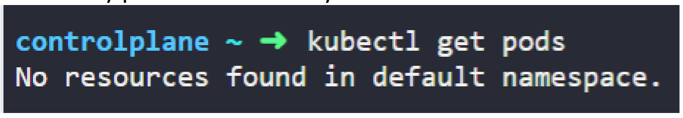
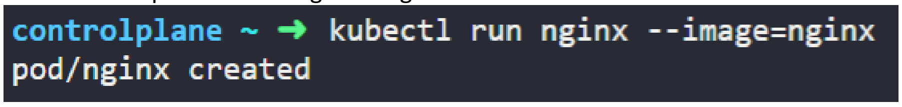
How many pods exist on the system? 0



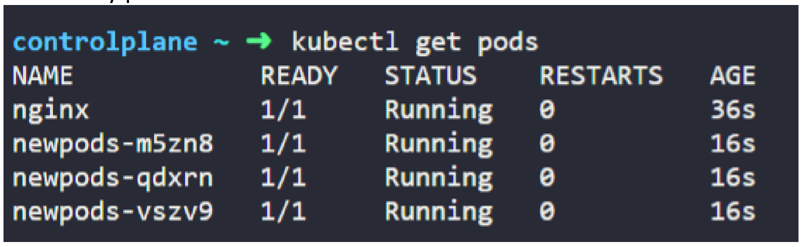
```
controlplane ~ ➜  kubectl get pods
No resources found in default namespace.
```

Create a new pod with the nginx image.



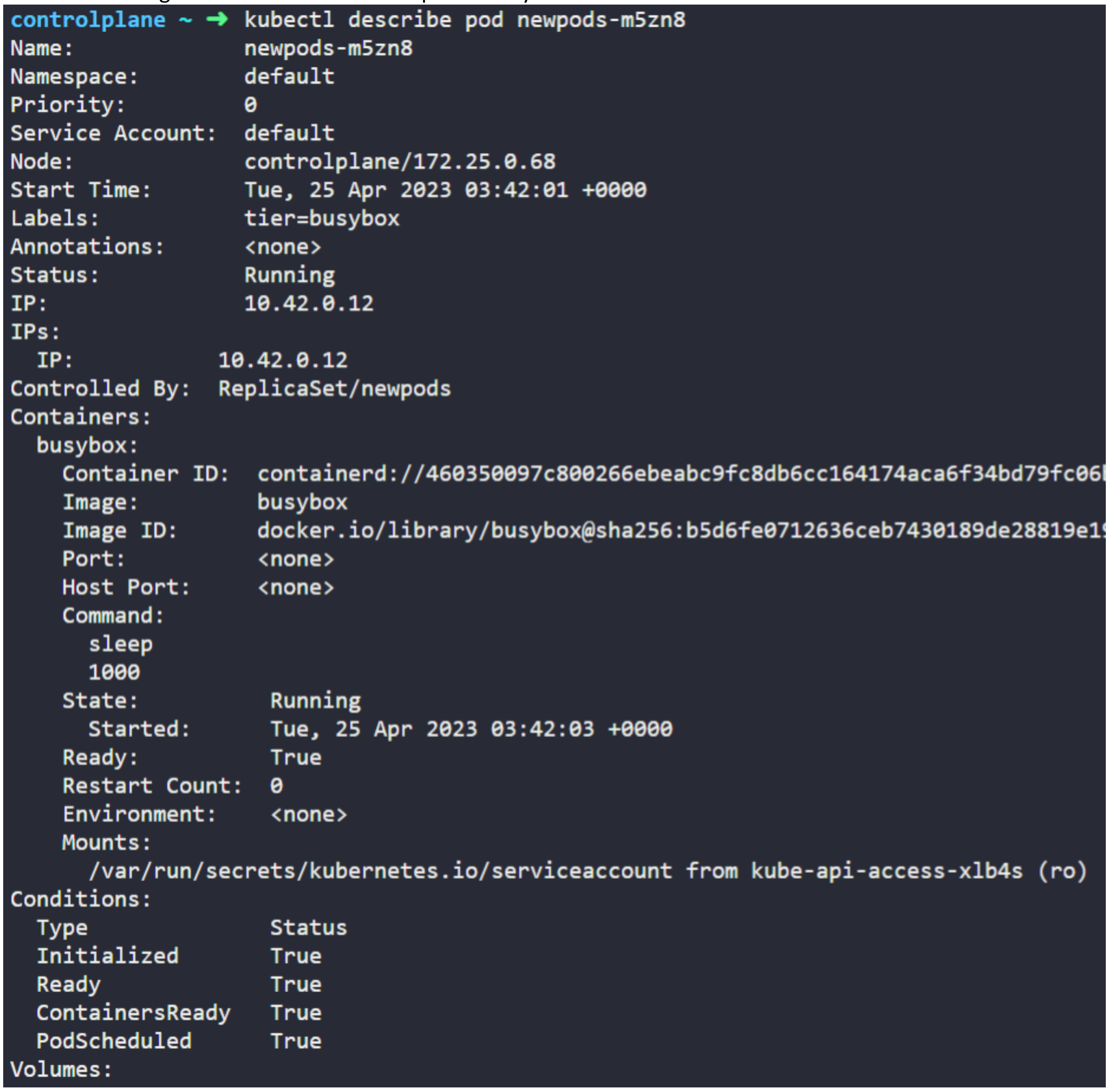
```
controlplane ~ ➜  kubectl run nginx --image=nginx
pod/nginx created
```

How many pods are created now? 4



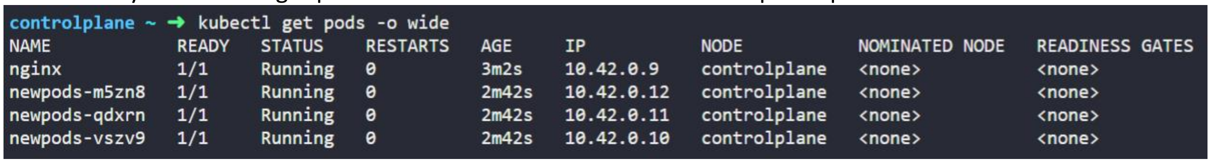
```
controlplane ~ ➜  kubectl get pods
NAME            READY   STATUS    RESTARTS   AGE
nginx           1/1     Running   0          36s
newpods-m5zn8   1/1     Running   0          16s
newpods-qdxrn   1/1     Running   0          16s
newpods-vszv9   1/1     Running   0          16s
```

What is the image used to create the new pods? busybox



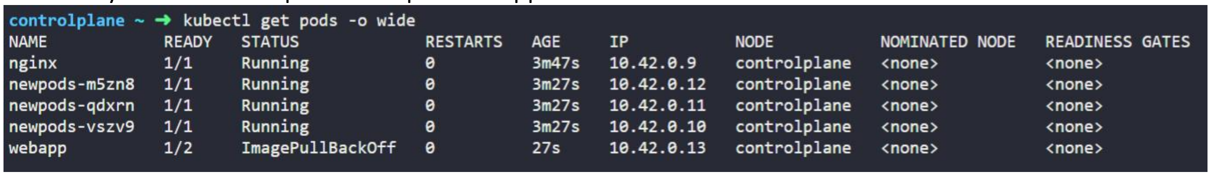
```
controlplane ~ ➜  kubectl describe pod newpods-m5zn8
Name:             newpods-m5zn8
Namespace:        default
Priority:         0
Service Account:  default
Node:             controlplane/172.25.0.68
Start Time:       Tue, 25 Apr 2023 03:42:01 +0000
Labels:           tier=busybox
Annotations:      <none>
Status:           Running
IP:               10.42.0.12
IPs:
  IP:           10.42.0.12
Controlled By:  ReplicaSet/newpods
Containers:
  busybox:
    Container ID:  containerd://460350097c800266ebeabc9fc8db6cc164174aca6f34bd79fc06
    Image:         busybox
    Image ID:      docker.io/library/busybox@sha256:b5d6fe0712636ceb7430189de28819e1
    Port:          <none>
    Host Port:     <none>
    Command:
      sleep
      1000
    State:          Running
      Started:      Tue, 25 Apr 2023 03:42:03 +0000
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-xlb4s (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
```

Which nodes are these pods placed on? Controlplane
Run the command kubectl describe pod newpods-<id> and look at the node field.
Alternatively run kubectl get pods -o wide and check for the node the pod is placed on.



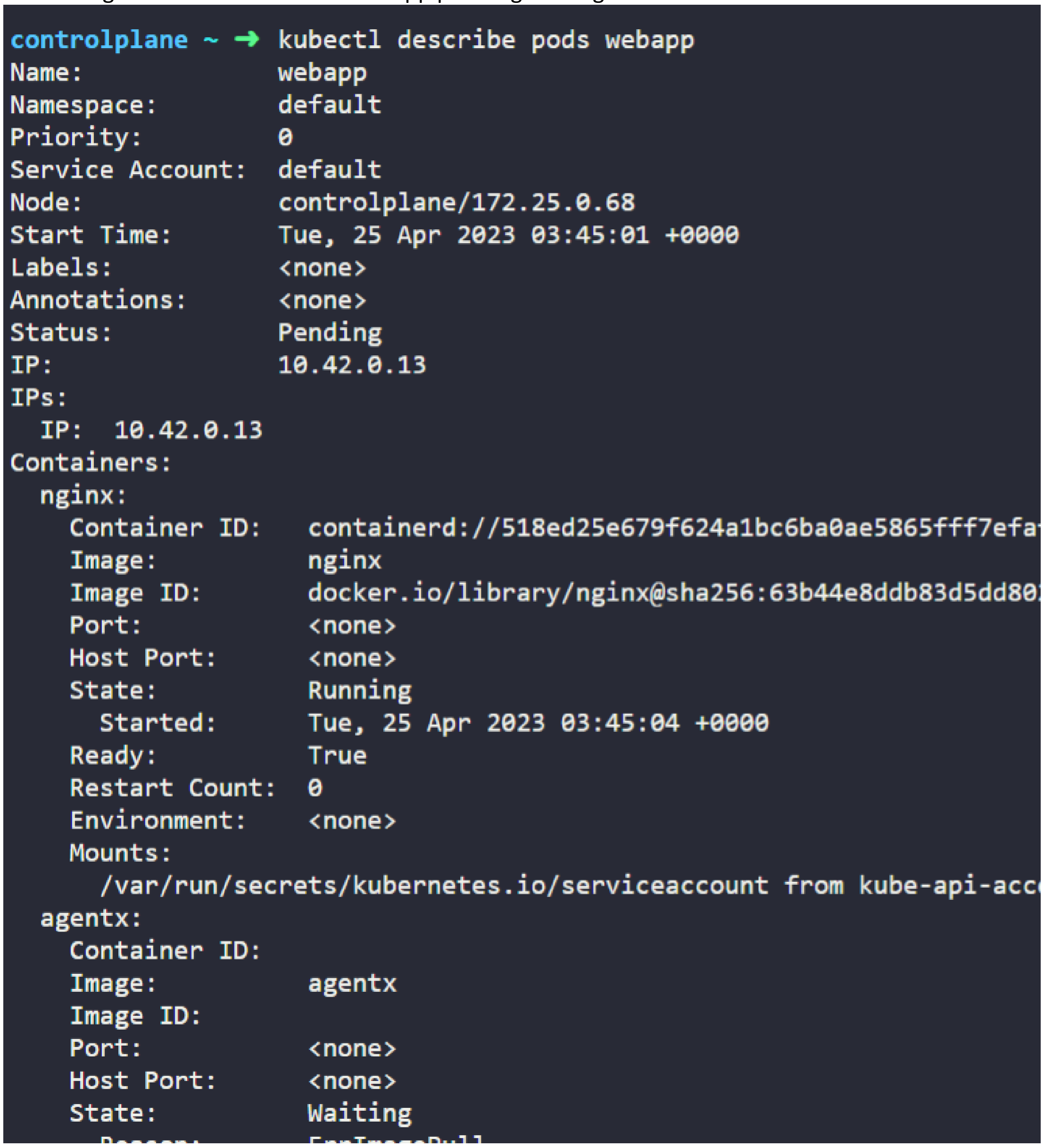
```
controlplane ~ ➜ kubectl get pods -o wide
NAME            READY   STATUS    RESTARTS   AGE     IP           NODE           NOMINATED NODE   READINESS GATES
nginx           1/1     Running   0          3m2s    10.42.0.9    controlplane   <none>           <none>
newpods-m5zn8   1/1     Running   0          2m42s   10.42.0.12   controlplane   <none>           <none>
newpods-qdxrn   1/1     Running   0          2m42s   10.42.0.11   controlplane   <none>           <none>
newpods-vszv9   1/1     Running   0          2m42s   10.42.0.10   controlplane   <none>           <none>
```

How many containers are part of the pod webapp? 2



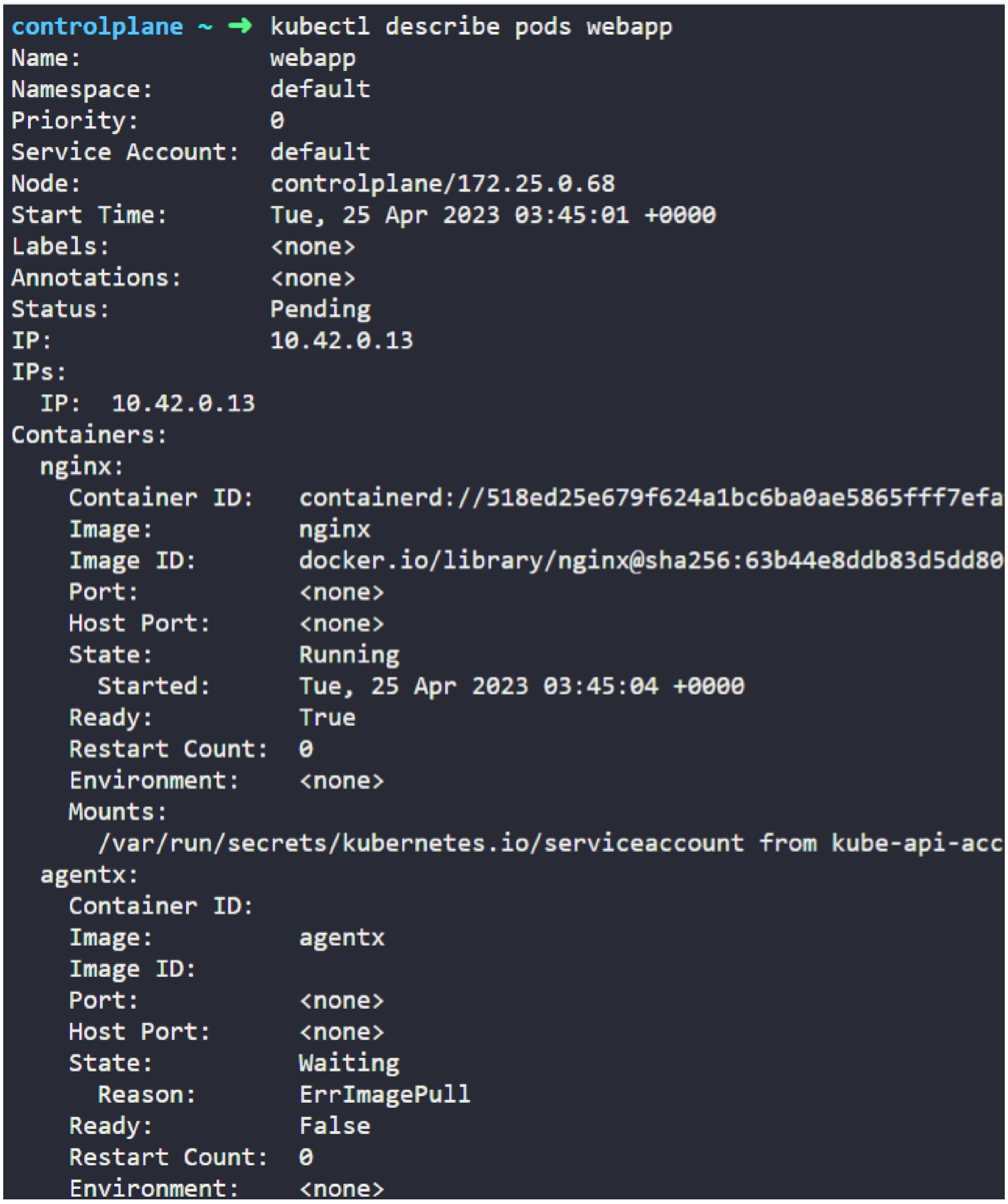
```
controlplane ~ ➜ kubectl get pods -o wide
NAME            READY   STATUS             RESTARTS   AGE     IP           NODE           NOMINATED NODE   READINESS GATES
nginx           1/1     Running            0          3m47s   10.42.0.9    controlplane   <none>           <none>
newpods-m5zn8   1/1     Running            0          3m27s   10.42.0.12   controlplane   <none>           <none>
newpods-qdxrn   1/1     Running            0          3m27s   10.42.0.11   controlplane   <none>           <none>
newpods-vszv9   1/1     Running            0          3m27s   10.42.0.10   controlplane   <none>           <none>
webapp          1/2     ImagePullBackOff   0          27s     10.42.0.13   controlplane   <none>           <none>
```

What images are used in the new webapp pod? Nginx & agentx



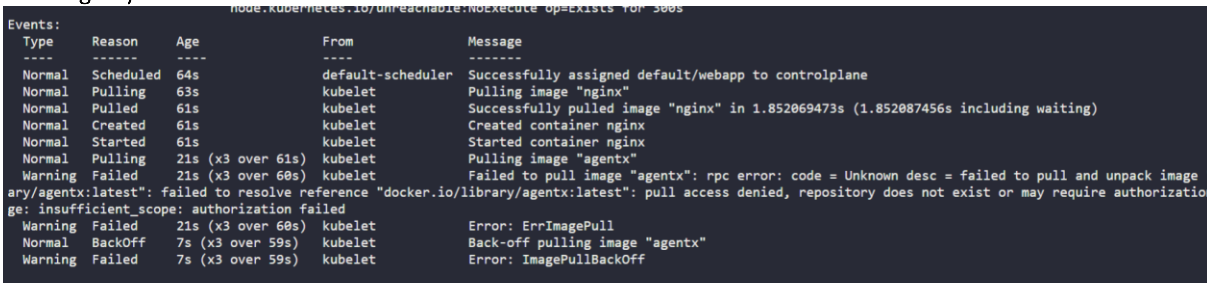
```
controlplane ~ ➜ kubectl describe pods webapp
Name:             webapp
Namespace:        default
Priority:         0
Service Account:  default
Node:             controlplane/172.25.0.68
Start Time:       Tue, 25 Apr 2023 03:45:01 +0000
Labels:           <none>
Annotations:      <none>
Status:           Pending
IP:               10.42.0.13
IPs:
  IP:  10.42.0.13
Containers:
  nginx:
    Container ID:   containerd://518ed25e679f624a1bc6ba0ae5865fff7efa
    Image:          nginx
    Image ID:       docker.io/library/nginx@sha256:63b44e8ddb83d5dd80
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Tue, 25 Apr 2023 03:45:04 +0000
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-acc
  agentx:
    Container ID:
    Image:          agentx
    Image ID:
    Port:           <none>
    Host Port:      <none>
    State:          Waiting
      Reason:       ErrImagePull
```

What is the state of the container agentx in the pod webapp? Error or waiting



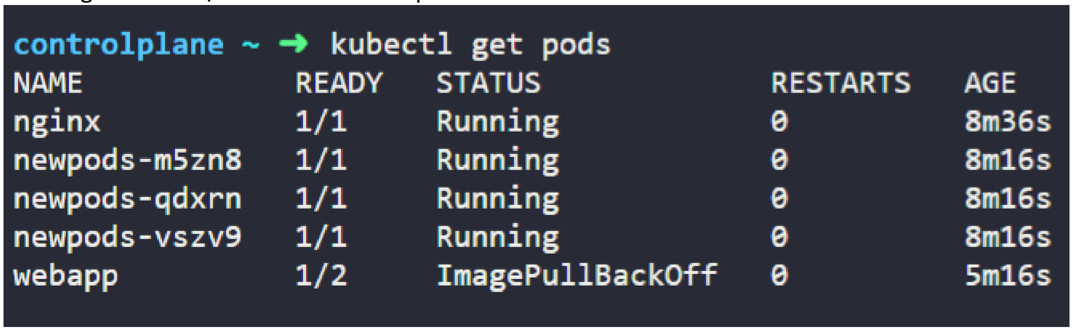
```
controlplane ~ ➜  kubectl describe pods webapp
Name:             webapp
Namespace:        default
Priority:         0
Service Account:  default
Node:             controlplane/172.25.0.68
Start Time:       Tue, 25 Apr 2023 03:45:01 +0000
Labels:           <none>
Annotations:      <none>
Status:           Pending
IP:               10.42.0.13
IPs:
  IP:  10.42.0.13
Containers:
  nginx:
    Container ID:   containerd://518ed25e679f624a1bc6ba0ae5865fff7efa
    Image:          nginx
    Image ID:       docker.io/library/nginx@sha256:63b44e8ddb83d5dd80
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Tue, 25 Apr 2023 03:45:04 +0000
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-acc
  agentx:
    Container ID:
    Image:          agentx
    Image ID:
    Port:           <none>
    Host Port:      <none>
    State:          Waiting
      Reason:       ErrImagePull
    Ready:          False
    Restart Count:  0
    Environment:    <none>
```

Why do you think the container agentx in pod webapp is in error? Docker image doesn't exist
Try to figure it out from the events section of the pod.
Run the command kubectl describe pod webapp and look under the events section.
An image by that name does not exist in DockerHub.



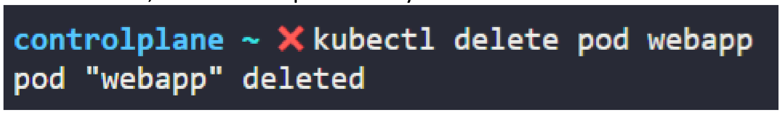
```
                              node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type     Reason     Age                From               Message
  ----     ------     ----               ----               -------
  Normal   Scheduled  64s                default-scheduler  Successfully assigned default/webapp to controlplane
  Normal   Pulling    63s                kubelet            Pulling image "nginx"
  Normal   Pulled     61s                kubelet            Successfully pulled image "nginx" in 1.852069473s (1.852087456s including waiting)
  Normal   Created    61s                kubelet            Created container nginx
  Normal   Started    61s                kubelet            Started container nginx
  Normal   Pulling    21s (x3 over 61s)  kubelet            Pulling image "agentx"
  Warning  Failed     21s (x3 over 60s)  kubelet            Failed to pull image "agentx": rpc error: code = Unknown desc = failed to pull and unpack image
ary/agentx:latest": failed to resolve reference "docker.io/library/agentx:latest": pull access denied, repository does not exist or may require authorizatio
ge: insufficient_scope: authorization failed
  Warning  Failed     21s (x3 over 60s)  kubelet            Error: ErrImagePull
  Normal   BackOff    7s (x3 over 59s)   kubelet            Back-off pulling image "agentx"
  Warning  Failed     7s (x3 over 59s)   kubelet            Error: ImagePullBackOff
```

What does the READY column in the output of the kubectl get pods command indicate?
Running container / total container in pod



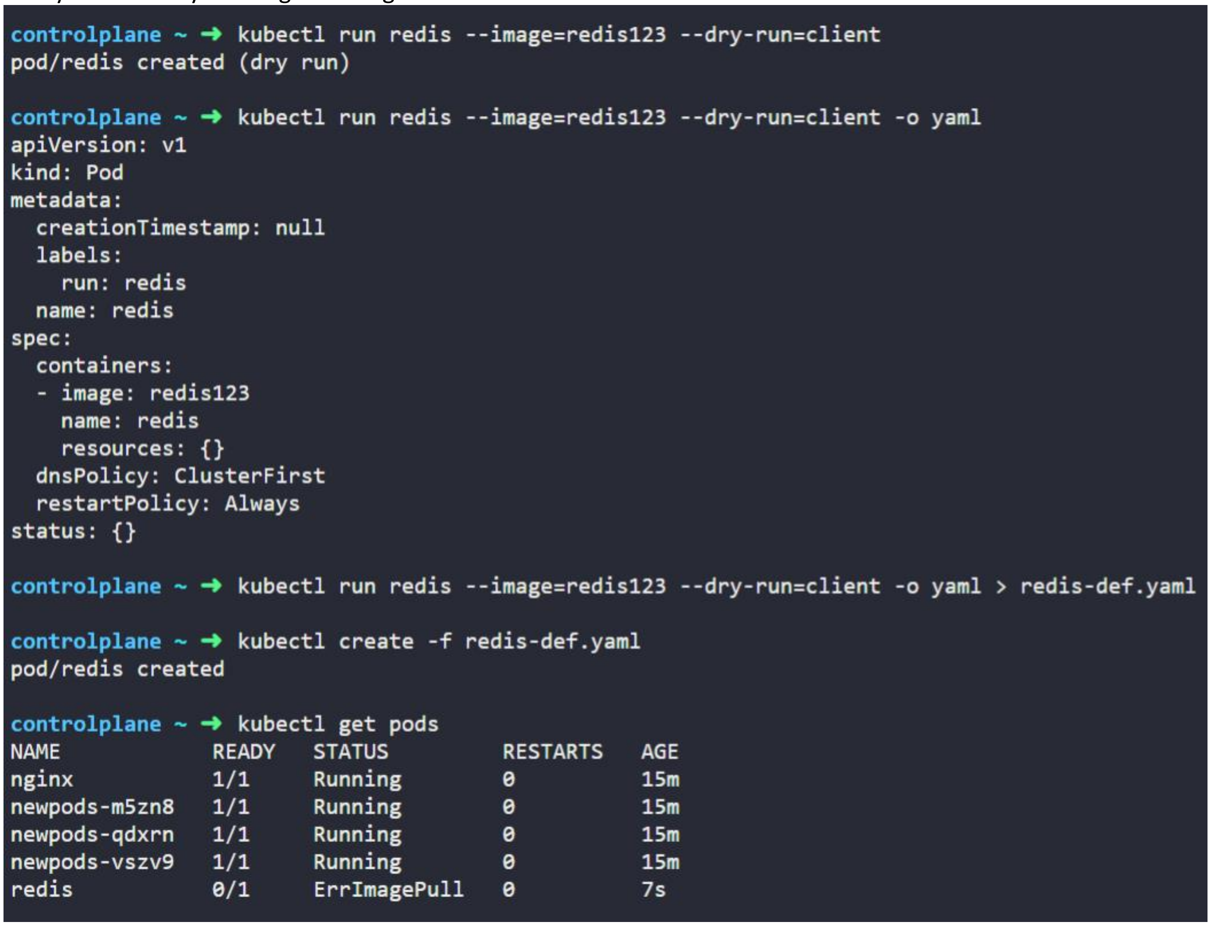
```
controlplane ~ ➜  kubectl get pods
NAME              READY   STATUS             RESTARTS   AGE
nginx             1/1     Running            0          8m36s
newpods-m5zn8     1/1     Running            0          8m16s
newpods-qdxrn     1/1     Running            0          8m16s
newpods-vszv9     1/1     Running            0          8m16s
webapp            1/2     ImagePullBackOff   0          5m16s
```

Delete the webapp Pod.
Once deleted, wait for the pod to fully terminate



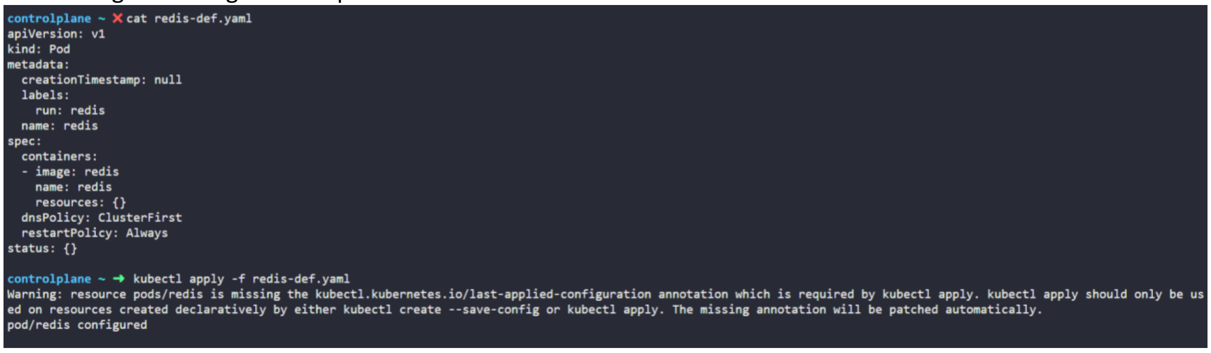
```
controlplane ~ ✖ kubectl delete pod webapp
pod "webapp" deleted
```

Create a new pod with the name redis and with the image redis123.
Use a pod-definition YAML file. And yes the image name is wrong!
We use kubectl run command with --dry-run=client -o yaml option to create a manifest file :-
After that, using kubectl create -f command to create a resource from the manifest file :-
Verify the work by running kubectl get command :-



```
controlplane ~ ➜  kubectl run redis --image=redis123 --dry-run=client
pod/redis created (dry run)

controlplane ~ ➜  kubectl run redis --image=redis123 --dry-run=client -o yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: redis
  name: redis
spec:
  containers:
  - image: redis123
    name: redis
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Always
status: {}

controlplane ~ ➜  kubectl run redis --image=redis123 --dry-run=client -o yaml > redis-def.yaml

controlplane ~ ➜  kubectl create -f redis-def.yaml
pod/redis created

controlplane ~ ➜  kubectl get pods
NAME             READY   STATUS         RESTARTS   AGE
nginx            1/1     Running        0          15m
newpods-m5zn8    1/1     Running        0          15m
newpods-qdxrn    1/1     Running        0          15m
newpods-vszv9    1/1     Running        0          15m
redis            0/1     ErrImagePull   0          7s
```

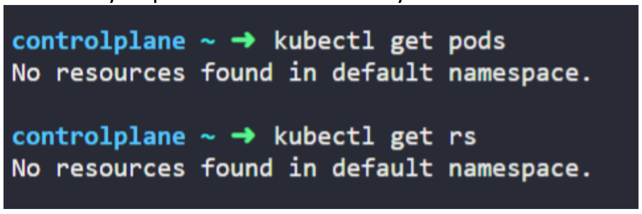
Now change the image on this pod to redis.



```
controlplane ~ ✗ cat redis-def.yaml
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: redis
  name: redis
spec:
  containers:
  - image: redis
    name: redis
    resources: {}
  dnsPolicy: ClusterFirst
  restartPolicy: Always
status: {}

controlplane ~ ➜ kubectl apply -f redis-def.yaml
Warning: resource pods/redis is missing the kubectl.kubernetes.io/last-applied-configuration annotation which is required by kubectl apply. kubectl apply should only be us
ed on resources created declaratively by either kubectl create --save-config or kubectl apply. The missing annotation will be patched automatically.
pod/redis configured
```

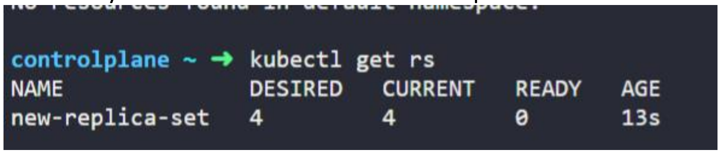
REPLICASETS

How many ReplicaSets exist on the system?



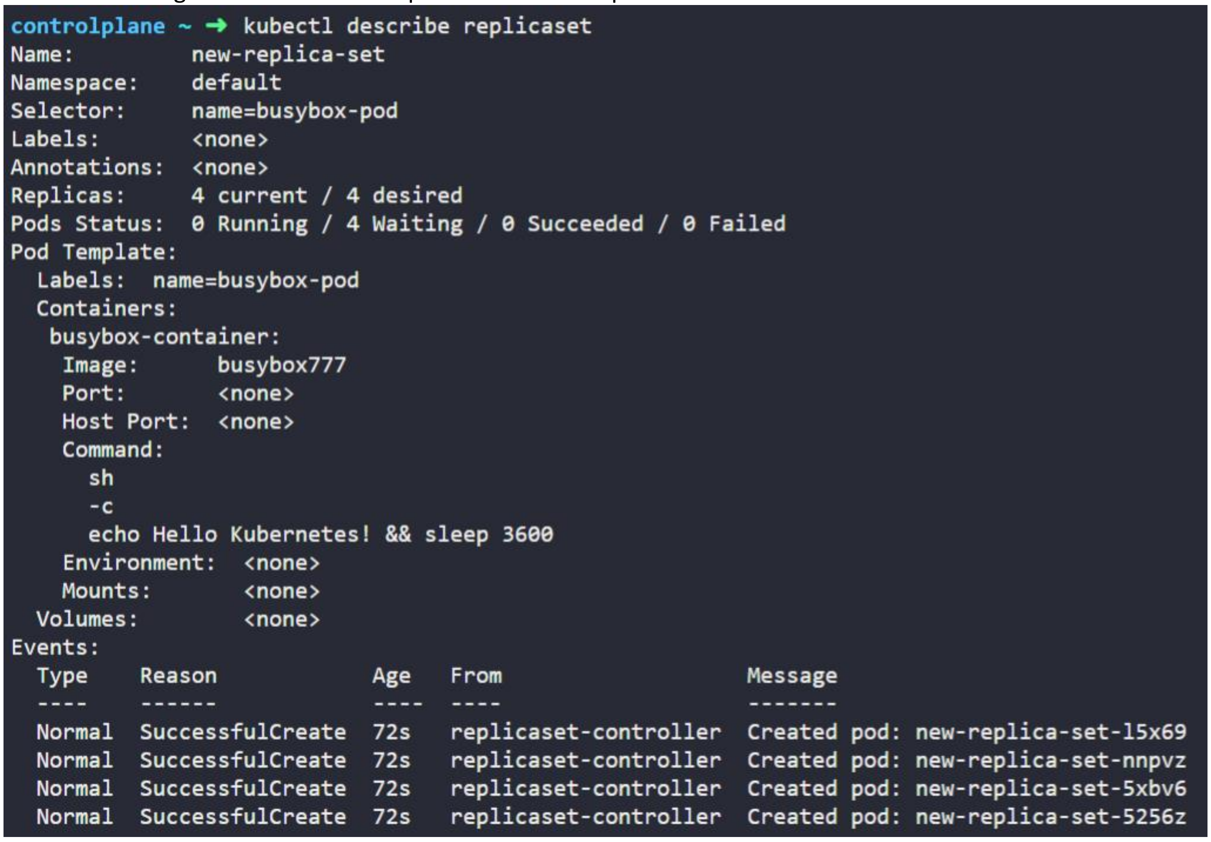
```
controlplane ~ ➜ kubectl get pods
No resources found in default namespace.

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

How many PODs are DESIRED in the new-replica-set?



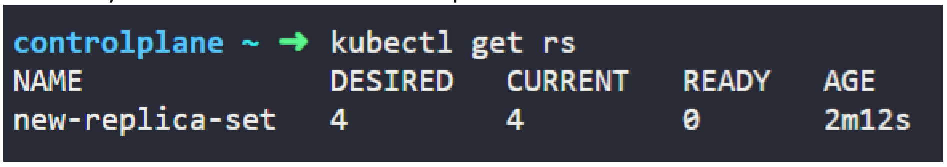
```
No resources found in default namespace.

controlplane ~ ➜ kubectl get rs
NAME              DESIRED   CURRENT   READY   AGE
new-replica-set   4         4         0       13s
```

What is the image used to create the pods in the new-replica-set?



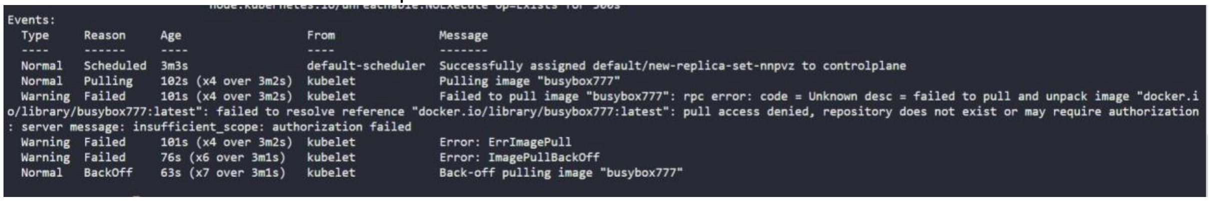
```
controlplane ~ ➜ kubectl describe replicaset
Name:         new-replica-set
Namespace:    default
Selector:     name=busybox-pod
Labels:       <none>
Annotations:  <none>
Replicas:     4 current / 4 desired
Pods Status:  0 Running / 4 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:  name=busybox-pod
  Containers:
   busybox-container:
    Image:      busybox777
    Port:       <none>
    Host Port:  <none>
    Command:
      sh
      -c
      echo Hello Kubernetes! && sleep 3600
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Events:
  Type    Reason            Age   From                   Message
  ----    ------            ----  ----                   -------
  Normal  SuccessfulCreate  72s   replicaset-controller  Created pod: new-replica-set-l5x69
  Normal  SuccessfulCreate  72s   replicaset-controller  Created pod: new-replica-set-nnpvz
  Normal  SuccessfulCreate  72s   replicaset-controller  Created pod: new-replica-set-5xbv6
  Normal  SuccessfulCreate  72s   replicaset-controller  Created pod: new-replica-set-5256z
```

How many PODs are READY in the new-replica-set?



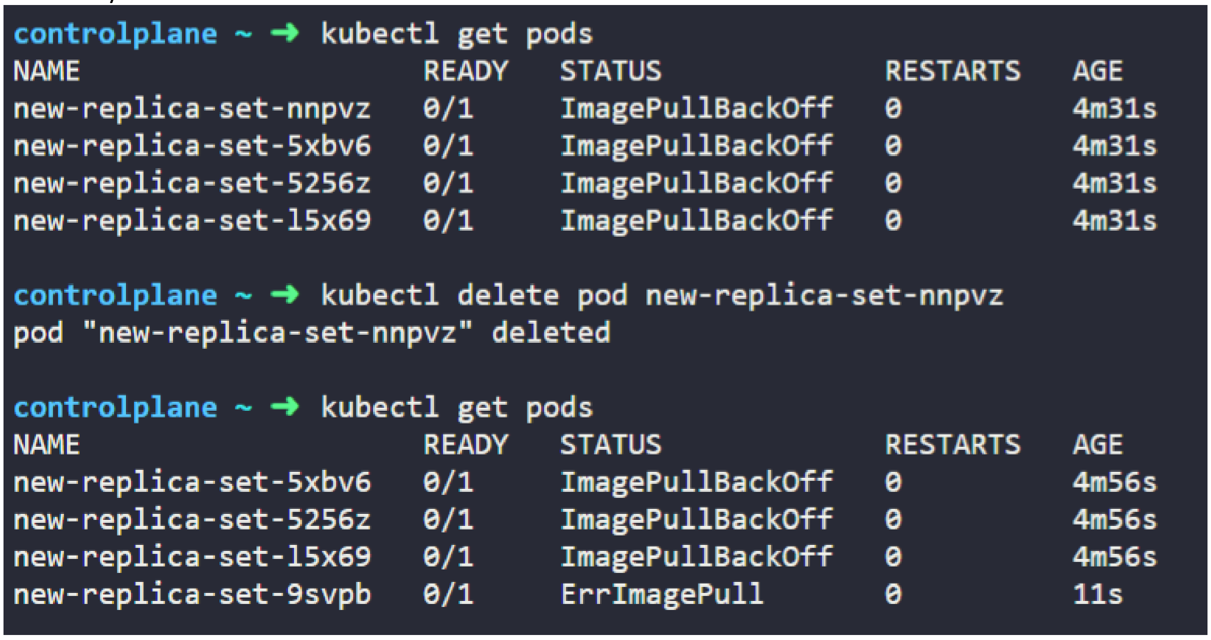
```
controlplane ~ ➜ kubectl get rs
NAME              DESIRED   CURRENT   READY   AGE
new-replica-set   4         4         0       2m12s
```

Why do you think the PODs are not ready? Image not exist
Run the command: kubectl describe pods and look at under the Events section.



```
Events:
  Type     Reason     Age                 From               Message
  ----     ------     ----                ----               -------
  Normal   Scheduled  3m3s                default-scheduler  Successfully assigned default/new-replica-set-nnpvz to controlplane
  Normal   Pulling    102s (x4 over 3m2s)  kubelet            Pulling image "busybox777"
  Warning  Failed     101s (x4 over 3m2s)  kubelet            Failed to pull image "busybox777": rpc error: code = Unknown desc = failed to pull and unpack image "docker.i
o/library/busybox777:latest": failed to resolve reference "docker.io/library/busybox777:latest": pull access denied, repository does not exist or may require authorization
: server message: insufficient_scope: authorization failed
  Warning  Failed     101s (x4 over 3m2s)  kubelet            Error: ErrImagePull
  Warning  Failed     76s (x6 over 3m1s)   kubelet            Error: ImagePullBackOff
  Normal   BackOff    63s (x7 over 3m1s)   kubelet            Back-off pulling image "busybox777"
```

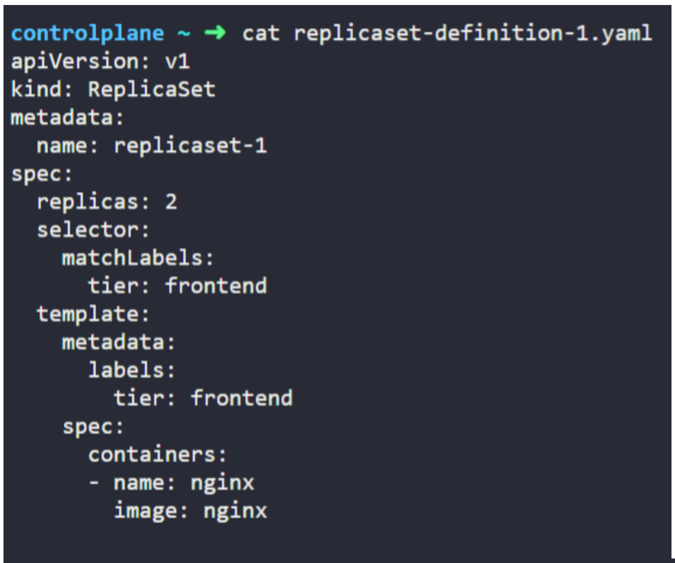
Delete any one of the 4 PODs.
How many PODs exist now?



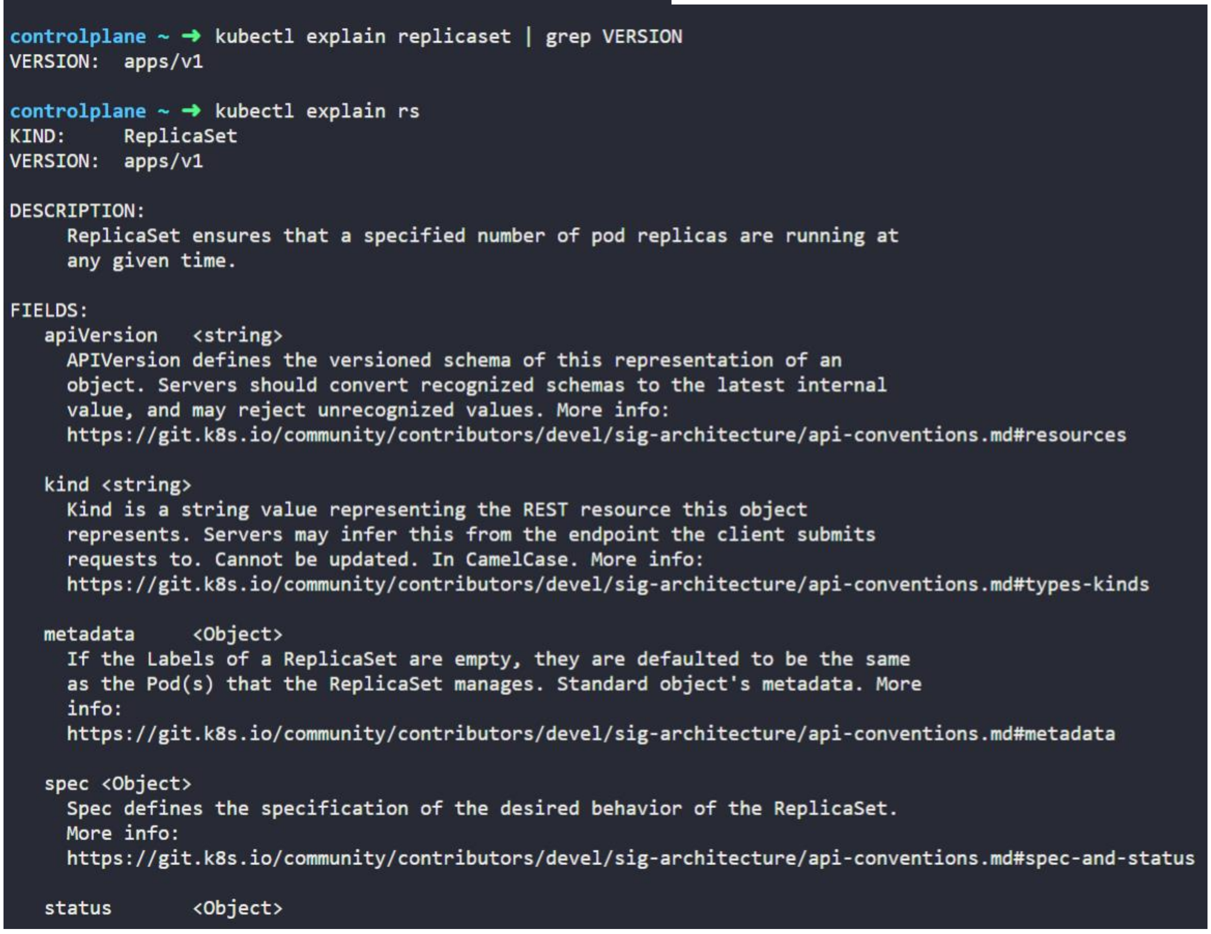
```
controlplane ~ ➜ kubectl get pods
NAME                    READY   STATUS             RESTARTS   AGE
new-replica-set-nnpvz   0/1     ImagePullBackOff   0          4m31s
new-replica-set-5xbv6   0/1     ImagePullBackOff   0          4m31s
new-replica-set-5256z   0/1     ImagePullBackOff   0          4m31s
new-replica-set-l5x69   0/1     ImagePullBackOff   0          4m31s

controlplane ~ ➜ kubectl delete pod new-replica-set-nnpvz
pod "new-replica-set-nnpvz" deleted

controlplane ~ ➜ kubectl get pods
NAME                    READY   STATUS             RESTARTS   AGE
new-replica-set-5xbv6   0/1     ImagePullBackOff   0          4m56s
new-replica-set-5256z   0/1     ImagePullBackOff   0          4m56s
new-replica-set-l5x69   0/1     ImagePullBackOff   0          4m56s
new-replica-set-9svpb   0/1     ErrImagePull       0          11s
```

Why are there still 4 PODs, even after you deleted one?
Replicaset ensures desired number of pods always run .

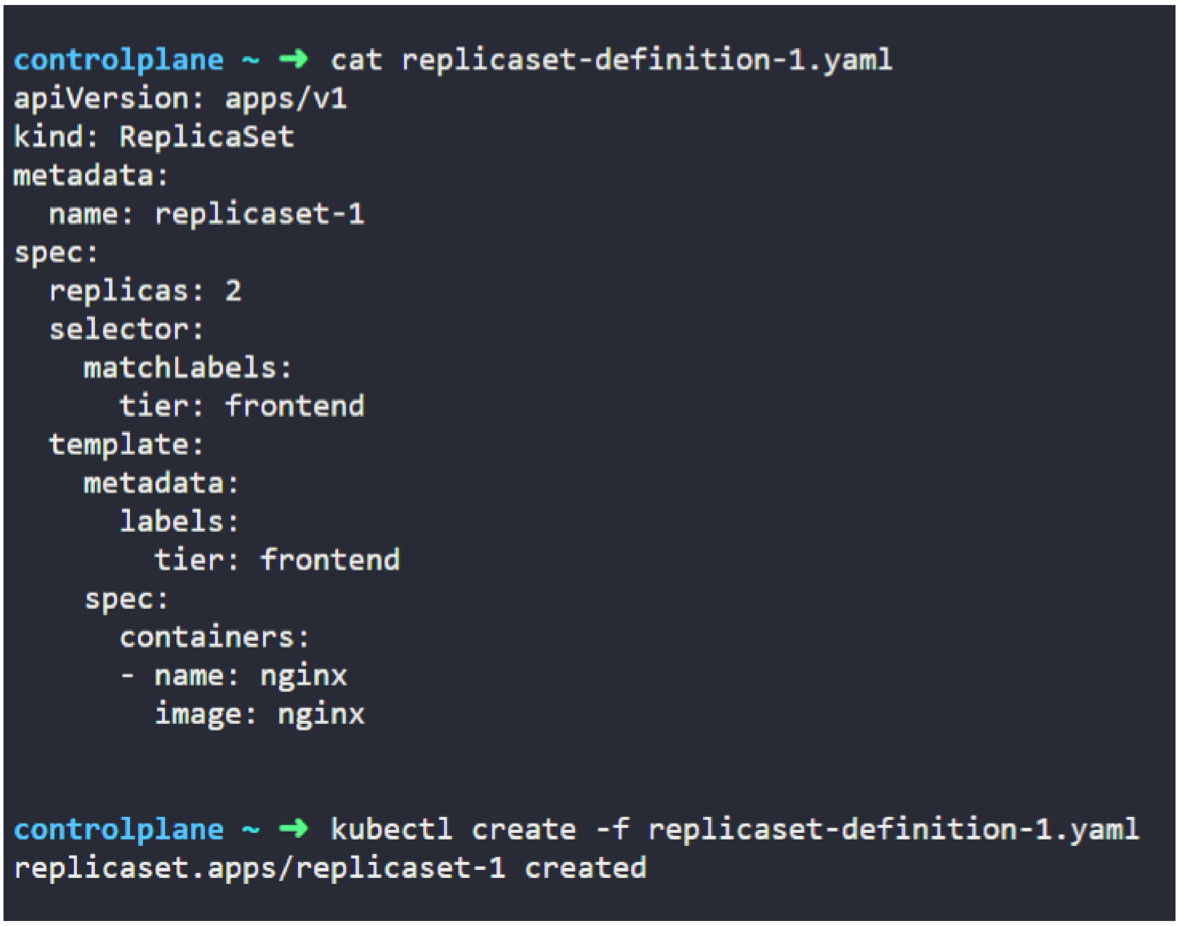
Create a ReplicaSet using the replicaset-definition-1.yaml file located at /root/.
There is an issue with the file, so try to fix it.
Run the command: kubectl explain replicaset | grep VERSION and correct the apiVersion for ReplicaSet.
Then run the command: kubectl create -f /root/replicaset-definition-1.yaml



```
controlplane ~ ➜ cat replicaset-definition-1.yaml
apiVersion: v1
kind: ReplicaSet
metadata:
  name: replicaset-1
spec:
  replicas: 2
  selector:
    matchLabels:
      tier: frontend
  template:
    metadata:
      labels:
        tier: frontend
    spec:
      containers:
      - name: nginx
        image: nginx
```

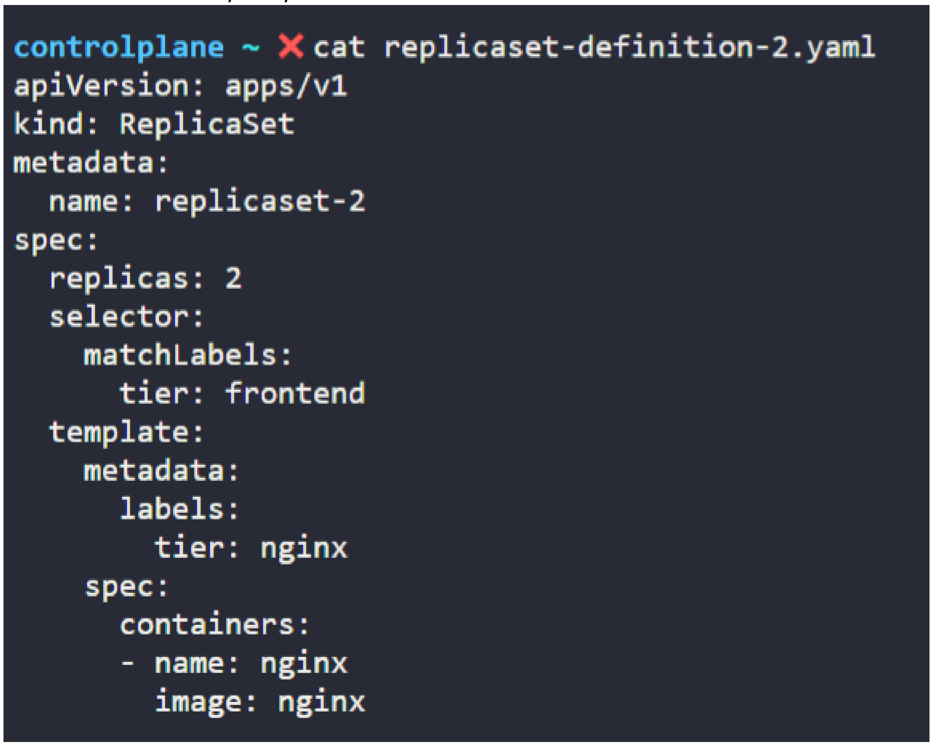


```
controlplane ~ ➜ kubectl explain replicaset | grep VERSION
VERSION:  apps/v1

controlplane ~ ➜ kubectl explain rs
KIND:     ReplicaSet
VERSION:  apps/v1

DESCRIPTION:
     ReplicaSet ensures that a specified number of pod replicas are running at
     any given time.

FIELDS:
   apiVersion   <string>
     APIVersion defines the versioned schema of this representation of an
     object. Servers should convert recognized schemas to the latest internal
     value, and may reject unrecognized values. More info:
     https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources

   kind <string>
     Kind is a string value representing the REST resource this object
     represents. Servers may infer this from the endpoint the client submits
     requests to. Cannot be updated. In CamelCase. More info:
     https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds

   metadata     <Object>
     If the Labels of a ReplicaSet are empty, they are defaulted to be the same
     as the Pod(s) that the ReplicaSet manages. Standard object's metadata. More
     info:
     https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

   spec <Object>
     Spec defines the specification of the desired behavior of the ReplicaSet.
     More info:
     https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#spec-and-status

   status       <Object>
```

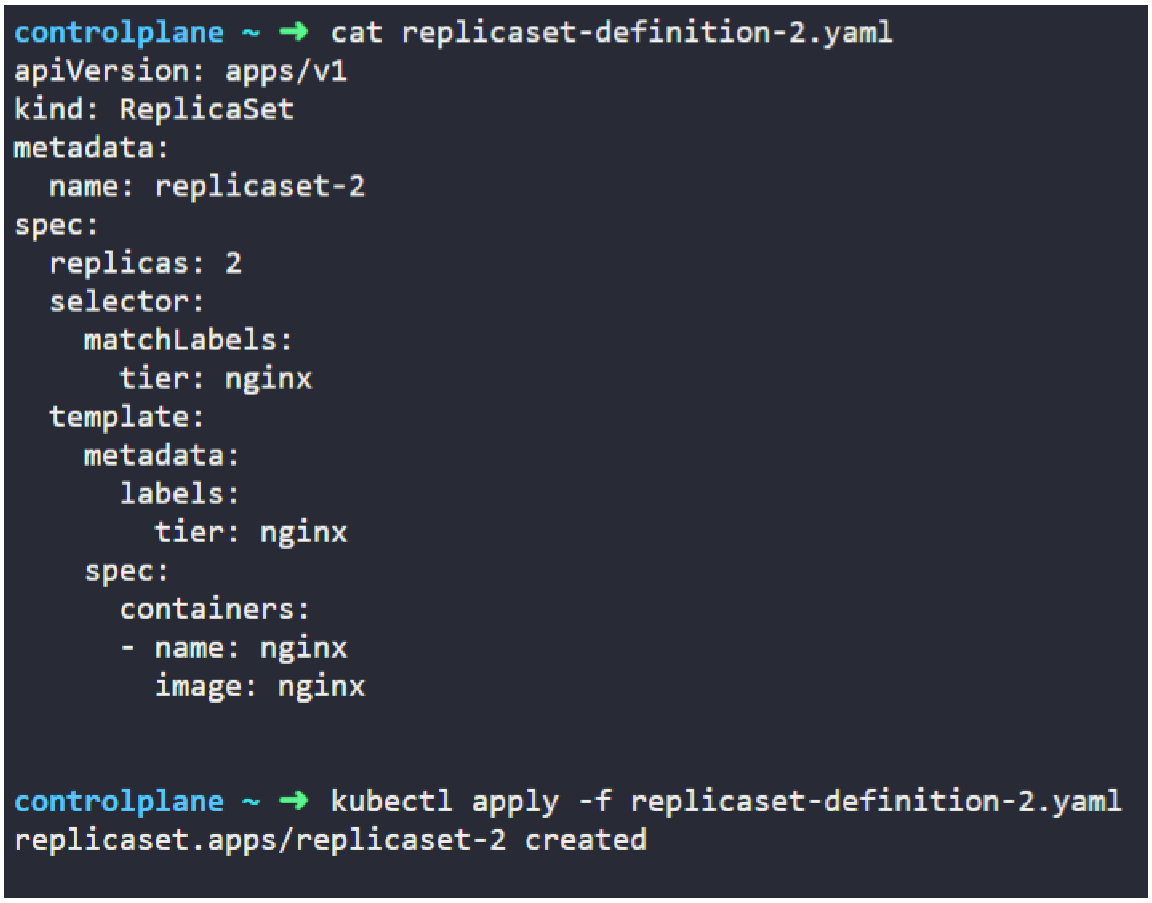


```
controlplane ~ ➜ cat replicaset-definition-1.yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: replicaset-1
spec:
  replicas: 2
  selector:
    matchLabels:
      tier: frontend
  template:
    metadata:
      labels:
        tier: frontend
    spec:
      containers:
      - name: nginx
        image: nginx


controlplane ~ ➜ kubectl create -f replicaset-definition-1.yaml
replicaset.apps/replicaset-1 created
```

Fix the issue in the replicaset-definition-2.yaml file and create a ReplicaSet using it.
This file is located at /root/.

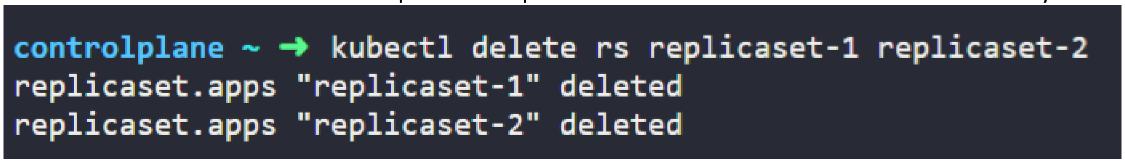


```
controlplane ~ ✗ cat replicaset-definition-2.yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: replicaset-2
spec:
  replicas: 2
  selector:
    matchLabels:
      tier: frontend
  template:
    metadata:
      labels:
        tier: nginx
    spec:
      containers:
      - name: nginx
        image: nginx
```



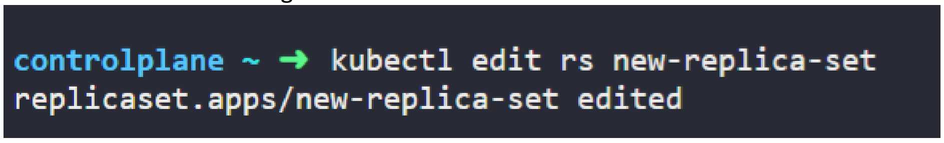
```
controlplane ~ ➜ cat replicaset-definition-2.yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: replicaset-2
spec:
  replicas: 2
  selector:
    matchLabels:
      tier: nginx
  template:
    metadata:
      labels:
        tier: nginx
    spec:
      containers:
      - name: nginx
        image: nginx


controlplane ~ ➜ kubectl apply -f replicaset-definition-2.yaml
replicaset.apps/replicaset-2 created
```

Delete the two newly created ReplicaSets - replicaset-1 and replicaset-2
Run the command: kubectl delete replicaset <replicaset-name> or kubectl delete -f <file-name>.yaml

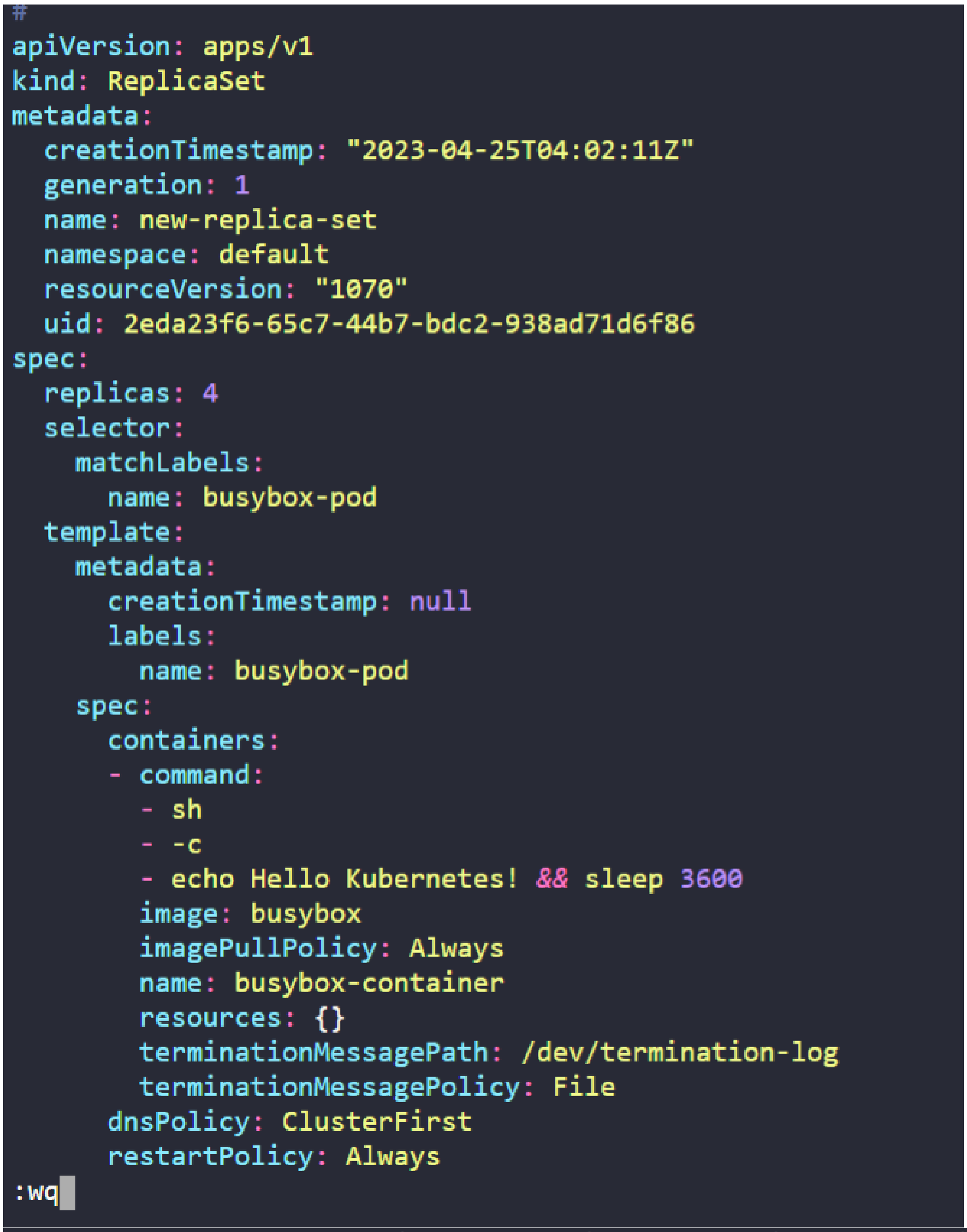


```
controlplane ~ ➜ kubectl delete rs replicaset-1 replicaset-2
replicaset.apps "replicaset-1" deleted
replicaset.apps "replicaset-2" deleted
```

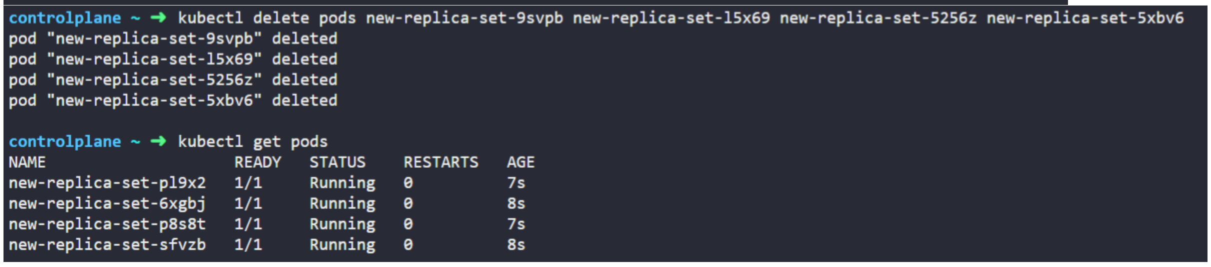
Fix the original replica set new-replica-set to use the correct busybox image.
Either delete and recreate the ReplicaSet or Update the existing ReplicaSet and then delete all PODs, so new ones with the correct image will be created.



```
controlplane ~ ➜ kubectl edit rs new-replica-set
replicaset.apps/new-replica-set edited
```

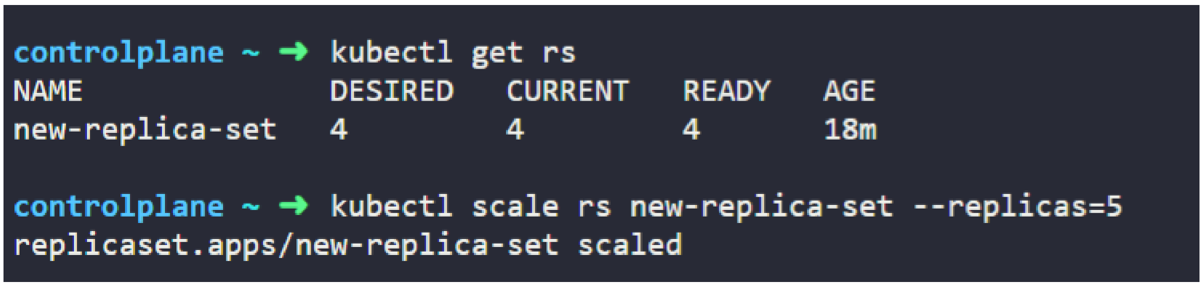


```
#
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  creationTimestamp: "2023-04-25T04:02:11Z"
  generation: 1
  name: new-replica-set
  namespace: default
  resourceVersion: "1070"
  uid: 2eda23f6-65c7-44b7-bdc2-938ad71d6f86
spec:
  replicas: 4
  selector:
    matchLabels:
      name: busybox-pod
  template:
    metadata:
      creationTimestamp: null
      labels:
        name: busybox-pod
    spec:
      containers:
      - command:
        - sh
        - -c
        - echo Hello Kubernetes! && sleep 3600
        image: busybox
        imagePullPolicy: Always
        name: busybox-container
        resources: {}
        terminationMessagePath: /dev/termination-log
        terminationMessagePolicy: File
      dnsPolicy: ClusterFirst
      restartPolicy: Always
:wq
```



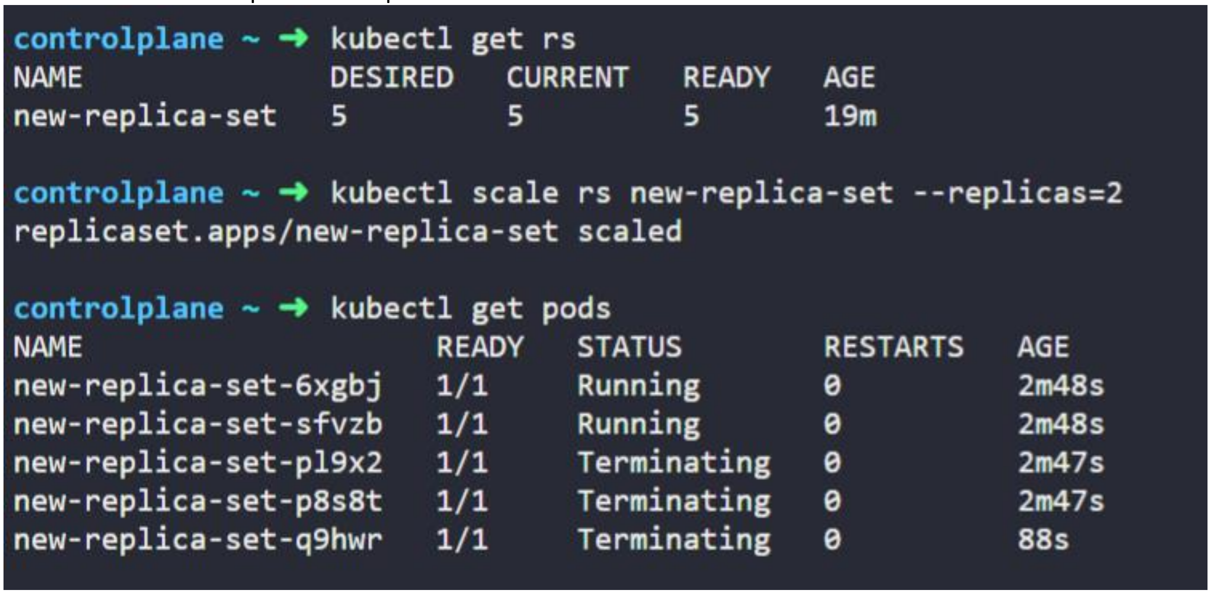
```
controlplane ~ ➜ kubectl delete pods new-replica-set-9svpb new-replica-set-l5x69 new-replica-set-5256z new-replica-set-5xbv6
pod "new-replica-set-9svpb" deleted
pod "new-replica-set-l5x69" deleted
pod "new-replica-set-5256z" deleted
pod "new-replica-set-5xbv6" deleted

controlplane ~ ➜ kubectl get pods
NAME                    READY   STATUS    RESTARTS   AGE
new-replica-set-pl9x2   1/1     Running   0          7s
new-replica-set-6xgbj   1/1     Running   0          8s
new-replica-set-p8s8t   1/1     Running   0          7s
new-replica-set-sfvzb   1/1     Running   0          8s
```

Scale the ReplicaSet to 5 PODs.
Use kubectl scale command or edit the replicaset using kubectl edit replicaset.
Run the command: kubectl edit replicaset new-replica-set, modify the replicas and then save the file OR run: kubectl scale rs new-replica-set --replicas=5 to scale up to 5 PODs.

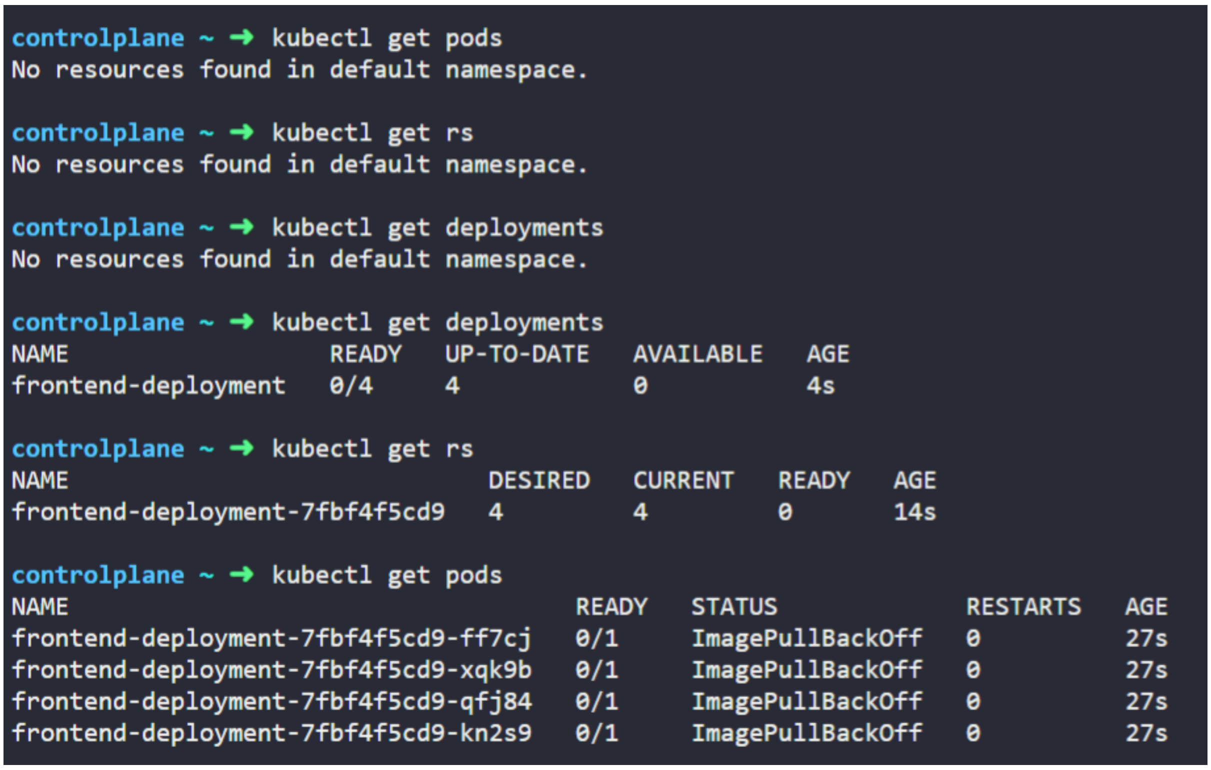


```
controlplane ~ ➜  kubectl get rs
NAME              DESIRED   CURRENT   READY   AGE
new-replica-set   4         4         4       18m

controlplane ~ ➜  kubectl scale rs new-replica-set --replicas=5
replicaset.apps/new-replica-set scaled
```

Now scale the ReplicaSet down to 2 PODs.
Use the kubectl scale command or edit the replicaset using kubectl edit replicaset.
Run the command: kubectl edit replicaset new-replica-set, modify the replicas and then save the file OR run: kubectl scale rs new-replica-set --replicas=2 to scale down to 2 PODs.

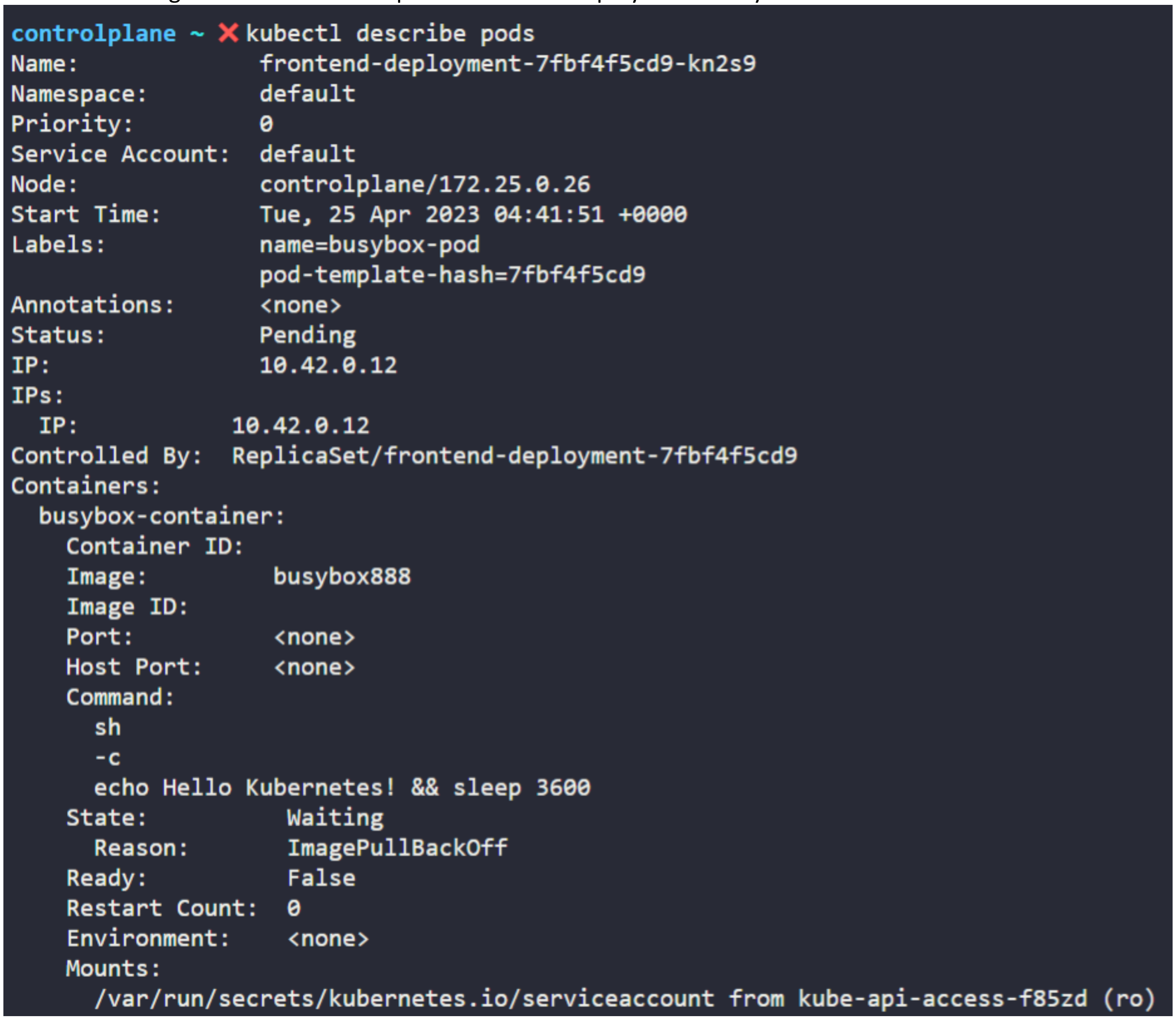


```
controlplane ~ ➜  kubectl get rs
NAME              DESIRED   CURRENT   READY   AGE
new-replica-set   5         5         5       19m

controlplane ~ ➜  kubectl scale rs new-replica-set --replicas=2
replicaset.apps/new-replica-set scaled

controlplane ~ ➜  kubectl get pods
NAME                    READY   STATUS        RESTARTS   AGE
new-replica-set-6xgbj   1/1     Running       0          2m48s
new-replica-set-sfvzb   1/1     Running       0          2m48s
new-replica-set-pl9x2   1/1     Terminating   0          2m47s
new-replica-set-p8s8t   1/1     Terminating   0          2m47s
new-replica-set-q9hwr   1/1     Terminating   0          88s
```



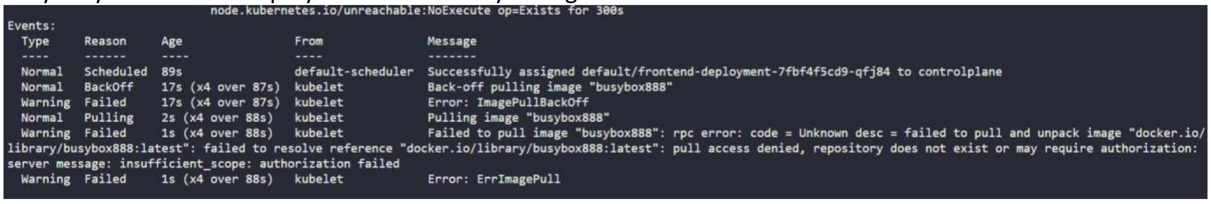
```
controlplane ~ ➜  kubectl get pods
No resources found in default namespace.

controlplane ~ ➜  kubectl get rs
No resources found in default namespace.

controlplane ~ ➜  kubectl get deployments
No resources found in default namespace.

controlplane ~ ➜  kubectl get deployments
NAME                  READY   UP-TO-DATE   AVAILABLE   AGE
frontend-deployment   0/4     4            0           4s

controlplane ~ ➜  kubectl get rs
NAME                             DESIRED   CURRENT   READY   AGE
frontend-deployment-7fbf4f5cd9   4         4         0       14s

controlplane ~ ➜  kubectl get pods
NAME                                   READY   STATUS             RESTARTS   AGE
frontend-deployment-7fbf4f5cd9-ff7cj   0/1     ImagePullBackOff   0          27s
frontend-deployment-7fbf4f5cd9-xqk9b   0/1     ImagePullBackOff   0          27s
frontend-deployment-7fbf4f5cd9-qfj84   0/1     ImagePullBackOff   0          27s
frontend-deployment-7fbf4f5cd9-kn2s9   0/1     ImagePullBackOff   0          27s
```

What is the image used to create the pods in the new deployment? Busybox888



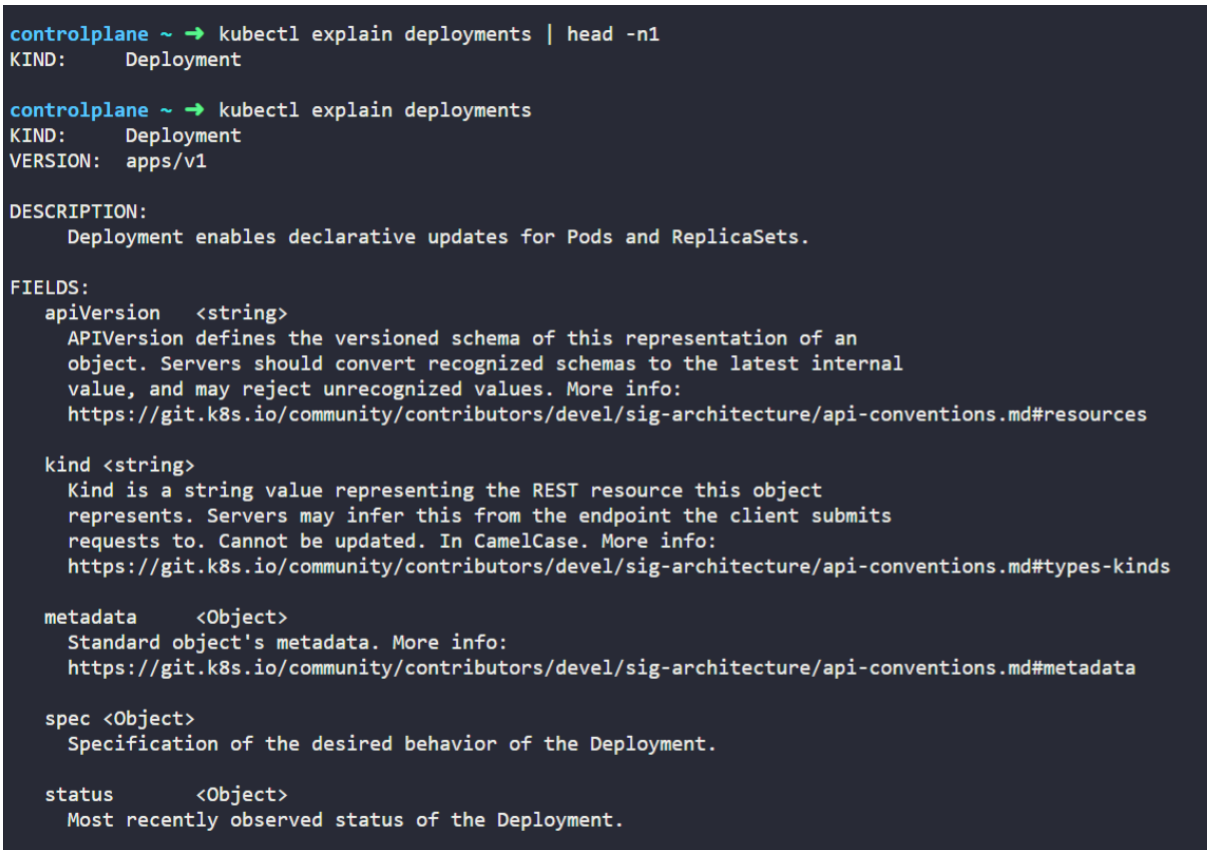
```
controlplane ~ ✗ kubectl describe pods
Name:             frontend-deployment-7fbf4f5cd9-kn2s9
Namespace:        default
Priority:         0
Service Account:  default
Node:             controlplane/172.25.0.26
Start Time:       Tue, 25 Apr 2023 04:41:51 +0000
Labels:           name=busybox-pod
                  pod-template-hash=7fbf4f5cd9
Annotations:      <none>
Status:           Pending
IP:               10.42.0.12
IPs:
  IP:           10.42.0.12
Controlled By:  ReplicaSet/frontend-deployment-7fbf4f5cd9
Containers:
  busybox-container:
    Container ID:
    Image:          busybox888
    Image ID:
    Port:           <none>
    Host Port:      <none>
    Command:
      sh
      -c
      echo Hello Kubernetes! && sleep 3600
    State:          Waiting
      Reason:       ImagePullBackOff
    Ready:          False
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-f85zd (ro)
```

Why do you think the deployment is not ready? Image not exist



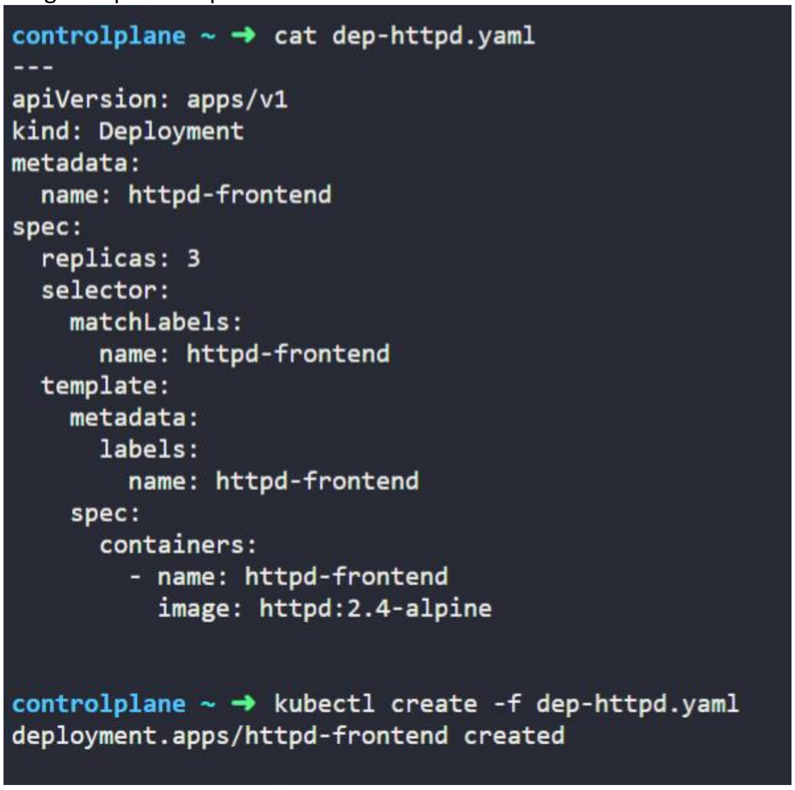
```
                             node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type     Reason     Age                From               Message
  ----     ------     ----               ----               -------
  Normal   Scheduled  89s                default-scheduler  Successfully assigned default/frontend-deployment-7fbf4f5cd9-qfj84 to controlplane
  Normal   BackOff    17s (x4 over 87s)  kubelet            Back-off pulling image "busybox888"
  Warning  Failed     17s (x4 over 87s)  kubelet            Error: ImagePullBackOff
  Normal   Pulling    2s (x4 over 88s)   kubelet            Pulling image "busybox888"
  Warning  Failed     1s (x4 over 88s)   kubelet            Failed to pull image "busybox888": rpc error: code = Unknown desc = failed to pull and unpack image "docker.io/
library/busybox888:latest": failed to resolve reference "docker.io/library/busybox888:latest": pull access denied, repository does not exist or may require authorization:
server message: insufficient_scope: authorization failed
  Warning  Failed     1s (x4 over 88s)   kubelet            Error: ErrImagePull
```

Create a new Deployment using the deployment-definition-1.yaml file located at /root/.
There is an issue with the file, so try to fix it.



```
controlplane ~ ➜ kubectl explain deployments | head -n1
KIND:     Deployment

controlplane ~ ➜ kubectl explain deployments
KIND:     Deployment
VERSION:  apps/v1

DESCRIPTION:
     Deployment enables declarative updates for Pods and ReplicaSets.

FIELDS:
   apiVersion   <string>
     APIVersion defines the versioned schema of this representation of an
     object. Servers should convert recognized schemas to the latest internal
     value, and may reject unrecognized values. More info:
     https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources

   kind <string>
     Kind is a string value representing the REST resource this object
     represents. Servers may infer this from the endpoint the client submits
     requests to. Cannot be updated. In CamelCase. More info:
     https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#types-kinds

   metadata     <Object>
     Standard object's metadata. More info:
     https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#metadata

   spec <Object>
     Specification of the desired behavior of the Deployment.

   status       <Object>
     Most recently observed status of the Deployment.
```

Create a new Deployment with the below attributes using your own deployment definition file.
Name: httpd-frontend;
Replicas: 3;
Image: httpd:2.4-alpine



```
controlplane ~ ➜ cat dep-httpd.yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: httpd-frontend
spec:
  replicas: 3
  selector:
    matchLabels:
      name: httpd-frontend
  template:
    metadata:
      labels:
        name: httpd-frontend
    spec:
      containers:
        - name: httpd-frontend
          image: httpd:2.4-alpine


controlplane ~ ➜ kubectl create -f dep-httpd.yaml
deployment.apps/httpd-frontend created
```