CURSO: DEVOPS SENIOR

Módulo 4: Observabilidad avanzada

Ejercicio Práctico 2

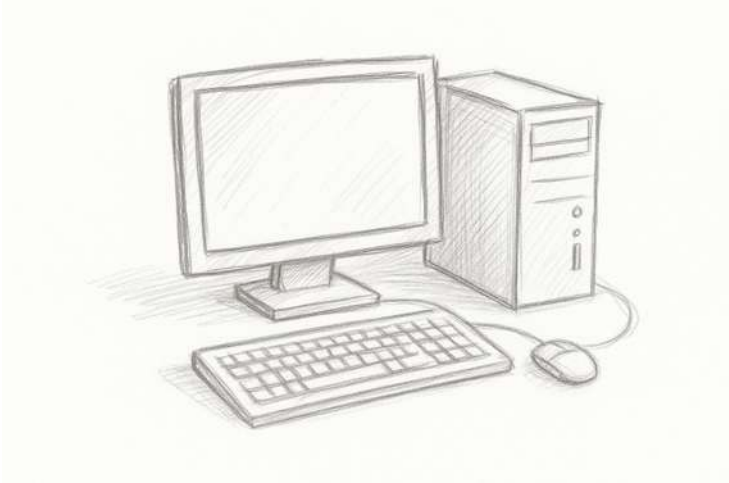
Trazabilidad Distribuida con OpenTelemetry + Jaeger + Tempo en una App de Microservicios

Objetivo:

Instrumentar una aplicación de microservicios con OpenTelemetry para capturar trazas distribuidas, visualizar el flujo entre servicios con Jaeger y Tempo, y detectar cuellos de botella o fallas de comunicación.

Resultado esperado:

Comprender cómo fluye una petición a través de múltiples servicios, cómo se instrumenta código para trazabilidad real, y cómo visualizar e interpretar cuellos de botella.



INSTRUCCIONES:

- **Entorno sugerido:**
    - Use Play with Docker o un entorno local con Docker Compose.
    - Clone este repositorio base gratuito: https://github.com/open-telemetry/opentelemetry-demo (o usar una app propia dividida en frontend, backend y DB).
- **Actividad guiada:**
    - Lance la aplicación demo con OpenTelemetry Collector, Jaeger y/o Tempo integrados (en modo Docker).
    - Simule peticiones HTTP desde el frontend.
- **Visualice en Jaeger o Tempo:**
    - Tiempos de respuesta por servicio.
    - Dependencias entre servicios.
    - Servicios con mayor latencia.
- Instrumentación adicional (manual):
    - Agregue trazas personalizadas en el backend usando el SDK de OpenTelemetry para Node.js o Python.
    - Use atributos (span.setAttribute) y eventos (span.addEvent) para marcar pasos relevantes (ej: acceso a DB, fallos).
- **Feedback técnico esperado:**
    - Mapas de servicios correctamente generados.
    - Líneas de tiempo de ejecución que evidencian el flujo real.
    - Capacidad de detectar “outliers” o tiempos muertos.
    - Opcional: integración con Grafana Tempo
    - Use Grafana como frontend para Tempo y correlaciona métricas y trazas.