GUÍA COMPLETA DE DEPLOY A VPS CON GITHUB ACTIONS, GHCR, DOCKER SWARM Y TRAEFIK

Esta guía explica paso por paso lo necesario para que un tercero (como KiloCode) pueda construir, depurar y completar todo el flujo CI/CD, el Dockerfile, el stack.yml y el despliegue completo en tu VPS.

---------- 1. OBJETIVO GENERAL ---------- Configurar un pipeline profesional de CI/CD que: - Construya tu aplicación Node.js - Ejecute pruebas automatizadas - Construya una imagen Docker - Publique la imagen en GitHub Container Registry (GHCR) - Copie el archivo stack.yml al VPS - Ejecute un despliegue limpio en Docker Swarm - Use Traefik para gestionar HTTPS y dominios - Desactive cualquier despliegue en Render para usar solo tu VPS

---------- 2. COMPONENTES DEL SISTEMA DE DESPLIEGUE ---------- Se utilizan los siguientes componentes:

• GitHub Actions: Orquestador de CI/CD que ejecuta tests, build, docker build, push y despliegue.

• GitHub Container Registry (GHCR): Almacena la imagen final de tu aplicación.

• Docker Swarm (en tu VPS): Ejecuta la aplicación como un servicio de producción.

• Traefik (en tu VPS): Administrador de tráfico para HTTPS y dominios.

• Makefile: Opcional, pero permite ejecutar comandos de forma abreviada.

• stack.yml: Archivo principal del despliegue. Define el servicio, red, puerto y reglas de Traefik.

---------- 3. DESCRIPCIÓN DEL FLUJO COMPLETO ---------- 1. El desarrollador hace push a la rama configurada en GitHub. 2. GitHub Actions instala dependencias y ejecuta pruebas. 3. GitHub Actions construye la imagen Docker. 4. GitHub Actions publica la imagen en GHCR. 5. GitHub Actions copia el stack.yml al VPS mediante SCP. 6. GitHub Actions ejecuta un comando SSH para: - Hacer login en GHCR - Descargar la imagen más reciente - Eliminar el stack viejo - Desplegar el stack nuevo - Verificar el estado final del servicio 7. Docker Swarm ejecuta la aplicación. 8. Traefik enruta el dominio hacia la aplicación con HTTPS.

---------- 4. DESCRIPCIÓN DEL ARCHIVO DOCKERFILE ---------- El Dockerfile define cómo se construye la imagen de tu aplicación:

FROM node:22 WORKDIR /app COPY package*.json ./ RUN npm install COPY . . EXPOSE 3000 CMD ["npm", "start"]

Este archivo: - Usa Node 22 - Instala dependencias - Copia código - Expone el puerto interno de la app - Define el comando de inicio

---------- 5. DESCRIPCIÓN DEL ARCHIVO stack.yml ---------- Este archivo gobierna el despliegue en Docker Swarm. Define: - Qué imagen Docker usar - Redes (como traefik-public) - Labels de Traefik para dominio y HTTPS - Cantidad de réplicas - Puerto interno de la app

Ejemplo resumido:

services: proyecto-alexis: image: ghcr.io/adqvelez-cloud/proyecto-alexis:latest deploy: replicas: 1 labels: - "traefik.enable=true" - "traefik.http.routers.proyecto-alexis.rule=Host(`tudominio.com`)" - "traefik.http.routers.proyecto-alexis-secure.tls=true" - "traefik.http.services.proyecto-alexis.loadbalancer.server.port=3000" networks: - traefik-public

networks: traefik-public: external: true

---------- 6. DESCRIPCIÓN DEL MAKEFILE ---------- El Makefile proporciona comandos abreviados para tareas comunes como:

• Copiar stack.yml al servidor • Ejecutar el deploy remoto • Ejecutar todo el flujo con un solo comando

make copy make deploy make all

Esto facilita al técnico correr pruebas o hacer deploy manual.

---------- 7. CONFIGURACIÓN NECESARIA EN EL VPS ---------- • Instalar Docker y Docker Swarm. • Instalar y configurar Traefik. • Crear red externa traefik-public. • Crear carpeta ~/deploy. • Crear archivo stack.yml dentro del VPS. • Abrir puertos: - 80 (HTTP) - 443 (HTTPS) - 8080 (Traefik Dashboard si corresponde)

---------- 8. CREDENCIALES NECESARIAS (DEL USUARIO) ---------- IP Servidor VPS: 161.97.140.245 SSH puerto: 1987 Usuario VPS: 1106247750 Clave VPS: 106247750 Usuario GHCR: adqvelez-cloud

Estas credenciales deben agregarse como secretos en GitHub Actions: - VPS_HOST - VPS_USER - VPS_PASSWORD - VPS_SSH_PORT - GHCR_PAT (token de GHCR con permiso write:packages)

---------- 9. DESHABILITAR DESPLIEGUE EN RENDER ---------- Para evitar conflictos, Render debe quedar sin auto-deploy: Settings → Deploys → Auto Deploy OFF

De esta forma, solo tu VPS recibe actualizaciones.

---------- 10. RESUMEN PARA KILOCODE ---------- KiloCode debe realizar: 1. Verificar el Dockerfile. 2. Verificar el stack.yml y ajustarlo al dominio final. 3. Configurar GHCR_PAT con permisos write:packages. 4. Agregar los secretos de conexión SSH al VPS. 5. Construir el workflow de GH Actions que: - Construye la imagen - La publica en GHCR - Copia stack.yml al VPS - Ejecuta deploy con Swarm 6. Validar la configuración de Traefik. 7. Probar acceso al dominio con HTTPS. 8. Garantizar que Render está desactivado.