Dev-Ops Day 4

Use-Case

1. CREATE A HELLO WORLD APPLICATION

2. HOST IN DOCKER HUB

3. USING KUBECTL DEPLOY IT IN A POD

4. TEST THE APPLICATION

- minikube tunnel

- Create a kubernetes deployment
- kubectl create deployment hello-minikube1 --image=k8s.gcr.io/echoserver:1.4
- Create a kubernetes service type LoadBalancer
- kubectl expose deployment hello-minikube1 --type=LoadBalancer --port=8080
- Check external IP
- kubectl get svc

Kubectl

- // List all pods in plain-text output format.
- $ kubectl get pods
- / List all replication controllers and services together in plain-text output format.
- $ kubectl get rc,services

- // Display the details of the node with name <node-name>.
- $ kubectl describe nodes <node-name>

- // Display the details of the pod with name <pod-name>.
- $ kubectl describe pods/<pod-name>

- // Get output from running 'date' from pod <pod-name>. By default, output is from the first container.
- $ kubectl exec <pod-name> date

- // Get an interactive TTY and run /bin/bash from pod <pod-name>. By default, output is from the first container.
- $ kubectl exec -ti <pod-name> /bin/bash

- // Return a snapshot of the logs from pod <pod-name>.
- $ kubectl logs <pod-name>

- // Start streaming the logs from pod <pod-name>. This is similiar to the 'tail -f' Linux command.
- $ kubectl logs -f <pod-name>

KUBECTL

- kubectl [command] [TYPE] [NAME] [flags]
- command: Specifies the operation that you want to perform on one or more resources, for example create, get, describe, delete.
- TYPE: Specifies the resource type
- NAME: Specifies the name of the resource. Names are case-sensitive. If the name is omitted, details for all resources are displayed, for example $ kubectl get pods.
- flags: Specifies optional flags. For example, you can use the -s or --server flags to specify the address and port of the Kubernetes API server.

Thank You