**Taller Docker – Kubernet Parte 1**

Objetivo: Comprender, analizar y reflexionar sobre los principios, componentes y estrategias fundamentales que permiten garantizar la resiliencia, el balanceo de carga y el monitoreo eficiente en sistemas distribuidos de tipo escritorio.

1. Define qué es un sistema distribuido y explica por qué se vuelve más complejo gestionar la resiliencia en estos entornos comparado con sistemas centralizados.
2. Explica con tus palabras qué es el balanceo de carga en aplicaciones de escritorio cliente-servidor y qué tipos existen.
3. ¿Qué diferencias existen entre alta disponibilidad y tolerancia a fallos? Da un ejemplo concreto para cada uno en un sistema de base de datos distribuida.
4. Describe cómo funcionaría una política de failover en un clúster local con tres nodos que ejecutan servicios replicados.
5. ¿Qué papel cumple la replicación activa-activa en la recuperación ante fallos?
6. Analiza un caso donde la ausencia de monitoreo en un sistema distribuido provocó una falla mayor. Describe cómo se habría evitado.
7. Define qué son los logs estructurados y cómo ayudan a la observabilidad en aplicaciones distribuidas de escritorio.
8. ¿Qué ventajas tiene usar herramientas como Prometheus y Grafana en lugar de monitoreo manual mediante logs locales?
9. Compara los conceptos de reintentos automáticos y circuit breakers en entornos distribuidos. ¿Cuál aplicarías si un servicio de escritorio tarda en responder?
10. Explica cómo aplicarías un esquema de canary deployment para un sistema de atención al cliente instalado en múltiples nodos locales.
11. ¿Por qué es importante aplicar políticas de afinidad en la distribución de procesos en entornos físicos multi-nodo?
12. Expón cómo implementarías un plan de recuperación ante desastres para una aplicación de escritorio crítica conectada por red local.
13. ¿Cómo se interpreta un perfil de rendimiento (CPU, memoria) en aplicaciones distribuidas? ¿Qué decisiones operativas pueden derivarse de él?
14. Describe las consecuencias de una mala configuración de timeouts en la comunicación entre procesos distribuidos.
15. Explica el concepto de fallback y diseña un escenario realista en donde su uso sea más ventajoso que detener el proceso ante errores.