Informe de Migración a Arquitectura de Microservicios – EcoMarket SPA

Integrantes: Ricardo Gaete, Damián Vergara, Felipe Villalón  
Instituto Profesional Duoc UC – Sede Maipú

# Índice

1. Introducción

2. Arquitectura de Microservicios

3. Estructura del Proyecto

4. Base de Datos

5. Implementación de Servicios REST

6. Implementación de Vistas

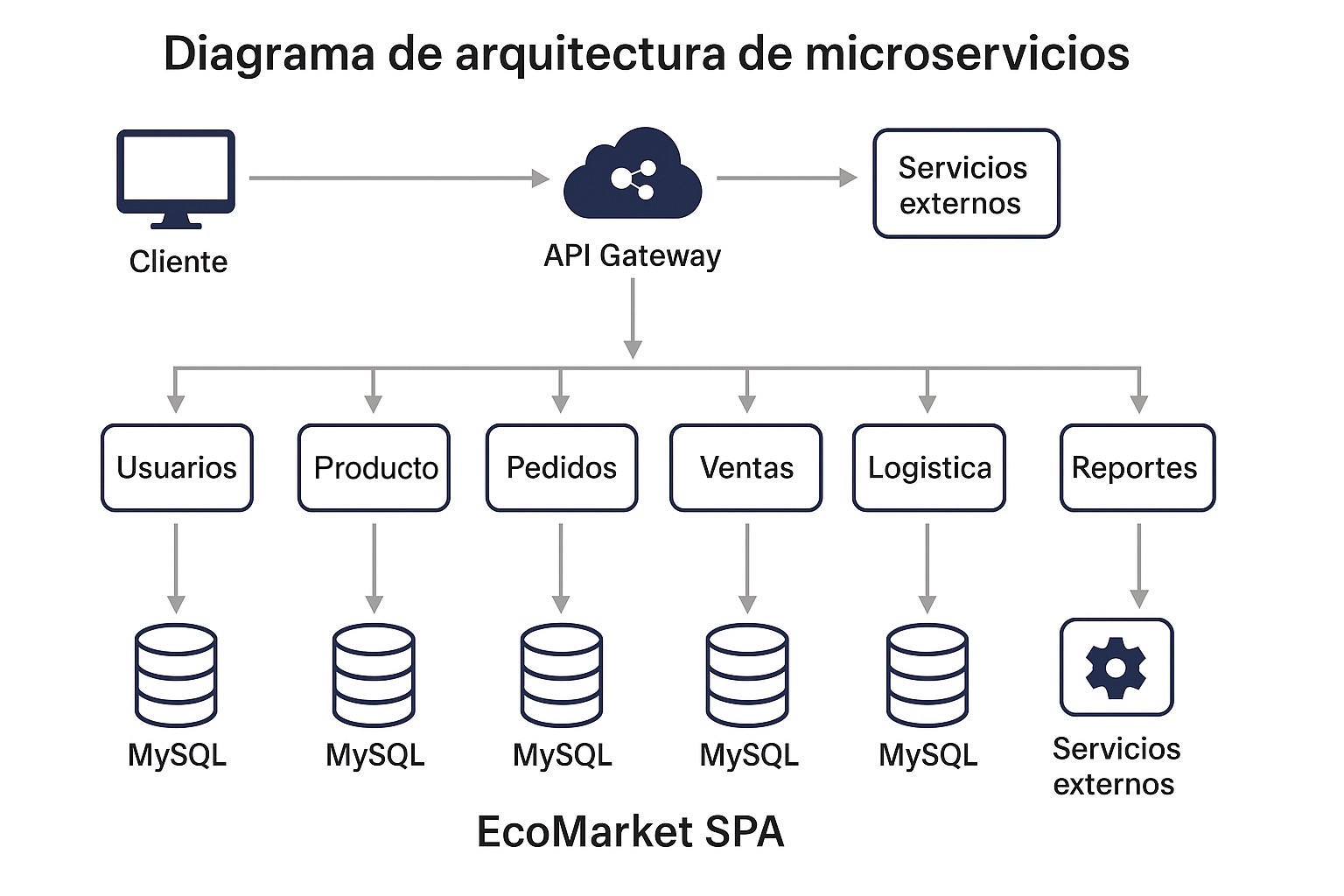
7. Control de Versiones con Git y GitHub

8. Conclusión

# 1. Introducción

EcoMarket SPA es una empresa chilena que comercializa productos ecológicos y sostenibles. Debido a los problemas de escalabilidad, disponibilidad y mantenimiento del sistema monolítico, se ha decidido migrar a una arquitectura basada en microservicios. Esta nueva arquitectura fue desarrollada con Spring Boot y Maven, usando buenas prácticas de desarrollo y control de versiones con Git y GitHub.  
  
El presente informe documenta el proceso completo de rediseño del sistema, desde su arquitectura, estructura del código, integración de servicios RESTful, gestión de bases de datos, pruebas y despliegue.

# 2. Arquitectura de Microservicios

La nueva arquitectura distribuye las funcionalidades en varios microservicios independientes que se comunican a través de APIs REST. Cada servicio gestiona un dominio específico de la aplicación:  
  
- usuarios-service: autenticación y gestión de usuarios  
- producto-service: gestión de productos y stock  
- pedidos-service: procesamiento y seguimiento de pedidos  
  
Cada servicio puede desplegarse, escalarse y actualizarse de manera autónoma.  
  


# 3. Estructura del Proyecto

Cada microservicio es un proyecto Spring Boot independiente con la siguiente estructura:  
  
src/main/java/com/ecomarket/[servicio]/  
├── controller s # Endpoints REST  
├── services # save, delete y find  
├── entities # Entidades JPA  
└── Aplicacion.java # Clase principal  
  
Dependencias clave en pom.xml:  
  
<dependency>spring-boot-starter-web</dependency>  
<dependency>spring-boot-starter-data-jpa</dependency>  
<dependency>mysql-connector-java</dependency>

# 4. Base de Datos

Se utilizó MySQL como sistema gestor de bases de datos. Cada microservicio tiene su propia base de datos para mantener la independencia y permitir escalabilidad horizontal.  
  
  
[Imagen de base de datos]

# 5. Implementación de Servicios REST

Los microservicios fueron desarrollados como APIs RESTful utilizando Spring Boot, implementando las operaciones básicas CRUD (Crear, Leer, Actualizar, Eliminar). Cada servicio está conectado a su propia base de datos en MySQL mediante Spring Data JPA.  
  
1. usuarios-service  
Este servicio permite gestionar los datos de los usuarios registrados en EcoMarket.  
  
Endpoints disponibles:  
- GET /api/usuarios → Lista todos los usuarios registrados.  
- POST /api/usuarios → Registra un nuevo usuario.  
- PUT /api/usuarios/{id} → Actualiza información de un usuario existente.  
- DELETE /api/usuarios/{id} → Elimina un usuario por su ID.  
  
2. inventario-service  
Encargado de la gestión de productos, stock y categorías disponibles en la tienda.  
  
Endpoints disponibles:  
- GET /api/inventario → Muestra el catálogo de productos.  
- POST /api/inventario → Añade un nuevo producto.  
- PUT /api/inventario/{id} → Modifica el stock o precio de un producto.  
- DELETE /api/inventario/{id} → Elimina un producto del inventario.  
  
3. pedidos-service  
Permite a los usuarios generar y consultar sus pedidos, incluyendo el estado de procesamiento.  
  
Endpoints disponibles:  
- GET /api/pedidos → Lista todos los pedidos registrados.  
- POST /api/pedidos → Crea un nuevo pedido con productos seleccionados.  
- PUT /api/pedidos/{id} → Cambia el estado de un pedido.  
- DELETE /api/pedidos/{id} → Cancela o elimina un pedido.  
  
  
[imagen de Postman]

# 6. Implementación de Vistas

Se desarrollaron vistas web básicas (HTML o React) para mostrar resultados de las operaciones GET y validar la conexión entre frontend y backend.  
  
Ejemplos de vistas:  
- Página de listado de pedidos  
- Detalle de usuario y productos  
  
[imagen del navegador]

# 7. Control de Versiones con Git y GitHub

Para el control de versiones se utilizó Git y como repositorio remoto GitHub, empleando ramas y pull requests para el trabajo colaborativo.  
  
Comandos usados:  
  
git init  
git config user.name  
git config user.email

Git config user.password  
git remote add origin   
git push -u origin main  
  
Fragmento del README.md:  
  
usuarios-service  
- GET /api/usuarios  
- POST /api/usuarios  
  
pedidos-service  
- GET /api/pedidos  
- POST /api/pedidos  
  
inventario-service  
- GET /api/inventario  
- POST /api/inventario  
  
[imagen del GitHub]

# 8. Conclusión

La migración del sistema monolítico a una arquitectura de microservicios ha mejorado la escalabilidad, mantenibilidad y disponibilidad del sistema de EcoMarket SPA. Las tecnologías utilizadas (Spring Boot, MySQL, Maven, Git) proporcionaron una base sólida para una solución moderna y robusta. La separación de responsabilidades entre microservicios permite una evolución independiente de cada módulo y prepara el camino para futuras integraciones como balanceo de carga, autenticación JWT o despliegue en la nube.