Multi Container Pods

This lab consists of a list of exercises to demonstrate and understand the most commonly used kubernetes commands and concepts to understand `multi-container pod` design patterns.

Learning Outcomes

After completing the lab, you will be able to understand and use Kubernetes concepts based on use-case scenarios in the following domains:

1. Multi container pod design patterns

Start the minikube

1. Start minikube locally `minikube start --driver=virtualbox`
2. Verify the kubectl context `kubectl config get-contexts` is set to minikube. If not, set it to minikube `kubectl config use-context minikube`

> Create all manifest resources in the directory `~/workspace/kubernetes-manifests/competencies`. Watch out for the right file names in the solution section.

Multi-Container Pod Design patterns

1. Create a pod that defines an application container which writes the current date to a log file every five seconds. The sidecar container is nginx serving that log file in a shared directory.

   ▼ Click to see solution

   `~/workspace/kubernetes-manifests/competencies/pods/14.yaml`

```yaml
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: alpine
  name: alpine
spec:
  volumes:
    - name: log-date-vol
      emptyDir: {}
  containers:
  - image: alpine
    name: alpine
    imagePullPolicy: IfNotPresent
    command: ["/bin/sh"]
    args: ["-c", "while true; do date >> /etc/kal-director
y/date-file.txt; sleep 5; done"]
    volumeMounts:
      - name: log-date-vol
        mountPath: /etc/kal-directory
  - image: nginx
    name: nginx
    imagePullPolicy: IfNotPresent
    volumeMounts:
      - name: log-date-vol
        mountPath: /etc/kal-directory
```

```
kubectl apply -f ~/workspace/kubernetes-manifests/competen
cies/pods/14.yaml
```

```
kubectl exec -it alpine -c nginx -- cat /etc/kal-director
y/date-file.txt
```

```
kubectl delete po alpine
```

2. Create a pod that defines an application container which writes the process to a file every 10 seconds. Create a adapter container that writes the datetime and the top 3 processes consuming maximum memory in a report file.

   HINT: Use the command `ps aux` and sort it based on memory usage . The output format may slightly vary based on the OS kernel.

   Sample Output

```
Sat Nov 7 11:18:11 UTC 2020
USER      COMMAND           PID %CPU %MEM   VSZ   RSS   STAR
TED      TIME
student   java            16213  2.7  3.7  5900  2976  07:3
4:42 00:06:47
student   gnome-shell      2373 21.7  1.7  2612  1565  Feb
22 2-04:02:42
```

```
student   chrome          14101  1.3  1.1  2512  596  13:0
9:55 00:17:56
```

▼ Click to see solution

`~/workspace/kubernetes-manifests/competencies/pods/15.yaml`

```
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: multi-container-pod
  name: multi-container-pod
spec:
  volumes:
    - name: shared-vol
      emptyDir: {}
  containers:
    - image: ubuntu
      name: ubuntu
      imagePullPolicy: IfNotPresent
      command: ["/bin/sh"]
      args: ["-c", "while true; do ps aux --sort=-pmem > /
logs/output.txt; sleep 10; done"]
      volumeMounts:
        - name: shared-vol
          mountPath: /logs
    - image: alpine
      name: alpine
      imagePullPolicy: IfNotPresent
      command: ["/bin/sh"]
      args: ["-c", "while true; do date > /logs/report.tx
t; cat /logs/output.txt | head -4 >> /logs/report.txt; sle
ep 10; done"]
      volumeMounts:
        - name: shared-vol
          mountPath: /logs
```

```
kubectl apply -f ~/workspace/kubernetes-manifests/competen
cies/pods/15.yaml
```

```
kubectl get po multi-container-pod
```

```
kubectl exec -it multi-container-pod -c alpine -- cat /log
s/report.txt
```

```
kubectl delete po  multi-container-pod
```

3. Create a pod that defines an application container which writes the current date and memory usage to a log file every five seconds. The adapter container will inspect the contents of the app's log file, reformat it, and write the correctly formatted output to a new file

   HINT: Use the command `free -tw --giga` for printing and choose the appropriate image similar to `ubuntu`

   Adapter Input Sample

```
Thu Mar 4 11:26:28 GMT 2021
              total        used        free      shared
buffers       cache   available
Mem:             32           9          10           0
0          11          22
Swap:             2           0           2
Total:           34           9          12
```

   Adapter Output Sample

```
Date: Thu Mar  4 09:36:42 GMT 2021
Total Memory: 34GB
Free Memory: 12GB
```

▼ Click to see solution
`~/workspace/kubernetes-manifests/competencies/pods/16.yaml`

```
apiVersion: v1
kind: Pod
metadata:
  labels:
    run: multi-container-pod
  name: multi-container-pod
spec:
  volumes:
    - name: shared-vol
      emptyDir: {}
  containers:
    - image: ubuntu
      name: ubuntu
      imagePullPolicy: IfNotPresent
      command: ["/bin/sh"]
      args: ["-c", "while true; do date > /logs/output.tx
t; free -tw --giga >> /logs/output.txt; sleep 10; done"]
```

```
      volumeMounts:
        - name: shared-vol
          mountPath: /logs
    - image: alpine
      name: alpine
      imagePullPolicy: IfNotPresent
      command: ["/bin/sh"]
      args: ["-c", "while true; do echo  'Date: ' $(cat /logs/output.txt |  head -1) > /logs/report.txt; echo 'Total Memory:' $(cat /logs/output.txt |  grep Total: | tr -s ' ' | cut -d ' ' -f 2) GB >> /logs/report.txt; echo 'Free Memory:' $(cat /logs/output.txt |  grep Total: | tr -s ' ' | cut -d ' ' -f 3) GB >> /logs/report.txt; sleep 10; done"]
      volumeMounts:
        - name: shared-vol
          mountPath: /logs
```

```
kubectl apply -f ~/workspace/kubernetes-manifests/competencies/pods/16.yaml
```

```
kubectl get po multi-container-pod
```

```
kubectl exec -it multi-container-pod -c alpine -- cat /logs/report.txt
```

```
kubectl delete po  multi-container-pod
```