REPLICASET

About Replicaset:-

To create a multiple set of same pods is called replicaset, same set of image container can be run with multiple copies and its giving high availability. Label is very important for replicaset, to identify the running pods and same label cannot be used. To create more than one identical copy, can use replicaset and for high availability replicaset can be used and incase if not going to modify the image then replicaset can be used.

Actual readme:-

A replicaSet purpose is to maintain a stable set of replica Pods running at any given time. As such, it is often used to guarantee the availability of a specified number of identical Pods.

How a ReplicaSet works:-

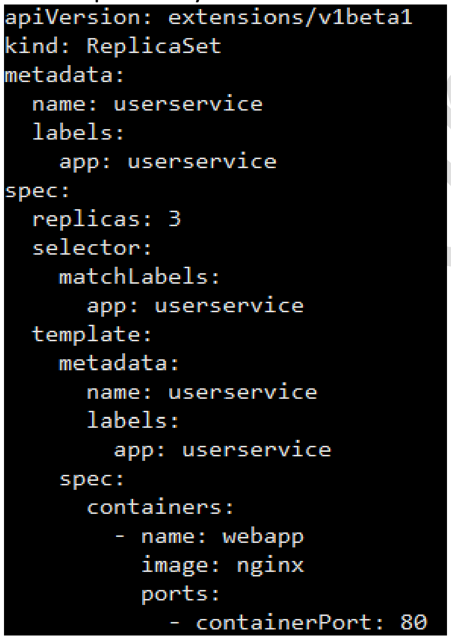
A ReplicaSet is defined with fields, including a selector that specifies how to identify Pods it can acquire, a number of replicas indicating how many Pods it should be maintaining, and a pod template specifying the data of new Pods it should create to meet the number of replicas criteria. A ReplicaSet then fulfills its purpose by creating and deleting Pods as needed to reach the desired number. When a ReplicaSet needs to create new Pods, it uses its Pod template.

When to use a ReplicaSet**:-**

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

###Create a replicaset###

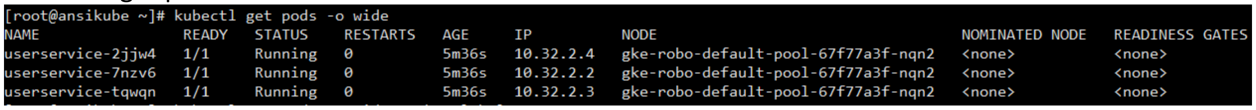
# cat replicaset.yml



```
apiVersion: extensions/v1beta1
kind: ReplicaSet
metadata:
  name: userservice
  labels:
    app: userservice
spec:
  replicas: 3
  selector:
    matchLabels:
      app: userservice
  template:
    metadata:
      name: userservice
      labels:
        app: userservice
    spec:
      containers:
        - name: webapp
          image: nginx
          ports:
            - containerPort: 80
```

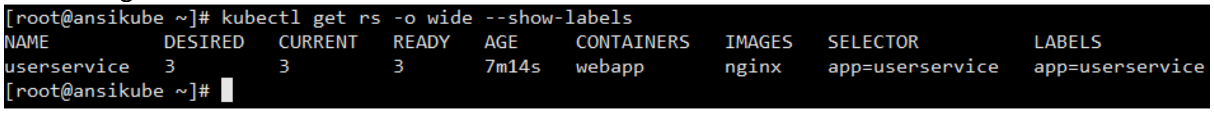
# kubectl apply -f replicaset.yml

kubectl get pods -o wide



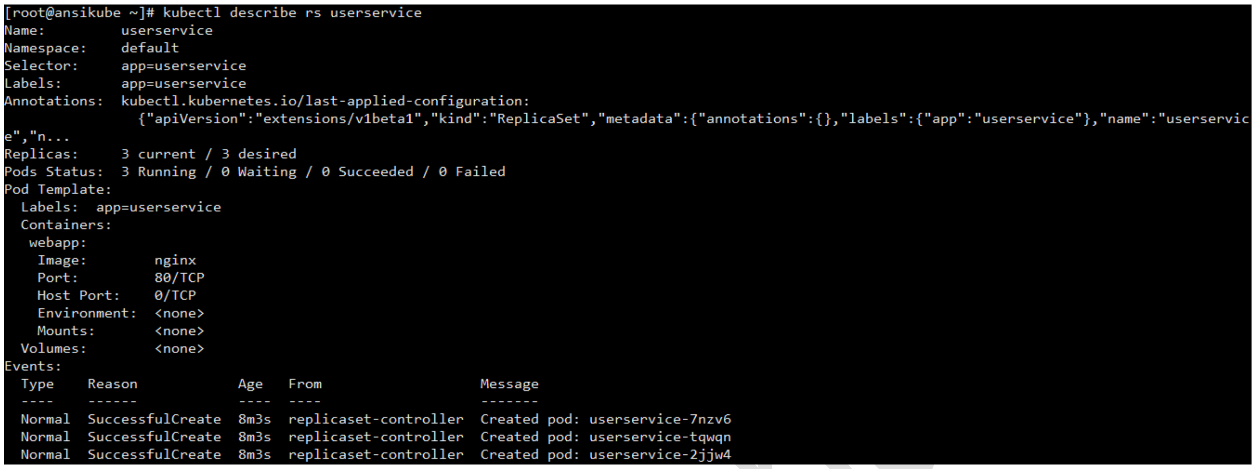
```
[root@ansikube ~]# kubectl get pods -o wide
NAME                READY   STATUS    RESTARTS   AGE    IP          NODE                                  NOMINATED NODE   READINESS GATES
userservice-2jjw4   1/1     Running   0          5m36s  10.32.2.4   gke-robo-default-pool-67f77a3f-nqn2   <none>           <none>
userservice-7nzv6   1/1     Running   0          5m36s  10.32.2.2   gke-robo-default-pool-67f77a3f-nqn2   <none>           <none>
userservice-tqwqn   1/1     Running   0          5m36s  10.32.2.3   gke-robo-default-pool-67f77a3f-nqn2   <none>           <none>
```

kubectl get rs -o wide --show-labels



```
[root@ansikube ~]# kubectl get rs -o wide --show-labels
NAME          DESIRED   CURRENT   READY   AGE     CONTAINERS   IMAGES   SELECTOR          LABELS
userservice   3         3         3       7m14s   webapp       nginx    app=userservice   app=userservice
[root@ansikube ~]#
```

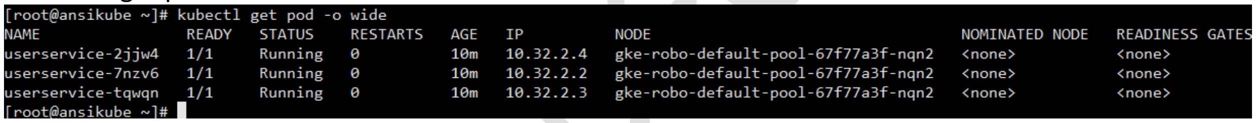
# kubectl describe rs userservice



```
[root@ansikube ~]# kubectl describe rs userservice
Name:         userservice
Namespace:    default
Selector:     app=userservice
Labels:       app=userservice
Annotations:  kubectl.kubernetes.io/last-applied-configuration:
                {"apiVersion":"extensions/v1beta1","kind":"ReplicaSet","metadata":{"annotations":{},"labels":{"app":"userservice"},"name":"userservic
e","n...
Replicas:     3 current / 3 desired
Pods Status:  3 Running / 0 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:  app=userservice
  Containers:
   webapp:
    Image:        nginx
    Port:         80/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Events:
  Type    Reason            Age   From                   Message
  ----    ------            ----  ----                   -------
  Normal  SuccessfulCreate  8m3s  replicaset-controller  Created pod: userservice-7nzv6
  Normal  SuccessfulCreate  8m3s  replicaset-controller  Created pod: userservice-tqwqn
  Normal  SuccessfulCreate  8m3s  replicaset-controller  Created pod: userservice-2jjw4
```

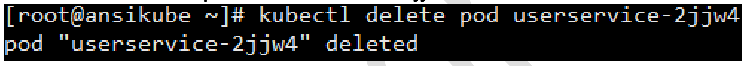
<u>###Testing, delete one container and check whether its recreate from image or not###</u>

kubectl get pod -o wide



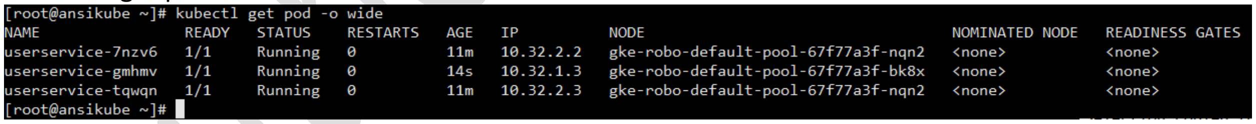
```
[root@ansikube ~]# kubectl get pod -o wide
NAME                READY   STATUS    RESTARTS   AGE   IP          NODE                                  NOMINATED NODE   READINESS GATES
userservice-2jjw4   1/1     Running   0          10m   10.32.2.4   gke-robo-default-pool-67f77a3f-nqn2   <none>           <none>
userservice-7nzv6   1/1     Running   0          10m   10.32.2.2   gke-robo-default-pool-67f77a3f-nqn2   <none>           <none>
userservice-tqwqn   1/1     Running   0          10m   10.32.2.3   gke-robo-default-pool-67f77a3f-nqn2   <none>           <none>
[root@ansikube ~]#
```

kubectl delete pod userservice-2jjw4



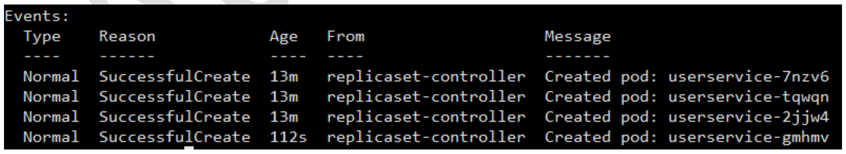
```
[root@ansikube ~]# kubectl delete pod userservice-2jjw4
pod "userservice-2jjw4" deleted
```

kubectl get pod -o wide



```
[root@ansikube ~]# kubectl get pod -o wide
NAME                READY   STATUS    RESTARTS   AGE   IP          NODE                                  NOMINATED NODE   READINESS GATES
userservice-7nzv6   1/1     Running   0          11m   10.32.2.2   gke-robo-default-pool-67f77a3f-nqn2   <none>           <none>
userservice-gmhmv   1/1     Running   0          14s   10.32.1.3   gke-robo-default-pool-67f77a3f-bk8x   <none>           <none>
userservice-tqwqn   1/1     Running   0          11m   10.32.2.3   gke-robo-default-pool-67f77a3f-nqn2   <none>           <none>
[root@ansikube ~]#
```

# kubectl describe rs userservice



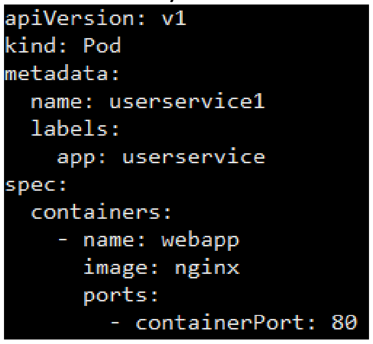
```
Events:
  Type    Reason            Age   From                   Message
  ----    ------            ----  ----                   -------
  Normal  SuccessfulCreate  13m   replicaset-controller  Created pod: userservice-7nzv6
  Normal  SuccessfulCreate  13m   replicaset-controller  Created pod: userservice-tqwqn
  Normal  SuccessfulCreate  13m   replicaset-controller  Created pod: userservice-2jjw4
  Normal  SuccessfulCreate  112s  replicaset-controller  Created pod: userservice-gmhmv
```

Note:- you can see the information about newly created pod, when the old pod got deleted then immediately new pod gets created, this is the main feature of replicaset.

<u>##Create a pod with existing label name of replicaset##</u>

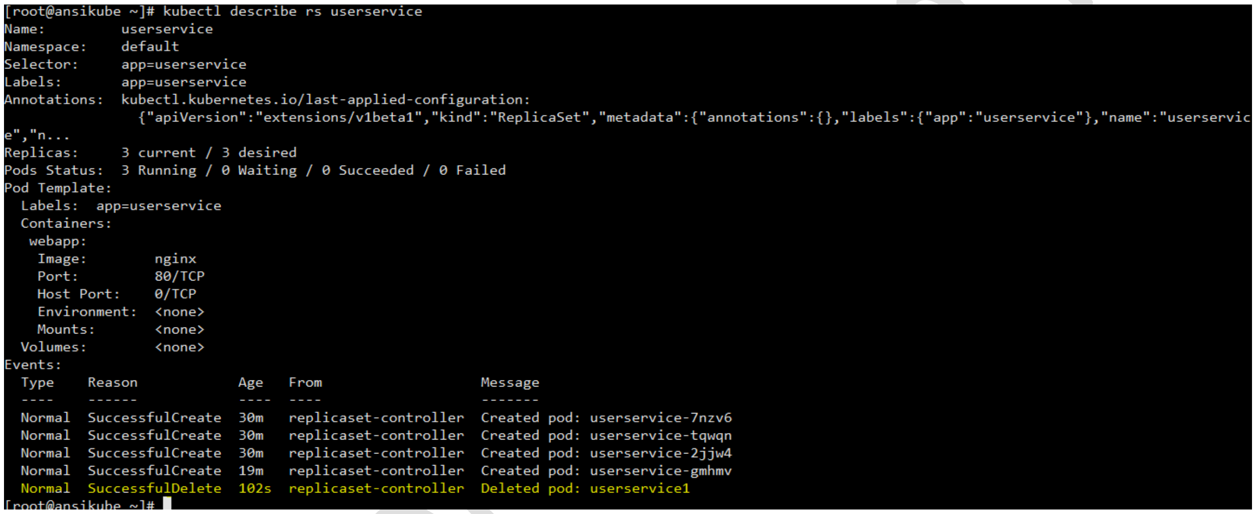
Going to create a pod by specifying the existing replicaset label name.

# cat demors.yml



```
apiVersion: v1
kind: Pod
metadata:
  name: userservice1
  labels:
    app: userservice
spec:
  containers:
    - name: webapp
      image: nginx
      ports:
        - containerPort: 80
```

# kubectl apply -f demors.yml
# kubectl get pods
kubectl describe rs userservice



```
[root@ansikube ~]# kubectl describe rs userservice
Name:         userservice
Namespace:    default
Selector:     app=userservice
Labels:       app=userservice
Annotations:  kubectl.kubernetes.io/last-applied-configuration:
                {"apiVersion":"extensions/v1beta1","kind":"ReplicaSet","metadata":{"annotations":{},"labels":{"app":"userservice"},"name":"userservic
e","n...
Replicas:     3 current / 3 desired
Pods Status:  3 Running / 0 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:  app=userservice
  Containers:
   webapp:
    Image:        nginx
    Port:         80/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Events:
  Type    Reason            Age   From                   Message
  ----    ------            ----  ----                   -------
  Normal  SuccessfulCreate  30m   replicaset-controller  Created pod: userservice-7nzv6
  Normal  SuccessfulCreate  30m   replicaset-controller  Created pod: userservice-tqwqn
  Normal  SuccessfulCreate  30m   replicaset-controller  Created pod: userservice-2jjw4
  Normal  SuccessfulCreate  19m   replicaset-controller  Created pod: userservice-gmhmv
  Normal  SuccessfulDelete  102s  replicaset-controller  Deleted pod: userservice1
[root@ansikube ~]#
```

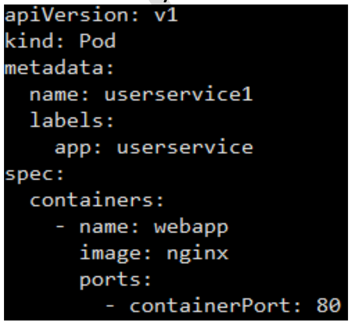
Note:- pod will not be created with existing replicaset label, because replicaset only have 3 desired, which we have mentioned, so 4th pod will not allow to create with the existing label, and it will get deleted.

<u>@@Delete the replicaset and create a pod first with label and then create replicaset.</u>

First pod will be created with label “userservice” and then with same label create the repilcaset with 3 desired.

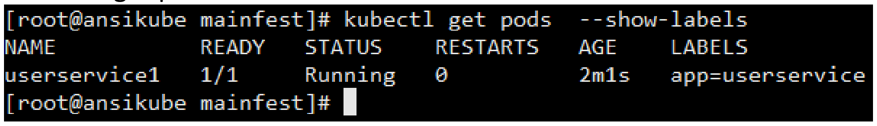
#kubectl delete -f replicaset.yml
#kubectl get pods -o wide
No resources found.

#cat demors.yml



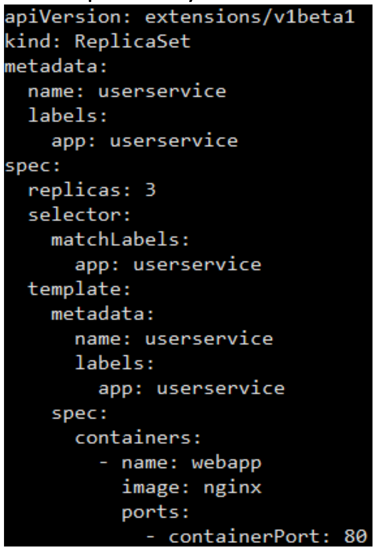
```
apiVersion: v1
kind: Pod
metadata:
  name: userservice1
  labels:
    app: userservice
spec:
  containers:
    - name: webapp
      image: nginx
      ports:
        - containerPort: 80
```

#kubectl apply -f demors.yml
#kubectl get pods --show-labels



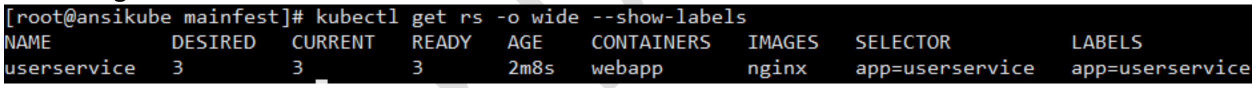
```
[root@ansikube mainfest]# kubectl get pods  --show-labels
NAME            READY   STATUS    RESTARTS   AGE    LABELS
userservice1    1/1     Running   0          2m1s   app=userservice
[root@ansikube mainfest]#
```

#cat replicaset.yml



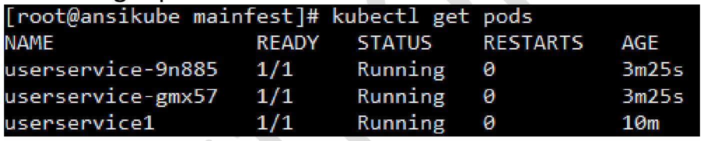
```
apiVersion: extensions/v1beta1
kind: ReplicaSet
metadata:
  name: userservice
  labels:
    app: userservice
spec:
  replicas: 3
  selector:
    matchLabels:
      app: userservice
  template:
    metadata:
      name: userservice
      labels:
        app: userservice
    spec:
      containers:
        - name: webapp
          image: nginx
          ports:
            - containerPort: 80
```

#kubectl apply -f replicaset.yml
#kubectl get rs -o wide --show-labels



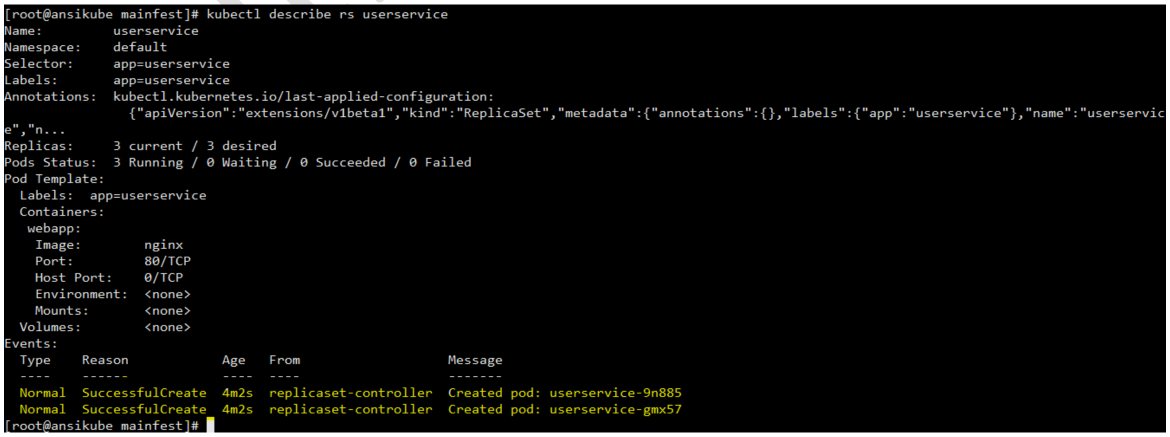
```
[root@ansikube mainfest]# kubectl get rs -o wide --show-labels
NAME          DESIRED   CURRENT   READY   AGE    CONTAINERS   IMAGES   SELECTOR          LABELS
userservice   3         3         3       2m8s   webapp       nginx    app=userservice   app=userservice
```

#kubectl get pods



```
[root@ansikube mainfest]# kubectl get pods
NAME                READY   STATUS    RESTARTS   AGE
userservice-9n885   1/1     Running   0          3m25s
userservice-gmx57   1/1     Running   0          3m25s
userservice1        1/1     Running   0          10m
```

#kubectl describe rs userservice



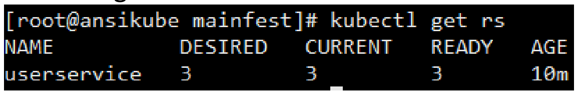
```
[root@ansikube mainfest]# kubectl describe rs userservice
Name:         userservice
Namespace:    default
Selector:     app=userservice
Labels:       app=userservice
Annotations:  kubectl.kubernetes.io/last-applied-configuration:
                {"apiVersion":"extensions/v1beta1","kind":"ReplicaSet","metadata":{"annotations":{},"labels":{"app":"userservice"},"name":"userservic
e","n...
Replicas:     3 current / 3 desired
Pods Status:  3 Running / 0 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:  app=userservice
  Containers:
   webapp:
    Image:        nginx
    Port:         80/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Events:
  Type    Reason            Age   From                   Message
  ----    ------            ----  ----                   -------
  Normal  SuccessfulCreate  4m2s  replicaset-controller  Created pod: userservice-9n885
  Normal  SuccessfulCreate  4m2s  replicaset-controller  Created pod: userservice-gmx57
[root@ansikube mainfest]#
```

Note:- Here replicaset has created only 2 pods instead of 3 desired, because already 1 pod is available with same label, so the conclusion is in replicaset all the pods been identified with labels only.

<u>@@Scaling in the replicaset. To modify the replicaset by using scaling option. this is called as manual scaling.</u>

Scaling out a Deployment will ensure new Pods are created and scheduled to Nodes with available resources. Scaling will increase the number of Pods to the new desired state. Kubernetes also supports autoscaling of Pods, but it is outside of the scope of this tutorial. Scaling to zero is also possible, and it will terminate all Pods of the specified Deployment.

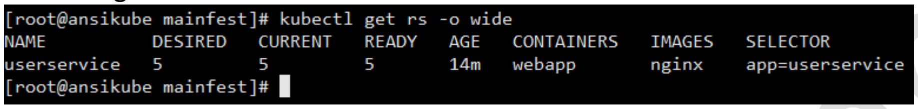
#kubectl get rs



```
[root@ansikube mainfest]# kubectl get rs
NAME          DESIRED   CURRENT   READY   AGE
userservice   3         3         3       10m
```

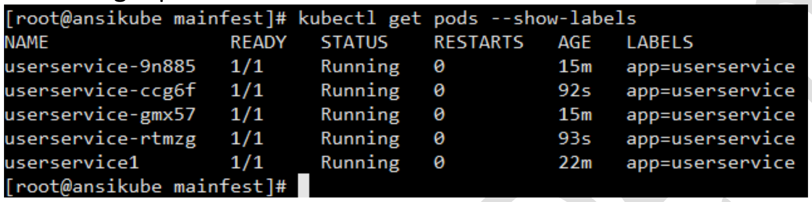
#kubectl scale --current-replicas=3 --replicas=5 rs/userservice

#kubectl get rs -o wide



```
[root@ansikube mainfest]# kubectl get rs -o wide
NAME          DESIRED   CURRENT   READY   AGE   CONTAINERS   IMAGES   SELECTOR
userservice   5         5         5       14m   webapp       nginx    app=userservice
[root@ansikube mainfest]#
```

#kubectl get pods --show-labels

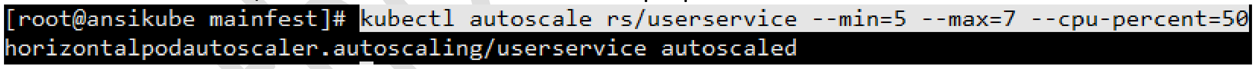


```
[root@ansikube mainfest]# kubectl get pods --show-labels
NAME                READY   STATUS    RESTARTS   AGE   LABELS
userservice-9n885   1/1     Running   0          15m   app=userservice
userservice-ccg6f   1/1     Running   0          92s   app=userservice
userservice-gmx57   1/1     Running   0          15m   app=userservice
userservice-rtmzg   1/1     Running   0          93s   app=userservice
userservice1        1/1     Running   0          22m   app=userservice
[root@ansikube mainfest]#
```

<u>@Autoscaling</u>

Autoscaling method, depends on resource availability, replicaset will get increase or decrease the pods. it's called as horizontal pod autoscaling. when the cpu get usage increased then only additional pod will create with settings of autoscaling. This is called adhoc or command line method.

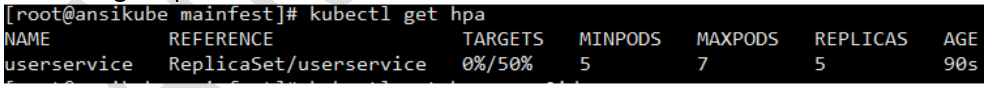
#kubectl autoscale rs/userservice --min=5 --max=7 --cpu-percent=50



```
[root@ansikube mainfest]# kubectl autoscale rs/userservice --min=5 --max=7 --cpu-percent=50
horizontalpodautoscaler.autoscaling/userservice autoscaled
```

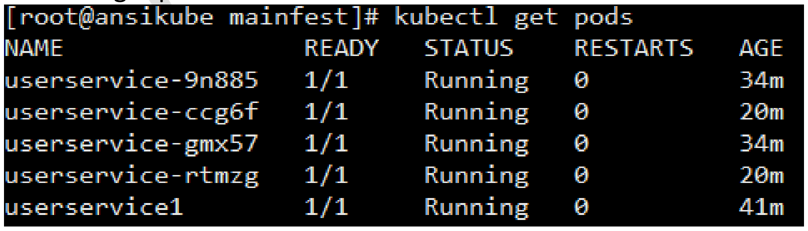
<u>: - horizontal pod autoscaler (hpa)</u>

#kubectl get hpa



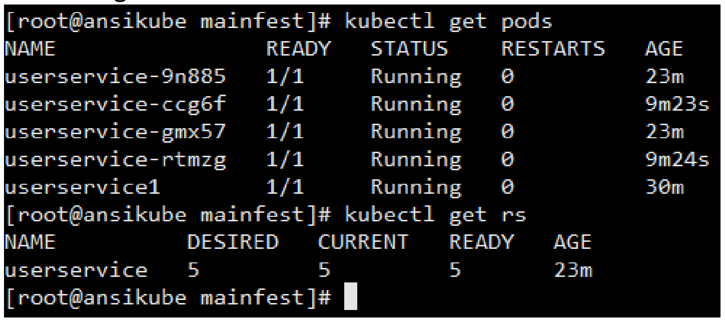
```
[root@ansikube mainfest]# kubectl get hpa
NAME          REFERENCE                 TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
userservice   ReplicaSet/userservice    0%/50%    5         7         5          90s
```

#kubectl get pods



```
[root@ansikube mainfest]# kubectl get pods
NAME                READY   STATUS    RESTARTS   AGE
userservice-9n885   1/1     Running   0          34m
userservice-ccg6f   1/1     Running   0          20m
userservice-gmx57   1/1     Running   0          34m
userservice-rtmzg   1/1     Running   0          20m
userservice1        1/1     Running   0          41m
```

#kubectl get rs

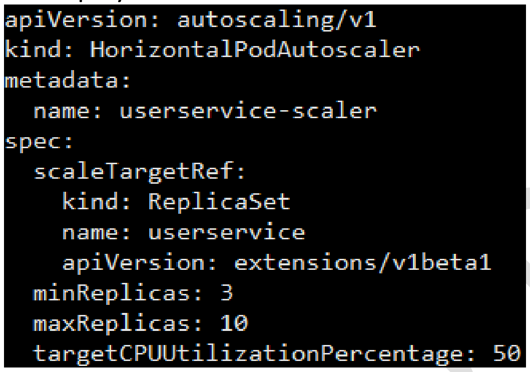


```
[root@ansikube mainfest]# kubectl get pods
NAME                READY   STATUS    RESTARTS   AGE
userservice-9n885   1/1     Running   0          23m
userservice-ccg6f   1/1     Running   0          9m23s
userservice-gmx57   1/1     Running   0          23m
userservice-rtmzg   1/1     Running   0          9m24s
userservice1        1/1     Running   0          30m
[root@ansikube mainfest]# kubectl get rs
NAME          DESIRED   CURRENT   READY   AGE
userservice   5         5         5       23m
[root@ansikube mainfest]#
```

<u>##Mainfest method or yml method to create autoscaling the replicaset##</u>
Delete the existing hpa and create new hpa with manifest yml for testing purpose.

# kubectl delete hpa userservice
#kubectl explain hpa

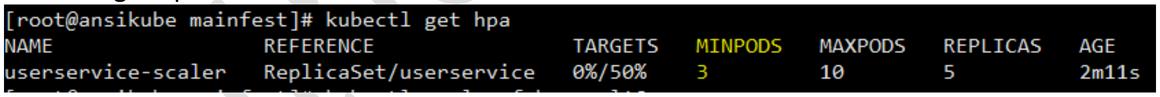
#cat hpa.yml



```
apiVersion: autoscaling/v1
kind: HorizontalPodAutoscaler
metadata:
  name: userservice-scaler
spec:
  scaleTargetRef:
    kind: ReplicaSet
    name: userservice
    apiVersion: extensions/v1beta1
  minReplicas: 3
  maxReplicas: 10
  targetCPUUtilizationPercentage: 50
```

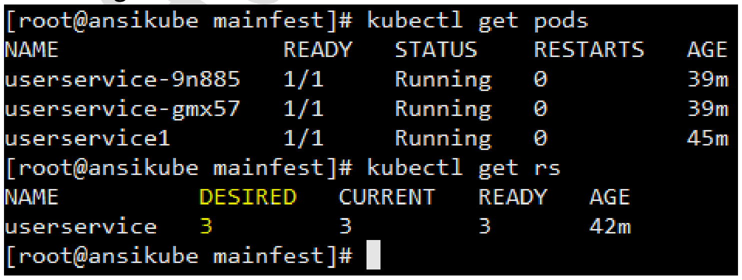
#kubectl apply -f hpa.yml

#kubectl get hpa



```
[root@ansikube mainfest]# kubectl get hpa
NAME                 REFERENCE                 TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
userservice-scaler   ReplicaSet/userservice    0%/50%    3         10        5          2m11s
```

#kubectl get pods
#kubectl get rs



```
[root@ansikube mainfest]# kubectl get pods
NAME                READY   STATUS    RESTARTS   AGE
userservice-9n885   1/1     Running   0          39m
userservice-gmx57   1/1     Running   0          39m
userservice1        1/1     Running   0          45m
[root@ansikube mainfest]# kubectl get rs
NAME          DESIRED   CURRENT   READY   AGE
userservice   3         3         3       42m
[root@ansikube mainfest]#
```

Note: - with manifest yml file, hpa has been created with min 3 and max 10 pods, so the autoscale will work whenever the resource been utilized and if you see previous output of pods, which is decrease from minpods 5 to minpods 3.