

Eco Market

Integrantes: Miguel Laguela

Rodrigo Valdes

Matias Torres

Asignatura: Full Stack 1

Fecha entrega: 26/05/2025

**Índice**

1. **introduccion …………………………………………. page 3**
2. **diagrama de microsevicio……………………...….. page 3**
3. **estructura del proyecto…………………………….. page 4**
4. **base de datos…………………………..…………….. page 6**
5. **implementación de los servicios…………………..page 8**
6. **implementación de vistas……………………….…..page 13**
7. **git hub………………………………………...…….…..page 15**
8. **conclusion………………………………………….…..page 15**

**1. Introduccion:**

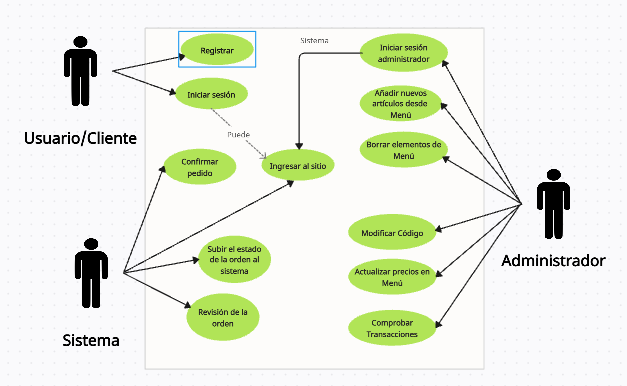
La empresa Ecomarket Spa es una compañía chilena que vende productos ecológicos, la cual ha sido sorprendida con un aumento significativo en su crecimiento desde la tienda local hacia otras tiendas al exterior. Debido a este desarrollo es que se refleja lo acotado de su sistema monolítico, presentando problemas de servicio al cliente, rendimiento y disponibilidad.

Por ende, tenemos la determinación de abordar los problemas con un servicio de arquitectura basada en microservicios, usando elementos tales como spring boot y la base de datos de mysql workbench. El framework spring boot gestiona las entidades dentro de la compañía con el uso de tres api rest (interface) y en caso de la base de datos sería el almacenamiento de información que la app de servicio necesita

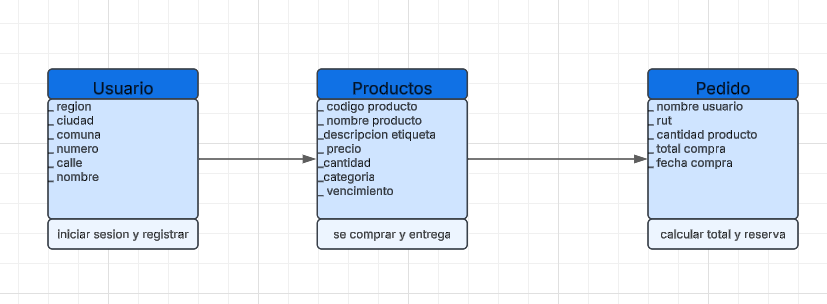
El objetivo es llegar a una solución definitiva siendo esta la migración de una arquitectura de monolítica a microservicio y ahora en la segunda parte utilizando spring y mysql para llegar al objetivo planteado.

**2. Diagrama de microservicio:**

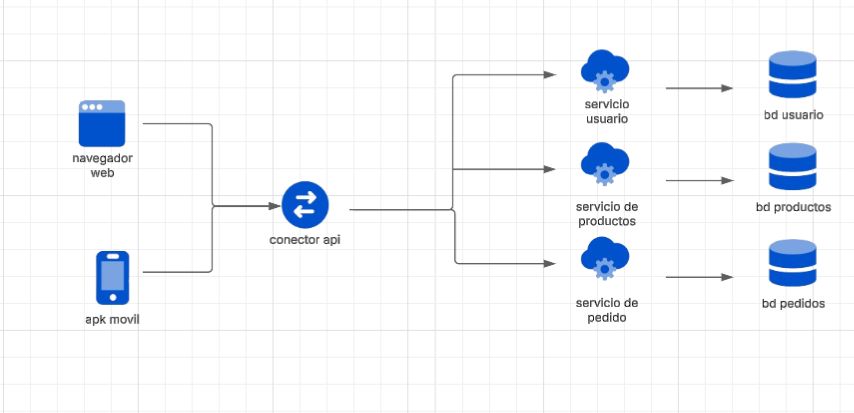
* Diagrama de Casos de Uso



* Diagrama de clases



* Diagrama de Despliegue



**3. Estructura del Proyecto:**

su estructura va primero con maven, dependencias, componentes que se implementan

pom.xml -> aquí se encuentran las dependencias del trabajo,Spring Jpa, Spring Web, Spring DevTools, crea un ejecutable de la app y su conector a base de datos mysql

com.trabajo.springboot.ecomarket.ecomarket.EcomarketApplication -> esta ruta es la clase principal que se inicia de los microservicios

com.trabajo.springboot.ecomarket.ecomarket.controllers.ProductoController.java -> es un controlador que ocupa la interface producto dentro del servicio para realizar acciones como eliminar crear o modificar los datos de la tabla producto.

com.trabajo.springboot.ecomarket.ecomarket.entities.Producto.java -> es la entidad de jpa en la cual se encuentra la tabla de producto, aquí se encuentran los atributos como por ejemplo nombre,descripcion,precio,stock y los mapas de la tabla

com.trabajo.springboot.ecomarket.ecomarket.repository.ProductoRepository.java -> es una interfaz con JpaRepository, tiene ciertos metodos por defecto para la entidad de Producto en este caso

com.trabajo.springboot.ecomarket.ecomarket.service.ProductoServiceImpl.java -> es una clase de servicio para gestion de la entidad Producto

com.trabajo.springboot.ecomarket.ecomarket.controllers.PedidoController.java -> es un controlador que ocupa la interface pedida dentro de servicio para realizar acciones como eliminar crear o modificar los datos de la tabla pedido.

com.trabajo.springboot.ecomarket.ecomarket.entities.Pedido.java -> es la entidad de jpa en la cual se encuentra la tabla de pedido, aquí se encuentran los atributos como por ejemplo fecha creacion,estado,direccion envió y los mapas de la tabla

com.trabajo.springboot.ecomarket.ecomarket.repository.PedidoRepository.java -> es una interfaz con JpaRepository, tiene ciertos métodos por defecto para la entidad de Pedido en este caso

com.trabajo.springboot.ecomarket.ecomarket.service.PedidoServiceImpl.java -> es una clase de servicio para gestión de la entidad Pedido

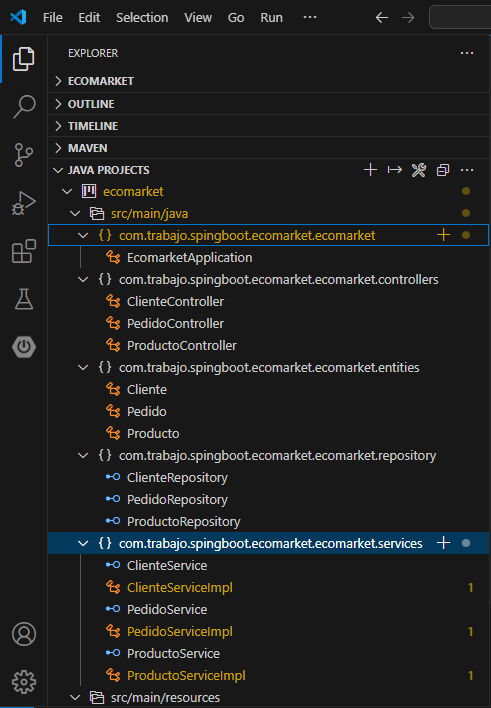
com.trabajo.springboot.ecomarket.ecomarket.controllers.ClienteController.java -> es un controlador que ocupa la interface pedida dentro de servicio para realizar acciones como eliminar crear o modificar los datos de la tabla cliente.

com.trabajo.springboot.ecomarket.ecomarket.entities.Cliente.java -> es la entidad de jpa en la cual se encuentra la tabla de cliente, aquí se encuentran los atributos como por ejemplo nombre, email, dirección y los mapas de la tabla

com.trabajo.springboot.ecomarket.ecomarket.repository.ClienteRepository.java -> es una interfaz con JpaRepository, tiene ciertos métodos por defecto para la entidad de Pedido en este caso

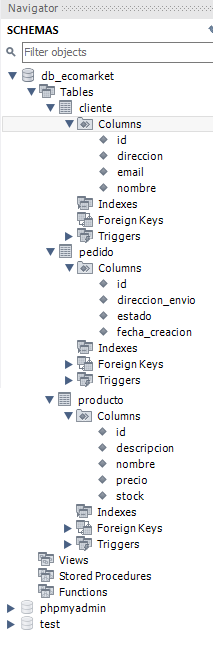
com.trabajo.springboot.ecomarket.ecomarket.service.ClienteServiceImpl.java -> es una clase de servicio para gestión de la entidad Cliente

src/main/resources/application.properties -> aquí es donde se hace la conexión con mysql ósea la base de datos



**4. Base de datos:**

El motor utilizado de base de datos es mysql ya que este tiene una conexión con spring boot



La tabla productos contiene:

id: clave primaria y es incremental

nombre: nombre del producto tipo String

descripción: el detalle de cada producto tipo String

precio: el precio del producto tipo Integer

stock: la cantidad disponible dentro de la tienda tipo Integer

La tabla cliente contiene:

id: clave primaria y es incremental

nombre: nombre del producto tipo String

dirección: la dirección del cliente tipo String

email: el correo del cliente tipo String

La tabla pedido contiene:

id: clave primaria y es incremental

fecha de creación: la fecha de creación del pedido tipo local date time

estado: el estado del pedido tipo String

dirección envió: la dirección de envío tipo String

**5. Implementacion de los servicios:**

Post -> /api/nombre tabla: crea nueva información de alguna de las tablas

Get -> /api/nombre tabla: se pide lista de la tabla

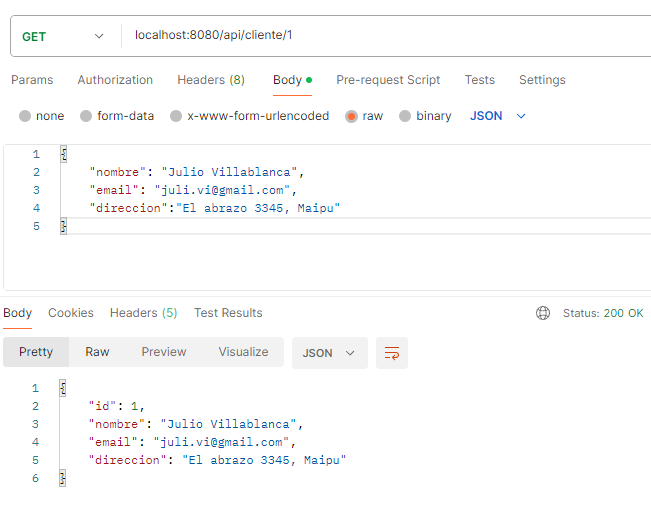
Get -> /api/nombre tabla/id: se puede pedir información de la tabla a través de digito numérico de la id

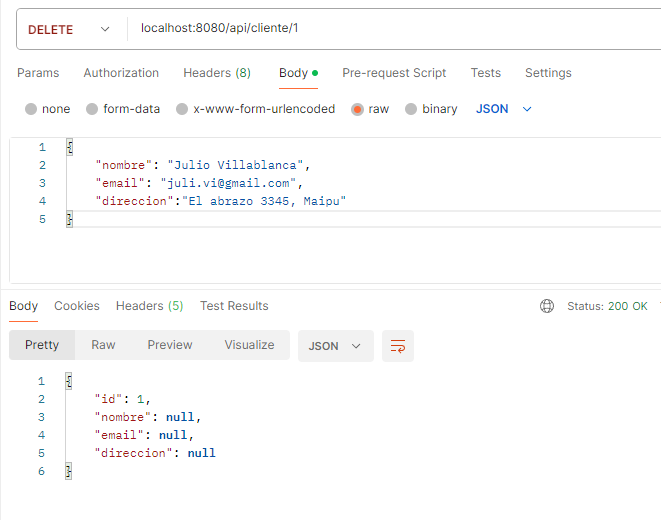
Delete -> /api/nombre tabla/id: elimina cualquier información dentro de una tabla en específico a través de su id

Ejemplo: Generación de un cliente (Post -> /api/cliente)

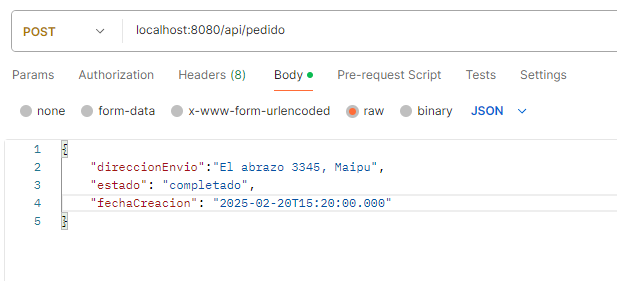


Ejemplo: Mostrar un cliente en la id 1 (Get -> /api/cliente/1)

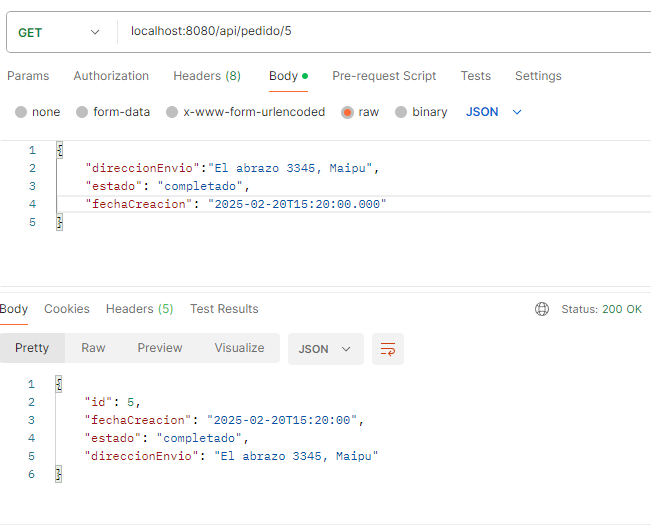


Ejemplo: Eliminar un cliente en la id 1 (Delete -> /api/cliente/1) 

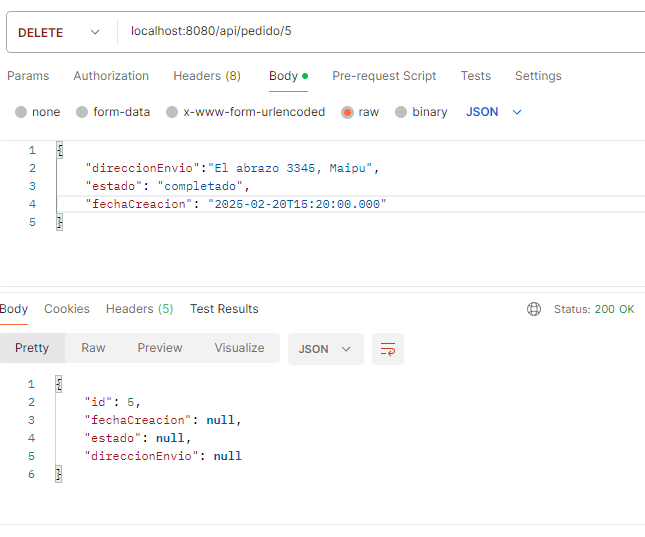
Ejemplo: Generación de un pedido (Post -> /api/pedido)



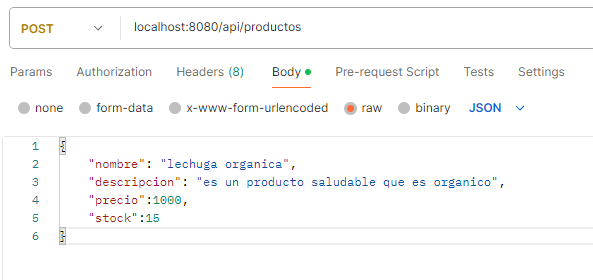
Ejemplo: Mostrar un pedido en la id 3 (Get -> /api/pedido/5)



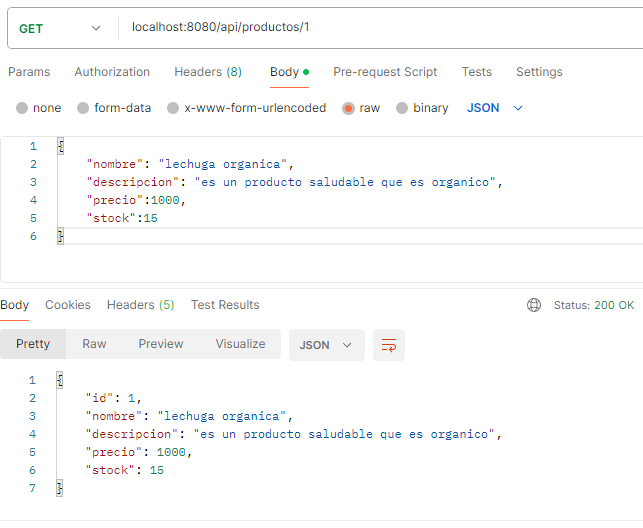
Ejemplo: Eliminar un pedido (Delete -> /api/pedido/3)



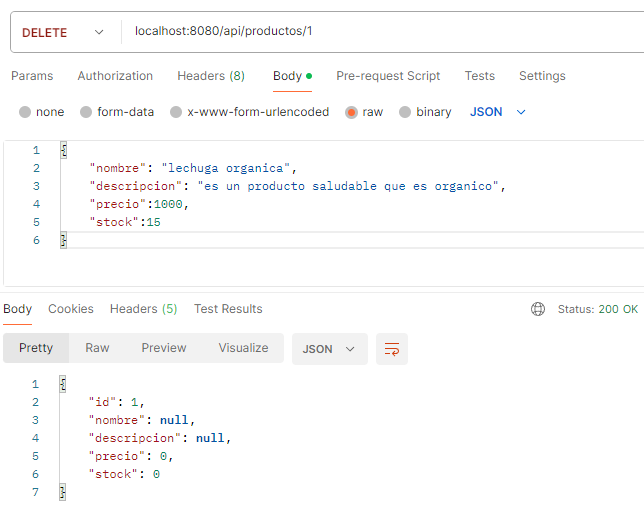
Ejemplo: Generación de un producto (Post -> /api/productos)



Ejemplo: Petición de un producto (Get -> /api/productos/1)



Ejemplo: Eliminar un producto (Delete -> /api/productos/1)

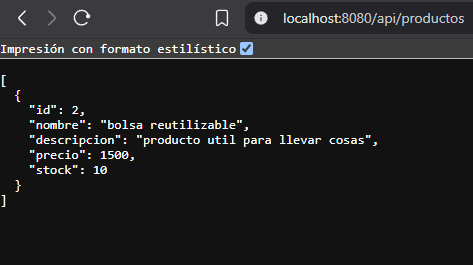


capturas para el api rest de los otros dos microservicios (pedidos y cliente)

**6. Implementacion de vistas:**

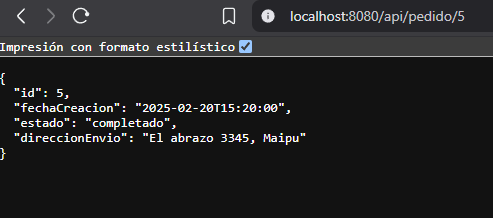
Dentro del programa para una mayor comodidad y facilidad se puede ver los datos a través de petición get con vistas desde la web de manera local

Ejemplo: Vista para lista de productos



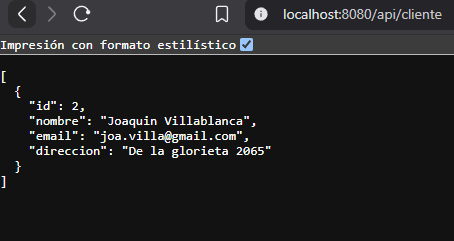
al acceder a la url localhost:8080/api/productos la aplicación backend debería pedir los datos con un get al endpoint de cada uno de los microservicios /api/productos y daría la respuesta dentro del navegador yendo a buscar la información que contiene productos.

Ejemplo: Vista de pedido



al acceder a la url localhost:8080/api/pedido/5 la aplicación backend debería pedir los datos con un get al endpoint de cada microservicio en este caso sería del /api/pedido/5 y sale en el navegador.

Ejemplo: Vista de cliente



al acceder a la url localhost:8080/api/cliente la aplicación backend debería pedir los datos con un get al endpoint de cada microservicio y en este caso del /api/cliente y mostraría en el navegador la información que contiene.

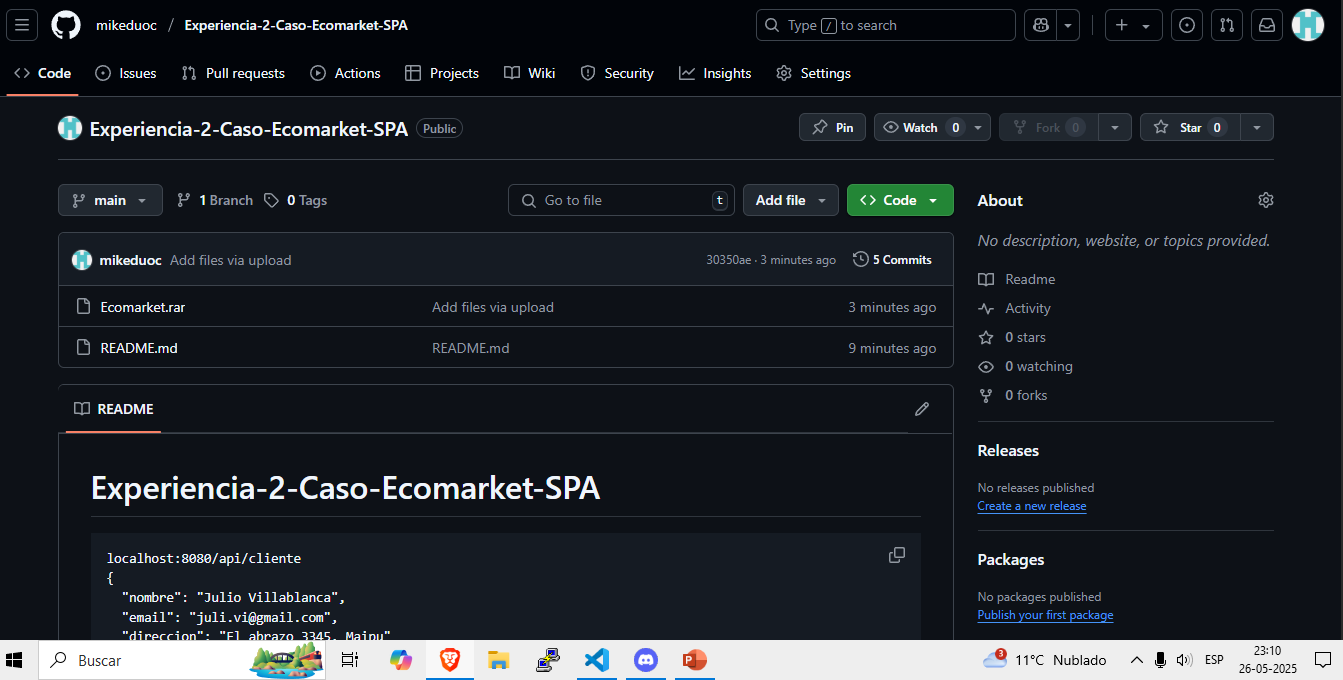
**7. Git hub:**

github es para usarlo en la colaboración de código además de subirlo a la web como repositorio

git init -> Inicia repositorio de forma local en git

git add -> agrega los archivos dentro del repositorio

git commit: guarda los cambios totales de forma local



**8. Conclusion:**

En esta segunda etapa sobre el proyecto de mejora del proceso microservicios dentro de la compañía chilena EcoMarket Spa, se pudo lograr una arquitectura de microservicios usando SpringBoot y MySql WorkBench. Dentro de estos se creó las tres api rest las cuales permiten conectar a crud y hacer actividades sobre las entidades fundamentales dentro de la compañía.

El uso de maven fue importante en el ámbito de las dependencias dentro del proyecto. La arquitectura fue reflejada por el orden de código. La utilización de GitHub dejó mantener un uso útil de las versiones y la plataforma para poder estar interconectados entre todos los creadores del microservicio.

Es realmente cierto que ayuda a entablar un piso, que permita el mejoramiento próximo del proyecto de negocio de EcoMarket Spa y eso finalmente llevará a aumentar la comunicación entre cada microservicio, mejorará la seguridad, rendimiento y disponibilidad, reduciendo la probabilidad de fallos más adelante.