 Principales comandos de “Kubernetes”

|  |
| --- |
| kubectl apply -f “fichero.yaml” |

* Aplica en Kubernetes la configuración especificada en “fichero.yaml”.

|  |
| --- |
| kubectl create deployment midespliegue --image=sergarb1/flaskparakubernetes --port=5000 |

* Crea un despliegue basado en una imagen dada y en el puerto 5000.

|  |
| --- |
| kubectl expose deployment midespliegue --type=LoadBalancer --name=midespliegue-http |

* Crea un servicio de tipo “LoadBalancer” exponeniendo “midespliegue”.

|  |
| --- |
| kubectl get pods; kubectl get services; kubectl get deployments |

* Muestra información de pods, servicios o despliegues.

|  |
| --- |
| kubectl scale deployment midespliegue --replicas=3 |

* Escala horizontalmente un despliegue a 3 réplicas.

|  |
| --- |
| kubectl autoscale deployment midespliegue --min=5 --max=10 |

* Configura autoescalado horizontal, aceptando entre 5 y 10 réplicas.

|  |
| --- |
| kubectl delete pod/deployment/service/autoscale nombre |

* Permite eliminar un pod, despliegue, servicio o autoescalado.

 Principales comandos de “MniKube”

|  |
| --- |
| minikube start |

* Inicia la máquina virtual que contiene MiniKube y pone el cluster Kubernetes en marcha

|  |
| --- |
| minikube service miservicio |

* Nos permite acceder a un servicio dentro de MiniKube desde la máquina local.

|  |
| --- |
| minikube tunnel |

* Mientras esté en ejecución, expone un servicio dentro de MiniKube a la máquina local

 Ejemplo de fichero YAML despliegue/servicio/persistencia con Kubernetes

|  |
| --- |
| *#Definimos la información del servicio* apiVersion: v1 kind: Service metadata:  name: wordpress  labels:  app: wordpress spec:  ports:  *#El servicio se expone en el puerto 80*  - port: 80  selector:  app: wordpress  tier: frontend  *#Aplicamos balanceo de carga para facilitar su escalado horizontal*  type: LoadBalancer **---** *#Definimos un volumen persistente* apiVersion: v1 kind: PersistentVolumeClaim metadata:  name: wp-pv-claim  labels:  app: wordpress spec:  *#Indica que solo puede ser montado para lectura/escritura por un nodo. Para el resto lectura.*  *#En este caso, se usa para modificar un fichero de configuración.*  accessModes:  - ReadWriteOnce  resources:  requests:  storage: 20Gi **---** *#definimos el despliegue* apiVersion: apps/v1 kind: Deployment metadata:  name: wordpress  labels:  app: wordpress spec:  selector:  matchLabels:  app: wordpress  tier: frontend  strategy:  type: Recreate  template:  metadata:  labels:  app: wordpress  tier: frontend  spec:  *#Imagen*  containers:  - image: wordpress:4.8-apache  name: wordpress  *#Indicamos variables de entorno*  env:  - name: WORDPRESS\_DB\_HOST  value: wordpress-mysql  - name: WORDPRESS\_DB\_PASSWORD  value: CEFIREdocker  ports:  - containerPort: 80  name: wordpress  volumeMounts:  - name: wordpress-persistent-storage  mountPath: /var/www/html  volumes:  - name: wordpress-persistent-storage  persistentVolumeClaim:  claimName: wp-pv-claim |