**MulticontainerPod**

Create a pod with two containers:

1. Container1: image = nginx
2. Container2: image = tomcat

| **apiVersion**: v1  **kind**: Pod  **metadata**:  **name**: multicontainer-pod  **spec**:  **containers**:  **- name**: nginx  **image**: nginx  **- name**: tomcat  **image**: tomcat |
| --- |

Verify that pod is running

kubectl get pods

Access both applications inside the pods:

curl <pod-IP>

curl <pod-IP>:8080

**CustomCommands**

| **apiVersion**: v1  **kind**: Pod  **metadata**:  **name**: pod-custom-command  **spec**:  **containers**:  **- name**: hello  **image**: alpine  **command**: ['sh', '-c', 'echo "Hello, Kubernetes!" && sleep 3600'] |
| --- |

**Resource Limits**

Pod Container : image nginx

Resource:

request: cpu 500m

Limit: 2 cpus

| **apiVersion**: v1  **kind**: Pod  **metadata**:  **creationTimestamp**: null  **labels**:  **run**: resource-limit  **name**: resource-limit  **spec**:  **containers**:  **- image**: vish/stress  **name**: resource-limit  **ports**:  **- containerPort**: 80  **resources**:  **limits**:  **cpu**: "1"  **requests**:  **cpu**: "0.5"  **args**: ["-cpus", "2"] # This is argument to the command in image and not a resource limit |
| --- |

**ReplicaSet**

| **apiVersion**: apps/v1  **kind**: ReplicaSet  **metadata**:  **name**: myreplicaset  **spec**:  **replicas**: 3  **selector**:  **matchLabels**:  **app**: prod  **template**:  **metadata**:  **name**: pods-for-rs  **labels**:  **app**: prod  **spec**:  **containers**:  **- name**: nginx  **image**: nginx |
| --- |

**Deployment**

Image: httpd

replicas: 5

kubectl create deployment mydep --image=httpd --port 80 --replicas=5 --dry-run -o yaml > deployment.yaml

| **apiVersion**: apps/v1  **kind**: Deployment  **metadata**:  **creationTimestamp**: null  **labels**:  **app**: mydep  **name**: mydep  **spec**:  **replicas**: 5  **selector**:  **matchLabels**:  **app**: mydep  **template**:  **metadata**:  **creationTimestamp**: null  **labels**:  **app**: mydep  **spec**:  **containers**:  **- image**: httpd  **name**: httpd  **ports**:  **- containerPort**: 80 |
| --- |

Update the deployment to use “httpd:2.4.10” image

kubectl set image deployment “deploymentname” “containername”=httpd:2.4.10 --record

Check the rollout history

kubectl rollout history deployments “deploymentname”

Scale the deployment to 10 replicas

kubectl scale deployment “deploymentname” --replicas=10

**Commands**:

| echo 'labsuser ALL=(ALL) NOPASSWD:ALL' > /etc/sudoers.d/labsuser  whoami  docker ps  kubectl version  kubeadm init  kubectl get nodes  kubeadm init -h  kubectl get nodes  kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml  kubectl get nodes  journalctl -u kubelet  kubelet -h  kubelet -h | grep version  kubelet --version  kubectl --version  kubeadm --version  kubeadm -h  kubeadm version  echo 'source <(kubectl completion bash)' >>~/.bashrc  source .bashrc  kubectl get pods -A  kubectl get pod  kubectl get pod -A  kubectl get namespaces  kubectl create ns simplilearn  kubectl get namespaces  kubectl run --name mypod --image nginx --namespace simplilearn  kubectl run mypod --image nginx --namespace simplilearn  kubectl get pods  kubectl get pods --namespace simplilearn  kubectl get pods --all-namespaces  kubectl get pods -A  kubectl api-resources  kubectl api-resources --namespaced  kubectl api-resources --namespace false  kubectl api-resources --namespace=false  kubectl api-resources -h  kubectl api-resources --namespaced=false  history  ll  vim pod\_multicontainer.yaml  kubectl apply -f pod\_multicontainer.yaml  kubectl get pods  kubectl describe pod multicontainer-pod  kubectl get pods -o wide  curl 192.168.235.131  curl 192.168.235.131:8080  cat pod\_multicontainer.yaml  kubectl get pods -o wide  docker run ubuntu  docker ps -a  docker inspect ubuntu:latest  docker run ubuntu cat /etc/lsb-release  docker run ubuntu echo hostname  docker run ubuntu hostname  docker pull busybox  docker inspect busybox:latest  vim pod-custom-command.yaml  kubectl apply -f pod-custom-command.yaml  kubectl get pods  kubectl describe pod pod-custom-command  cat pod-custom-command.yaml  kubectl get pods  kubectl logs multicontainer-pod  kubectl logs multicontainer-pod nginx  kubectl logs multicontainer-pod tomcat  kubectl logs multicontainer-pod tomcat -f  kubectl logs -h  kubectl logs multicontainer-pod --all-containers  kubectl exec multicontainer-pod  kubectl exec multicontainer-pod -h  kubectl exec multicontainer-pod nginx ls -l /var/www/html  kubectl exec multicontainer-pod nginx -- ls -l /var/www/html  kubectl exec multicontainer-pod nginx -- ls -l /etc  kubectl exec multicontainer-pod -c tomact -- ls -l /etc  kubectl exec multicontainer-pod -c tomcat -- ls -l /etc  kubectl exec -it multicontainer-pod -c tomcat  kubectl exec -it multicontainer-pod -c tomcat -- bash  kubectl run anotherpod --image nginx --port 80 --dry-run  kubectl run anotherpod --image nginx --port 80 --dry-run -o yaml  kubectl run anotherpod --image nginx --port 80 --dry-run -o yaml > anotherpod.yaml  ll  vim anotherpod.yaml  kubectl describe nodes worker1  vim anotherpod.yaml  kubectl apply -f anotherpod.yaml  kubectl describe pod anotherpod  kubectl run resource-limit --image vish/stress --port 80 --dry-run -o yaml > pod-resource-limit.yaml  vim pod-resource-limit.yaml  kubectl apply -f pod-resource-limit.yaml  kubectl get pods -o wide  kubectl describe node worker1  kubectl logs -h  kubectl logs resource-limit --alsologtostderr  kubectl describe node worker1  vim pod-resource-limit.yaml  kubectl get pods  kubectl delete pod anotherpod  kubectl get pods  vim replicaset.yaml  kubectl explain replicaset  kubectl explain replicaset.spec  vim replicaset.yaml  kubectl apply -f replicaset.yaml  kubectl get rs  kubectl get pods  kubectl delete pod myreplicaset-s5swq  kubectl get pods  kubectl get pods -o wide  vim replicaset.yaml  cat pod-resource-limit.yaml  vim replicaset.yaml  kubectl delete pod --all  kubectl create deployment --name myde --image=nginx:1.19.0 --port 80 --replicas=3 --dry-run -o yaml > deployment.yaml  kubectl create deployment mydep --image=nginx:1.19.0 --port 80 --replicas=3 --dry-run -o yaml > deployment.yaml  vim deployment.yaml  kubectl apply -d deployment.yaml  kubectl apply -f deployment.yaml  kubectl get deployments.apps  kubectl get rs  kubectl get pods  kubectl delete rs myreplicaset  kubectl get pods  kubectl get pods -o wide  kubectl get deployments.apps -o wide  kubectl set image deployment mydep  kubectl set image deployment mydep nginx=nginx:1.20.0 --record  kubectl get pods  cat deployment.yaml  kubectl get deployments.apps -o wide  kubectl get rs  kubectl rollout  kubectl rollout deployment mydep  kubectl rollout deployments mydep  kubectl rollout history deployments mydep  kubectl set image deployment mydep nginx=nginx:1.21.0 --record -h  kubectl set image deployment mydep nginx=nginx:1.21.0 --record  kubectl rollout status deployment mydep  kubectl rollout history deployments mydep  kubectl rollout undo deployment mydep  kubectl rollout history deployments mydep  kubectl get rs  kubectl get deployments.apps -o wide  kubectl scale deployment mydep --replicas=9  kubectl get deployments.apps -o wide  kubectl get pods -o wide  kubectl rollout pause deployment mydep  kubectl rollout unpause deployment mydep  kubectl rollout resume deployment mydep |
| --- |