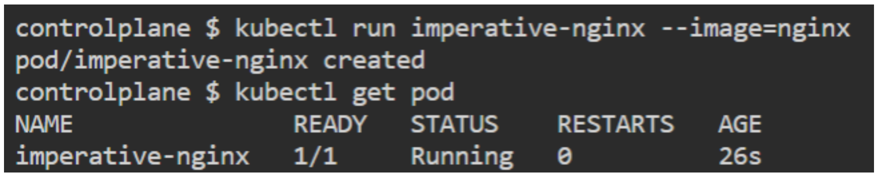
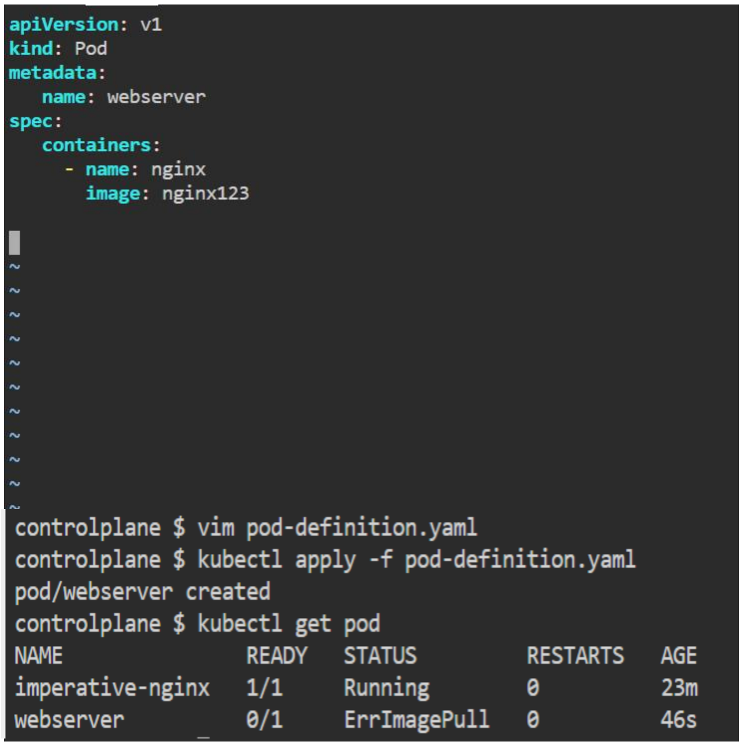
Lab1

1- Create a pod with the name “imperative-nginx” and with the image nginx and latest tag. using Imperative command (not yaml).



```
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          26s
```

2- Create a pod with the name webserver and with the image “nginx123”Use a pod-definition YAML file.

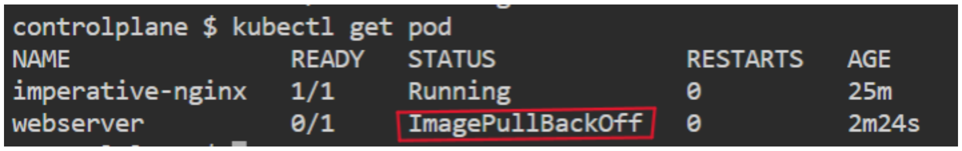


```
apiVersion: v1
kind: Pod
metadata:
   name: webserver
spec:
   containers:
     - name: nginx
       image: nginx123

~
~
~
~
~
~
~
~
~
~
~
controlplane $ vim pod-definition.yaml
controlplane $ kubectl apply -f pod-definition.yaml
pod/webserver created
controlplane $ kubectl get pod
NAME               READY   STATUS         RESTARTS   AGE
imperative-nginx   1/1     Running        0          23m
webserver          0/1     ErrImagePull   0          46s
```

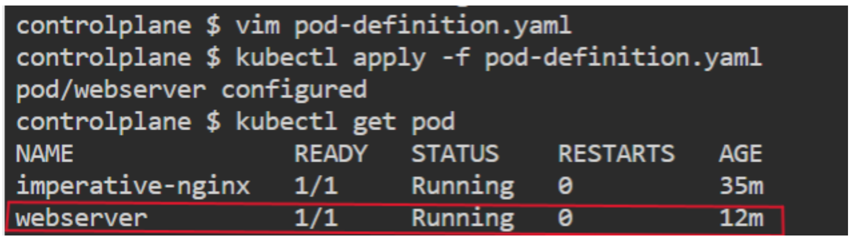
3- What is the nginx pod status?

Error in the image pulling.



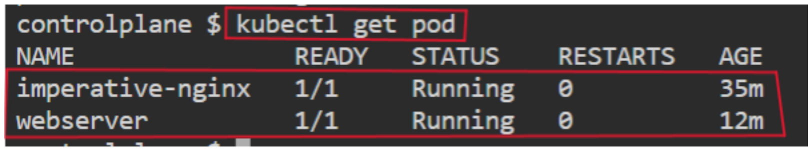
```
controlplane $ kubectl get pod
NAME               READY   STATUS             RESTARTS   AGE
imperative-nginx   1/1     Running            0          25m
webserver          0/1     ImagePullBackOff   0          2m24s
```

4- Change the nginx pod image to “nginx” check the status again



```
controlplane $ vim pod-definition.yaml
controlplane $ kubectl apply -f pod-definition.yaml
pod/webserver configured
controlplane $ kubectl get pod
NAME               READY   STATUS    RESTARTS   AGE
imperative-nginx   1/1     Running   0          35m
webserver          1/1     Running   0          12m
```

5- How many pods are running in the system? Type the command to show this

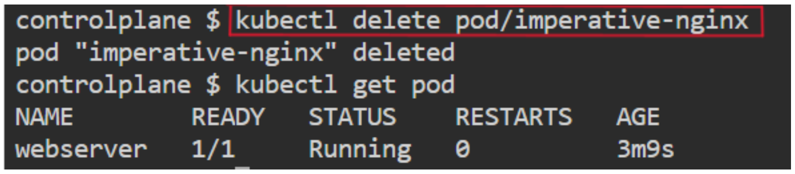


```
controlplane $ kubectl get pod
NAME               READY   STATUS    RESTARTS   AGE
imperative-nginx   1/1     Running   0          35m
webserver          1/1     Running   0          12m
```

6- What does READY column in the output of get pods command indicate?

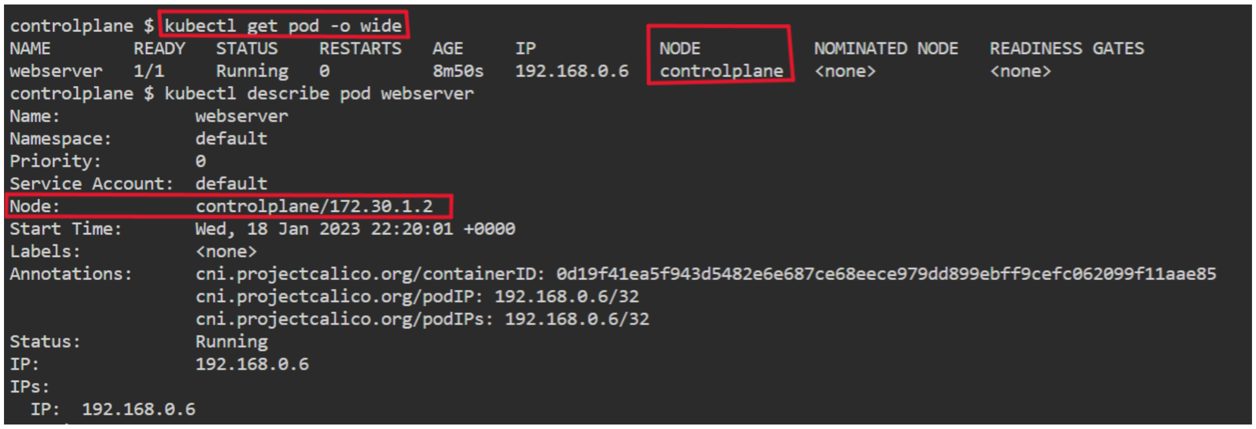
The container numbers in the pod are ready.

7- Delete first pod named imperative-nginx you just created. Type the command to do this



```
controlplane $ kubectl delete pod/imperative-nginx
pod "imperative-nginx" deleted
controlplane $ kubectl get pod
NAME        READY   STATUS    RESTARTS   AGE
webserver   1/1     Running   0          3m9s
```

8- Which node is pod named webserver running on (list two commands to do this)

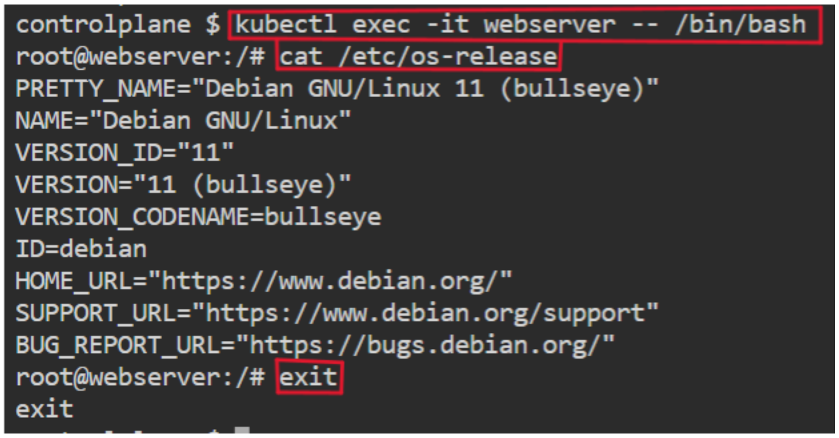


```
controlplane $ kubectl get pod -o wide
NAME        READY   STATUS    RESTARTS   AGE     IP            NODE           NOMINATED NODE   READINESS GATES
webserver   1/1     Running   0          8m50s   192.168.0.6   controlplane   <none>           <none>
controlplane $ kubectl describe pod webserver
Name:             webserver
Namespace:        default
Priority:         0
Service Account:  default
Node:             controlplane/172.30.1.2
Start Time:       Wed, 18 Jan 2023 22:20:01 +0000
Labels:           <none>
Annotations:      cni.projectcalico.org/containerID: 0d19f41ea5f943d5482e6e687ce68eece979dd899ebff9cefc062099f11aae85
                  cni.projectcalico.org/podIP: 192.168.0.6/32
                  cni.projectcalico.org/podIPs: 192.168.0.6/32
Status:           Running
IP:               192.168.0.6
IPs:
  IP:  192.168.0.6
```

9- Get a shell to the running container i.e ssh into it (figure out the command)

10- Run cat /etc/os-release inside the container

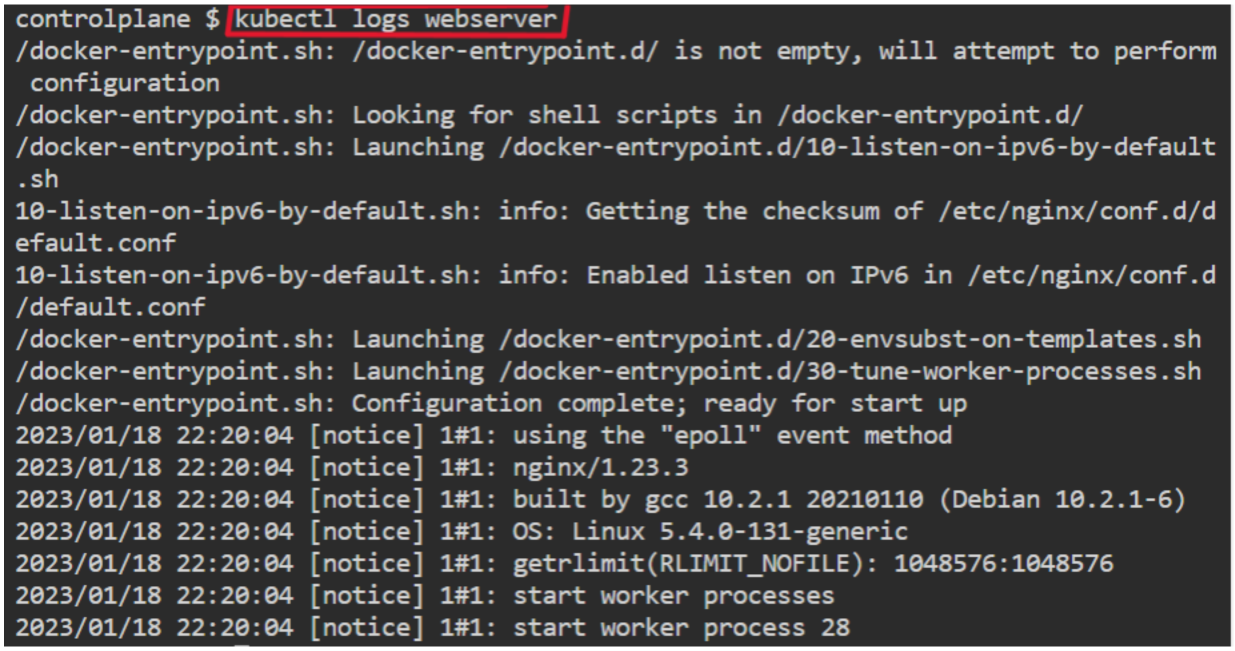
11- Exit from the shell (/bin/bash) session



```
controlplane $ kubectl exec -it 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
```

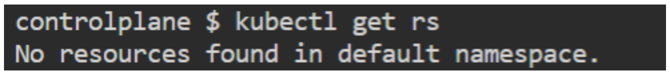
12- Get logs of pod, what are logs and what they are used for?

they are used for keep track of what our pod/application is doing or to keep track of users, new requests, etc. And for troubleshooting; whenever something goes wrong or our application crashes, we check the logs.



```
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/d
efault.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
2023/01/18 22:20:04 [notice] 1#1: using the "epoll" event method
2023/01/18 22:20:04 [notice] 1#1: nginx/1.23.3
2023/01/18 22:20:04 [notice] 1#1: built by gcc 10.2.1 20210110 (Debian 10.2.1-6)
2023/01/18 22:20:04 [notice] 1#1: OS: Linux 5.4.0-131-generic
2023/01/18 22:20:04 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2023/01/18 22:20:04 [notice] 1#1: start worker processes
2023/01/18 22:20:04 [notice] 1#1: start worker process 28
```

13- How many ReplicaSets exist on the system?



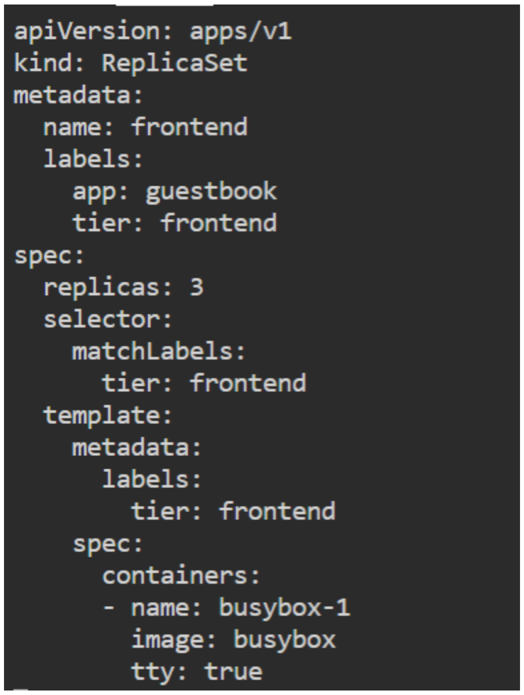
```
controlplane $ kubectl get rs
No resources found in default namespace.
```

14- create a ReplicaSet

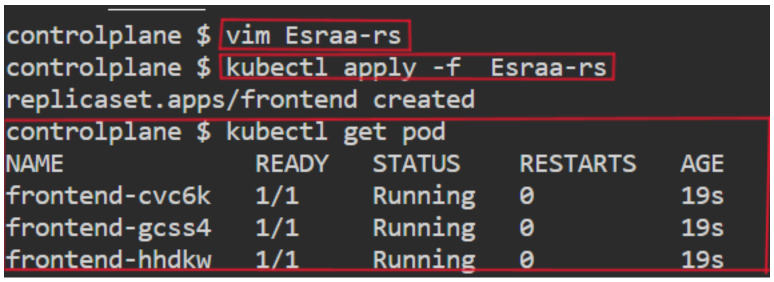
withname= replica-set-1

image= busybox

replicas= 3

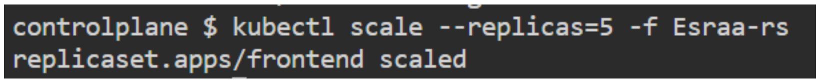


```
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: frontend
  labels:
    app: guestbook
    tier: frontend
spec:
  replicas: 3
  selector:
    matchLabels:
      tier: frontend
  template:
    metadata:
      labels:
        tier: frontend
    spec:
      containers:
      - name: busybox-1
        image: busybox
        tty: true
```



```
controlplane $ vim Esraa-rs
controlplane $ kubectl apply -f  Esraa-rs
replicaset.apps/frontend created
controlplane $ kubectl get pod
NAME             READY   STATUS    RESTARTS   AGE
frontend-cvc6k   1/1     Running   0          19s
frontend-gcss4   1/1     Running   0          19s
frontend-hhdkw   1/1     Running   0          19s
```

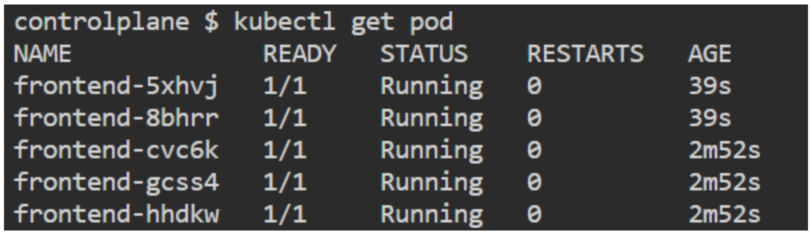
15- Scale the ReplicaSet replica-set-1 to 5 PODs.



```
controlplane $ kubectl scale --replicas=5 -f Esraa-rs
replicaset.apps/frontend scaled
```

16- How many PODs are READY in the replica-set-1?

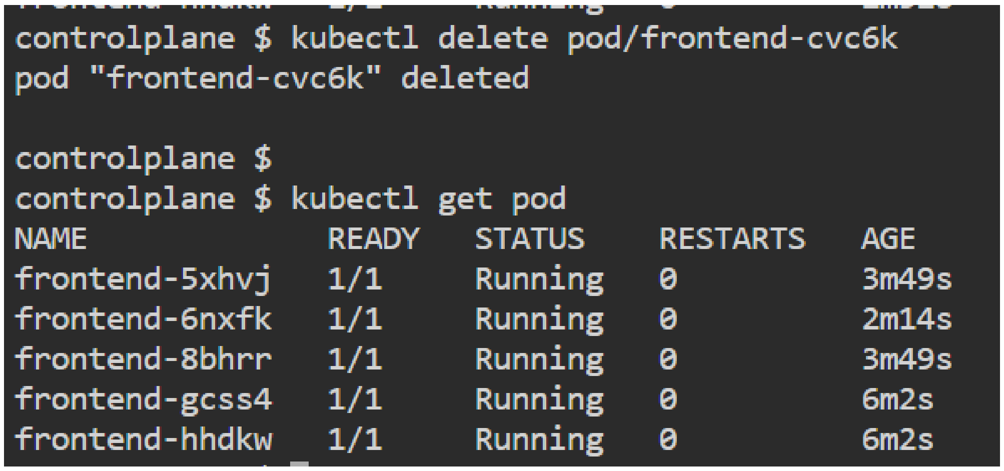
5 pods are ready.



```
controlplane $ kubectl get pod
NAME             READY   STATUS    RESTARTS   AGE
frontend-5xhvj   1/1     Running   0          39s
frontend-8bhrr   1/1     Running   0          39s
frontend-cvc6k   1/1     Running   0          2m52s
frontend-gcss4   1/1     Running   0          2m52s
frontend-hhdkw   1/1     Running   0          2m52s
```

17- Delete any one of the 5 PODs then check How many PODs exist now? Why are there still 5 PODs, even after you deleted one?

Because one of the replicaset features is to keep the number of running pods equals to the desired replicas in the yaml file so once the pod is deleted another one is created.



```
controlplane $ kubectl delete pod/frontend-cvc6k
pod "frontend-cvc6k" deleted

controlplane $
controlplane $ kubectl get pod
NAME             READY   STATUS    RESTARTS   AGE
frontend-5xhvj   1/1     Running   0          3m49s
frontend-6nxfk   1/1     Running   0          2m14s
frontend-8bhrr   1/1     Running   0          3m49s
frontend-gcss4   1/1     Running   0          6m2s
frontend-hhdkw   1/1     Running   0          6m2s
```