

# Trabajo Práctico 2: Orquestación y Observabilidad

DevOps | UTN FRRe | 2025

## [Trabajo Práctico 2: Orquestación y Observabilidad](#)

[Objetivo del Trabajo Práctico](#)

[Introducción al escenario](#)

[Pautas](#)

[Entregables](#)

[Fecha de Entrega](#)

[Requerimientos Funcionales](#)

[Requerimientos No Funcionales](#)

[1. PARTE 1: Observabilidad](#)

[2. PARTE 2: Orquestación](#)

[Prueba requerida en el Coloquio](#)

[Rúbrica](#)

## Objetivo del Trabajo Práctico

El objetivo de este trabajo práctico es extender las capacidades del proyecto desarrollado en el TP1 incorporando orquestación de servicios y observabilidad completa mediante métricas, logs y trazas.

El grupo podrá reutilizar el código del TP1 o, si lo considera oportuno, crear una nueva aplicación que cumpla con los objetivos planteados, siempre manteniendo la arquitectura de servicios containerizados.

## Introducción al escenario

En el TP1 se desarrolló una aplicación web y un servicio API conectados a un servidor Redis, todos containerizados.

En este TP2, los estudiantes deberán evolucionar esta arquitectura agregando:

- Orquestación de servicios (con Kubernetes ligero o proxy con balanceo, hay 2 opciones de entrega).
- Observabilidad completa de contenedores y de la aplicación utilizando componentes basados en OpenTelemetry, además de Prometheus y Grafana para la visualización.

Este trabajo busca preparar al estudiante para escenarios reales de despliegue en infraestructura distribuida y monitoreable, donde los sistemas soporten falla de un nodo y continúen operando.

## Pautas

1. El trabajo se realiza en grupos de hasta 2 o 3 personas.
2. Se toma como base el TP1, pero los grupos pueden crear una nueva aplicación si lo desean.
3. El código debe estar en un repositorio Github grupal.
4. Se evaluará el funcionamiento de manera práctica en el coloquio.
5. La rúbrica tiene un total de 100 puntos.

## Entregables

1. Aplicación funcionando con orquestación y observabilidad (demostración en el coloquio).
2. Repositorio Github del grupo.
3. Paneles en Grafana con visualización de métricas.
4. Configuraciones de orquestación según opción elegida.
5. Informe o slide deck resumen: arquitectura, decisiones, problemas y mejoras.

## Fecha de Entrega

Fecha de Entrega Semana del 1 al 5 de Dic de Forma Virtual, en fecha a coordinar con la catedra (Coloquio Grupal)

- Registrar coloquio en el siguiente calendario  
[https://docs.google.com/spreadsheets/d/1RS5qhR7xMp5aeK159s5Tb0jg5JYf4Xb\\_QYFU5kjsgAE/edit?usp=sharing](https://docs.google.com/spreadsheets/d/1RS5qhR7xMp5aeK159s5Tb0jg5JYf4Xb_QYFU5kjsgAE/edit?usp=sharing)

## Requerimientos Funcionales

### 1. Aplicación base

La aplicación deberá ser la misma del TP1 o una versión nueva, pero debe exponer al menos una acción que permita generar carga controlada (por ejemplo, un botón que dispare un algoritmo que aumente el uso de memoria al 100%).

## Requerimientos No Funcionales

### 1. PARTE 1: Observabilidad

La aplicación debe contar con:

#### A. **Logs y trazas**

Todos los servicios deben emitir logs estructurados. Deben existir trazas que permitan seguir el recorrido de la aplicación (por ejemplo, al enviar una tarea, procesarla, leer desde Redis, etc.).

Se debe utilizar OpenTelemetry, acorde al lenguaje del proyecto.

#### B. **Métricas, Debe existir al menos:**

Un indicador de contenedor:

Uso de CPU y/o Uso de memoria

(el grupo puede usar cAdvisor, node-exporter o exporters similares)

Un indicador de la aplicación:

Ejemplo:

- Total de tareas cargadas
- Cantidad de requests procesados
- Tiempo de respuesta
- Objetos en cache, etc.

#### C. **Visualización**

Las métricas deben poder visualizarse en Grafana mediante gráficos y paneles. Sencillos pero deben ser graficos y/o indicadores numericos

#### D. **Stack recomendado**

OpenTelemetry SDK / Auto-Instrumentation

Prometheus

Grafana

OTel Collector (opcional pero recomendado)

## 2. PARTE 2: Orquestación

Las opciones son mutuamente excluyentes: el grupo elige una.

### **OPCION 1 Orquestación con Kubernetes ligero**

Utilizar cualquiera de los siguientes:

- k3s
- rke2

Debe incluir:

- Despliegue de la app y Redis en pods.
- Servicio o Ingress para exponer la web.
- Hoja de configuración o YAMLS necesarios.

Demostración de High Availability

Al ejecutar la acción que consume memoria (hasta saturar 80–100%), el nodo debe quedar fuera de servicio, y otro nodo debe levantar automáticamente el contenedor para mantener el servicio.

### **OPCION 2 Reverse Proxy + Balanceador**

Implementar: Traefik, NGINX, HAProxy u otro reverse proxy.

Debe hacer:

- Balanceo de carga entre al menos dos instancias de la app.
- Detección de falla basada en Memoria > 80% o Caída de health-check (o sea siempre detectando un valor o endpoint a verificar por el proxy reverso)
- Remover un nodo enfermo de la rotación.

Ejemplo de demo solicitada

- Ejecutar en un servidor la acción que eleva la memoria a 100%
- El balanceador debe desviarse automáticamente hacia el otro nodo.

## Prueba requerida en el Coloquio

En ambos casos (opción 1 u opción 2), se debe:

- Levantar el entorno completo.
- Ejecutar una acción desde la app que eleve la memoria al 100% en un nodo/pod.
- Demostrar que el servicio sigue funcionando

## Rúbrica

	<b>Envío de datos a OpenTelemetry</b> Logs estructurados + métricas + trazas funcionales recolectadas correctamente.	<b>Visualización en Grafana</b> Dashboard completo mostrando métricas de contenedores y de la aplicación	<b>Orquestación – Opción 1 u Opción 2</b> Despliegue correcto, balanceo, nodos funcionando, configs entregadas	<b>Alta disponibilidad (HA) ante memoria &gt;80%</b> Se simula la saturación y el sistema se recupera automáticamente.	
<b>Puntos</b>	<b>25</b>	<b>25</b>	<b>20</b>	<b>20</b>	<b>10</b>

Entrega Excelente (100%)	Se aborda el tema. Se presenta la idea y se profundiza la misma o agregando valor. La entrega se realiza en tiempo y forma. El trabajo está estructurado y completado al 100. Se detalla en el coloquio de principio a fin el proceso completo y los problemas en el mismo. Se extendió lo que se propuso como TP. Se mejoró tareas previamente realizadas. El trabajo se presentó con todos los lineamientos propuestos.
Terminado Satisfactorio (80%)	Se aborda el tema, pero se encuentra en un 75% el punto abordado. La entrega se realiza pero existen puntos faltantes para completar la idea de la funcionalidad requerida. Se entrega en tiempo y forma. No se extendió en lo que se propuso como ideas en el TP. El trabajo o ítem faltó algunos puntos a tener en cuenta para completarlo.
Basico (60%)	Se aborda el tema pero con un nivel escaso de comprensión y de realización. Se encuentra realizado al 50% del ítem solicitado. No se extendió en mejorar o perfeccionar. Se encuentra deficiente la organización del trabajo. No se detalla en el coloquio parte del ítem o se argumenta. No se presentan todos los lineamientos propuestos para el ítem.
No realizado/Escaso (0%)	Solo se menciona el tema o no se aborda. No presenta información relacionada al ítem solicitado. O se realizó pero con error en el abordaje para su funcionamiento o publicación. No se estructuró el trabajo. La entrega no se realizó en tiempo y forma