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](#_Toc197822658)

[Entregables del Proyecto Final: Gestión de Eventos y Participantes 2](#_Toc197822659)

[1. Código Fuente 2](#_Toc197822660)

[1.1. Repositorios de Microservicios: 2](#_Toc197822661)

[2. Artefactos Contenerizados 3](#_Toc197822662)

[2.1. Imágenes Docker: 3](#_Toc197822663)

[2.2. Docker Compose: 5](#_Toc197822664)

[3. Configuración para Kubernetes 5](#_Toc197822665)

[3.1. Archivos YAML: 5](#_Toc197822666)

[4. microservicios. 5](#_Toc197822667)

[4.1. Manifiestos completos: 5](#_Toc197822668)

[5. Bases de Datos 6](#_Toc197822669)

[5.1. Scripts de Inicialización: 6](#_Toc197822670)

[5.2. Volúmenes: 6](#_Toc197822671)

[6. Documentación 6](#_Toc197822672)

[6.1. Manual de Despliegue: 6](#_Toc197822673)

[6.2. Diagrama de Arquitectura: 6](#_Toc197822674)

[7. Resultados 6](#_Toc197822675)

[7.1. Validación del Despliegue: 6](#_Toc197822676)

[7.2. Registro de Imágenes: 6](#_Toc197822677)

# 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.

Se creó el siguiente repositorio Git para todos los entregables del presente proyecto: <https://github.com/jonathan0284/Curso_Docker_K8S.git>

### Código Fuente

### Repositorios de Microservicios:

* Código fuente de los dos Micro servicios: gestión de eventos y gestión de inscripciones (participantes).

Ms-eventos: <https://github.com/jonathan0284/Curso_Docker_K8S/tree/main/Microservicios/ms-eventos>

Ms-participantes:

<https://github.com/jonathan0284/Curso_Docker_K8S/tree/main/Microservicios/ms-participantes>

* Estructura del proyecto conforme a las mejores prácticas (separación en capas: controlador, servicio, repositorio, entidades, etc.).

|  |
| --- |
| Ms-eventos |
|  |
| Ms-participantes |
|  |

* Uso de control de versiones (Git) con un historial claro de commits: <https://github.com/jonathan0284/Curso_Docker_K8S/activity>

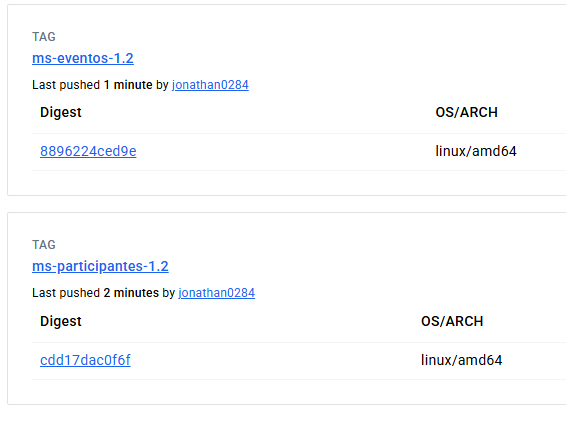
### Artefactos Contenerizados

### Imágenes Docker:

* Imágenes Docker de ambos microservicios, construidas y publicadas en un repositorio como Docker Hub.

|  |
| --- |
| Comando para compilar los Microservicios  =============================  .\mvnw clean install  Crear las imágenes Docker  =====================  docker build -t ms-eventos:1.2 .  docker build -t ms-participantes:1.2 .  Crear y levantar los contenedores  ==========================  docker run -d --name ms-participantes --network dki-network -p 7075:7075 ms-participantes:1.2  docker run -d --name ms-eventos --network dki-network -p 9090:9090 ms-eventos:1.2  Subir Imagenes a Docker HUB  ===========================  docker tag ms-participantes:1.2 jonathan0284/trabajo-final:ms-participantes-1.2  docker push jonathan0284/trabajo-final:ms-participantes-1.2  docker tag ms-eventos:1.2 jonathan0284/trabajo-final:ms-eventos-1.2  docker push jonathan0284/trabajo-final:ms-eventos-1.2 |

<https://hub.docker.com/r/jonathan0284/trabajo-final/tags>



* Imágenes optimizadas según las mejores prácticas (Dockerfile eficiente).

|  |
| --- |
| Dockerfile: Ms-eventos |
| FROM openjdk:21-jdk-slim  WORKDIR /app  COPY target/ms-participantes-0.0.1-SNAPSHOT.jar app.jar  EXPOSE 7075  ENTRYPOINT ["java", "-jar", "app.jar"] |

|  |
| --- |
| Dockerfile: Ms-participantes |
| FROM openjdk:21-jdk-slim  WORKDIR /app  COPY target/ms-participantes-0.0.1-SNAPSHOT.jar app.jar  EXPOSE 7075  ENTRYPOINT ["java", "-jar", "app.jar"] |

### Docker Compose:

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

|  |
| --- |
| services:  dki-oradb:  image: container-registry.oracle.com/database/express:21.3.0-xe  hostname: dki-oradb  container\_name: dki-oradb  environment:  ORACLE\_PWD: "dkpassword"  ORACLE\_SID: "XEPDB1"  ORACLE\_PDB: "XEPDB1"  SYSDBA\_USER: "sys"  SYSDBA\_PASSWORD: "dkpasswords"  DKUSER: "dkuser"  DKUSER\_PASSWORD: "dkpassword"  ORACLE\_CHARACTERSET: AL32UTF8  NLS\_LANG: AMERICAN\_AMERICA.AL32UTF8  ports:  - "1521:1521"  - "5500:5500"  volumes:  - dki-volume:/opt/oracle/oradata  - ./scripts:/opt/oracle/scripts/startup  networks:  - dki-network  healthcheck:  test: ["CMD", "sh", "-c", "echo 'SELECT 1 FROM DUAL;' | sqlplus system/dkpassword@localhost:1521/XE"]  interval: 30s  timeout: 10s  retries: 5  start\_period: 60s  ms-eventos:  image: jonathan0284/trabajo-final:ms-eventos-1.2  container\_name: ms-eventos  build:  context: ../ms-eventos  ports:  - "9090:9090"  networks:  - dki-network  depends\_on:  dki-oradb:  condition: service\_healthy  healthcheck:  test: ["CMD", "curl", "-f", "http://127.0.0.1:9090/eventos"]  interval: 30s  timeout: 15s  retries: 5  start\_period: 30s  ms-participantes:  image: jonathan0284/trabajo-final:ms-participantes:1.2  container\_name: ms-participantes  build:  context: ../ms-participantes  ports:  - "7075:7075"  environment:  - MS\_EVENTOS\_URL=http://ms-eventos:9090  networks:  - dki-network  depends\_on:  dki-oradb:  condition: service\_healthy  ms-eventos:  condition: service\_healthy  healthcheck:  test: ["CMD", "curl", "-f", "http://127.0.0.1:7075/participante"]  interval: 30s  timeout: 15s  retries: 5  start\_period: 30s  volumes:  dki-volume:  networks:  dki-network:  driver: bridge |

### 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 en 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 Postman o CURL.

### Registro de Imágenes:

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