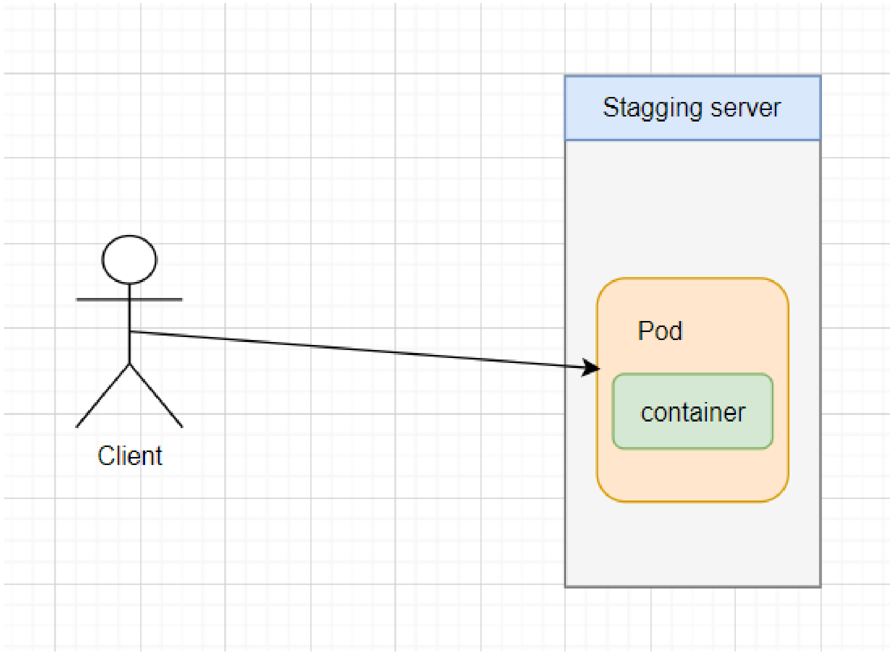
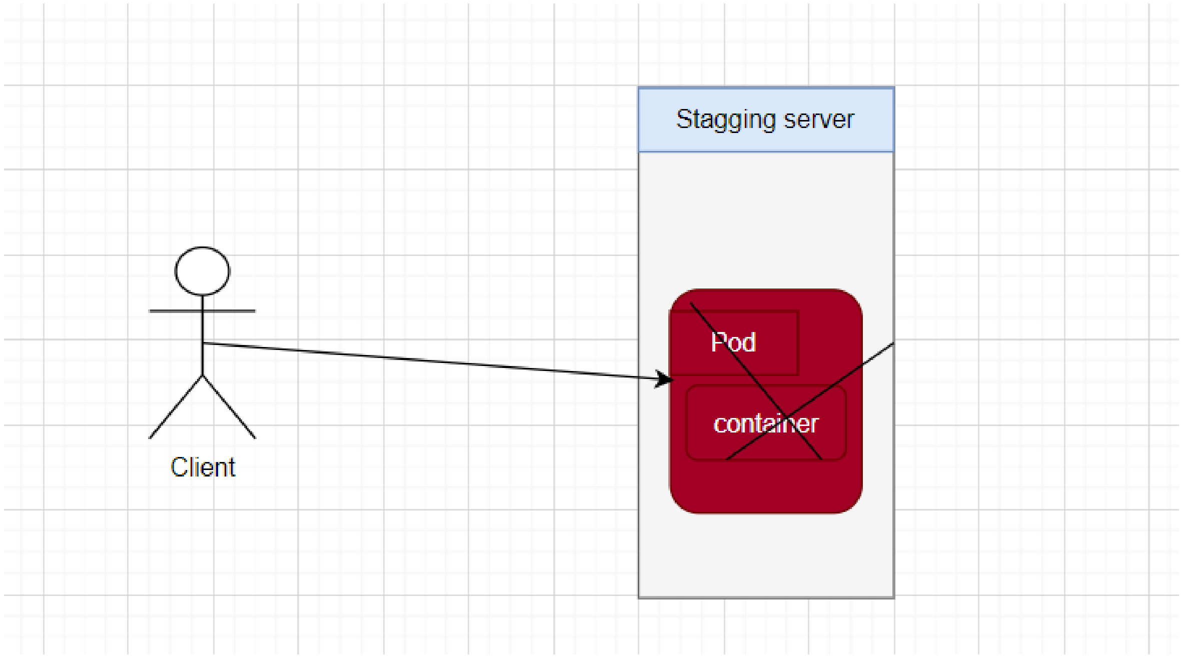
We have client trying to access application that encapsulated on pod.

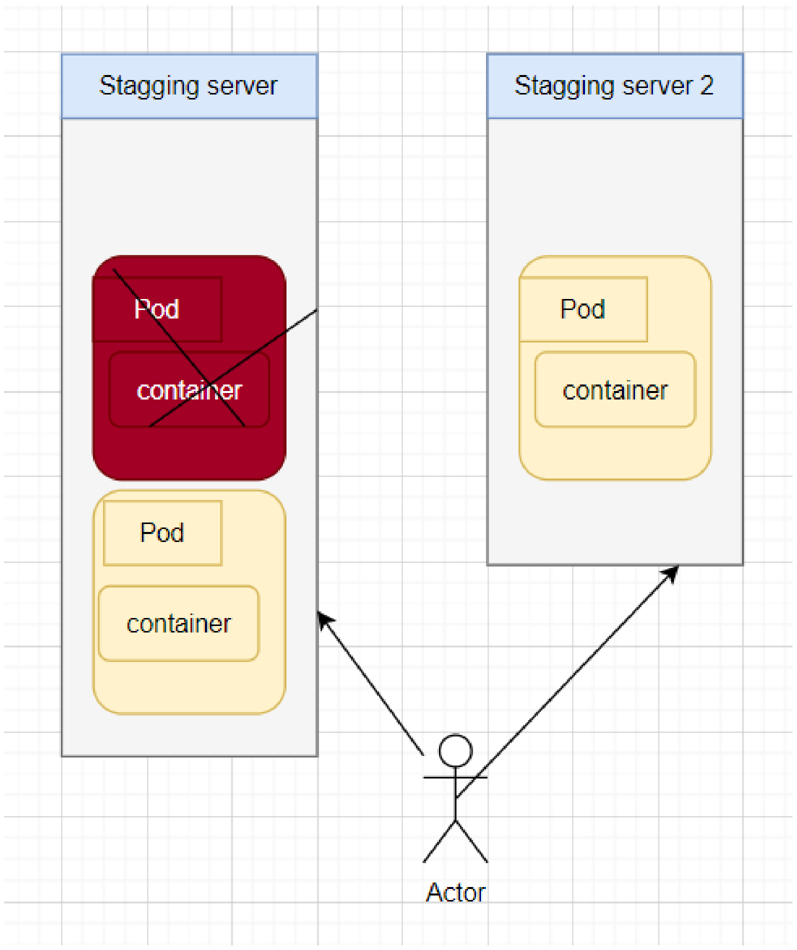


What happed if application Crashed and not been accessiable ?



So we cannot access our application.

Is this Legal to each time check app is up && running or crashed even if we have deploy multilpe instance of same app on differenent nodes ? No.



We need some component to mange this process. -----------> Replication Controller.

Replicatiion Controller ----------

k8s object used to create multiple instances of single pod and ensure that desired number of pods is running all time

 so if there is any crash or pod deleted it will starts an~~d~~ pod with same specification. -> (achive high availability).

It can span over multiple nodes.

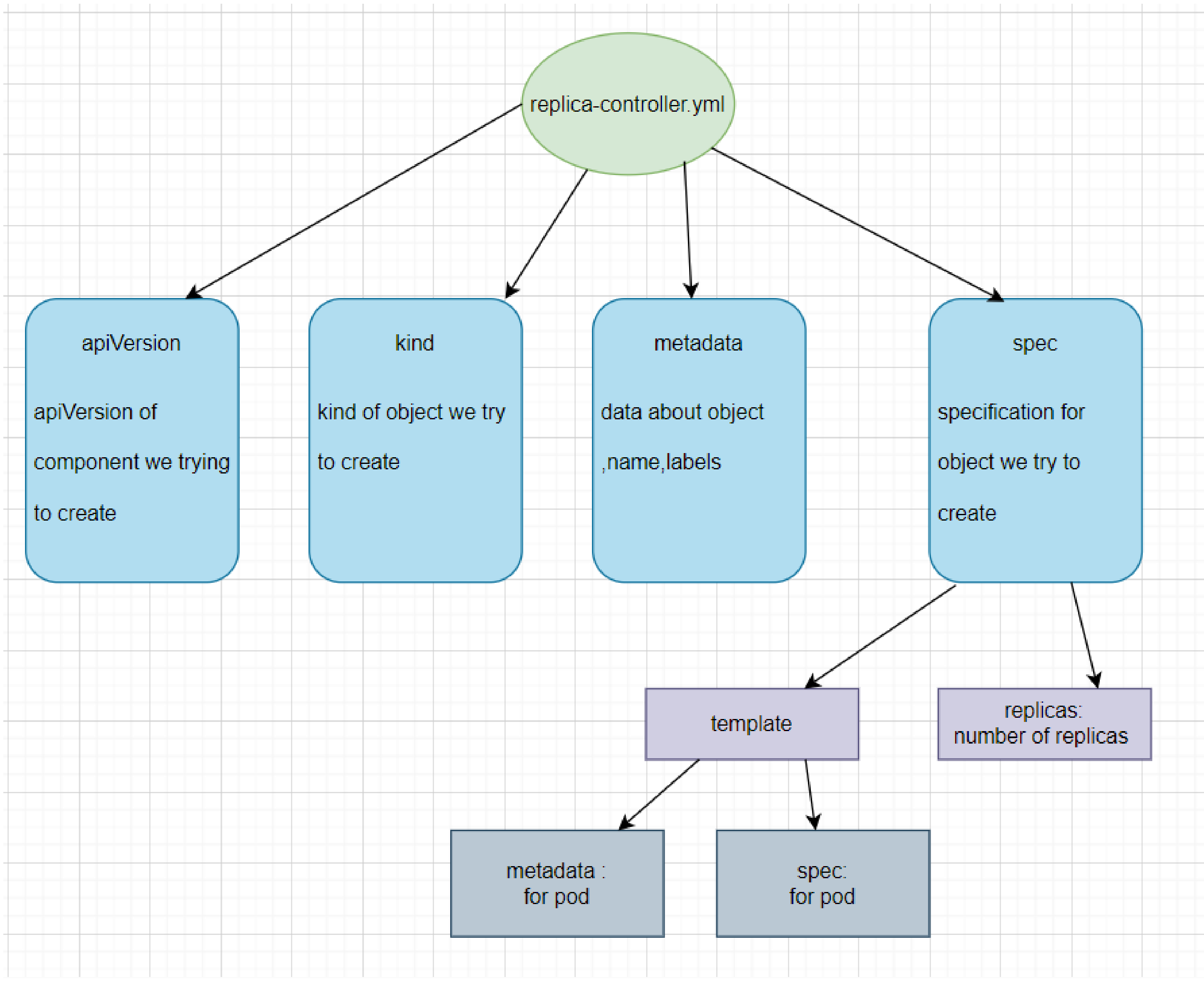
yaml file for replication controller consist of 4 mandatory sections {

apiVersion ------------------> api_version of object we try to create,

 kind ------------------> kind of object we try to create ,

metadata ----------------> data about object name and labels for this object ,

spec --------->specification for object we create consist of 1)templae for pod 2)replicas number of replicas of this pod}



replica-controller.yml
apiVersion
apiVersion of component we trying to create
kind
kind of object we try to create
metadata
data about object ,name,labels
spec
specification for object we try to create
template
replicas: number of replicas
metadata : for pod
spec: for pod

```
vi  myReplica.yml
apiVersion:  v1
Kind: ReplicationController
```

```
metadata:
    name: replicat-controller-1
spec:
    template:
            #metadata && spec section on pod
definition
            metadata:
                 name: webapp
                 labels:
                    app: web
            spec:
                 containers:
                     - name: container_name
                       image: image_name
             # this metadata and spec related to pod
definition.

    replicas: num_of_replicas
```

```
:wq
```

So how to create Replication controller from yaml file.

```
$ kubectl create -f  file_name
```

How to list replicationcontroller ?

```
$ kubectl get replicationcontrollers
```

Another way to create multiple instances of pod is replicaset .

Replicaset -> newest , require selectors(mandatory).

Selector used for restrict which pods that replicaset will manage.

Replicaset can manages pods that not created as part of replicaset creation.by it has labels that match selector on replicset.

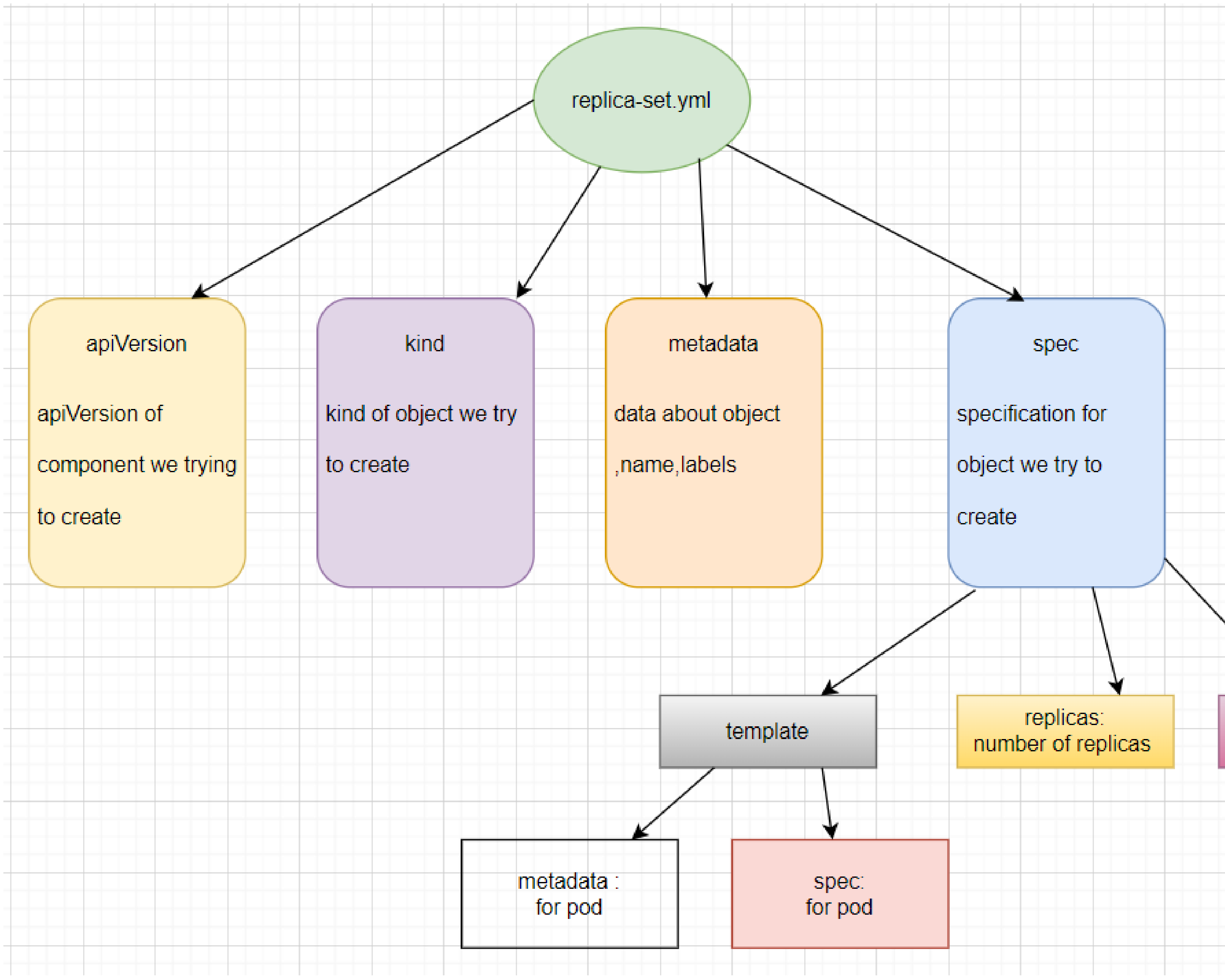
This is selector on replicaset definition.

```
selector:
    matchLabels:
          label1: zxd
```

and this pod created before replicaset had been created but has labels matches to labels defined on selector on replicaset so replicaset will manage this pod. --> need for review

```
metadata:
    name: pod1
     labels:
          label1: zxd
```

for replicationcontroller selector is optional but if not specified is assume as labels on pod definition.


replica-set.yml
apiVersion
apiVersion of component we trying to create
kind
kind of object we try to create
metadata
data about object ,name,labels
spec
specification for object we try to create
template
replicas: number of replicas
metadata : for pod
spec: for pod

```
vi  myReplicaSet.yml
apiVersion:  apps/v1
Kind: ReplicationSet
```

```
metadata:
    name: replicat-set-1
spec:
    template:
            #metadata && spec section on pod
definition
            metadata:
                name: webapp
                labels:
                  app: web
            spec:
                containers:
                   - name: container_name
                     image: image_name
             # this metadata and spec related to pod
definition.

    replicas: num_of_replicas
    selector:
            matchLabels:

                    label: value
:wq
```

If we have 3 running pods that have labels match selector identified on Replicaset && Replication Controller what will happen?

------------>Replicaset && Replication configuration replicas: 3

for Replication controller --------------> will not create new instances as we have number of replicas desired is running and match labels.

For replicaset ----------> it will run new 3 instances for this replica and mange old pods if there match between labels.

How to Scale replicaset (decrease or increase) ?

1) update yaml file number of replicas && kubectl replace -f file_name --force

delete old pods and replace them with new pods

2)kubectl scale --replicas=number -f file_name

3)kubectl scale --replicas=number replicaset replicaset_name

commands -------------->.....

$ kubectl create -f file_name ----------------------> create k8s object

$ kubectl get replicaset

$ kubectl delete replicaset replicaset_name

$kubectl scale --replicas=number -f file_name

$kubectl scale --replicas=number replicaset replicaset_name

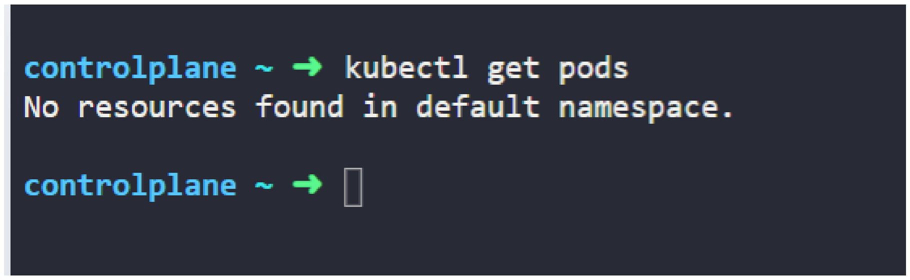
Note : To create replicaset labels on pod definition on metadata section for pod must match labels on selector section

----

Labs ------------>

Q1) How many PODs exist on the system? 0 pods

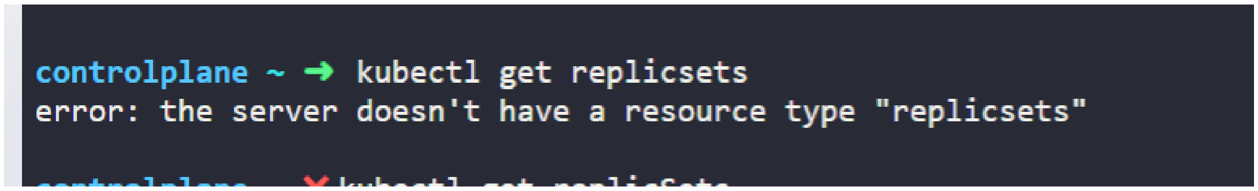
$ kubectl get pods



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

controlplane ~ ➜  
```

Q2) How many ReplicaSets exist on the system?

$ kubectl get replicasets

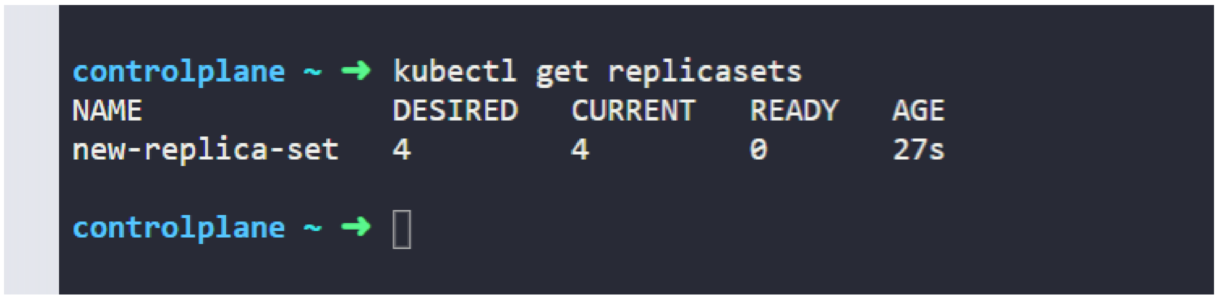


```
controlplane ~ ➜  kubectl get replicsets
error: the server doesn't have a resource type "replicsets"

controlplane ~ ✗ kubectl get replicaSets
```

Q3) How about now? How many ReplicaSets do you see?

Desired ----------> refers to number of replicas to be created

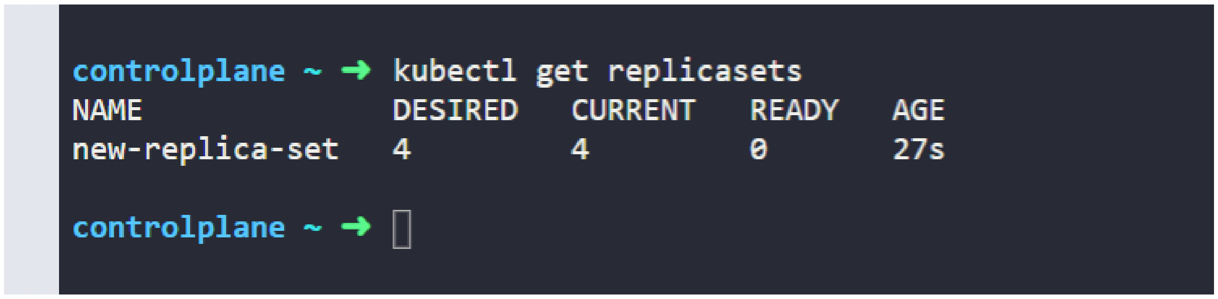
Current --------> refer to number of replicas has been created.



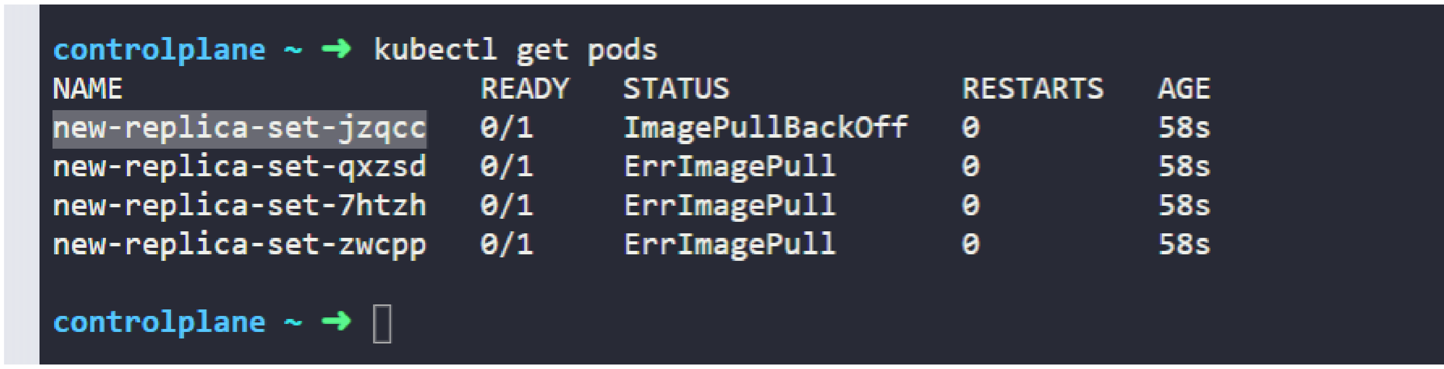
```
controlplane ~ ➜ kubectl get replicasets
NAME              DESIRED   CURRENT   READY   AGE
new-replica-set   4         4         0       27s

controlplane ~ ➜ 
```

Q4) How many PODs are DESIRED in the new-replica-set?   4 pods

$ kubectl get replicasets



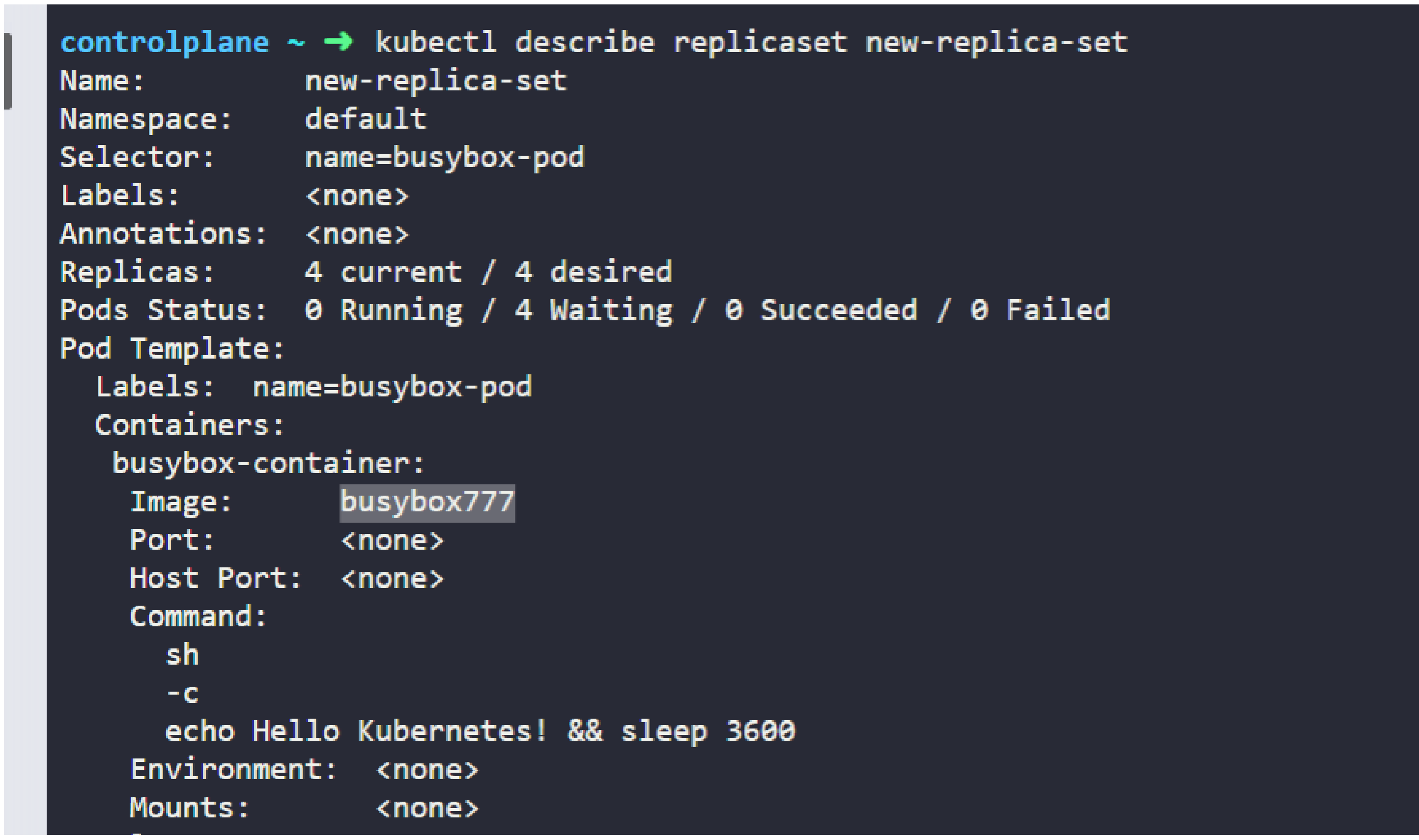
```
controlplane ~ ➜ kubectl get replicasets
NAME              DESIRED   CURRENT   READY   AGE
new-replica-set   4         4         0       27s

controlplane ~ ➜ 
```



```
controlplane ~ ➜ kubectl get pods
NAME                    READY   STATUS             RESTARTS   AGE
new-replica-set-jzqcc   0/1     ImagePullBackOff   0          58s
new-replica-set-qxzsd   0/1     ErrImagePull       0          58s
new-replica-set-7htzh   0/1     ErrImagePull       0          58s
new-replica-set-zwcpp   0/1     ErrImagePull       0          58s

controlplane ~ ➜ 
```

Q5) What is the image used to create the pods in the new-replica-set?

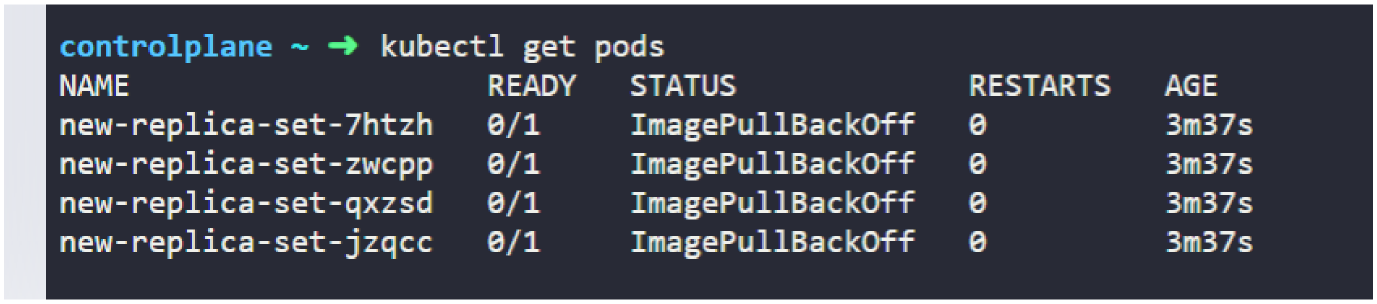
$ kubectl describe replicaset replicaset_name



```
controlplane ~ ➜ kubectl describe replicaset new-replica-set
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>
```

Q6) How many PODs are READY in the new-replica-set? 0

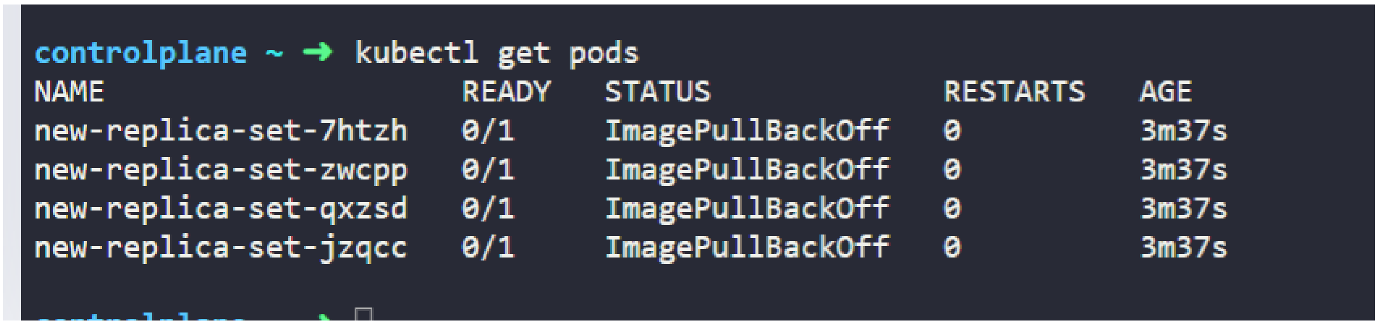
$ kubectl get pods



```
controlplane ~ ➜ kubectl get pods
NAME                    READY   STATUS             RESTARTS   AGE
new-replica-set-7htzh   0/1     ImagePullBackOff   0          3m37s
new-replica-set-zwcpp   0/1     ImagePullBackOff   0          3m37s
new-replica-set-qxzsd   0/1     ImagePullBackOff   0          3m37s
new-replica-set-jzqcc   0/1     ImagePullBackOff   0          3m37s
```

Q7) Why do you think the PODs are not ready?

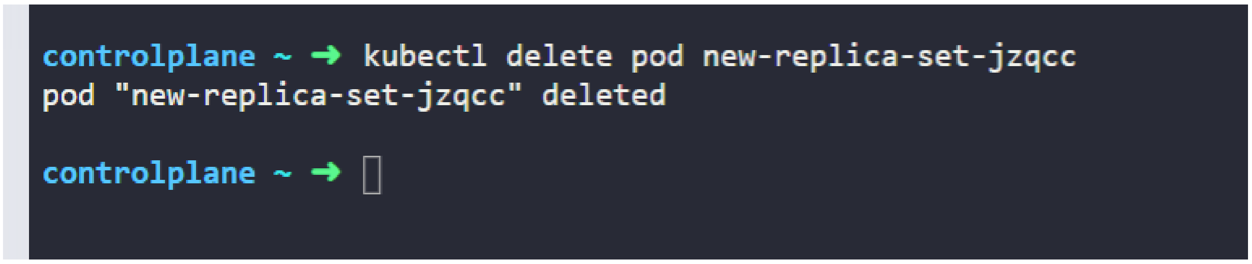
Image has an issue



```
controlplane ~ ➜ kubectl get pods
NAME                     READY   STATUS             RESTARTS   AGE
new-replica-set-7htzh    0/1     ImagePullBackOff   0          3m37s
new-replica-set-zwcpp    0/1     ImagePullBackOff   0          3m37s
new-replica-set-qxzsd    0/1     ImagePullBackOff   0          3m37s
new-replica-set-jzqcc    0/1     ImagePullBackOff   0          3m37s
```

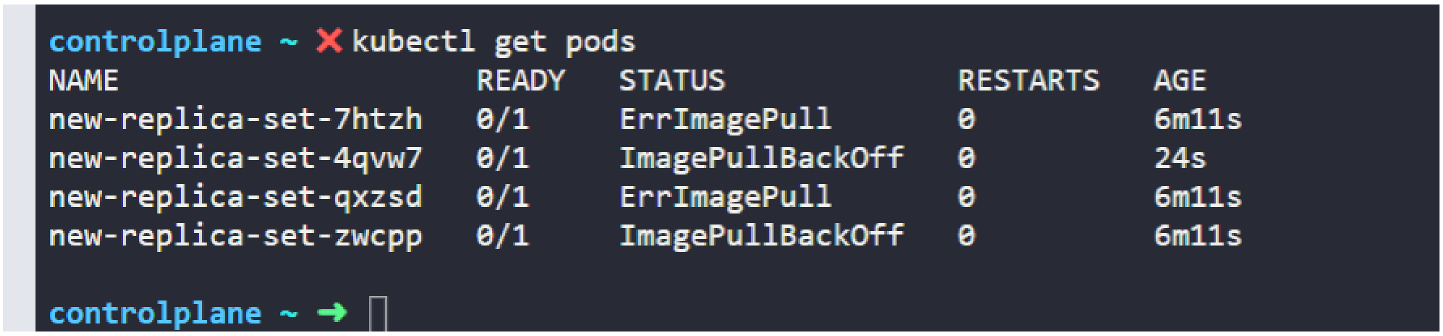
Q7) Delete any one of the 4 PODs.

$ kubectl delete pod pod_name



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

controlplane ~ ➜ 
```

Q8) How many PODs exist now? 4



```
controlplane ~ ✗ kubectl get pods
NAME                     READY   STATUS             RESTARTS   AGE
new-replica-set-7htzh    0/1     ErrImagePull       0          6m11s
new-replica-set-4qvw7    0/1     ImagePullBackOff   0          24s
new-replica-set-qxzsd    0/1     ErrImagePull       0          6m11s
new-replica-set-zwcpp    0/1     ImagePullBackOff   0          6m11s

controlplane ~ ➜ 
```

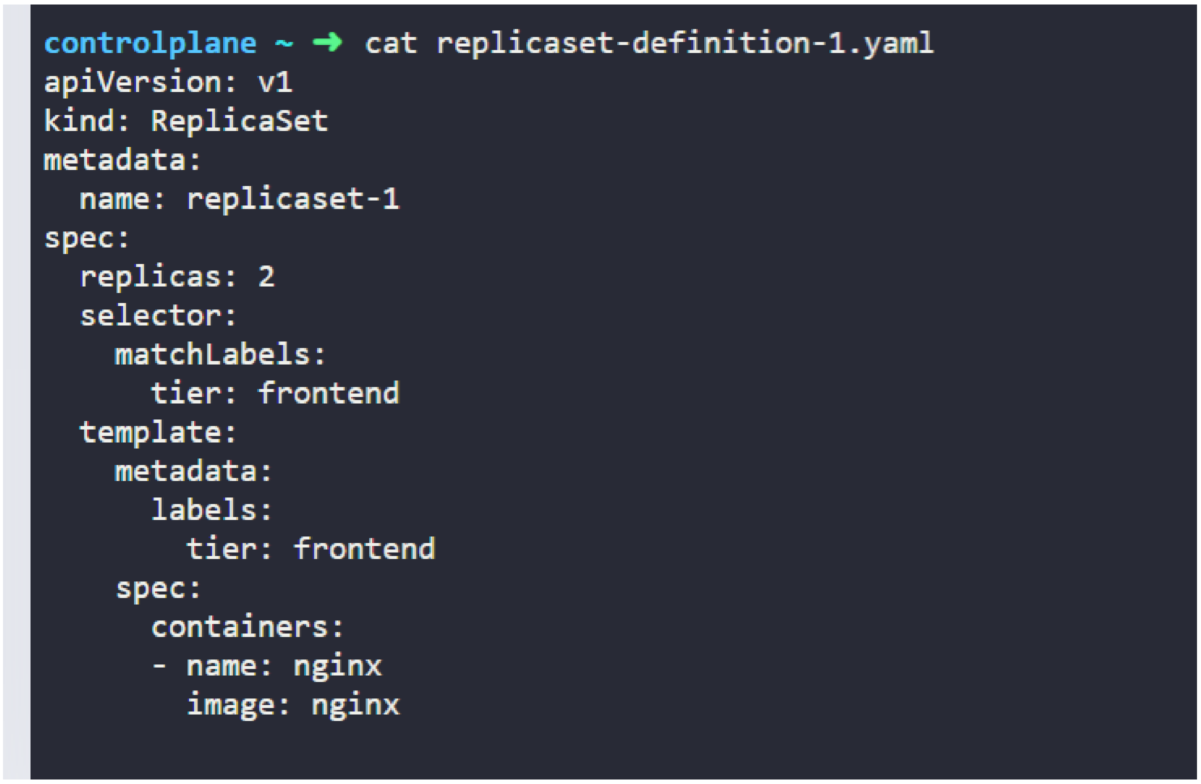
Q9) Why are there still 4 PODs, even after you deleted one?

Because replicaset make sure that currently number of pods equal to desired number.

Q10) Create a ReplicaSet using the replicaset-definition-1.yaml file located at /root/ there is an error fix it.

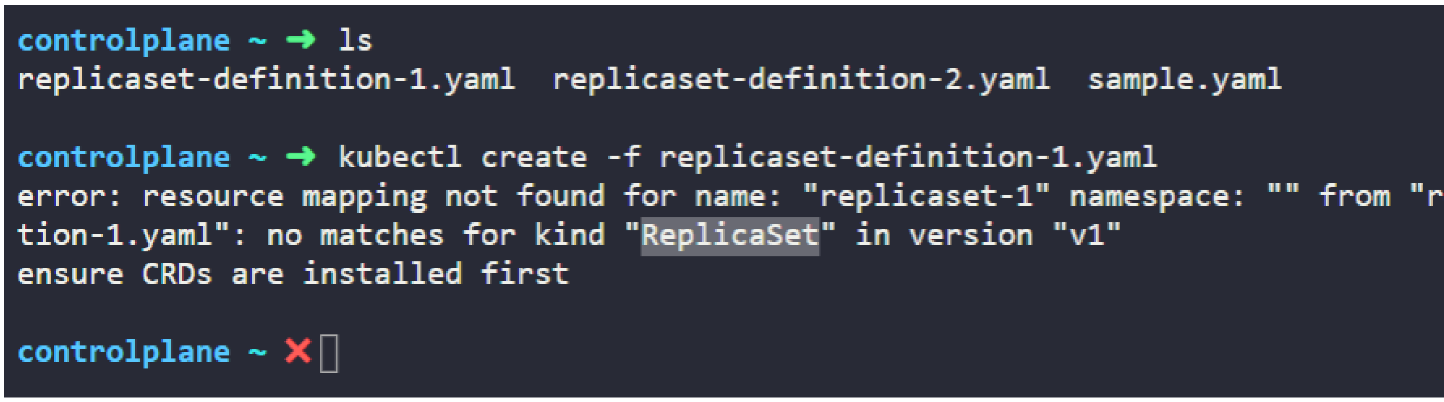
Before any change.

Based on Modification ?



```
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
```

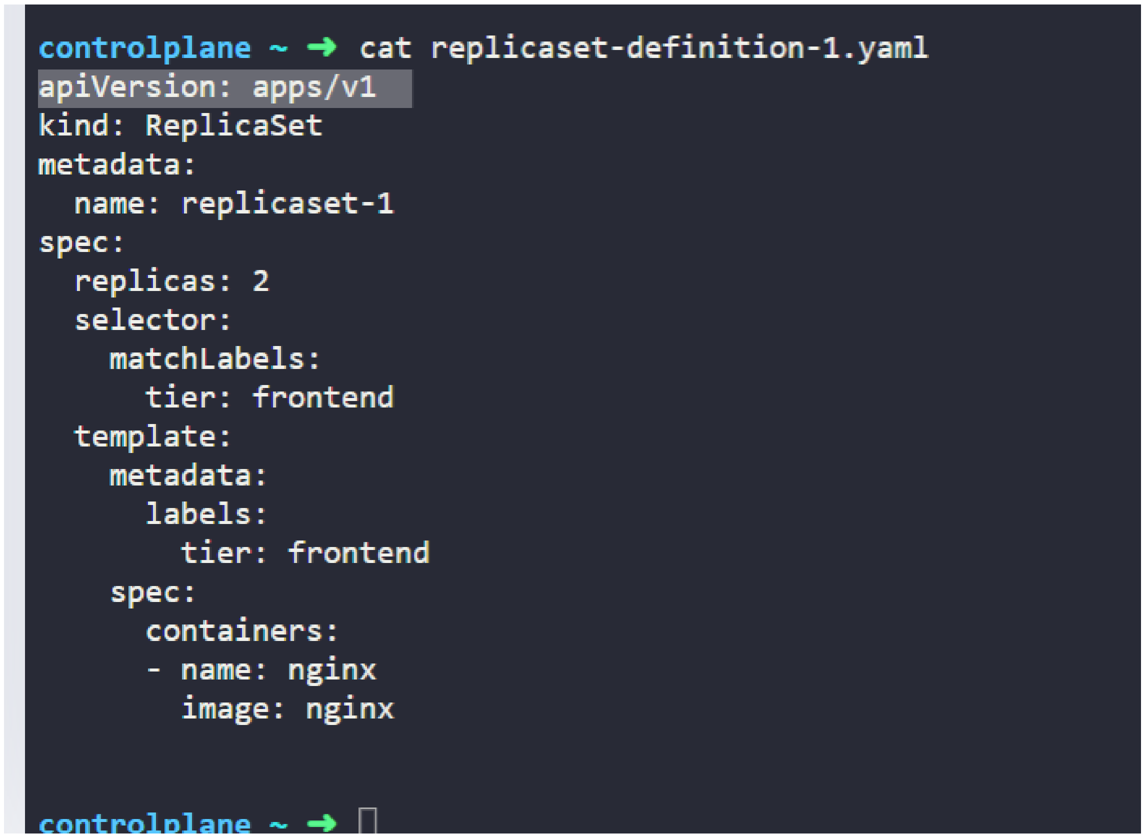
$ kubectl create -f file_name



```
controlplane ~ ➜  ls
replicaset-definition-1.yaml  replicaset-definition-2.yaml  sample.yaml

controlplane ~ ➜  kubectl create -f replicaset-definition-1.yaml
error: resource mapping not found for name: "replicaset-1" namespace: "" from "r
tion-1.yaml": no matches for kind "ReplicaSet" in version "v1"
ensure CRDs are installed first

controlplane ~ ✗
```

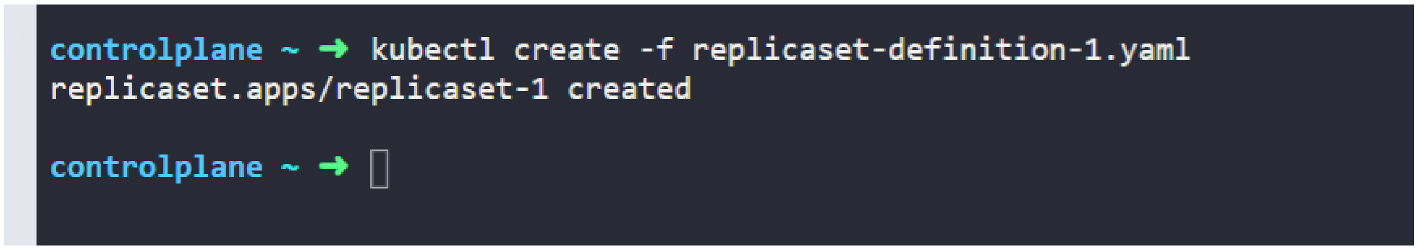
apiVersion of replicaset is not correct .

---------solution----

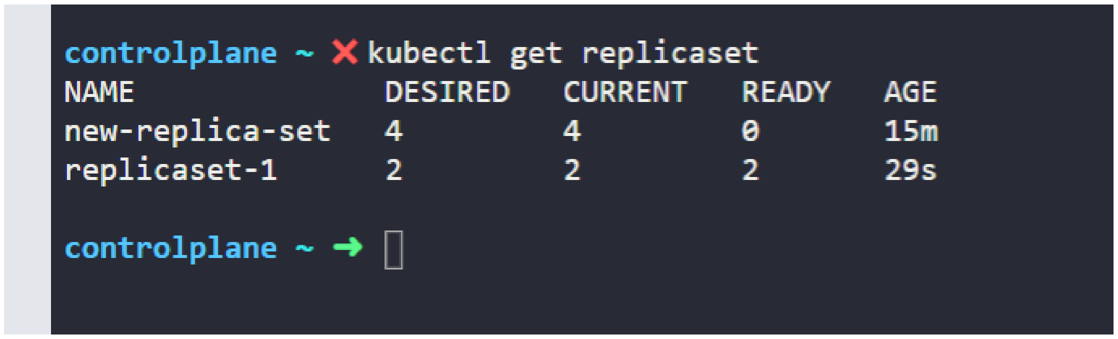


```
controlplane ~ ➜  cat replicaset-definition-1.yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: replicaset-1
spec:
  replicas: 2
  selector:
    matchLabels:
      tier: frontend
  template:
    metadata:
      labels:
        tier: frontend
    spec:
      containers:
      - name: nginx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


controlplane ~ ➜
```

Only modify apiVersion for Replicaset



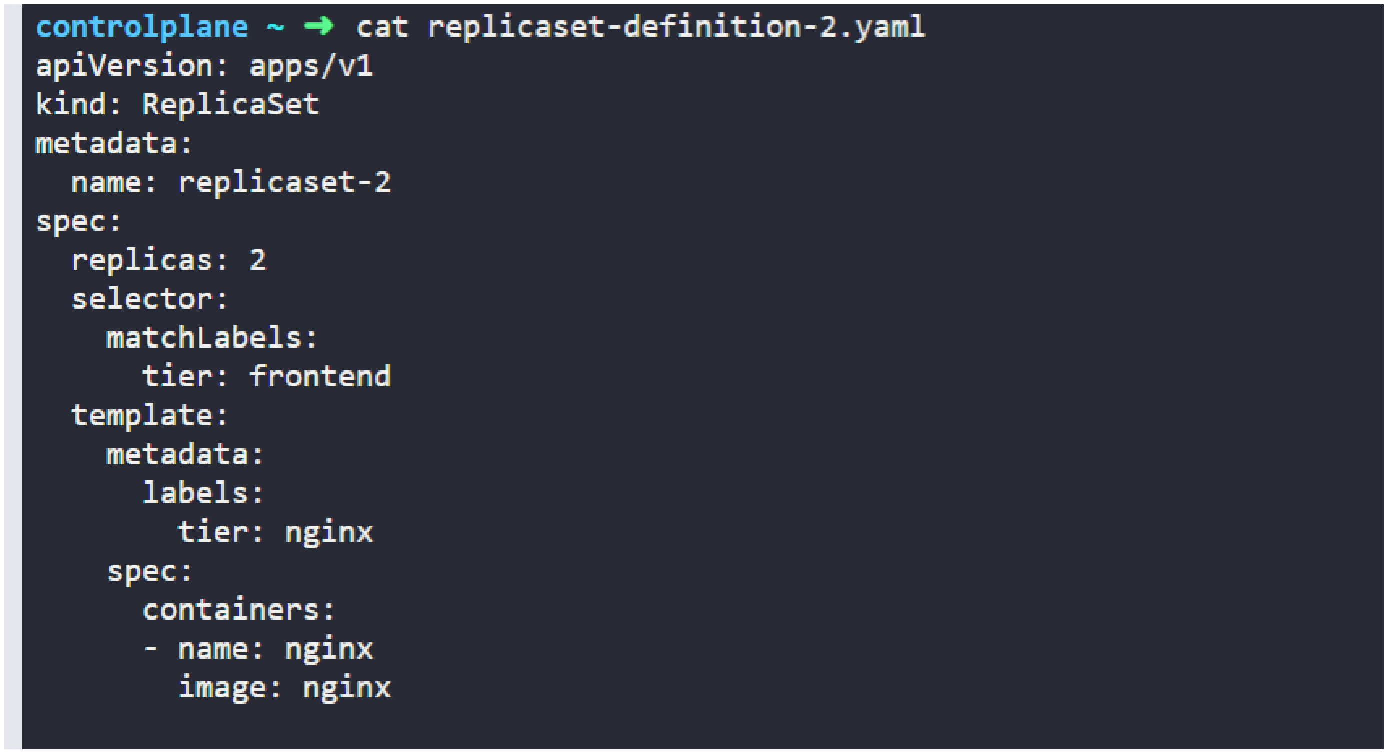
```
controlplane ~ ➜  kubectl create -f replicaset-definition-1.yaml
replicaset.apps/replicaset-1 created

controlplane ~ ➜  
```



```
controlplane ~ ✗ kubectl get replicaset
NAME              DESIRED   CURRENT   READY   AGE
new-replica-set   4         4         0       15m
replicaset-1      2         2         2       29s

controlplane ~ ➜  
```

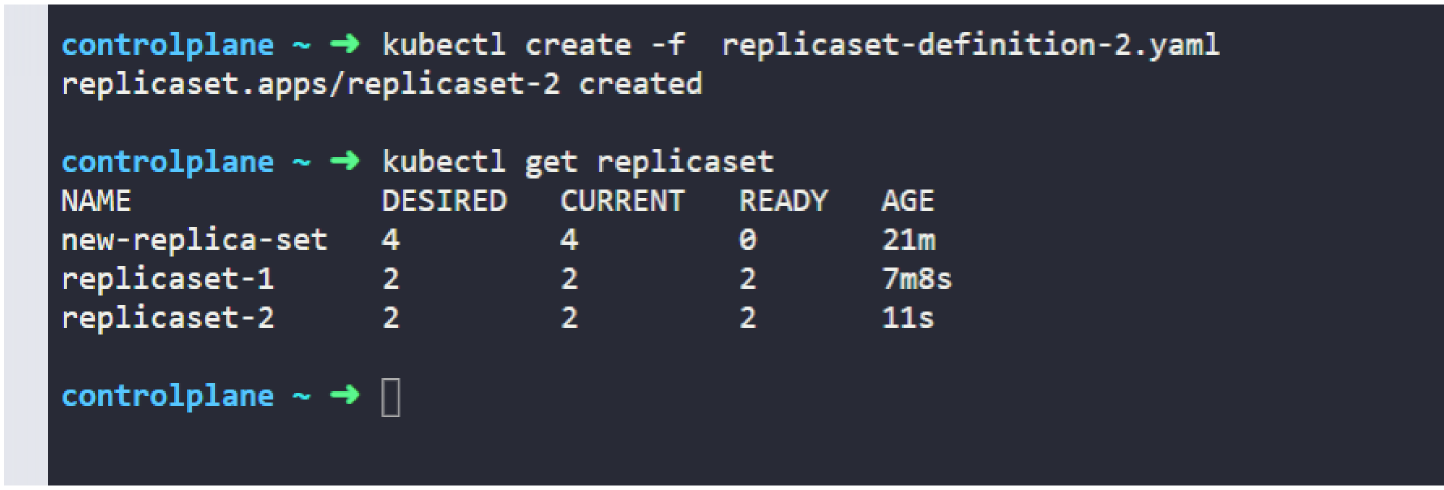
Q11) Fix the issue in the replicaset-definition-2.yaml file and create a ReplicaSet using it.

Before any modification ....

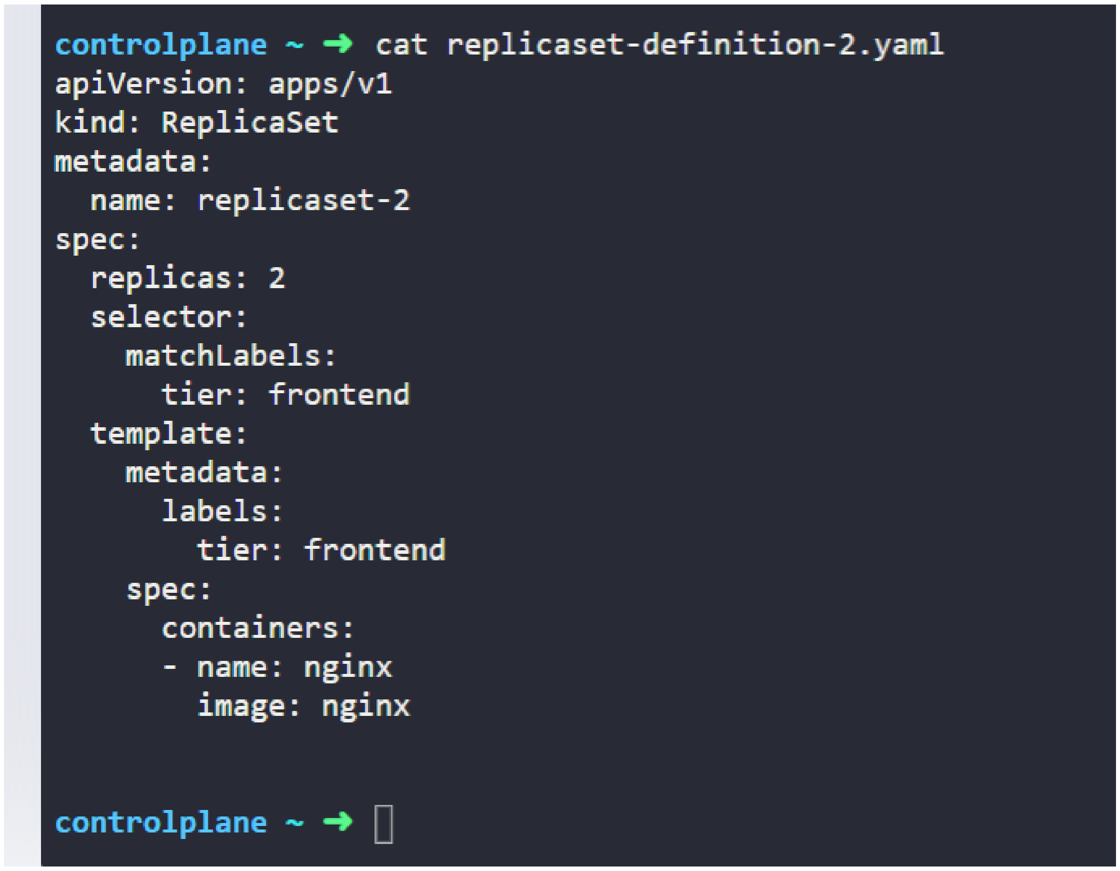


```
controlplane ~ ➜  cat replicaset-definition-2.yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: replicaset-2
spec:
  replicas: 2
  selector:
    matchLabels:
      tier: frontend
  template:
    metadata:
      labels:
        tier: nginx
    spec:
      containers:
      - name: nginx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

----------solution--------



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

controlplane ~ ➜  kubectl get replicaset
NAME              DESIRED   CURRENT   READY   AGE
new-replica-set   4         4         0       21m
replicaset-1      2         2         2       7m8s
replicaset-2      2         2         2       11s

controlplane ~ ➜  
```

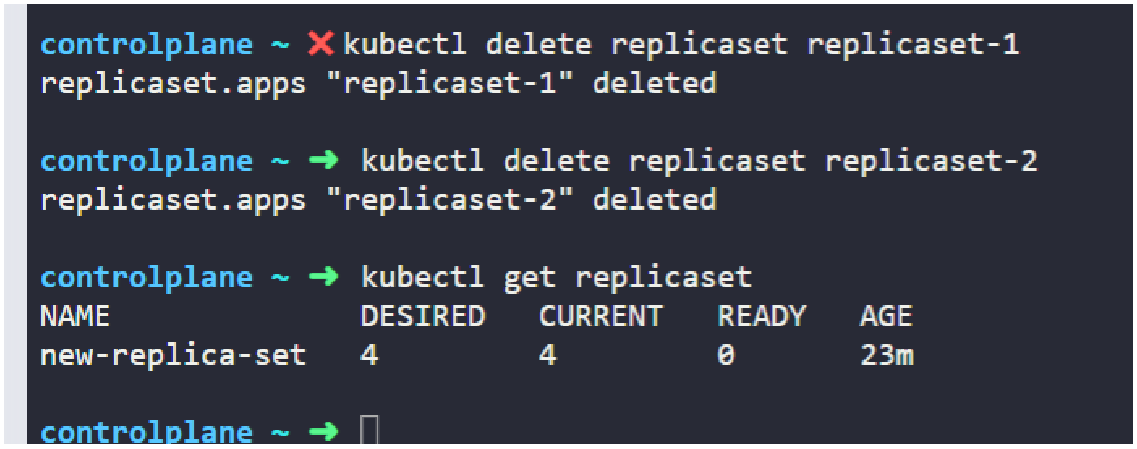


```
controlplane ~ ➜  cat replicaset-definition-2.yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: replicaset-2
spec:
  replicas: 2
  selector:
    matchLabels:
      tier: frontend
  template:
    metadata:
      labels:
        tier: frontend
    spec:
      containers:
      - name: nginx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


controlplane ~ ➜  
```

Solution : To create replicaset labels on pod definition on metadata section for pod must match labels on selector section

Q12) Delete the two newly created ReplicaSets - replicaset-1 and replicaset-2

$ kubectl delete replicaset replicaset_name



```
controlplane ~ ✗ kubectl delete replicaset replicaset-1
replicaset.apps "replicaset-1" deleted

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

controlplane ~ ➜  kubectl get replicaset
NAME              DESIRED   CURRENT   READY   AGE
new-replica-set   4         4         0       23m

controlplane ~ ➜
```

Q13) Fix the original replica set new-replica-set to use the correct busybox image.

The old replicaset has image on its configuration called busybox 123 that is not exist on dockerhub repo.

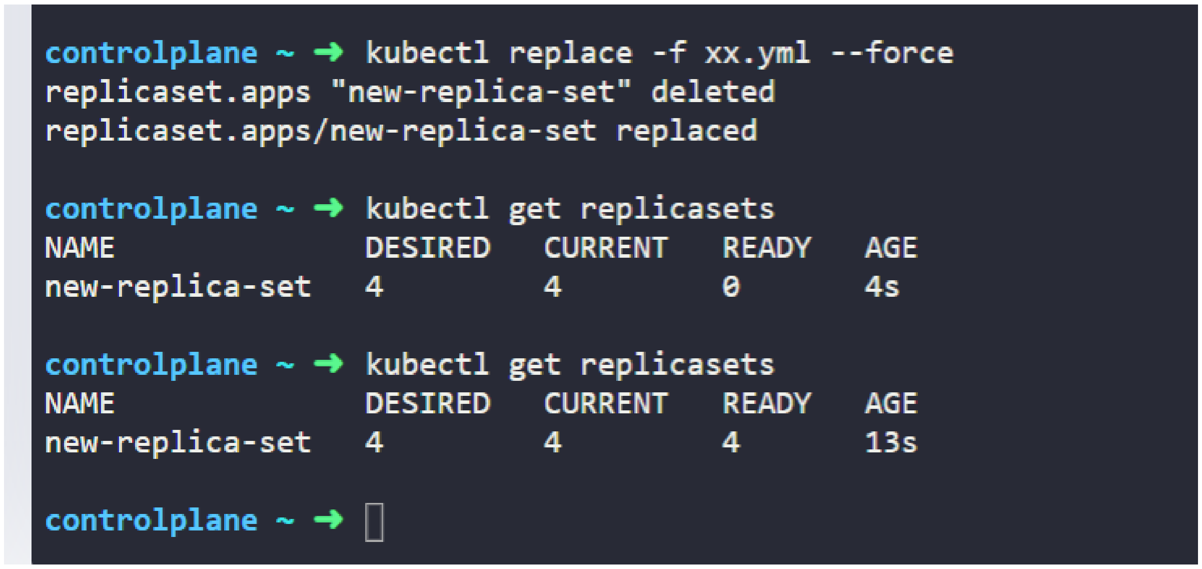
so i don't know where is location of old replicaset file but needs its configuration ?

 $ kubectl get replicaset replicaset_name -o yaml > text.yml

-o yaml -----------------> means get this k8s but output in yaml format and save it on text.yml .

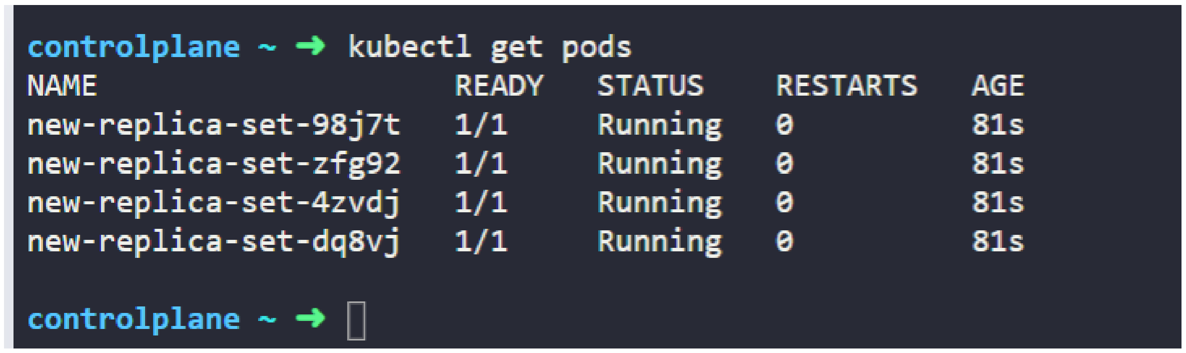
Then modify image name and perform

$ kubectl replace -f text.yml --force



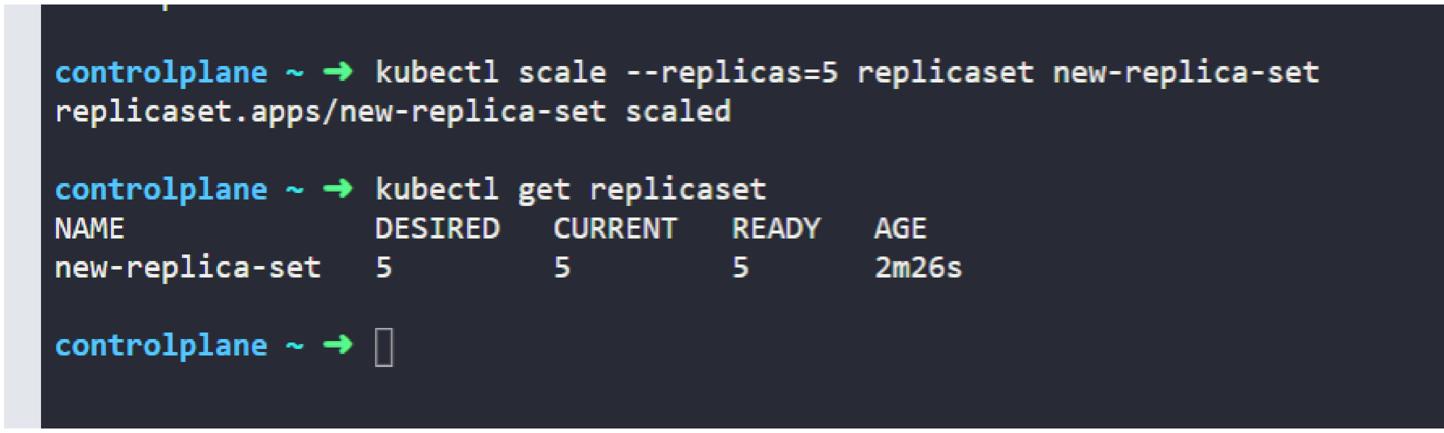
```
controlplane ~ ➜  kubectl replace -f xx.yml --force
replicaset.apps "new-replica-set" deleted
replicaset.apps/new-replica-set replaced

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

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

controlplane ~ ➜  
```

Q14) Scale the ReplicaSet to 5 PODs.



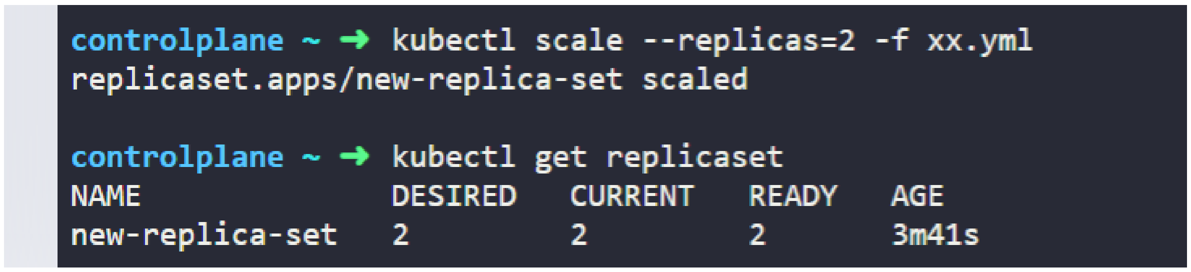
```
controlplane ~ ➜  kubectl get pods
NAME                    READY   STATUS    RESTARTS   AGE
new-replica-set-98j7t   1/1     Running   0          81s
new-replica-set-zfg92   1/1     Running   0          81s
new-replica-set-4zvdj   1/1     Running   0          81s
new-replica-set-dq8vj   1/1     Running   0          81s

controlplane ~ ➜  
```

$kubectl scale --replicas=5 replicaset replicaset_name


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

controlplane ~ ➜ kubectl get replicaset
NAME              DESIRED   CURRENT   READY   AGE
new-replica-set   5         5         5       2m26s

controlplane ~ ➜ 
```

Q15) Now scale the ReplicaSet down to 2 PODs.

$kubectl scale --replicas=5 -f file_name


```
controlplane ~ ➜ kubectl scale --replicas=2 -f xx.yml
replicaset.apps/new-replica-set scaled

controlplane ~ ➜ kubectl get replicaset
NAME              DESIRED   CURRENT   READY   AGE
new-replica-set   2         2         2       3m41s
```