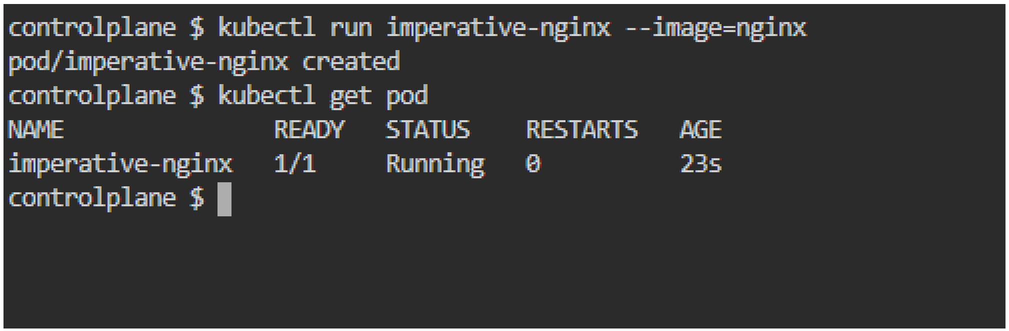
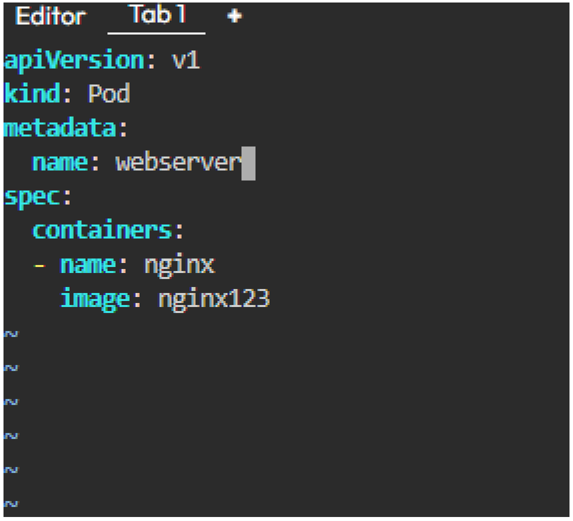
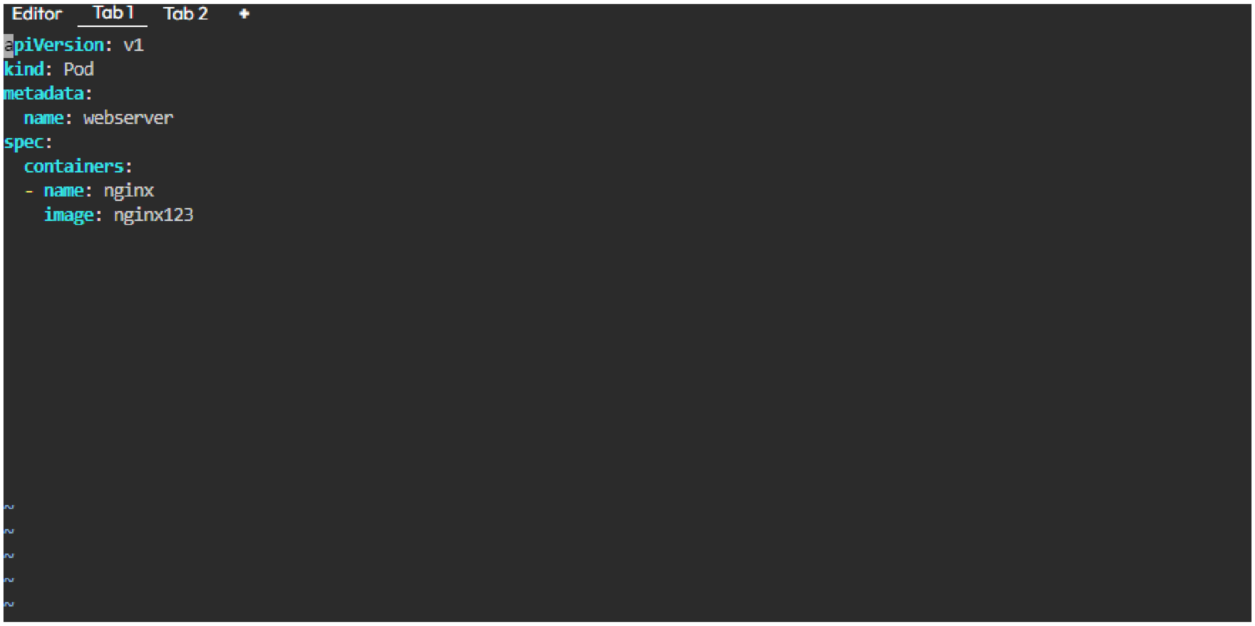


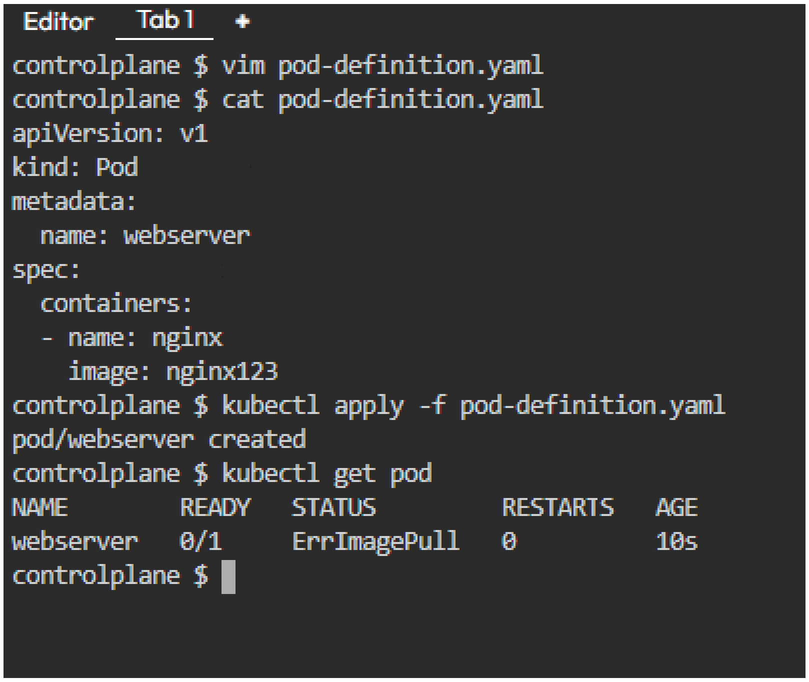
```
controlplane $ kubectl run imperative-nginx --image=nginx
pod/imperative-nginx created
controlplane $ kubectl get pod
NAME               READY   STATUS    RESTARTS   AGE
imperative-nginx   1/1     Running   0          23s
controlplane $ 
```



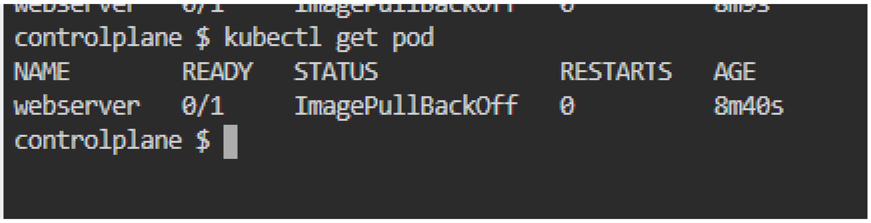
```
apiVersion: v1
kind: Pod
metadata:
  name: webserver
spec:
  containers:
  - name: nginx
    image: nginx123
~
~
~
~
~
~
```



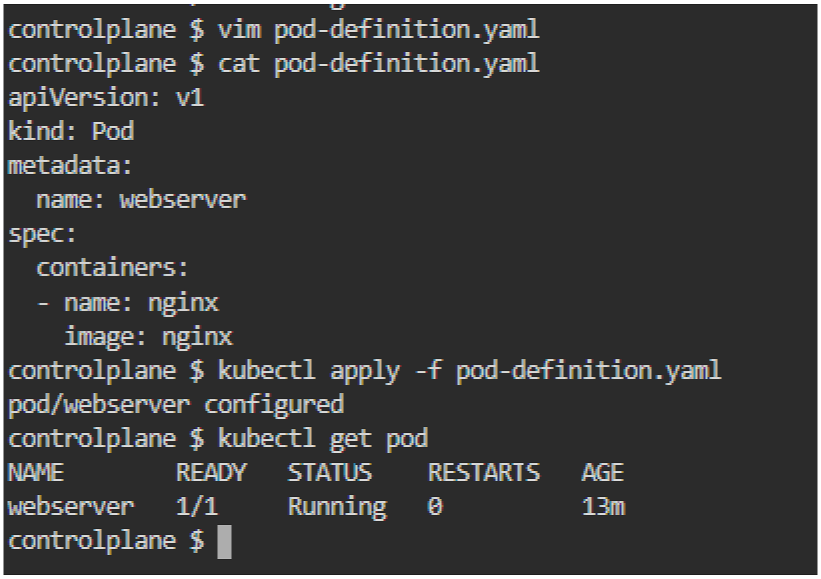
```
apiVersion: v1
kind: Pod
metadata:
  name: webserver
spec:
  containers:
  - name: nginx
    image: nginx123
~
~
~
~
~
```



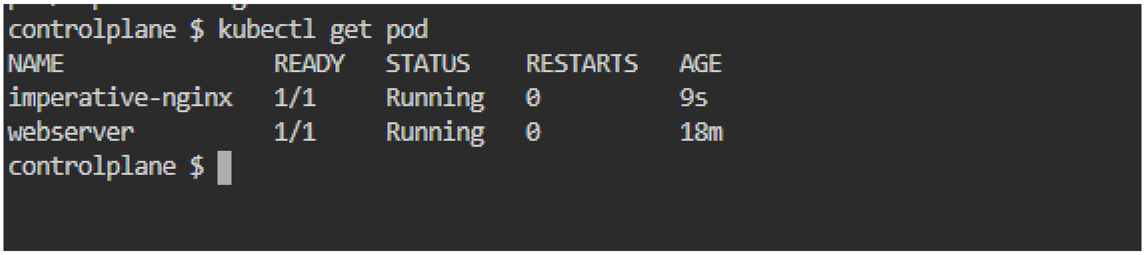
```
controlplane $ vim pod-definition.yaml
controlplane $ cat pod-definition.yaml
apiVersion: v1
kind: Pod
metadata:
  name: webserver
spec:
  containers:
  - name: nginx
    image: nginx123
controlplane $ kubectl apply -f pod-definition.yaml
pod/webserver created
controlplane $ kubectl get pod
NAME        READY   STATUS         RESTARTS   AGE
webserver   0/1     ErrImagePull   0          10s
controlplane $
```



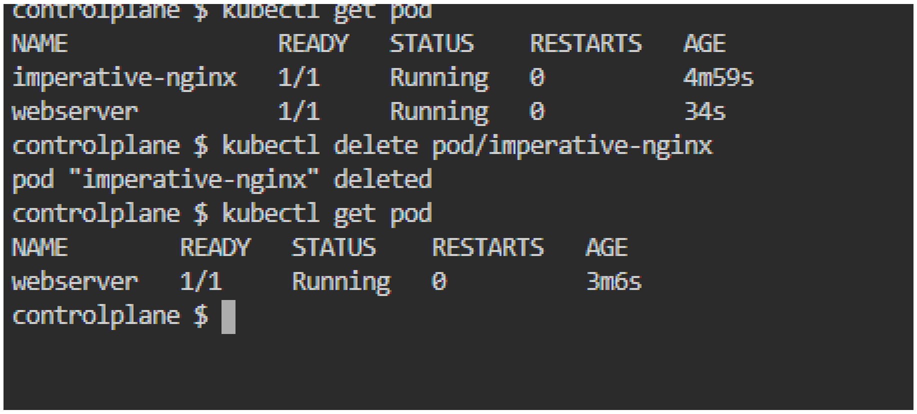
```
webserver   0/1     ImagePullBackOff   0          8m9s
controlplane $ kubectl get pod
NAME        READY   STATUS             RESTARTS   AGE
webserver   0/1     ImagePullBackOff   0          8m40s
controlplane $
```



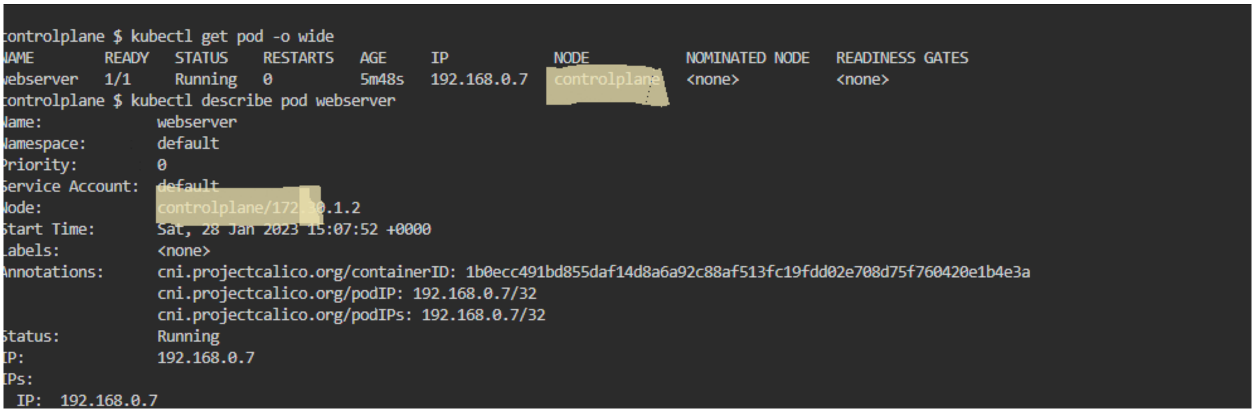
```
controlplane $ vim pod-definition.yaml
controlplane $ cat pod-definition.yaml
apiVersion: v1
kind: Pod
metadata:
  name: webserver
spec:
  containers:
  - name: nginx
    image: nginx
controlplane $ kubectl apply -f pod-definition.yaml
pod/webserver configured
controlplane $ kubectl get pod
NAME        READY   STATUS    RESTARTS   AGE
webserver   1/1     Running   0          13m
controlplane $
```



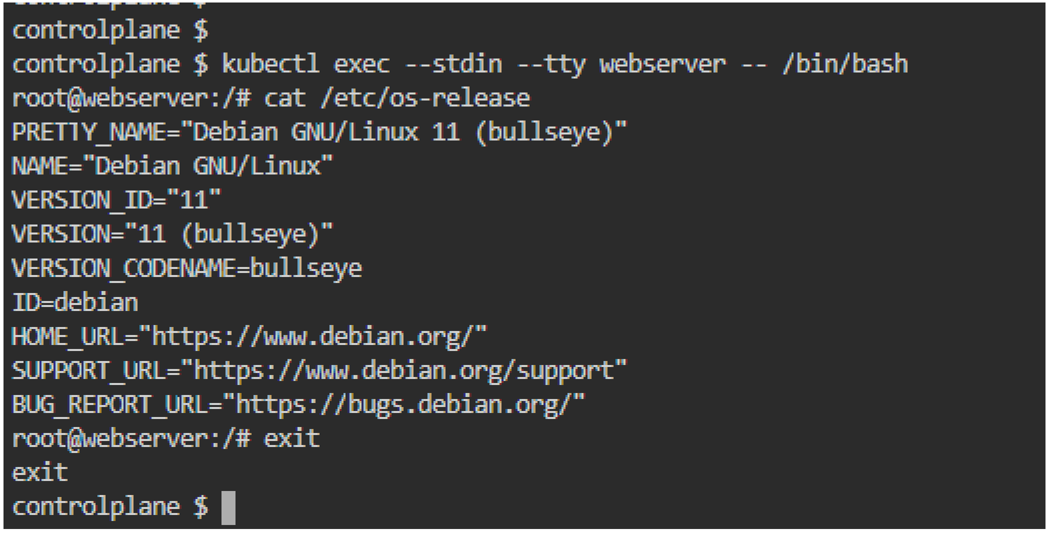
```
controlplane $ kubectl get pod
NAME               READY   STATUS    RESTARTS   AGE
imperative-nginx   1/1     Running   0          9s
webserver          1/1     Running   0          18m
controlplane $
```



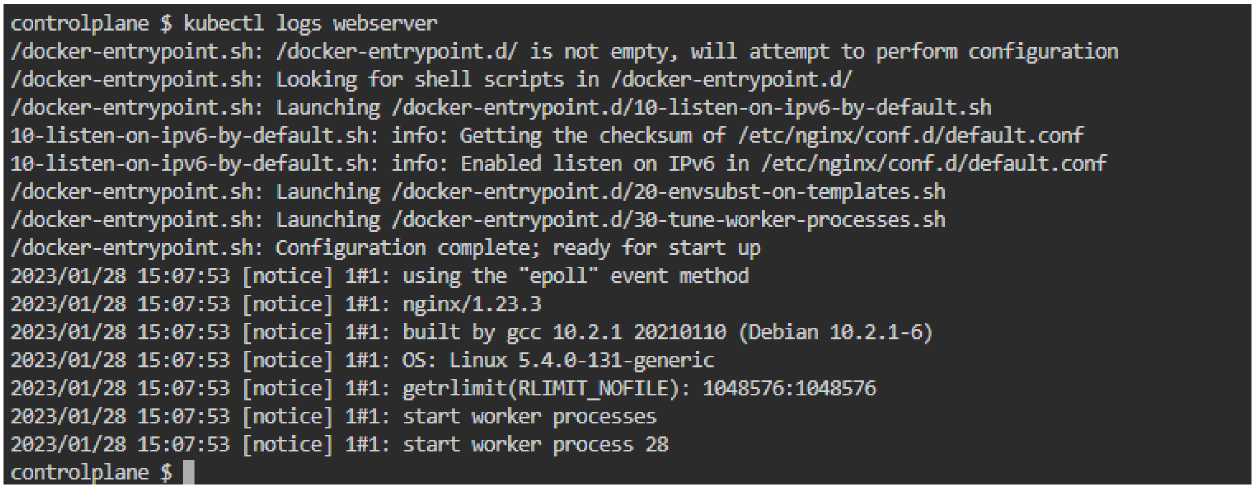
```
controlplane $ kubectl get pod
NAME               READY   STATUS    RESTARTS   AGE
imperative-nginx   1/1     Running   0          4m59s
webserver          1/1     Running   0          34s
controlplane $ kubectl delete pod/imperative-nginx
pod "imperative-nginx" deleted
controlplane $ kubectl get pod
NAME        READY   STATUS    RESTARTS   AGE
webserver   1/1     Running   0          3m6s
controlplane $
```



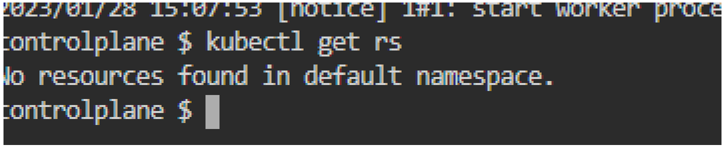
```
ontrolplane $ kubectl get pod -o wide
AME         READY   STATUS    RESTARTS   AGE     IP            NODE           NOMINATED NODE   READINESS GATES
ebserver    1/1     Running   0          5m48s   192.168.0.7   controlplane   <none>           <none>
ontrolplane $ kubectl describe pod webserver
ame:             webserver
amespace:        default
riority:         0
ervice Account:  default
ode:             controlplane/172.30.1.2
tart Time:       Sat, 28 Jan 2023 15:07:52 +0000
abels:           <none>
nnotations:      cni.projectcalico.org/containerID: 1b0ecc491bd855daf14d8a6a92c88af513fc19fdd02e708d75f760420e1b4e3a
                 cni.projectcalico.org/podIP: 192.168.0.7/32
                 cni.projectcalico.org/podIPs: 192.168.0.7/32
tatus:           Running
P:               192.168.0.7
Ps:
 IP:  192.168.0.7
```



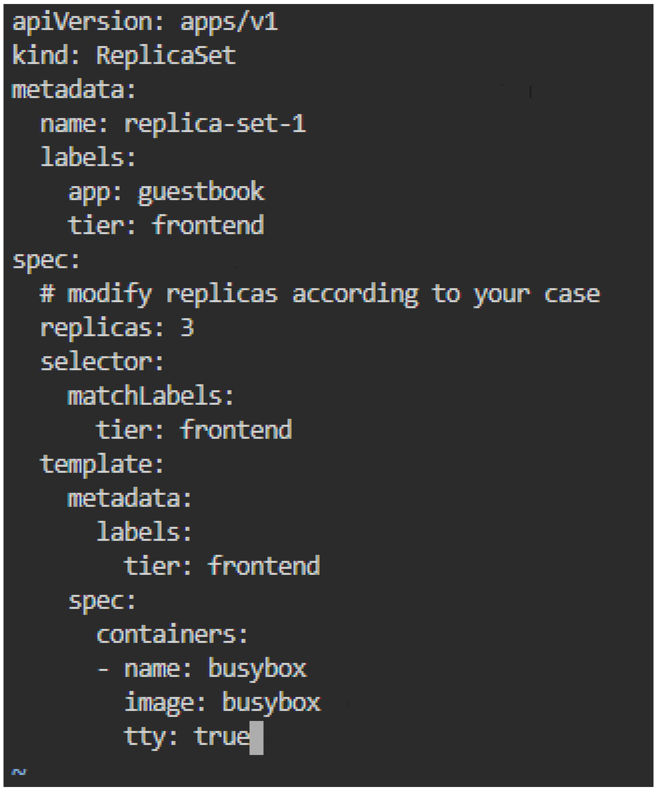
```
controlplane $
controlplane $ kubectl exec --stdin --tty webserver -- /bin/bash
root@webserver:/# cat /etc/os-release
PRETTY_NAME="Debian GNU/Linux 11 (bullseye)"
NAME="Debian GNU/Linux"
VERSION_ID="11"
VERSION="11 (bullseye)"
VERSION_CODENAME=bullseye
ID=debian
HOME_URL="https://www.debian.org/"
SUPPORT_URL="https://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"
root@webserver:/# exit
exit
controlplane $
```



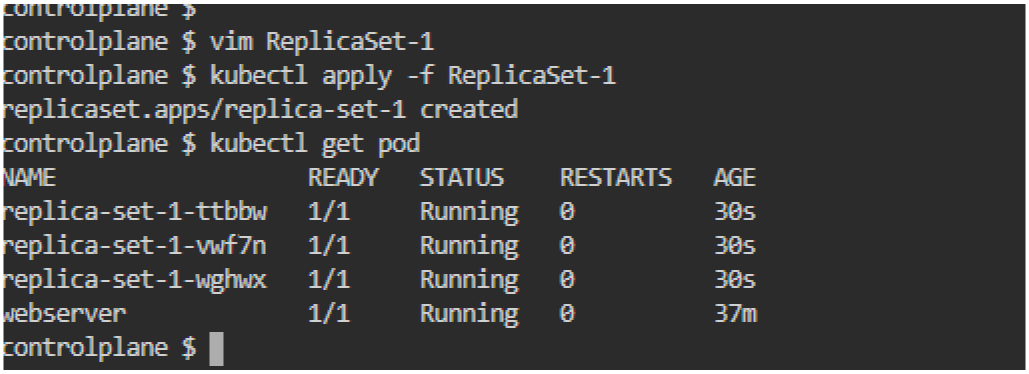
```
controlplane $ kubectl logs webserver
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2023/01/28 15:07:53 [notice] 1#1: using the "epoll" event method
2023/01/28 15:07:53 [notice] 1#1: nginx/1.23.3
2023/01/28 15:07:53 [notice] 1#1: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
2023/01/28 15:07:53 [notice] 1#1: OS: Linux 5.4.0-131-generic
2023/01/28 15:07:53 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2023/01/28 15:07:53 [notice] 1#1: start worker processes
2023/01/28 15:07:53 [notice] 1#1: start worker process 28
controlplane $
```



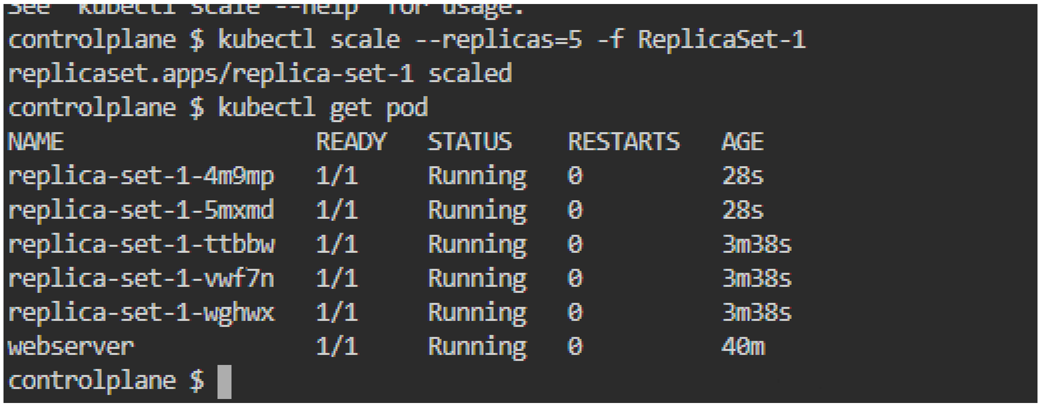
```
2023/01/28 15:07:53 [notice] 1#1: start worker proce
ontrolplane $ kubectl get rs
o resources found in default namespace.
ontrolplane $
```



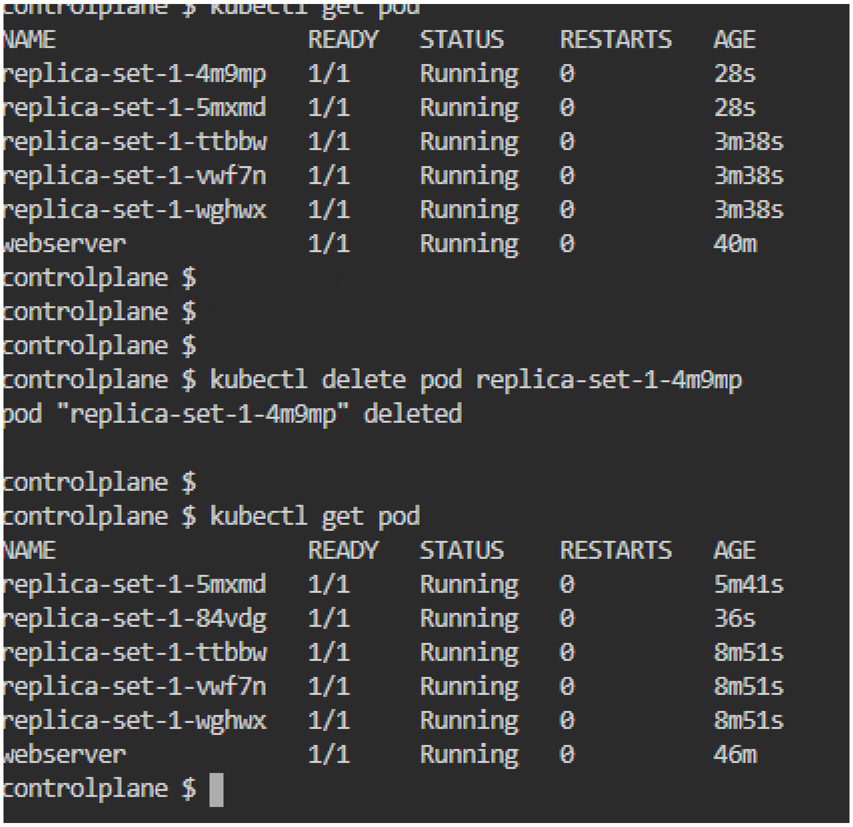
```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: replica-set-1
  labels:
    app: guestbook
    tier: frontend
spec:
  # modify replicas according to your case
  replicas: 3
  selector:
    matchLabels:
      tier: frontend
  template:
    metadata:
      labels:
        tier: frontend
    spec:
      containers:
      - name: busybox
        image: busybox
        tty: true
~
```



```
controlplane $
controlplane $ vim ReplicaSet-1
controlplane $ kubectl apply -f ReplicaSet-1
replicaset.apps/replica-set-1 created
controlplane $ kubectl get pod
NAME                  READY   STATUS    RESTARTS   AGE
replica-set-1-ttbbw   1/1     Running   0          30s
replica-set-1-vwf7n   1/1     Running   0          30s
replica-set-1-wghwx   1/1     Running   0          30s
webserver             1/1     Running   0          37m
controlplane $
```



```
See 'kubectl scale --help' for usage.
controlplane $ kubectl scale --replicas=5 -f ReplicaSet-1
replicaset.apps/replica-set-1 scaled
controlplane $ kubectl get pod
NAME                  READY   STATUS    RESTARTS   AGE
replica-set-1-4m9mp   1/1     Running   0          28s
replica-set-1-5mxmd   1/1     Running   0          28s
replica-set-1-ttbbw   1/1     Running   0          3m38s
replica-set-1-vwf7n   1/1     Running   0          3m38s
replica-set-1-wghwx   1/1     Running   0          3m38s
webserver             1/1     Running   0          40m
controlplane $
```



```
controlplane $ kubectl get pod
NAME                  READY   STATUS    RESTARTS   AGE
replica-set-1-4m9mp   1/1     Running   0          28s
replica-set-1-5mxmd   1/1     Running   0          28s
replica-set-1-ttbbw   1/1     Running   0          3m38s
replica-set-1-vwf7n   1/1     Running   0          3m38s
replica-set-1-wghwx   1/1     Running   0          3m38s
webserver             1/1     Running   0          40m
controlplane $
controlplane $
controlplane $
controlplane $ kubectl delete pod replica-set-1-4m9mp
pod "replica-set-1-4m9mp" deleted

controlplane $
controlplane $ kubectl get pod
NAME                  READY   STATUS    RESTARTS   AGE
replica-set-1-5mxmd   1/1     Running   0          5m41s
replica-set-1-84vdg   1/1     Running   0          36s
replica-set-1-ttbbw   1/1     Running   0          8m51s
replica-set-1-vwf7n   1/1     Running   0          8m51s
replica-set-1-wghwx   1/1     Running   0          8m51s
webserver             1/1     Running   0          46m
controlplane $
```

Qu6. The READY column means how many containers in each pod are ready

Qu12. LOG is the file extension for an automatically produced file that contains a record of events from certain software and operating systems. While they can contain a number of things, log files are often used to show all events associated with the system or application that created them. For example, your backup program might keep log files showing exactly what happened (or didn't happen) during a backup. Windows keeps all kinds of log files for its various services.

The point of a log file is to keep track of what's happening behind the scenes and if something should happen within a complex system, you have access to a detailed list of events that took place before the malfunction. Basically, whatever the application, server, or OS thinks needs to be recorded.

While most log files contain the .log file extension, sometimes applications may use the .txt extension or a different proprietary extension, instead.

Qu.16 5 pods

Qu.17 The deleted POD is a part of a replica set so we have to find 5 pods running over all time, So If one pod terminated, it will create another pod