Documentación del Proyecto de Microservicios en .NET y Docker

Introducción

Este proyecto implementa una arquitectura de microservicios con .NET 8, SQL Server y Docker. La solución se compone de dos microservicios:

- ProductService: Gestión de productos.
- InventoryService: Gestión del inventario, con validación de existencia de productos a través de ProductService.

Tecnologías utilizadas

- ASP.NET Core 8
- SQL Server 2022 (contenedor Docker)
- Docker y Docker Compose
- Swagger para documentación de APIs
- Polly para reintentos y manejo de fallos en comunicación HTTP
- API Key para autenticación entre microservicios

Arquitectura

- Cada microservicio es autónomo y expone su API REST.
- InventoryService se comunica con ProductService mediante HTTP usando JSON API estándar.
- Ambos servicios están orquestados con Docker Compose y conectados mediante una red compartida.

Componentes

ProductService

- CRUD de productos
- API protegida por API Key
- Swagger disponible en `/swagger`

InventoryService
- CRUD de inventario
- Consulta de productos en ProductService para validar existencia
- HttpClient con autenticación mediante API Key y política de reintento (Polly)

Docker Compose
```yaml
services:
  sqlserver:
    image: mcr.microsoft.com/mssql/server:2022-latest
    ports:
      - "1433:1433"
    ...

  productservice:
    build: ./ProductService
    ports:
      - "8080:80"
    ...

  inventoryservice:
    build: ./InventoryService
    ports:
```

```
      - "8081:80"

    ...
```

Pruebas

1. Ingresar a Swagger en `http://localhost:8080/swagger` para ProductService.

2. Crear un producto.

3. Ingresar a Swagger en `http://localhost:8081/swagger` para InventoryService.

4. Crear inventario validando que el producto exista mediante ProductService.

Seguridad

- Autenticación entre microservicios mediante API Key.

- Middleware personalizado en ProductService para validar el header `X-Api-Key`.

Conclusión

Este proyecto demuestra cómo implementar microservicios independientes con .NET y Docker, conectados por HTTP y autenticados de forma segura, utilizando buenas prácticas como separación de responsabilidades, resiliencia con Polly y despliegue contenedorizado.