LAB 5

Configure a Kubernetes Pod

Lab Objectives

In this lab we will configure a Kubernetes Pod hosting a mysql database from the command-line. We'll also log into the container we deploy into this lab using the mysql-client. We'll expose the pod in a follow-up lab.

Lab Structure - Overview

1. Run a command to deploy a pod from the command-line
2. Run a command to deploy a pod sourced from a YAML file
3. Run a command to exec a shell into a Kubernetes hosted container
4. Run a command to exec directly into the mysql-client from the command-line

Lab Overview

Conventions

Lab Guide Conventions

`reboot`	Any text a student needs to enter is printed like this.
`<your.ip>`	Any time a student needs to insert their own value, the text has brackets.
	Focuses the student's attention to a particular part of an image.
File	User Interface (UI) buttons and objects are bold.
Special Font	Unusual or important words or phrases are marked with italics.

Code Blocks

Blocks of sample code are set apart from the body and marked accordingly. It is recommended that students do not copy/paste text from the lab into their files. Extra formatting is often transferred in this process and can result in failed operations.

```
# ls -l /var/www/html/index.html
-rw-rw-r--  1 root root 1872 Jun 21 09:33 /var/www/html/index.html
# date
Wed Jun 21 09:33:42 EDT 200
```

1. Deploy a pod from the command-line

Step by Step Guide

This process will take approximately 10 minutes.

Step	Action
1.	Open a terminal console (iTerm, Terminal, PowerShell, Ubuntu Bash, Git Bash, etc).

2.	Show the nodes of the Kubernetes cluster: **`kubectl get nodes`**
	`$ kubectl get nodes`
	`NAME STATUS AGE VERSION`
	`minikube Ready 12h v1.6.0`

3.	Show information on the Kubernetes cluster: **`kubectl cluster-info`**
	`$ kubectl cluster-info`
	`Kubernetes master is running at https://192.168.64.13:8443`
	`KubeDNS is running at https://192.168.64.13:8443/api/v1/proxy/names...`

4.	Show all objects in the default kubernetes namespace: **`kubectl get all`**
	`$ kubectl get all`
	`NAME CLUSTER-IP EXTERNAL-IP PORT(S) AGE`
	`svc/kubernetes 10.0.0.1 <none> 443/TCP 12h`

5. Show all pods in the default namespace: **kubectl get pod**

```
$ kubectl get pod
No resources found.
```

6. Show pods in the kube-system namespace:
kubectl get pod --namespace=kube-system

```
$ kubectl get pod --namespace=kube-system
NAME                          READY     STATUS    RESTARTS   AGE
kube-addon-manager-minikube   1/1       Running   1          12h
kube-dns-v20-dzfn3            3/3       Running   3          12h
kubernetes-dashboard-r92px    1/1       Running   1          12h
```

7. Create a standalone nginx pod in the default namespace:
kubectl run nginx_pod_lab --image=nginx:1.11-alpine --port=80

```
$ kubectl run nginx_pod_lab --image=nginx:1.11-alpine --port=80
The Deployment "nginx_pod_lab" is invalid:
```

What happened? If there is an error, what went wrong?

8. Create a standalone nginx pod in the default namespace:
kubectl run nginx-pod-lab --image=nginx:1.11-alpine --port=80

```
$ kubectl run nginx-pod-lab --image=nginx:1.11-alpine --port=80
deployment "nginx-pod-lab" created
```

9. Show all pods in the default namespace: **kubectl get pod**

```
$ kubectl get pod
NAME                             READY     STATUS    RESTARTS   AGE
nginx-pod-lab-1856640016-mglns   1/1       Running   0          40s
```

10. Show newly create pod: **kubectl get pod nginx-pod-lab-<name>**

```
$ kubectl get pod nginx-pod-lab-1856640016-mglns
NAME                             READY     STATUS    RESTARTS   AGE
nginx-pod-lab-1856640016-mglns   1/1       Running   0          4m
```

11. **kubectl get pod nginx-pod-lab-<name> -o json**

12. **kubectl get pod nginx-pod-lab-<name> -o yaml**

13.	Cleanup the environment: `kubectl delete all --all`

2. Create a pod from a local manifest

Step by Step Guide

This process will take approximately 5 minutes.

Step	Action
1.	Open a terminal console (iTerm, Terminal, PowerShell, Ubuntu Bash, Git Bash, etc).

2.	In your working lab directory, create a file: **nginx-kube.yaml**
	With the following content (this manifest will deploy an nginx container)
	
	`apiVersion: v1`
	`kind: Pod`
	`metadata:`
	`  labels:`
	`  name: nginx-web`
	`  name: nginx-web`
	`spec:`
	`  containers:`
	`  - image: nginx:1.11-alpine`
	`  name: nginx-web`
	`  ports:`
	`  - containerPort: 80`
	`  name: http`
	`  protocol: TCP`

3.	Deploy a pod from the : **`kubectl create -f nginx-kube.yaml`**
	
	`$ kubectl create -f nginx-kube.yaml`
	`pod "nginx-web" created`

4.	Show the deployed pods : **`kubectl get pods`**
	Take note of the new pod that was created.
	
	`$ kubectl get pods`
	`NAME READY STATUS RESTARTS AGE`
	`nginx-pod-lab-1856640016-mglns 1/1 Running 0`
	`nginx-web 1/1 Running 0`

5.	Cleanup the environment: **`kubectl delete pods --all`**
6.	Recheck the environment: **`kubectl get all`**

4. Create a pod with two containers

Step by Step Guide

This process will take approximately 5 minutes.

Step	Action
1.	Open a terminal console (iTerm, Terminal, PowerShell, Ubuntu Bash, Git Bash, etc).
2.	Create a file called "**two-containers-one-pod.yaml**" with the following content: Can you determine what this manifest does (don't worry we'll cover it soon enough)?

```
apiVersion: v1
kind: Pod
metadata:
  name: two-containers
spec:

  restartPolicy: Never

  volumes:
  - name: shared-data
    emptyDir: {}

  containers:

  - name: nginx-container
    image: nginx
    volumeMounts:
    - name: shared-data
      mountPath: /usr/share/nginx/html

  - name: debian-container
    image: debian
    volumeMounts:
    - name: shared-data
      mountPath: /pod-data
    command: ["/bin/sh"]
    args: ["-c", "echo Hello from the debian container >
/pod-data/index.html"]
```

3.	Deploy the pod from the manifest `kubectl create -f two-containers-one-pod.yaml`

```
$ kubectl create -f two-containers-one-pod.yaml
pod "two-containers" created
```

4.	Show the deployed pods : **kubectl get pods** Take note of the new pod that was created.

```
$ kubectl get pods
NAME             READY     STATUS      RESTARTS   AGE
two-containers   1/2       Completed   0          0s
```

5. Show the deployed pods : **kubectl describe pods two-containers**
Take note of the events associated with the creation of the containers in the pod.

```
$ kubectl describe pods two-containers
Name:       two-containers
Namespace:  default
Node:       minikube/192.168.64.13
Start Time: Mon, 24 Apr 2017 10:18:23 -0700
Labels:           <none>
Annotations:<none>
Status:           Running
IP:
172.17.0.4
Controllers:<none>
Containers:
  nginx-container:
```

6. Log into the nginx container of the two-containers pod:
kubectl exec -it two-containers -c nginx-container -- /bin/bash

```
$ kubectl exec -it two-containers -c nginx-container -- /bin/bash
root@two-containers:/#
```

7. In the container, install curl and procps for the next steps:
apt-get update && apt-get install -y curl procps

8. Validate that nginx is running in the container: **ps aux**

```
root@two-containers:/# ps aux
USER       PID %CPU %MEM    VSZ   RSS COMMAND
root         1  0.0  0.1  31876  5280 nginx: master process nginx
nginx        5  0.0  0.0  32264  2964 nginx: worker process
```

9. Check the default nginx website using curl: **curl localhost**

```
root@two-containers:/# curl localhost
Hello from the debian container
```

5. Deploy a MySQL DB and connect to the client

Step by Step Guide

This process will take approximately 5 minutes.

Step	Action
1.	Open a terminal console (iTerm, Terminal, PowerShell, Ubuntu Bash, Git Bash, etc).

2.	Deploy a MySQL DB:
	`kubectl run mysql-demo --image=mysql:5.5 --port=3306 \`
	`--env="MYSQL_ROOT_PASSWORD=password"`
	
	`$ kubectl run mysql-demo --image=mysql:5.5 \`
	`--env="MYSQL_ROOT_PASSWORD=password" \`
	`--port=3306`
	
	`deployment "mysql-demo" created`

3.	Show all pods via **`kubectl get pod`**
	Note the name of the pod you just created.
	
	`$ kubectl get pods`
	`NAME READY STATUS RESTARTS AGE`
	`mysql-db-4111478071-9b67g 1/1 Running 0 4h`

4.	Use kubectl to log into the container (similiar to the **`docker exec -it`** command)
	`kubectl exec -it <mysql-pod-name> -- mysql -ppassword`
	
	`$ kubectl exec -it mysql-db-4111478071-9b67g -- mysql -ppassword`
	`Welcome to the MySQL monitor. Commands end with ; or \g.`
	`Your MySQL connection id is 2`
	`Server version: 5.5.55 MySQL Community Server (GPL)`
	
	`Copyright (c) 2000, 2017, Oracle and/or its affiliates. All rights reserved.`
	
	`Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.`
	
	`Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.`
	
	`mysql>`

Lab Complete!