**Tarea: Implementación de OpenTelemetry**

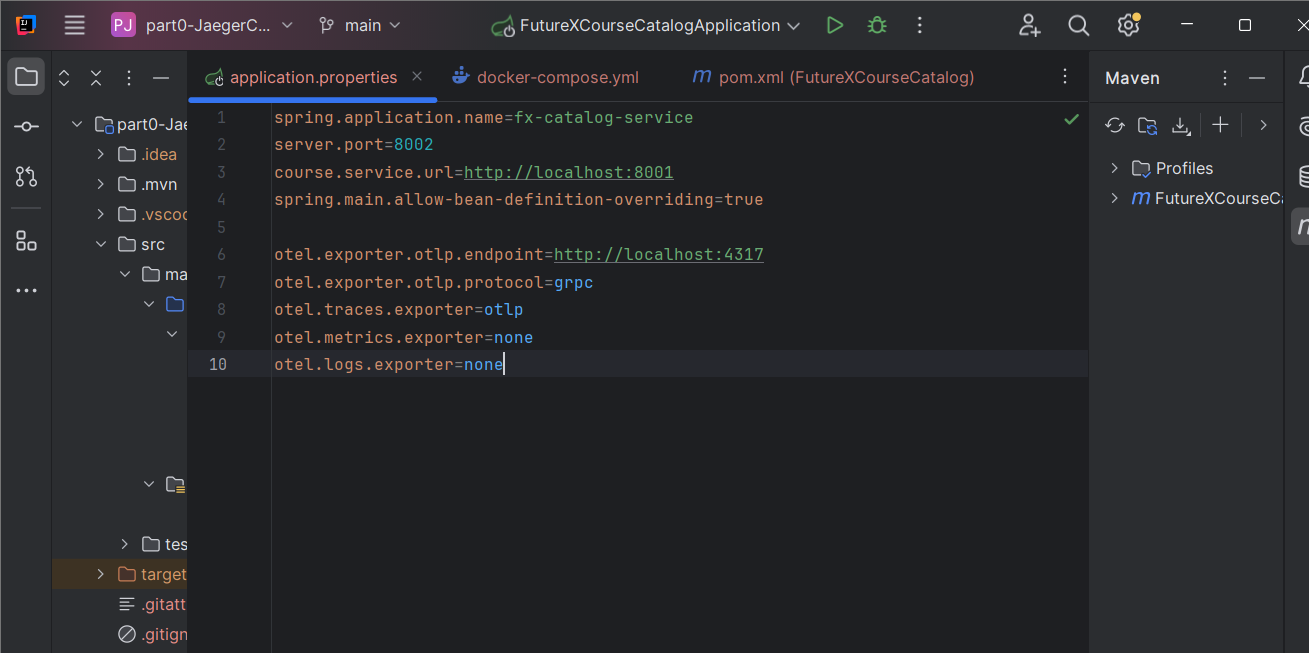
**David Donneys- A00396036**

Objetivo: Implementar OpenTelemetry en los microservicios FutureXCourseApp y FutureXCourseCatalog utilizando tres enfoques diferentes

Parte 1: Usando el Agente Java de OpenTelemetry

1. Descarguen el agente Java de OpenTelemetry. 2. Modifiquen el archivo application.properties en ambos servicios. 3. Ejecuten el archivo Docker Compose para Jaeger. 4. Ejecuten ambas aplicaciones con el agente Java. 5. Verifiquen las trazas en la UI de Jaeger.

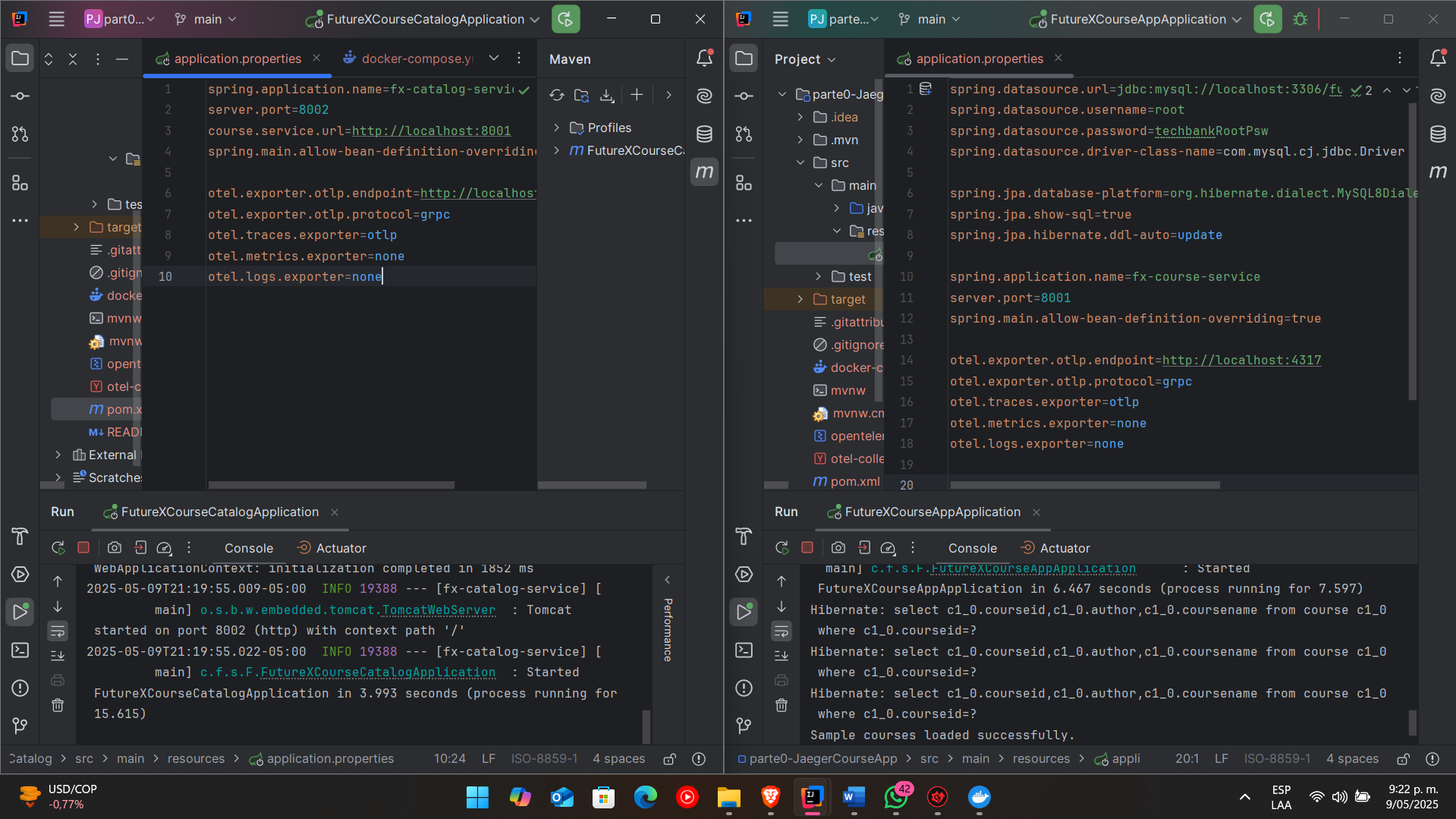
• Pantallazo del application.properties modificado



Texto

El contenido generado por IA puede ser incorrecto.

• Pantallazo de la aplicación ejecutándose con el agente



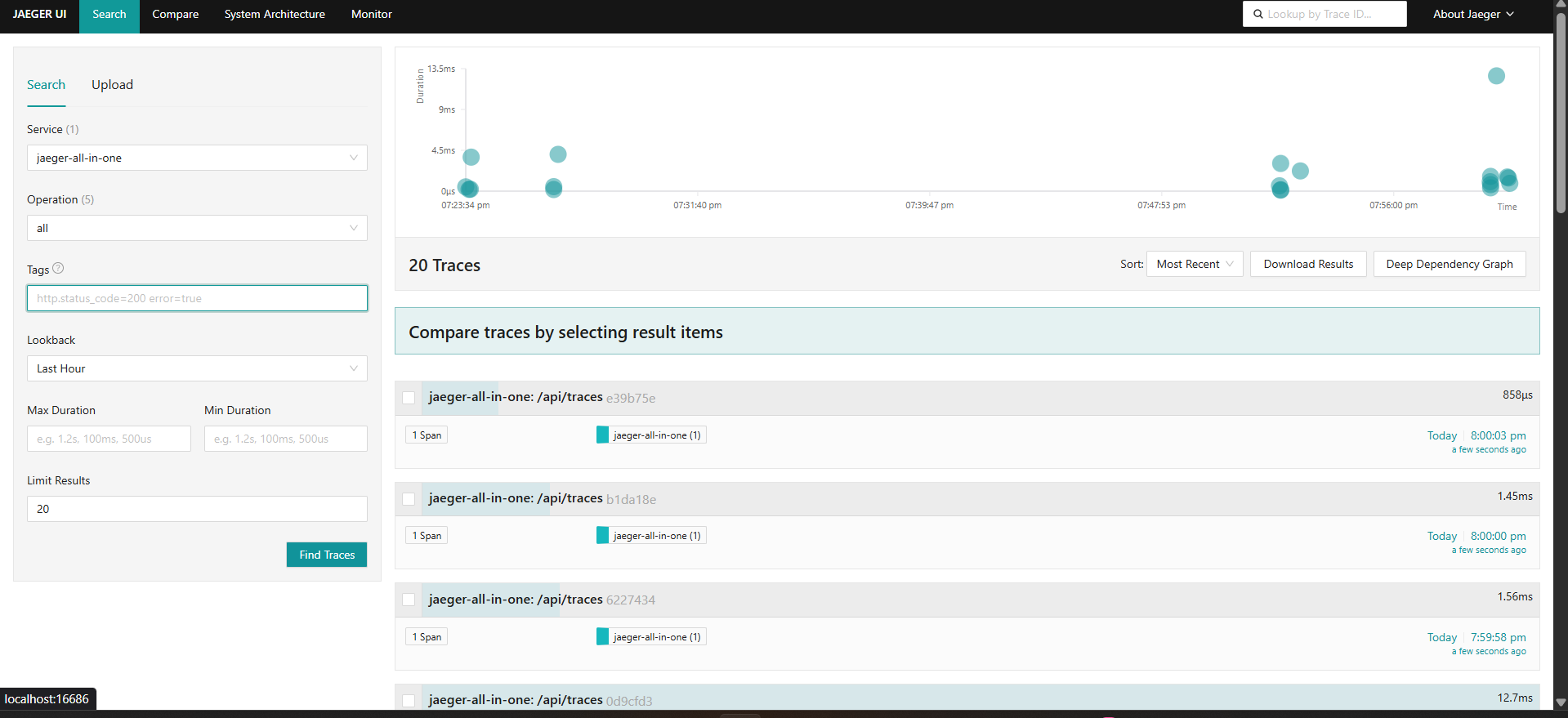
Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

Captura de pantalla de un celular

El contenido generado por IA puede ser incorrecto.

• Pantallazo de las trazas en la UI de Jaeger



Parte 2: Usando la API de OpenTelemetry

1. Actualicen las dependencias en los archivos pom.xml. 2. Creen la clase de configuración OpenTelemetryConfig. 3. Actualicen las clases principales de la aplicación. 4. Modifiquen los controladores para usar OpenTelemetry. 5. Ejecuten Jaeger y verifiquen las trazas.

• Pantallazo de la clase OpenTelemetryConfig

Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.

• Pantallazo de los controladores modificados

Texto

El contenido generado por IA puede ser incorrecto.

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

• Pantallazo de las trazas en Jaeger

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

Parte 3: Usando el Colector de OpenTelemetry

1. Configuren el Colector de OpenTelemetry. 2. Actualicen el archivo docker-compose.yml. 3. Modifiquen el archivo application.properties. 4. Actualicen la clase OpenTelemetryConfig. 5. Ejecuten los contenedores Docker y verifiquen las trazas.

• Pantallazo del archivo de configuración del Colector (yaml)

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

• Pantallazo del application.properties modificado

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

• Pantallazo de la clase OpenTelemetryConfig actualizada

Texto

El contenido generado por IA puede ser incorrecto.

• Pantallazo de las trazas en Jaeger pasando por el Colector

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

**Desafíos Encontrados y Soluciones Implementadas**

Durante la implementación de OpenTelemetry en los microservicios FutureXCourseApp y FutureXCourseCatalog, me enfrenté principalmente a tres desafíos técnicos significativos. Al configurar el entorno Docker para Jaeger y el Colector, encontré problemas de comunicación entre los servicios que impedían el envío correcto de las trazas; esto requirió varios ajustes en el archivo docker-compose.yml, especialmente en la configuración de puertos y dependencias entre contenedores, hasta lograr que todos los servicios pudieran comunicarse adecuadamente. Paralelamente, la configuración del pipeline en el Colector de OpenTelemetry resultó particularmente compleja, ya que requería una comprensión profunda de cómo definir correctamente los receptores, procesadores y exportadores para optimizar el flujo de telemetría; después de consultar extensivamente la documentación oficial y hacer varias pruebas, logré implementar una configuración que incluía procesamiento por lotes que mejoró significativamente el rendimiento. Finalmente, uno de los retos más frustrantes fue la gestión de dependencias y versiones, donde encontré incompatibilidades entre las bibliotecas de OpenTelemetry y mi entorno Spring Boot que generaban errores en tiempo de ejecución; la solución requirió una investigación meticulosa para identificar las versiones exactas compatibles entre sí, además de implementar el BOM (Bill of Materials) de OpenTelemetry en el pom.xml, lo que finalmente aseguró la coherencia entre todas las dependencias y permitió que la instrumentación funcionara correctamente.