70.00,'Almacenamiento'),(7,'Auriculares Sony WH-1000XM4','Auriculares
```

```
inalámbricos con cancelación de ruido',129.00,'Audio'),(8,'Impresora HP DeskJet 4155','Impresora multifuncional con Wi-Fi',145.00,'Oficina'),(9,'Router TP-Link Archer C6','Router inalámbrico doble banda',60.00,'Redes'),(10,'Memoria USB 64GB','Memoria flash USB 3.0',15.00,'Almacenamiento'),(11,'Cafe','Sobres',2.50,'Caja'),(12,'Teclado Mecánico RGB','Teclado gamer con switches azules',65.50,'Periféricos'),(13,'Silla Ergonómica Ejecutiva','Silla Ejecutiva con soporte lumbar y ajuste de altura',189.90,'Muebles'),(19,'Teclado Mecánico','Teclado gamer con switches azules',65.50,'Periféricos'),(22,'Monitor Samsung 32\"','Monitor LED 24\" Full HD',325.00,'Pantallas'),(23,'Monitor ASUS TUF Gaming','Monitor de 27 pulgadas con 165Hz',199.99,'Monitores');
/*!40000 ALTER TABLE `producto` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Temporary view structure for view `vInventario`
--

SET @saved_cs_client     = @@character_set_client;
/*!50503 SET character_set_client = utf8mb4 */;
/*!50001 CREATE VIEW `vInventario` AS SELECT
 1 AS `producto_id`,
 1 AS `producto`,
 1 AS `stock_actual`,
 1 AS `umbral_minimo`,
 1 AS `faltante`,
 1 AS `fecha_actualizacion`*/;
SET character_set_client = @saved_cs_client;

--
-- Temporary view structure for view `vOrdenes`
--

SET @saved_cs_client     = @@character_set_client;
/*!50503 SET character_set_client = utf8mb4 */;
/*!50001 CREATE VIEW `vOrdenes` AS SELECT
 1 AS `orden_id`,
 1 AS `nombre_cliente`,
 1 AS `correo`,
 1 AS `direccion_entrega`,
 1 AS `estado`,
 1 AS `fecha_creacion`,
 1 AS `fecha_venta`,
 1 AS `productos`,
 1 AS `total`*/;
SET character_set_client = @saved_cs_client;

--
-- Temporary view structure for view `vVentas`
--

SET @saved_cs_client     = @@character_set_client;
/*!50503 SET character_set_client = utf8mb4 */;
/*!50001 CREATE VIEW `vVentas` AS SELECT
 1 AS `producto_id`,
 1 AS `producto`,
 1 AS `total_vendido`,
 1 AS `total_ventas`,
 1 AS `cliente`,
 1 AS `numero_compras`,
 1 AS `total_gastado`,
 1 AS `fecha_venta`*/;
SET character_set_client = @saved_cs_client;

--
-- Dumping routines for database 'arka'
--

--
-- Final view structure for view `vInventario`
--

/*!50001 DROP VIEW IF EXISTS `vInventario`*/;
/*!50001 SET @saved_cs_client          = @@character_set_client */;
/*!50001 SET @saved_cs_results         = @@character_set_results */;
/*!50001 SET @saved_col_connection     = @@collation_connection */;
/*!50001 SET character_set_client      = utf8mb4 */;
/*!50001 SET character_set_results     = utf8mb4 */;
/*!50001 SET collation_connection      = utf8mb4_0900_ai_ci */;
/*!50001 CREATE ALGORITHM=UNDEFINED */
/*!50013 DEFINER=`admin`@`%` SQL SECURITY DEFINER */
/*!50001 VIEW `vInventario` AS select `p`.`id` AS `producto_id`,`p`.`nombre` AS `producto`,`i`.`stock_actual` AS
```

```
`stock_actual`,`i`.`umbral_minimo` AS `umbral_minimo`,(`i`.`umbral_minimo` - `i`.`stock_actual`) AS
`faltante`,`i`.`fecha_actualizacion` AS `fecha_actualizacion` from (`inventario` `i` join `producto` `p` on((`p`.`id` =
`i`.`producto_id`))) where (`i`.`stock_actual` < `i`.`umbral_minimo`) order by (`i`.`umbral_minimo` - `i`.`stock_actual`) desc */;
/*!50001 SET character_set_client      = @saved_cs_client */;
/*!50001 SET character_set_results     = @saved_cs_results */;
/*!50001 SET collation_connection      = @saved_col_connection */;

--
-- Final view structure for view `vOrdenes`
--

/*!50001 DROP VIEW IF EXISTS `vOrdenes`*/;
/*!50001 SET @saved_cs_client          = @@character_set_client */;
/*!50001 SET @saved_cs_results         = @@character_set_results */;
/*!50001 SET @saved_col_connection     = @@collation_connection */;
/*!50001 SET character_set_client      = utf8mb4 */;
/*!50001 SET character_set_results     = utf8mb4 */;
/*!50001 SET collation_connection      = utf8mb4_0900_ai_ci */;
/*!50001 CREATE ALGORITHM=UNDEFINED */
/*!50013 DEFINER=`admin`@`%` SQL SECURITY DEFINER */
/*!50001 VIEW `vOrdenes` AS select `o`.`id` AS `orden_id`,`o`.`nombre_cliente` AS `nombre_cliente`,`o`.`correo` AS
`correo`,`o`.`direccion_entrega` AS `direccion_entrega`,`o`.`estado` AS `estado`,`o`.`fecha_creacion` AS
`fecha_creacion`,`o`.`fecha_venta` AS `fecha_venta`,group_concat(`p`.`nombre` separator ', ') AS
`productos`,sum((`d`.`cantidad` * `d`.`precio_unitario`)) AS `total` from ((`orden` `o` left join `detalleorden` `d` on((`d`.`orden_id` =
`o`.`id`))) left join `producto` `p` on((`p`.`id` = `d`.`producto_id`))) group by `o`.`id`,`o`.`nombre_cliente`,`o`.`correo`,`o`.`estado`
order by `o`.`fecha_creacion` desc */;
/*!50001 SET character_set_client      = @saved_cs_client */;
/*!50001 SET character_set_results     = @saved_cs_results */;
/*!50001 SET collation_connection      = @saved_col_connection */;

--
-- Final view structure for view `vVentas`
--

/*!50001 DROP VIEW IF EXISTS `vVentas`*/;
/*!50001 SET @saved_cs_client          = @@character_set_client */;
/*!50001 SET @saved_cs_results         = @@character_set_results */;
/*!50001 SET @saved_col_connection     = @@collation_connection */;
/*!50001 SET character_set_client      = utf8mb4 */;
/*!50001 SET character_set_results     = utf8mb4 */;
/*!50001 SET collation_connection      = utf8mb4_0900_ai_ci */;
/*!50001 CREATE ALGORITHM=UNDEFINED */
/*!50013 DEFINER=`admin`@`%` SQL SECURITY DEFINER */
/*!50001 VIEW `vVentas` AS select `p`.`id` AS `producto_id`,`p`.`nombre` AS `producto`,sum(`d`.`cantidad`) AS
`total_vendido`,sum((`d`.`cantidad` * `d`.`precio_unitario`)) AS `total_ventas`,`o`.`nombre_cliente` AS `cliente`,count(distinct
`o`.`id`) AS `numero_compras`,sum(`o`.`subtotal`) AS `total_gastado`,cast(`o`.`fecha_venta` as date) AS `fecha_venta` from
((`detalleorden` `d` join `producto` `p` on((`p`.`id` = `d`.`producto_id`))) join `orden` `o` on((`o`.`id` = `d`.`orden_id`))) where
((`o`.`estado` = 'COMPLETADA') and (`o`.`fecha_venta` is not null)) group by `p`.`id`,`o`.`nombre_cliente`,cast(`o`.`fecha_venta`
as date) order by `fecha_venta` desc,`total_ventas` desc */;
/*!50001 SET character_set_client      = @saved_cs_client */;
/*!50001 SET character_set_results     = @saved_cs_results */;
/*!50001 SET collation_connection      = @saved_col_connection */;
SET @@SESSION.SQL_LOG_BIN = @MYSQLDUMP_TEMP_LOG_BIN;
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;

-- Dump completed on 2025-11-13 17:54:43
```

Codigo base

Estructura del microservicio

Para el desarrollo de los microservicios el diseño de software utilizado es el de Arquitectura Limpia como se muestra a continucacion:

```
com.enyoi.inventario
 ├── application
 │    ├── dto/
 │    │    ├── ProductoDTO.java
 │    │    ├── InventarioDTO.java
 │    │    └── HistoricoInventarioDTO.java
 │    └── mapper/
 │         └── InventarioMapper.java
 │    └── service/
 │         └── InventarioService.java
 ├── infrastructure/
 │    ├── entity/
 │    │    ├── ProductoEntity.java
 │    │    ├── InventarioEntity.java
 │    │    └── HistoricoInventarioEntity.java
 │    └── repository/
 │         ├── ProductoRepository.java
 │         ├── InventarioRepository.java
 │         └── HistoricoInventarioRepository.java
 └── controller/
     └── InventarioController.java
```

Codigo fuente

HistoricoInventarioEntity.java

```java
package com.enyoi.inventario.infrastructure.entity;

import lombok.Getter;
import lombok.Setter;
import org.springframework.data.annotation.Id;
import org.springframework.data.relational.core.mapping.Table;

import java.time.LocalDateTime;
```

```java
@Table("historico_inventario")
@Getter
@Setter
public class HistoricoInventarioEntity {
    @Id
    private int id;
    private int productoId;
    private int cantidadAnterior;
    private int cantidadNueva;
    private String motivo;
    private LocalDateTime fechaCambio;

}
```

HistoricoInventarioRepository.java

```java
package com.enyoi.inventario.infrastructure.repository;

import org.springframework.data.repository.reactive.ReactiveCrudRepository;
import com.enyoi.inventario.infrastructure.entity.HistoricoInventarioEntity;

public interface HistoricoInventarioRepository extends
ReactiveCrudRepository<HistoricoInventarioEntity, Integer> {
}
```

InventarioAppService.java

```java
package com.enyoi.inventario.application.service;

import io.r2dbc.spi.ConnectionFactory;
import org.springframework.data.r2dbc.core.R2dbcEntityTemplate;
import org.springframework.r2dbc.connection.R2dbcTransactionManager;
import org.springframework.stereotype.Service;
import org.springframework.transaction.ReactiveTransactionManager;
import org.springframework.transaction.reactive.TransactionalOperator;
import reactor.core.publisher.Mono;
import reactor.core.publisher.Flux;

import java.time.LocalDateTime;
import java.util.UUID;

import com.enyoi.inventario.application.dto.*;
import com.enyoi.inventario.application.mapper.InventarioMapper;
import com.enyoi.inventario.infrastructure.entity.*;
import com.enyoi.inventario.infrastructure.repository.*;

@Service
public class InventarioAppService {
    private final ProductoRepository productoRepo;
    private final InventarioRepository inventarioRepo;
    private final R2dbcEntityTemplate template;
    private final TransactionalOperator txOperator;

    public InventarioAppService(R2dbcEntityTemplate template, ProductoRepository productoRepo,
InventarioRepository inventarioRepo, ConnectionFactory connectionFactory) {
        this.productoRepo = productoRepo;
        this.inventarioRepo = inventarioRepo;
        this.template = template;
```

```
        ReactiveTransactionManager tm = new R2dbcTransactionManager(connectionFactory);
        this.txOperator = TransactionalOperator.create(tm);
    }

    public Mono<ProductoDTO> registrarProducto(ProductoInveDTO dto) {
        ProductoDTO prod = new ProductoDTO();
        prod.setNombre(dto.getNombre());
        prod.setDescripcion(dto.getDescripcion());
        prod.setPrecio(dto.getPrecio());
        prod.setCategoria(dto.getCategoria());

        ProductoEntity e = InventarioMapper.toEntity(prod);

        return productoRepo.existsByNombre(e.getNombre())
                .flatMap(exists -> {
                    if (exists) {
                        return Mono.error(new IllegalArgumentException(
                                "Ya existe un producto registrado con el nombre: " + e.getNombre()
                        ));
                    }

                    return template.insert(ProductoEntity.class)
                            .using(e)
                            .flatMap(saved -> {
                                InventarioEntity inv = new InventarioEntity();
                                inv.setProductoId(saved.getId());
                                inv.setStockActual(dto.getStock());
                                inv.setUmbralMinimo(dto.getUmbralMinimo());
                                inv.setFechaActualizacion(LocalDateTime.now());
                                return template.insert(InventarioEntity.class)
                                        .using(inv)
                                        .thenReturn(InventarioMapper.toDTO(saved));
                            });
                }).as(txOperator::transactional);
    }

    public Mono<AjusteResponseDTO> ajustarStock(int productoId, String accion, int cantidad, String
motivo) {
        return inventarioRepo.findByProductoId(productoId)
                .flatMap(inv -> {
                    int stockAnterior = inv.getStockActual();
                    int nuevoStock = stockAnterior;

                    // Lógica de acción                    if ("COMPRA".equalsIgnoreCase(accion)) {
                        nuevoStock += cantidad;
                    } else if ("VENTA".equalsIgnoreCase(accion)) {
                        nuevoStock -= cantidad;
                        if (nuevoStock < 0) {
                            return Mono.error(new IllegalArgumentException("La cantidad del producto supera el
stock actual"));
                        }
                    } else {
                        return Mono.error(new IllegalArgumentException("Acción no válida (solo COMPRA o
VENTA)"));
                    }

                    inv.setStockActual(nuevoStock);
                    inv.setFechaActualizacion(LocalDateTime.now());

                    // Guardar cambio en inventario                    int finalNuevoStock = nuevoStock;
                    return inventarioRepo.save(inv)
```

```
                    .flatMap(saved -> {
                        // Guardar en histórico                        HistoricoInventarioEntity h = new
HistoricoInventarioEntity();
                        h.setProductoId(saved.getProductoId());
                        h.setCantidadAnterior(stockAnterior);
                        h.setCantidadNueva(finalNuevoStock);
                        h.setMotivo(motivo + " (" + accion + ")");
                        h.setFechaCambio(LocalDateTime.now());

                        return template.insert(HistoricoInventarioEntity.class)
                                .using(h)
                                // ◈ Recuperar el producto para armar la
respuesta                                        .then(productoRepo.findById(saved.getProductoId()))
                                        .map(prod -> new AjusteResponseDTO(
                                                prod.getId(),
                                                prod.getNombre(),
                                                stockAnterior,
                                                finalNuevoStock,
                                                motivo,
                                                accion.toUpperCase(),
                                                LocalDateTime.now()
                                        ))
                                );
                    });
        });
    }

    public Flux<InventarioDTO> listarInventarioBajo() {
        return inventarioRepo.findAll()
                .filter(inv -> inv.getStockActual() < inv.getUmbralMinimo()).map(InventarioMapper::toDTO)
                .flatMap(inv ->
                        productoRepo.findById(inv.getProductoId())
                                .map(prod -> {
                                    InventarioDTO dto = new InventarioDTO();
                                    dto.setId(inv.getId());
                                    dto.setProductoId(prod.getId());
                                    dto.setStockActual(inv.getStockActual());
                                    dto.setUmbralMinimo(inv.getUmbralMinimo());
                                    dto.setFechaActualizacion(inv.getFechaActualizacion());

                                    // ◈ Asignamos también los datos del producto
dto.setNombreProducto(prod.getNombre());
                                    dto.setDescripcionProducto(prod.getDescripcion());
                                    dto.setPrecioProducto(prod.getPrecio());
                                    dto.setCategoriaProducto(prod.getCategoria());

                                    return dto;
                                })
                );
    }

    public Flux<ProductoDTO> listarProductos() {
        return productoRepo.findAll().map(InventarioMapper::toDTO);
    }

    public Mono<InventarioDTO> obtenerInventarioPorProductoId(int productoId) {
        return inventarioRepo.findByProductoId(productoId)
                .switchIfEmpty(Mono.error(new IllegalArgumentException(
                        "No se encontró inventario asociado al producto con ID: " + productoId)))
                .flatMap(inv ->
                        productoRepo.findById(productoId)
                                .map(prod -> {
```

```
                        InventarioDTO dto = new InventarioDTO();
                        dto.setId(inv.getId());
                        dto.setProductoId(prod.getId());
                        dto.setStockActual(inv.getStockActual());
                        dto.setUmbralMinimo(inv.getUmbralMinimo());
                        dto.setFechaActualizacion(inv.getFechaActualizacion());

                        // 🔷 Asignamos también los datos del producto
dto.setNombreProducto(prod.getNombre());
                        dto.setDescripcionProducto(prod.getDescripcion());
                        dto.setPrecioProducto(prod.getPrecio());
                        dto.setCategoriaProducto(prod.getCategoria());

                        return dto;
                    })
            );
    }

    public Flux<InventarioDTO> listarInventario() {
        return inventarioRepo.findAll()
                .flatMap(inv ->
                        productoRepo.findById(inv.getProductoId())
                                .map(prod -> {
                        InventarioDTO dto = new InventarioDTO();
                        dto.setId(inv.getId());
                        dto.setProductoId(prod.getId());
                        dto.setStockActual(inv.getStockActual());
                        dto.setUmbralMinimo(inv.getUmbralMinimo());
                        dto.setFechaActualizacion(inv.getFechaActualizacion());

                        // 🔷 Asignamos también los datos del producto
dto.setNombreProducto(prod.getNombre());
                        dto.setDescripcionProducto(prod.getDescripcion());
                        dto.setPrecioProducto(prod.getPrecio());
                        dto.setCategoriaProducto(prod.getCategoria());

                        return dto;
                    })
            );
    }
}
```

InventarioController.java

```
package com.enyoi.inventario.controller;

import com.enyoi.inventario.application.dto.*;
import jakarta.validation.Valid;
import org.springframework.validation.annotation.Validated;
import org.springframework.web.bind.annotation.*;
import org.springframework.http.MediaType;
import reactor.core.publisher.Mono;
import reactor.core.publisher.Flux;

import com.enyoi.inventario.application.service.InventarioAppService;

@RestController
@RequestMapping("/api")
@Validated
public class InventarioController {
```

```
    private final InventarioAppService service;

    public InventarioController(InventarioAppService service) {
        this.service = service;
    }

    /**     * Crea un nuevo producto (y su inventario inicial con stock 0)     */    @PostMapping(
            value = "/inventario/producto",
            consumes = MediaType.APPLICATION_JSON_VALUE,
            produces = MediaType.APPLICATION_JSON_VALUE     )
    public Mono<ProductoDTO> crearProducto(@Valid @RequestBody ProductoInveDTO dto) {
        return service.registrarProducto(dto);
    }

    /**     * Lista todos los productos registrados     */    @GetMapping(value = "/inventario/productos",
produces = MediaType.APPLICATION_JSON_VALUE)
    public Flux<ProductoDTO> listarProductos() {
        return service.listarProductos();
    }

    /**     * Obtiene los inventarios con stock bajo (menor al umbral)     */    @GetMapping(value =
"/inventario/bajo", produces = MediaType.APPLICATION_JSON_VALUE)
    public Flux<InventarioDTO> inventarioBajo() {
        return service.listarInventarioBajo();
    }

    /**     * Obtiene el inventario de un producto por su ID     */    @GetMapping(value =
"/inventario/{idProducto}", produces = MediaType.APPLICATION_JSON_VALUE)
    public Mono<InventarioDTO> obtenerInventario(@PathVariable int idProducto) {
        return service.obtenerInventarioPorProductoId(idProducto);
    }

    /**     * Ajusta el stock de un inventario específico     */    @PostMapping(value =
"/inventario/{idProducto}/ajustar", consumes = MediaType.APPLICATION_JSON_VALUE)
    public Mono<AjusteResponseDTO> ajustarStock(@PathVariable int idProducto, @Valid
@RequestBody AjusteInventarioDTO ajusteDto) {
        return service.ajustarStock(idProducto, ajusteDto.getAccion(), ajusteDto.getCantidad(),
ajusteDto.getMotivo());
    }

    /**     * Obtiene todo el invetario     */    @GetMapping(value = "/inventario", produces =
MediaType.APPLICATION_JSON_VALUE)
    public Flux<InventarioDTO> obtenerInventario() {
        return service.listarInventario();
    }
}
```

Link del repositorio de los microservicios:

https://github.com/MMRaul/JavaBackendDeveloper