Integrantes

Quintana Cubas Lenin Alexander  
Ponte Arica Anthony Rosemberg  
Ramirez Vasquez Jonathan  
Silva Ramos Juan Armando

Docker y Kubernetes Intermedio

Proyecto Final del Curso

Contenido

[Proyecto Curso Docker y Kubernetes 2](#_Toc197415614)

[Entregables del Proyecto Final: Gestión de Eventos y Participantes 2](#_Toc197415615)

[1. Código Fuente 2](#_Toc197415616)

[1.1. Repositorios de Microservicios: 2](#_Toc197415617)

[2. Artefactos Contenerizados 2](#_Toc197415618)

[2.1. Imágenes Docker: 2](#_Toc197415619)

[2.2. Docker Compose: 2](#_Toc197415620)

[3. Configuración para Kubernetes 2](#_Toc197415621)

[3.1. Archivos YAML: 2](#_Toc197415622)

[3.2. Manifiestos completos: 3](#_Toc197415623)

[4. Bases de Datos 3](#_Toc197415624)

[4.1. Scripts de Inicialización: 3](#_Toc197415625)

[4.2. Volúmenes: 3](#_Toc197415626)

[5. Documentación 3](#_Toc197415627)

[5.1. Manual de Despliegue: 3](#_Toc197415628)

[5.2. Diagrama de Arquitectura: 3](#_Toc197415629)

[6. Resultados 4](#_Toc197415630)

[6.1. Validación del Despliegue: 4](#_Toc197415631)

[6.2. Registro de Imágenes: 4](#_Toc197415632)

# Proyecto Curso Docker y Kubernetes

## Entregables del Proyecto Final: Gestión de Eventos y Participantes

Los entregables se organizan en categorías que abarcan desde el código fuente hasta la

documentación y los artefactos de despliegue

### Código Fuente

### Repositorios de Microservicios:

* Código fuente de los dos microservicios (gestión de eventos y gestión de
* inscripciones).
* Estructura del proyecto conforme a las mejores prácticas (separación en
* capas: controlador, servicio, repositorio, entidades, etc.).
* Uso de control de versiones (Git) con un historial claro de commits.

### Artefactos Contenerizados

### Imágenes Docker:

* Imágenes Docker de ambos microservicios, construidas y publicadas en un
* repositorio como Docker Hub.
* Imágenes optimizadas según las mejores prácticas (Dockerfile eficiente).

### Docker Compose:

* Archivo docker-compose.yml para levantar ambos microservicios junto con sus bases de datos en un entorno de desarrollo local.

### Configuración para Kubernetes

### Archivos YAML:

* **Deployments:** Configuración de pods y réplicas para ambos microservicios.
* **Services:** Definición de servicios ClusterIP o NodePort para los

### microservicios.

* **Ingress:** Configuración de rutas basadas en contexto para exponer las APIs externamente (Opcional, este tema se cubrió en enl nivel Básico)
* **ConfigMaps y Secrets:** Manejo de variables de entorno y datos sensibles.
* **Probes:** Liveness y readiness probes configuradas.

### Manifiestos completos:

* Conjunto de manifiestos YAML organizados y documentados en carpetas para cada microservicio.

### Bases de Datos

### Scripts de Inicialización:

* Scripts SQL para inicializar las bases de datos con tablas necesarias y datos de prueba.

### Volúmenes:

* Configuración de volúmenes en Kubernetes para garantizar la persistencia de los datos.

### Documentación

### Manual de Despliegue:

* Pasos detallados para:

▪ Contenerizar y construir las imágenes.

▪ Desplegar en Kubernetes utilizando los manifiestos YAML.

▪ Configurar Docker Compose para desarrollo local.

### Diagrama de Arquitectura:

* Representación gráfica de la solución, incluyendo:

▪ Estructura de microservicios.

▪ Comunicación entre ellos.

▪ Despliegue en Kubernetes.

### Resultados

### Validación del Despliegue:

* Evidencia del despliegue exitoso en Kubernetes (por ejemplo, capturas de pantalla o logs que muestren los pods corriendo).
* Pruebas realizadas en los servicios mediante Postmano o CURL.

### Registro de Imágenes:

* URL de las imágenes Docker publicadas (Docker Hub).