SISTEMAS DISTRIBUIDOS

Microservicios & RabbitMQ

> Implementación de servicios de notificaciones desacoplados y gestión de errores mediante Dead Letter Exchanges (DLX).

STACK TECNOLÓGICO

- Docker Compose
- RabbitMQ
- Python Consumers

CÓDIGO FUENTE

aek676/itsi-2026

Repositorio completo del proyecto

AUTOR

Anass El Jabiry Kaddir

EJERCICIOS

4.2 y 4.3

Ejercicio 2: Servicio Notifier

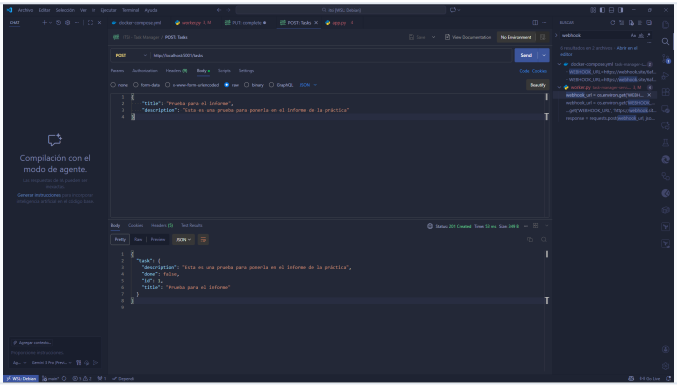
Desacoplamiento de servicios

02

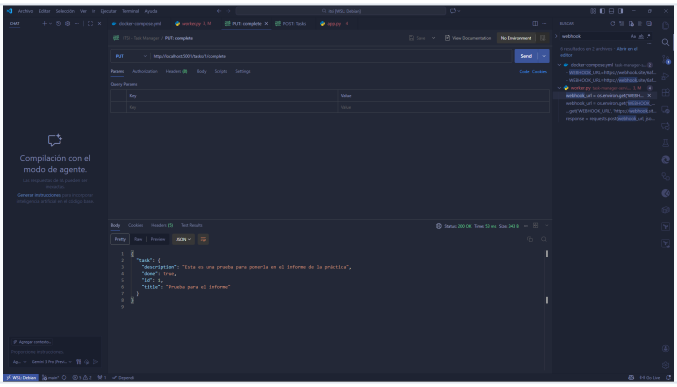
ENUNCIADO: SERVICIO DE NOTIFICACIONES

Crear un tercer servicio (`notifier`) que consuma de `task_completed` para enviar notificaciones simuladas (POST a Webhook) tras completar una tarea, desacoplando esta lógica del worker principal.

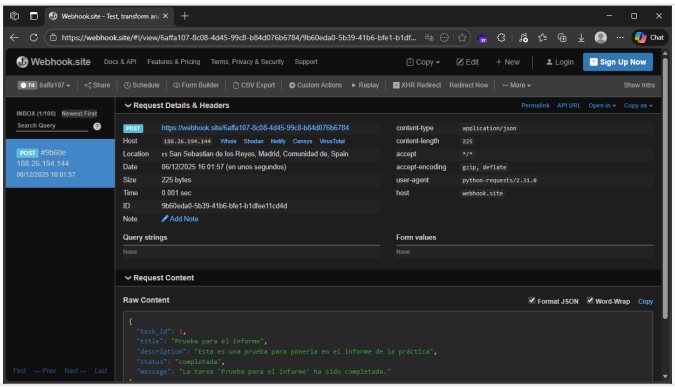
1. PETICIÓN POST (CREAR TAREA)



2. TAREA COMPLETADA (LOG WORKER)



3. WEBHOOK RECIBIDO



Resultado: Notificación enviada correctamente al endpoint externo sin bloquear el worker.

Ejercicio 3: Dead Letter Queue

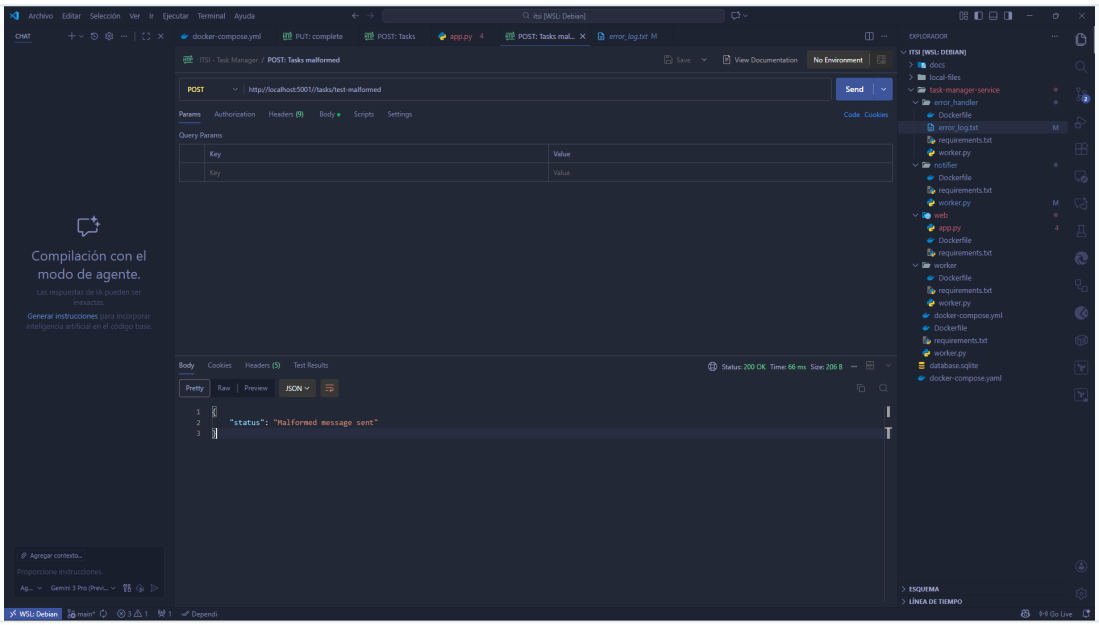
Resiliencia y manejo de errores

03

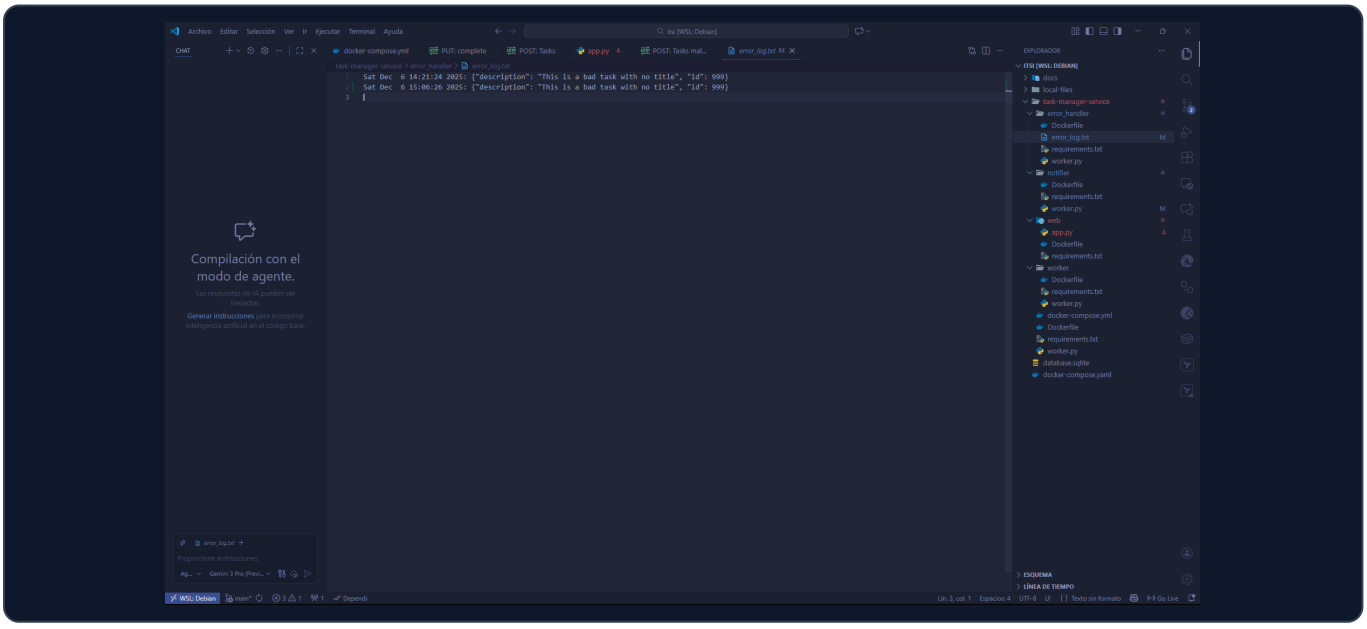
ENUNCIADO: PERSISTENCIA Y DLX

Implementar una *Dead Letter Exchange* (DLX). El worker debe rechazar (NACK) mensajes malformados (sin `title`). Estos deben redirigirse automáticamente a una cola de auditoría `tasks_failed`.

1. SIMULACIÓN DE ERROR (ENDPOINT MALFORMADO)



2. LOG DEL SISTEMA (RECHAZO Y DLX)



Análisis: El log confirma el rechazo (NACK) y el redireccionamiento exitoso a la cola de errores.