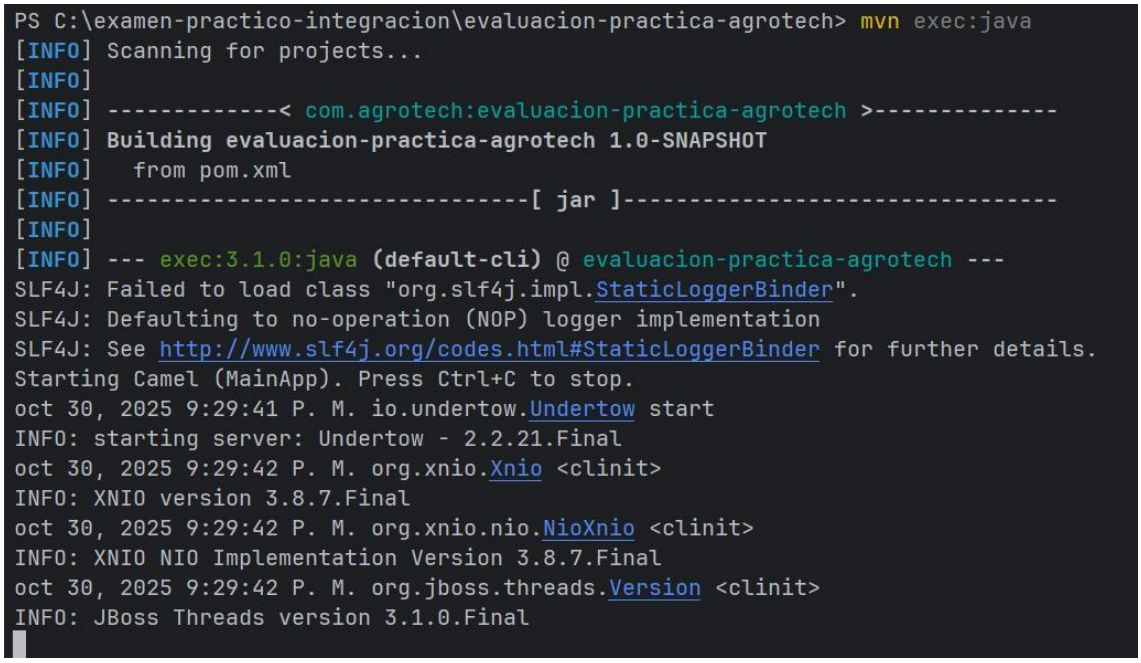
Evaluación Practica Integración de Sistemas

Integrante: José Sánchez

Evidencias:



Respuestas de las preguntas:

1. **¿Qué patrón aplicaste en cada fase del flujo y por qué?**

   File Transfer para la ingestión batch (leer sensores.csv, mover a processed y procesar filas) porque desacopla productor/consumidor y es simple para lotes; Shared Database para intercambio de estado entre AgroAnalyzer y FieldControl (ambos leen/escriben en lecturas) por rapidez de implementación; RPC simulado (direct request–reply en Camel) para modelar llamadas síncronas cuando el cliente necesita respuesta inmediata.

2. **¿Qué riesgos observas al usar una base de datos compartida?**

   Riesgos de DB compartida: conlleva acoplamiento fuerte entre servicios, problemas de gobernanza y migraciones, riesgo de contención y single point of failure, dificultades para escalar y gestionar permisos/seguridad, y mayor complejidad para probar y evolucionar sin romper consumidores.

3. **¿Cómo ayuda el RPC simulado a representar un flujo síncrono?**

RPC simulado y flujo síncrono: al usar rutas direct y requestBody en Camel se modela exactamente la semántica request–reply —el cliente bloquea hasta recibir respuesta o timeout— lo que permite probar comportamiento síncrono (timeouts, errores) sin necesidad de infraestructura remota, aunque no reproduce latencias/particiones de red reales.

4. **¿Qué limitaciones tienen los patrones clásicos frente a arquitecturas modernas?**

   Limitaciones de patrones clásicos vs modernos: enfoques como File Transfer o Shared DB son fáciles de implementar pero limitan escalabilidad, resiliencia y despliegue independiente; las arquitecturas modernas (event-driven, DB-per-service, mensajería) favorecen desacoplo, tolerancia a fallos y evolución independiente a costa de mayor complejidad operacional.