Scaling

What is Scaling?

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.

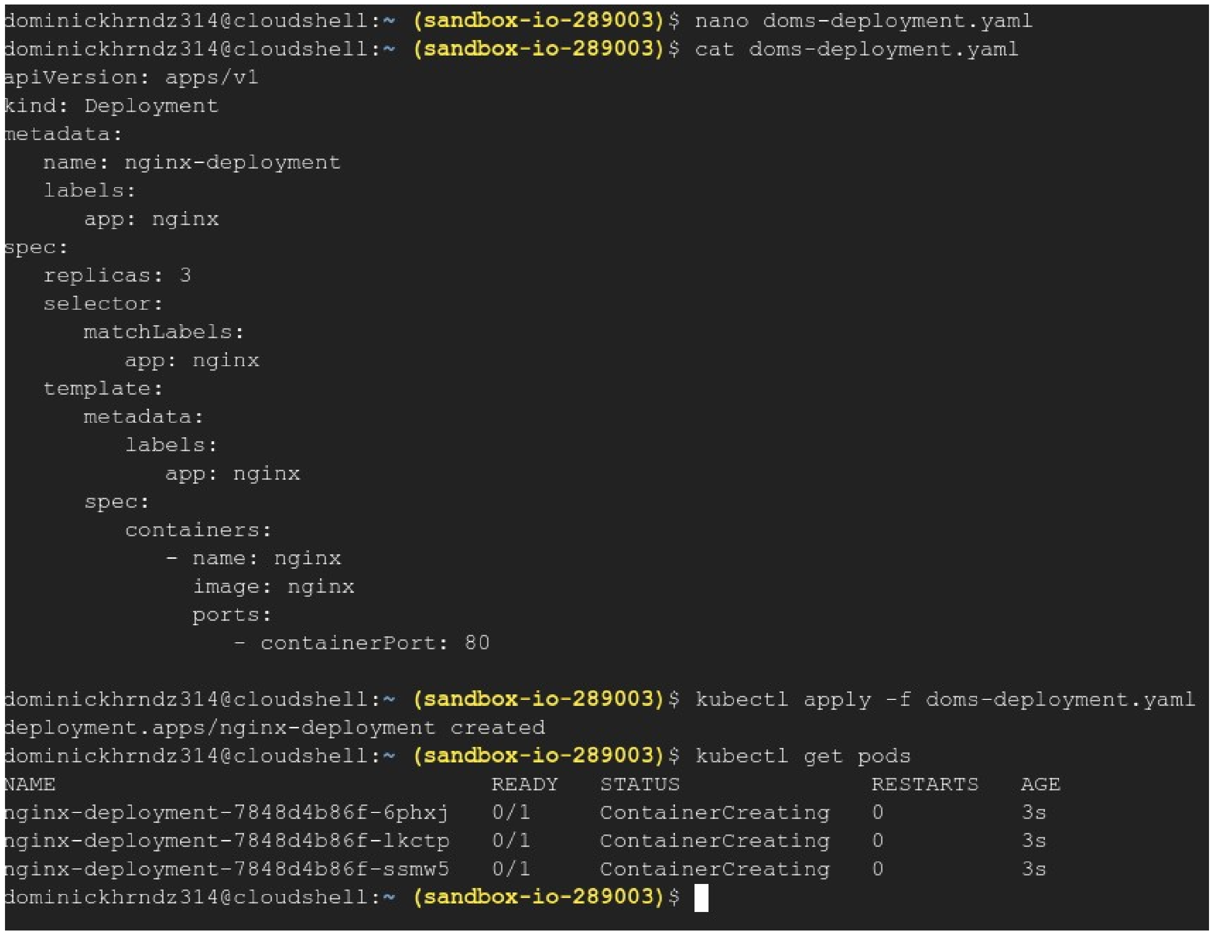
Running multiple instances of an application will require a way to distribute the traffic to all of them.

Scaling is accomplished by changing the number of replicas in a Deployment. Once you have multiple instances of an Application running, you would be able to do Rolling updates without downtime.

Hands On – Scale a deployment through both declarative and imperative methods

I. Declarative Scaling

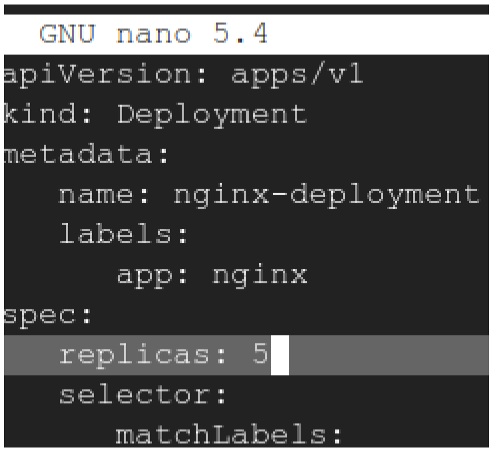
Let's first start by creating a deployment file, thus applying a declarative change. Create 3 pods running nginx, apply, and verify.



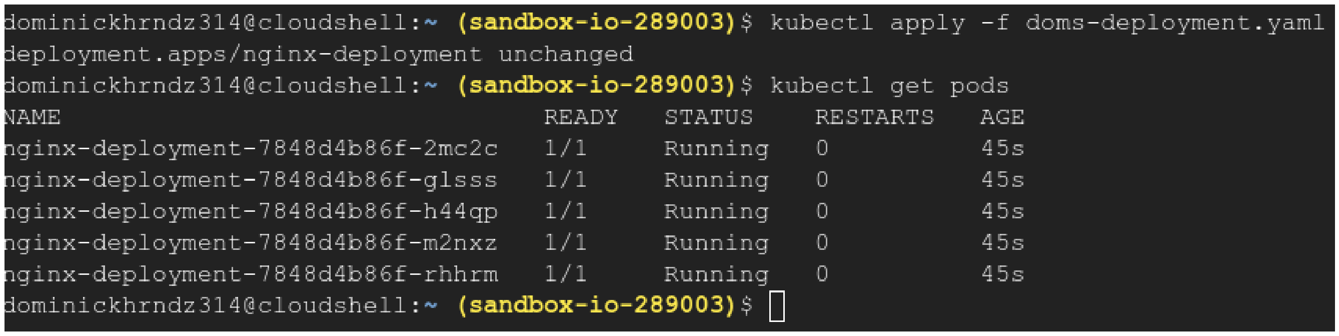
```
dominickhrndz314@cloudshell:~ (sandbox-io-289003)$ nano doms-deployment.yaml
dominickhrndz314@cloudshell:~ (sandbox-io-289003)$ cat doms-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
   name: nginx-deployment
   labels:
      app: nginx
spec:
   replicas: 3
   selector:
      matchLabels:
         app: nginx
   template:
      metadata:
         labels:
            app: nginx
      spec:
         containers:
            - name: nginx
              image: nginx
              ports:
                 - containerPort: 80

dominickhrndz314@cloudshell:~ (sandbox-io-289003)$ kubectl apply -f doms-deployment.yaml
deployment.apps/nginx-deployment created
dominickhrndz314@cloudshell:~ (sandbox-io-289003)$ kubectl get pods
NAME                                READY   STATUS              RESTARTS   AGE
nginx-deployment-7848d4b86f-6phxj   0/1     ContainerCreating   0          3s
nginx-deployment-7848d4b86f-lkctp   0/1     ContainerCreating   0          3s
nginx-deployment-7848d4b86f-ssmw5   0/1     ContainerCreating   0          3s
dominickhrndz314@cloudshell:~ (sandbox-io-289003)$
```

Now let’s scale the number of replicas to 5 using the declarative method. Go back into your file and change the *replicas* field from the 3 to 5. Save your changes, apply the config, and verify.



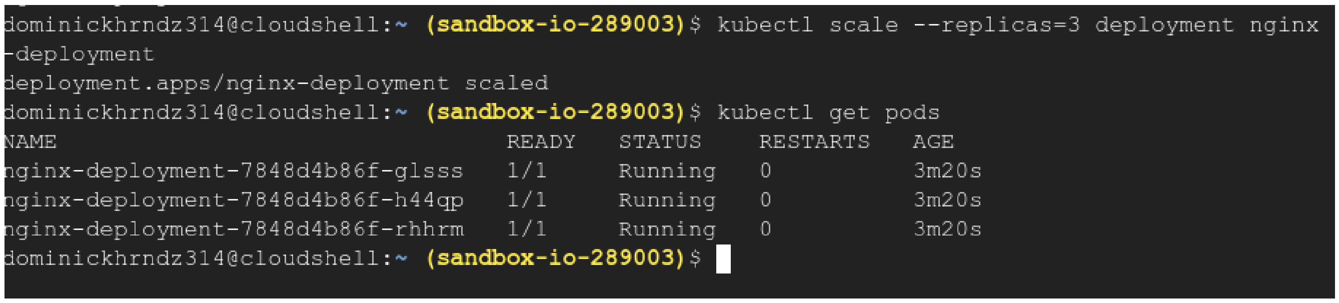
```
  GNU nano 5.4
apiVersion: apps/v1
kind: Deployment
metadata:
   name: nginx-deployment
   labels:
      app: nginx
spec:
   replicas: 5
   selector:
      matchLabels:
```



```
dominickhrndz314@cloudshell:~ (sandbox-io-289003)$ kubectl apply -f doms-deployment.yaml
deployment.apps/nginx-deployment unchanged
dominickhrndz314@cloudshell:~ (sandbox-io-289003)$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-7848d4b86f-2mc2c   1/1     Running   0          45s
nginx-deployment-7848d4b86f-glsss   1/1     Running   0          45s
nginx-deployment-7848d4b86f-h44qp   1/1     Running   0          45s
nginx-deployment-7848d4b86f-m2nxz   1/1     Running   0          45s
nginx-deployment-7848d4b86f-rhhrm   1/1     Running   0          45s
dominickhrndz314@cloudshell:~ (sandbox-io-289003)$
```

II. Imperative Scaling

Now lets use imperative commands to scale our deployment back down to 3 pods instead of 5. we can use the kubectl scale –replicas=3 deployment nginx command to accomplish this.



```
dominickhrndz314@cloudshell:~ (sandbox-io-289003)$ kubectl scale --replicas=3 deployment nginx
-deployment
deployment.apps/nginx-deployment scaled
dominickhrndz314@cloudshell:~ (sandbox-io-289003)$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
nginx-deployment-7848d4b86f-glsss   1/1     Running   0          3m20s
nginx-deployment-7848d4b86f-h44qp   1/1     Running   0          3m20s
nginx-deployment-7848d4b86f-rhhrm   1/1     Running   0          3m20s
dominickhrndz314@cloudshell:~ (sandbox-io-289003)$
```