Proyecto final

Daniel Fernando Esai Ramirez Callo

Computacion Tolerante a Fallas

07/12/2025

Descripción del Proyecto

Este proyecto implementa un **Sistema de Reservas** utilizando una arquitectura basada en **microservicios**, cada uno encargado de una parte del dominio: clientes, mesas, reservas y notificaciones.
Un **API Gateway** centraliza todas las peticiones para simplificar el consumo del sistema por parte del usuario final.

Los microservicios se comunican entre sí mediante solicitudes HTTP internas dentro de la red de Docker.

Este repositorio incluye:

- Código fuente de cada microservicio
- Dockerfiles
- Docker Compose listo para ejecutar
- Manifiestos opcionales para despliegue en Kubernetes
- Guide completa y detallada para desplegar, probar y depurar
- Presentación del proyecto incluida en /docs

Arquitectura del Proyecto

proyecto_reservas_full/

│— api-gateway/

│ ├— app.py

```
│   ├── Dockerfile
│   └── requirements.txt
│
├── ms-clientes/
├── ms-reservas/
├── ms-mesas/
├── ms-notificaciones/
│   (cada carpeta contiene: app.py, Dockerfile, requirements.txt)
│
├── docker-compose.yml
├── k8s/
├── docs/
│   └── presentacion.pptx
└── README.md
```

. Requisitos Previos

Antes de desplegar el proyecto asegúrate de tener instalado:

Requisito	**Versión recomendada**	**Comando para verificar**
Docker Desktop	Última	docker --version
Docker Compose	V2 o superior	docker compose version

Requisito	Versión recomendada	Comando para verificar
Python 3 (opcional)	3.9+	python --version
curl o Postman	cualquiera	---
Minikube / kubectl (opcional)	si usas Kubernetes	kubectl version

2. Preparación del Proyecto

1.Descarga o clona este repositorio.

2.Entra a la carpeta raíz:

cd proyecto_reservas_full

3. Verifica que todas las carpetas existan:

   ls -la

   Debes ver:

   api-gateway/

   ms-clientes/

   ms-reservas/

   ms-mesas/

   ms-notificaciones/

   docker-compose.yml

k8s/

docs/

Si falta algo, revisa el ZIP descargado.

3. Configuración Importante — SECRET_KEY

Los microservicios de **clientes** y **reservas** utilizan JWT.

Asegúrate de que ambos servicios tengan la misma variable SECRET_KEY.

Esto ya está configurado en docker-compose.yml así:

environment:

- SECRET_KEY=mi_clave_segura_2025

Puedes cambiar el valor si quieres.

4. Despliegue con Docker Compose

Este proyecto incluye un docker-compose.yml listo para funcionar.

Para construir las imágenes y levantar todos los microservicios:

docker compose up –build

Si estás usando el Docker Compose clásico:

docker-compose up –build

5. Verificación después de levantar servicios

Cuando todo arranca correctamente verás líneas como:

* Running on http://0.0.0.0:5000

* Running on http://0.0.0.0:5001

* Running on http://0.0.0.0:5002

* Running on http://0.0.0.0:5003

* Running on http://0.0.0.0:5004

Verifica contenedores:

docker ps

Si todos aparecen en estado *Up*, el sistema está funcional.

6. Endpoints disponibles

La comunicación externa se hace exclusivamente a través del:

📍 **API Gateway: http://localhost:5000**

6.1 Registrar Cliente

curl -X POST http://localhost:5000/register \

-H "Content-Type: application/json" \

-d
'{"email":"ana@example.com","nombre":"Ana","password":"12345"}'

6.2 Iniciar Sesión (Obtener Token JWT)

curl -X POST http://localhost:5000/login \

-H "Content-Type: application/json" \

-d '{"email":"ana@example.com","password":"12345"}'

Respuesta esperada:

{"token": "eyJhbGciOi..."}

Guarda ese token para request futuras.

6.3 Crear Reserva

TOKEN="PEGAR_TOKEN_AQUÍ"

```
curl -X POST http://localhost:5000/reservar \
-H "Content-Type: application/json" \
-H "Authorization: Bearer $TOKEN" \
-d '{"fecha":"2025-12-01","hora":"20:00","mesa_id":1}'
```

6.4 Consultar Reservas del Usuario

curl http://localhost:5000/mis-reservas/ana@example.com

7. Estructura de Microservicios

Cada microservicio incluye:

app.py Lógica del servicio

Dockerfile Cómo se empaqueta

requirements.txt Dependencias

Comunicación HTTP entre contenedores usando nombres como:

http://ms-mesas:5003

http://ms-clientes:5001

http://ms-notificaciones:5004

8. Cómo depurar errores comunes

"file not found" al construir

Causa: Estás fuera de la carpeta del proyecto.
Solución:

cd proyecto_reservas_full

Puerto ya en uso

Búscalo:

Linux/macOS:

sudo lsof -i :5000

Windows:

netstat -ano | findstr :5000

Mata el proceso o cambia el puerto.

JWT inválido

Asegúrate que **ms-clientes y ms-reservas** tengan la *misma SECRET_KEY*.

Servicio no responde

Ver logs:

docker compose logs ms-reservas --tail=100

Revisa errores y corrige.

9. Cómo detener y limpiar contenedores

Detener todo:

docker compose down

Limpiar todo (imágenes + volúmenes):

docker compose down --rmi all --volumes --remove-orphans

10. Despliegue Opcional en Kubernetes (Minikube)

10.1 Iniciar Minikube

minikube start

10.2 Construir imágenes dentro de Minikube

eval $(minikube docker-env)

docker build -t ms-clientes:1 ms-clientes/

docker build -t ms-reservas:1 ms-reservas/

docker build -t ms-mesas:1 ms-mesas/

docker build -t ms-notificaciones:1 ms-notificaciones/

docker build -t api-gateway:1 api-gateway/

10.3 Aplicar los manifiestos

kubectl apply -f k8s/

10.4 Obtener URL del API Gateway

minikube service api-gateway --url