Tasks:

1. You are tasked with creating a simple pod in your Kubernetes cluster. The pod should run a container using the busybox image.

2. Change the image name from busybox to nginx, also check that pod is running well.

3. Create a ReplicaSet named "app-replicaset" managing three replicas of an application pod using the nginx:v1 image.

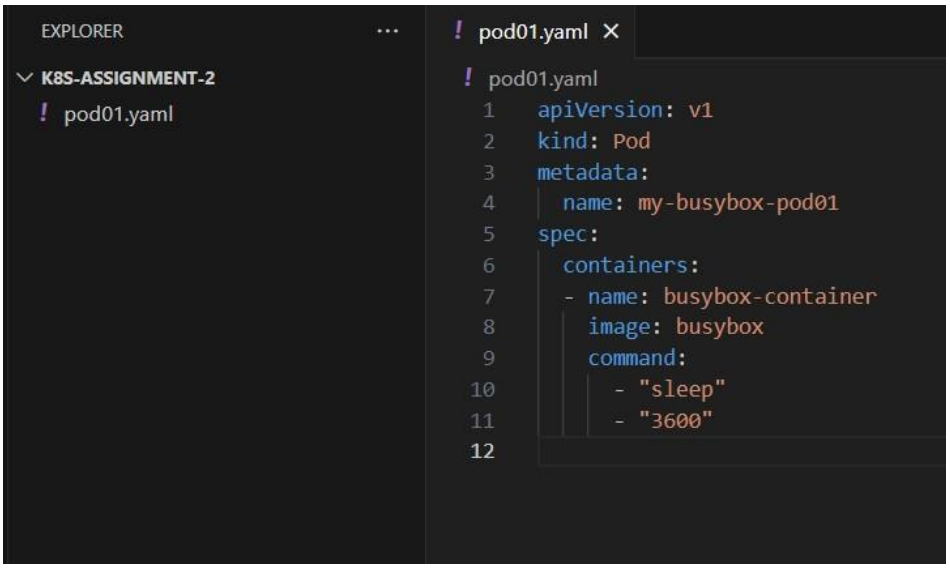
4. Create a Deployment named "app-deployment" managing four replicas of an application pod using the nginx:alpine image.

5. Explain how to automatically roll back to the previous version using the "app-deployment."

6. Describe the differences between a ClusterIP service and a LoadBalancer service, providing a use case for each.

Task 1: Creating a simple pod with busybox Image

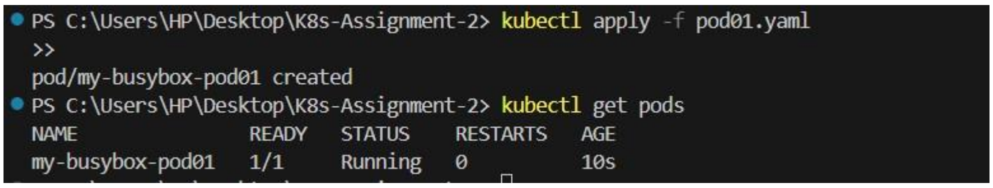
Creating a file named **busybox-pod01.yaml** with the following content:



```yaml
apiVersion: v1
kind: Pod
metadata:
  name: my-busybox-pod01
spec:
  containers:
  - name: busybox-container
    image: busybox
    command:
      - "sleep"
      - "3600"
```

Apply the configuration using the following command:

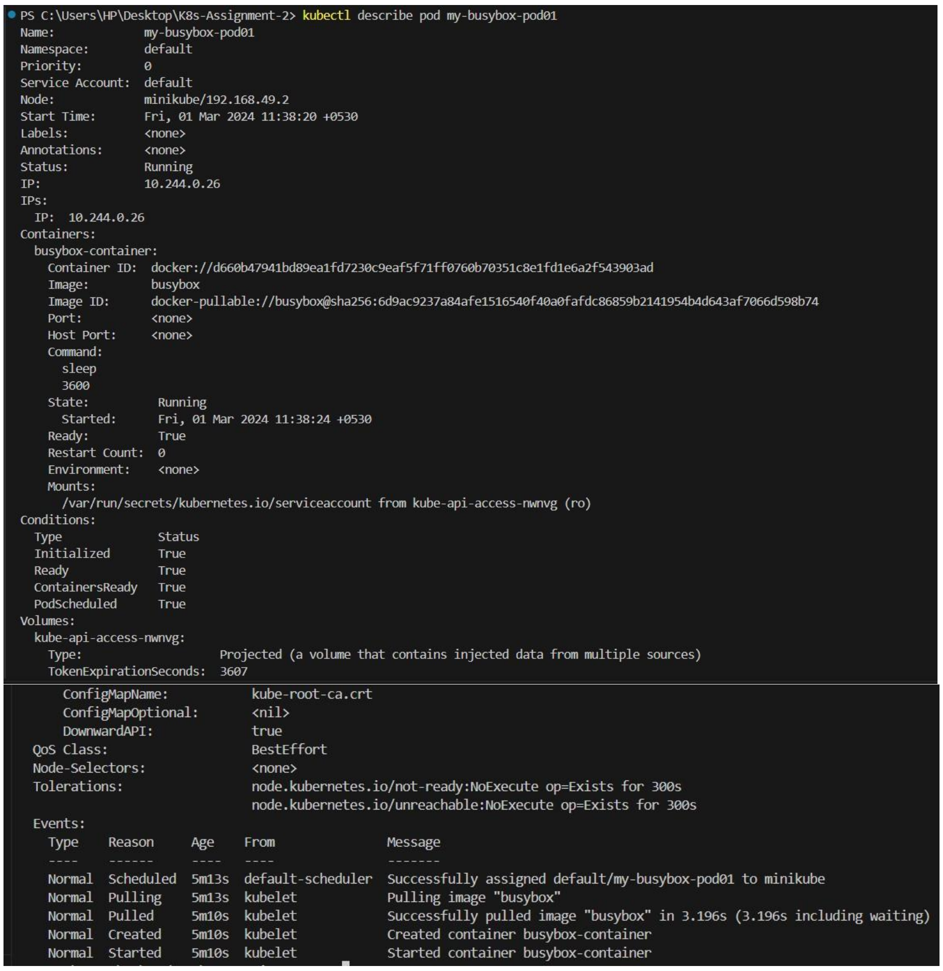
kubectl apply -f pod01.yaml



```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl apply -f pod01.yaml
>>
pod/my-busybox-pod01 created
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl get pods
NAME               READY   STATUS    RESTARTS   AGE
my-busybox-pod01   1/1     Running   0          10s
```

Get the description of the pod by running the below command:

kubectl describe pod my-busybox-pod01

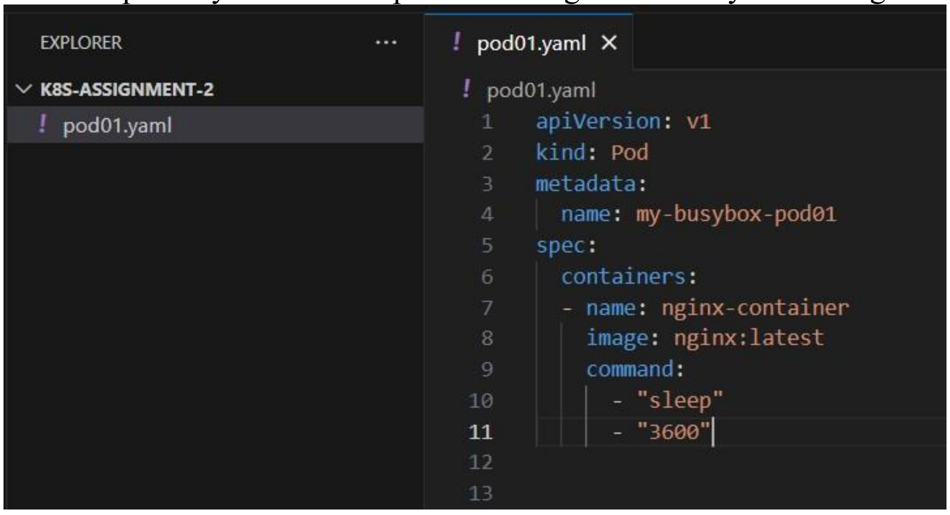


```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl describe pod my-busybox-pod01
Name:             my-busybox-pod01
Namespace:        default
Priority:         0
Service Account:  default
Node:             minikube/192.168.49.2
Start Time:       Fri, 01 Mar 2024 11:38:20 +0530
Labels:           <none>
Annotations:      <none>
Status:           Running
IP:               10.244.0.26
IPs:
  IP:  10.244.0.26
Containers:
  busybox-container:
    Container ID:  docker://d660b47941bd89ea1fd7230c9eaf5f71ff0760b70351c8e1fd1e6a2f543903ad
    Image:         busybox
    Image ID:      docker-pullable://busybox@sha256:6d9ac9237a84afe1516540f40a0fafdc86859b2141954b4d643af7066d598b74
    Port:          <none>
    Host Port:     <none>
    Command:
      sleep
      3600
    State:          Running
      Started:      Fri, 01 Mar 2024 11:38:24 +0530
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-nwnvg (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  kube-api-access-nwnvg:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
    DownwardAPI:             true
QoS Class:                   BestEffort
Node-Selectors:              <none>
Tolerations:                 node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                             node.kubernetes.io/unreachable:NoExecute op=Exists for 300s

Events:
  Type    Reason     Age    From               Message
  ----    ------     ----   ----               -------
  Normal  Scheduled  5m13s  default-scheduler  Successfully assigned default/my-busybox-pod01 to minikube
  Normal  Pulling    5m13s  kubelet            Pulling image "busybox"
  Normal  Pulled     5m10s  kubelet            Successfully pulled image "busybox" in 3.196s (3.196s including waiting)
  Normal  Created    5m10s  kubelet            Created container busybox-container
  Normal  Started    5m10s  kubelet            Started container busybox-container
```

Task 2: Changing pod image from busybox to nginx

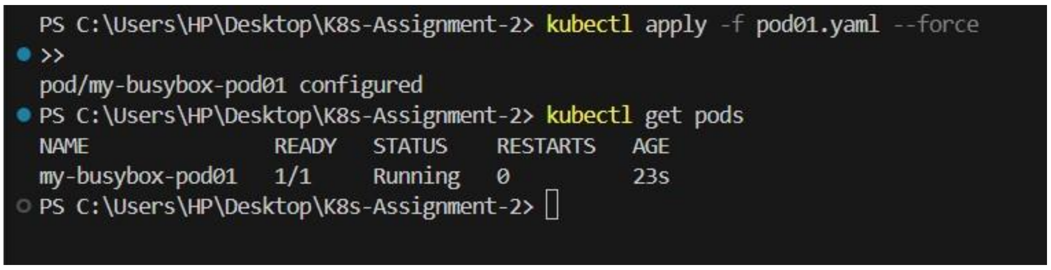
Edit the ‘pod01.yaml’ file to update the image from ‘busybox’ to ‘nginx’:



```yaml
apiVersion: v1
kind: Pod
metadata:
  name: my-busybox-pod01
spec:
  containers:
  - name: nginx-container
    image: nginx:latest
    command:
      - "sleep"
      - "3600"
```

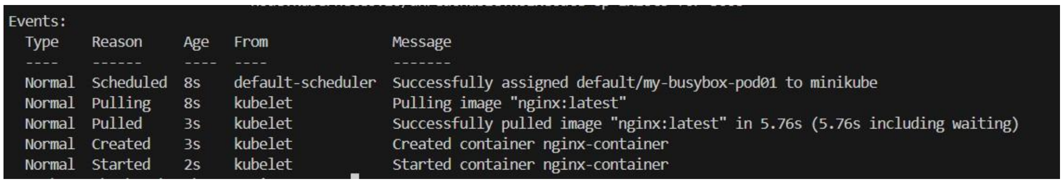
Apply the updated configuration:

kubectl apply -f pod01.yaml –force



```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl apply -f pod01.yaml --force
>>
pod/my-busybox-pod01 configured
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl get pods
NAME               READY   STATUS    RESTARTS   AGE
my-busybox-pod01   1/1     Running   0          23s
PS C:\Users\HP\Desktop\K8s-Assignment-2>
```

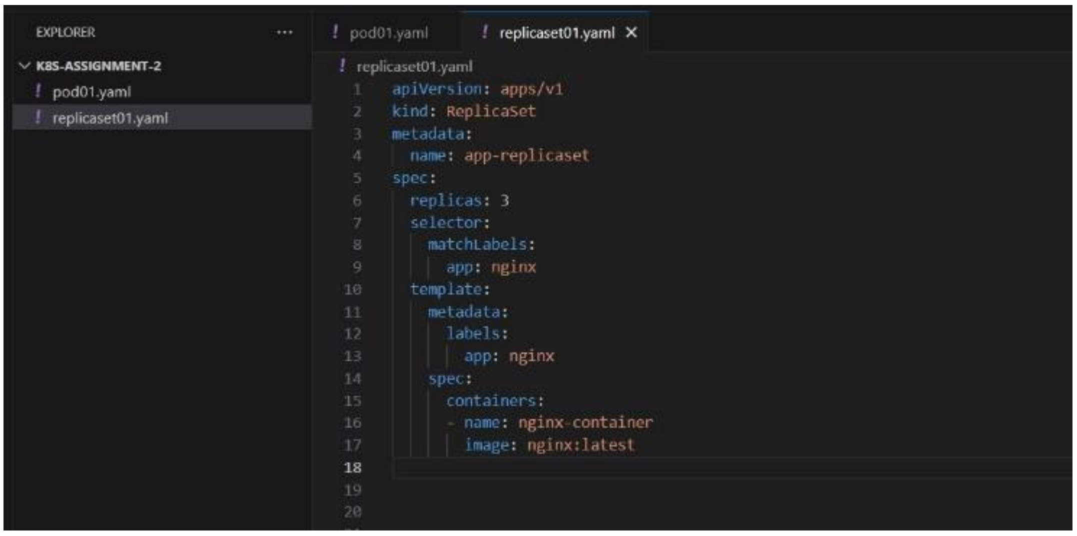
Screenshot of **“kubectl describe pod my-busybox-pod01”** command: Now, it is pulling the image “nginx:latest”.



```
Events:
  Type    Reason     Age   From               Message
  ----    ------     ----  ----               -------
  Normal  Scheduled  8s    default-scheduler  Successfully assigned default/my-busybox-pod01 to minikube
  Normal  Pulling    8s    kubelet            Pulling image "nginx:latest"
  Normal  Pulled     3s    kubelet            Successfully pulled image "nginx:latest" in 5.76s (5.76s including waiting)
  Normal  Created    3s    kubelet            Created container nginx-container
  Normal  Started    2s    kubelet            Started container nginx-container
```

Task 3: Creating a replicaset with nginx image

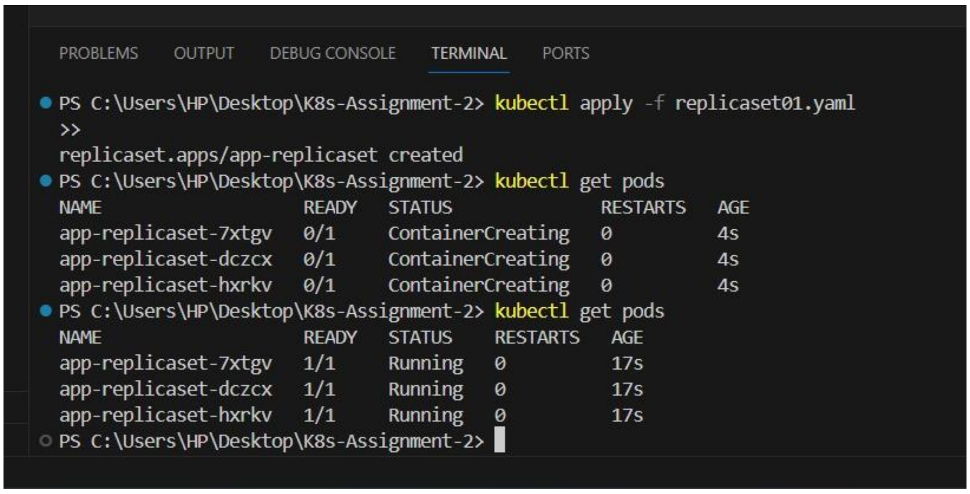
Create a file named ‘replicaset01.yaml’ with the following content:



```yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: app-replicaset
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx-container
        image: nginx:latest
```

Apply the configuration:

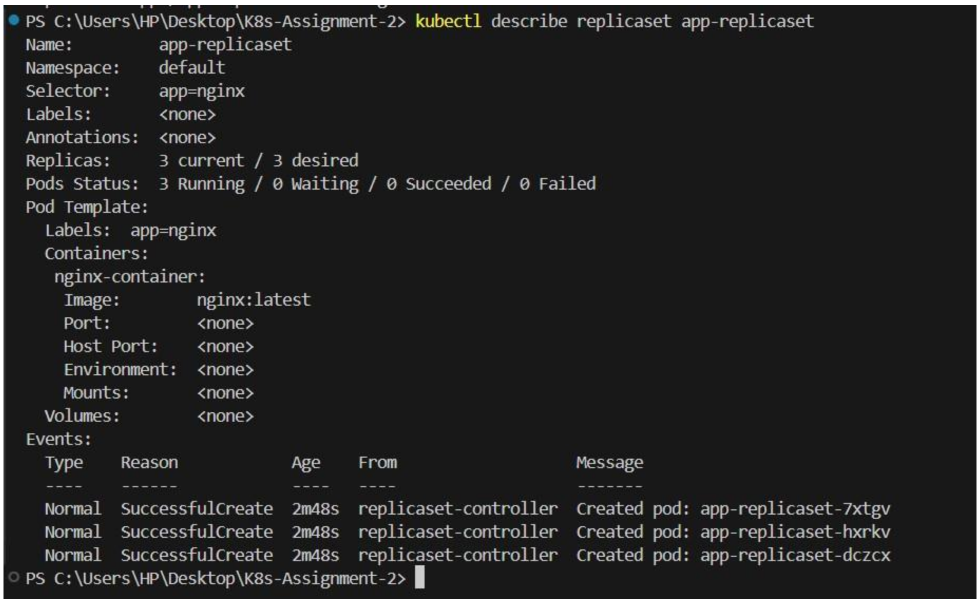
kubectl apply -f replicaset01.yaml



```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl apply -f replicaset01.yaml
>>
replicaset.apps/app-replicaset created
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl get pods
NAME                   READY   STATUS              RESTARTS   AGE
app-replicaset-7xtgv   0/1     ContainerCreating   0          4s
app-replicaset-dczcx   0/1     ContainerCreating   0          4s
app-replicaset-hxrkv   0/1     ContainerCreating   0          4s
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl get pods
NAME                   READY   STATUS    RESTARTS   AGE
app-replicaset-7xtgv   1/1     Running   0          17s
app-replicaset-dczcx   1/1     Running   0          17s
app-replicaset-hxrkv   1/1     Running   0          17s
PS C:\Users\HP\Desktop\K8s-Assignment-2>
```

Get the description of the replicaset by running the below command:

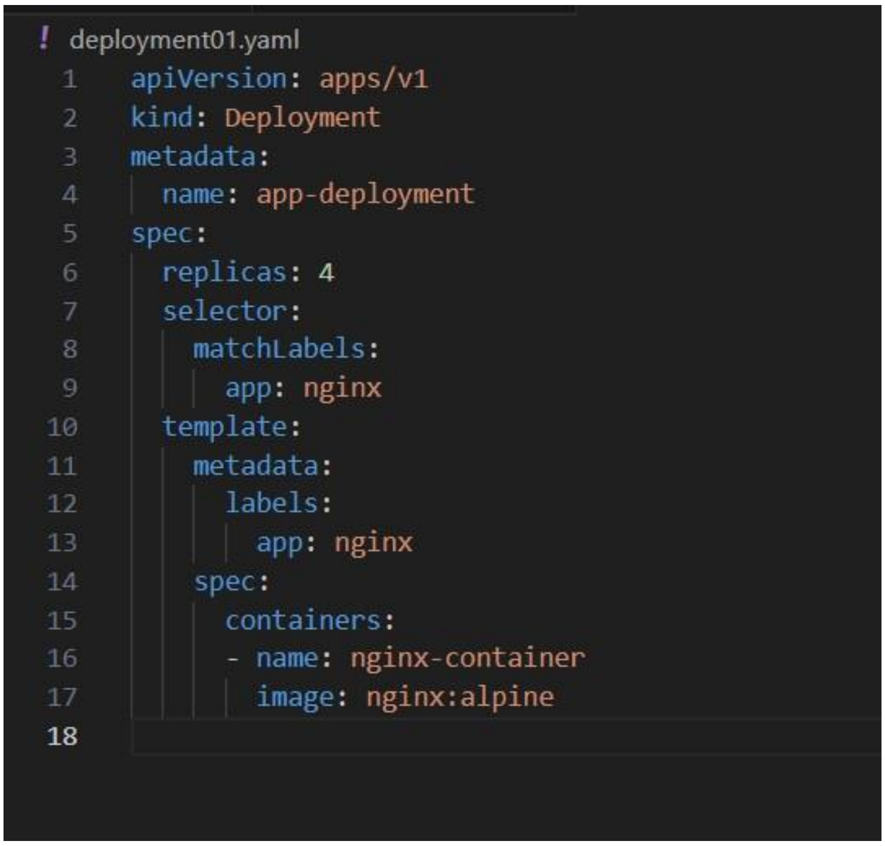
kubectl describe replicaset app-replicaset



```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl describe replicaset app-replicaset
Name:         app-replicaset
Namespace:    default
Selector:     app=nginx
Labels:       <none>
Annotations:  <none>
Replicas:     3 current / 3 desired
Pods Status:  3 Running / 0 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:  app=nginx
  Containers:
   nginx-container:
    Image:        nginx:latest
    Port:         <none>
    Host Port:    <none>
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Events:
  Type    Reason            Age   From                   Message
  ----    ------            ----  ----                   -------
  Normal  SuccessfulCreate  2m48s replicaset-controller  Created pod: app-replicaset-7xtgv
  Normal  SuccessfulCreate  2m48s replicaset-controller  Created pod: app-replicaset-hxrkv
  Normal  SuccessfulCreate  2m48s replicaset-controller  Created pod: app-replicaset-dczcx
PS C:\Users\HP\Desktop\K8s-Assignment-2>
```

Task 4: Creating a deployment with nginx:alpine image

Create a file named deployment01.yaml with the following content:

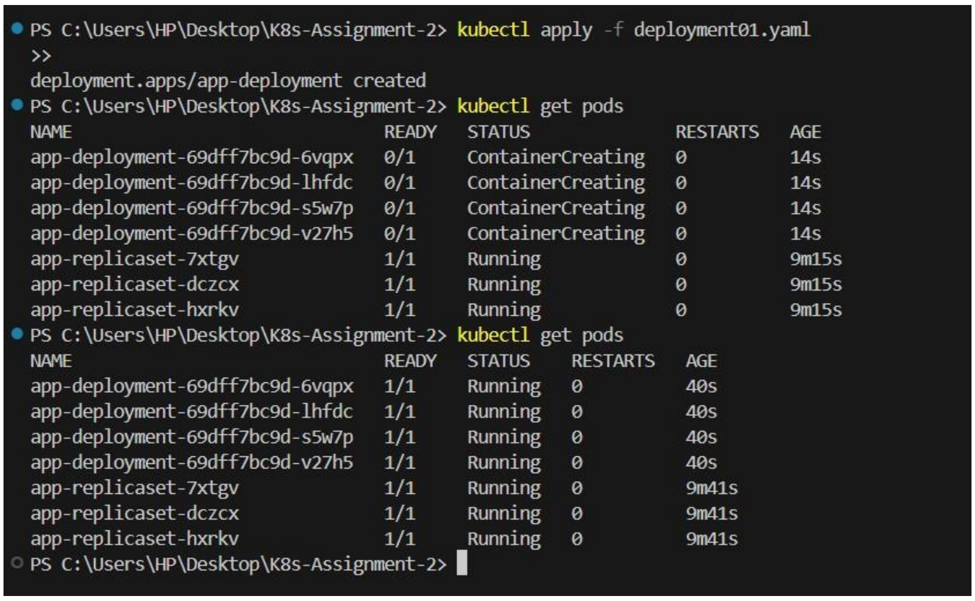


```
! deployment01.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: app-deployment
spec:
  replicas: 4
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx-container
        image: nginx:alpine

```

Apply the configuration:

kubectl apply -f deployment01.yaml

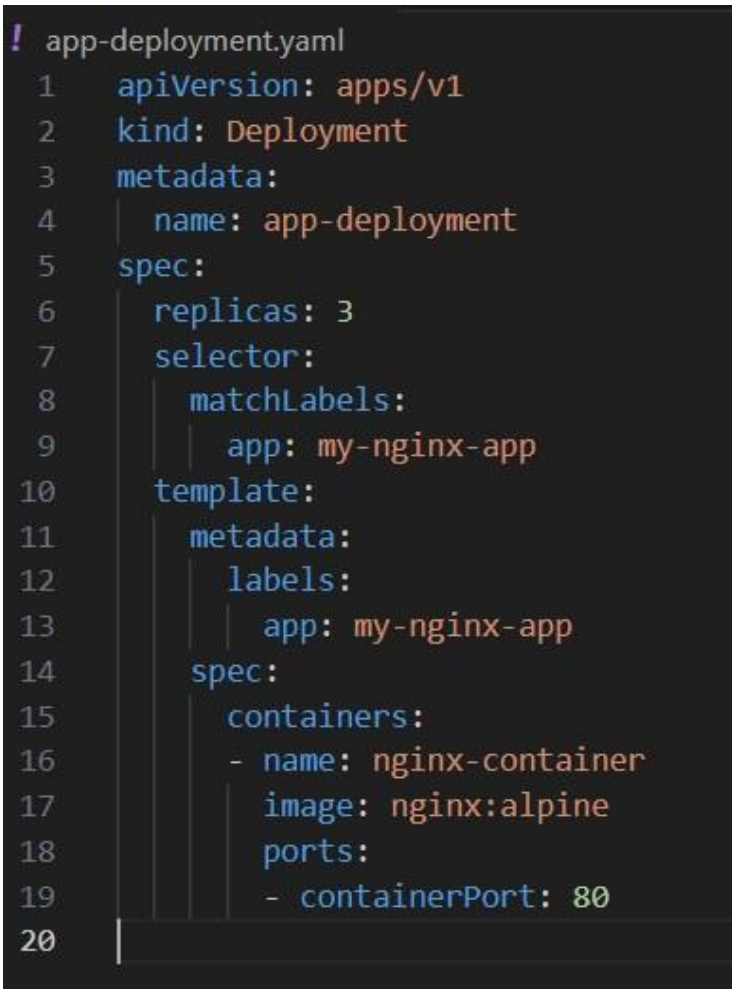


```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl apply -f deployment01.yaml
>>
deployment.apps/app-deployment created
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl get pods
NAME                              READY   STATUS              RESTARTS   AGE
app-deployment-69dff7bc9d-6vqpx   0/1     ContainerCreating   0          14s
app-deployment-69dff7bc9d-lhfdc   0/1     ContainerCreating   0          14s
app-deployment-69dff7bc9d-s5w7p   0/1     ContainerCreating   0          14s
app-deployment-69dff7bc9d-v27h5   0/1     ContainerCreating   0          14s
app-replicaset-7xtgv              1/1     Running             0          9m15s
app-replicaset-dczcx              1/1     Running             0          9m15s
app-replicaset-hxrkv              1/1     Running             0          9m15s
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl get pods
NAME                              READY   STATUS    RESTARTS   AGE
app-deployment-69dff7bc9d-6vqpx   1/1     Running   0          40s
app-deployment-69dff7bc9d-lhfdc   1/1     Running   0          40s
app-deployment-69dff7bc9d-s5w7p   1/1     Running   0          40s
app-deployment-69dff7bc9d-v27h5   1/1     Running   0          40s
app-replicaset-7xtgv              1/1     Running   0          9m41s
app-replicaset-dczcx              1/1     Running   0          9m41s
app-replicaset-hxrkv              1/1     Running   0          9m41s
PS C:\Users\HP\Desktop\K8s-Assignment-2>
```

Task 5: Rolling back to the previous version using the "app-deployment".

1. Deploy Initial Version:

   Save the following YAML as ‘app-deployment.yaml’ to deploy the initial version of the app:

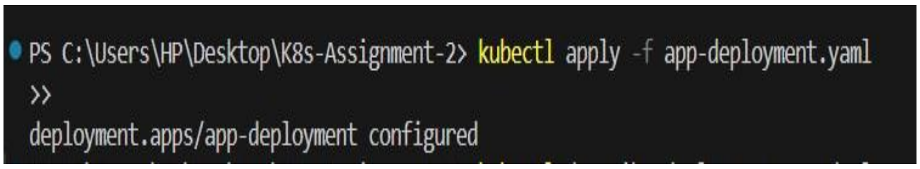


```yaml
! app-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: app-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-nginx-app
  template:
    metadata:
      labels:
        app: my-nginx-app
    spec:
      containers:
      - name: nginx-container
        image: nginx:alpine
        ports:
        - containerPort: 80

```

   Apply the deployment:

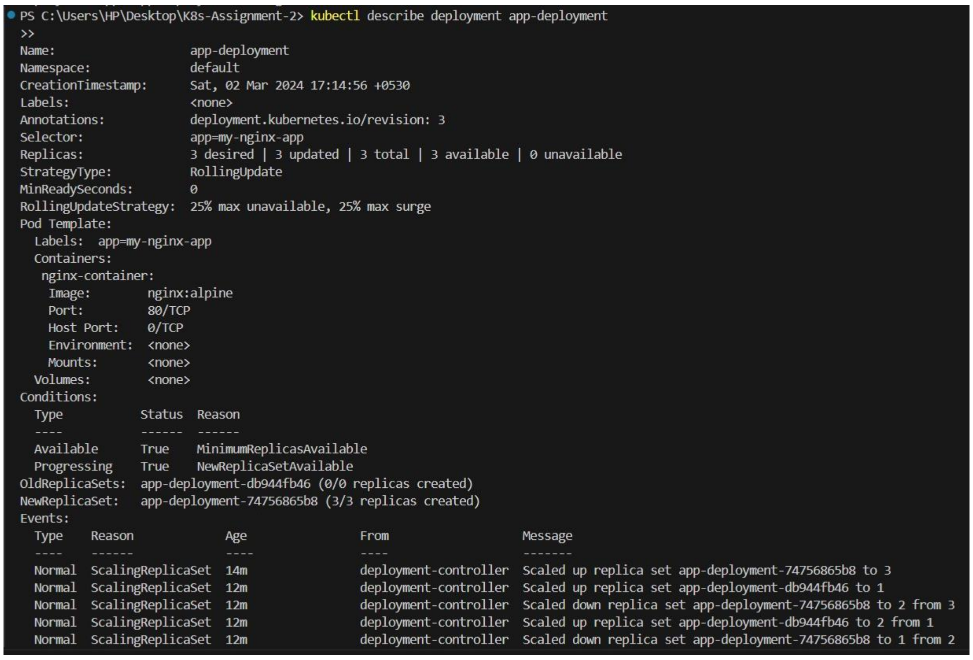
   kubectl apply -f app-deployment.yaml



```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl apply -f app-deployment.yaml
>>
deployment.apps/app-deployment configured
```

kubectl describe deployment app-deployment

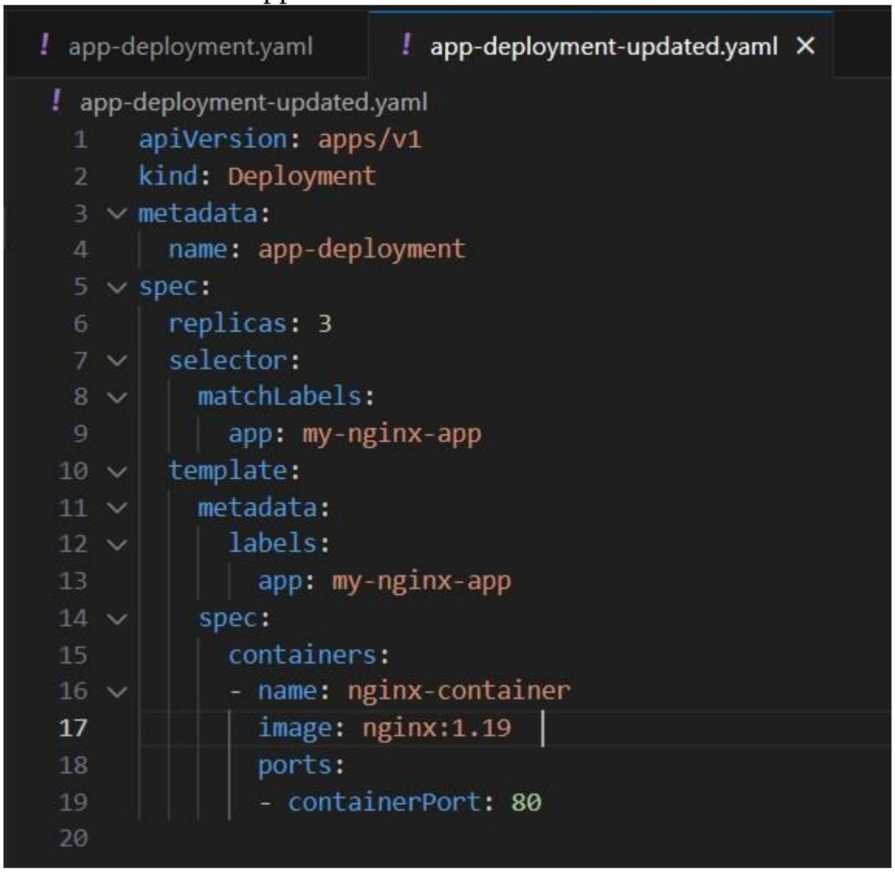
Description of the deployment as we can see that the image version is **nginx:alpine**.



```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl describe deployment app-deployment
>>
Name:                   app-deployment
Namespace:              default
CreationTimestamp:      Sat, 02 Mar 2024 17:14:56 +0530
Labels:                 <none>
Annotations:            deployment.kubernetes.io/revision: 3
Selector:               app=my-nginx-app
Replicas:               3 desired | 3 updated | 3 total | 3 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=my-nginx-app
  Containers:
   nginx-container:
    Image:        nginx:alpine
    Port:         80/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets:  app-deployment-db944fb46 (0/0 replicas created)
NewReplicaSet:   app-deployment-74756865b8 (3/3 replicas created)
Events:
  Type    Reason             Age                 From                   Message
  ----    ------             ----                ----                   -------
  Normal  ScalingReplicaSet  14m                 deployment-controller  Scaled up replica set app-deployment-74756865b8 to 3
  Normal  ScalingReplicaSet  12m                 deployment-controller  Scaled up replica set app-deployment-db944fb46 to 1
  Normal  ScalingReplicaSet  12m                 deployment-controller  Scaled down replica set app-deployment-74756865b8 to 2 from 3
  Normal  ScalingReplicaSet  12m                 deployment-controller  Scaled up replica set app-deployment-db944fb46 to 2 from 1
  Normal  ScalingReplicaSet  12m                 deployment-controller  Scaled down replica set app-deployment-74756865b8 to 1 from 2
```

2. Update the App to a New Version:

   Save the following updated YAML as ‘app-deployment-updated.yaml’ to deploy a new version of the app:

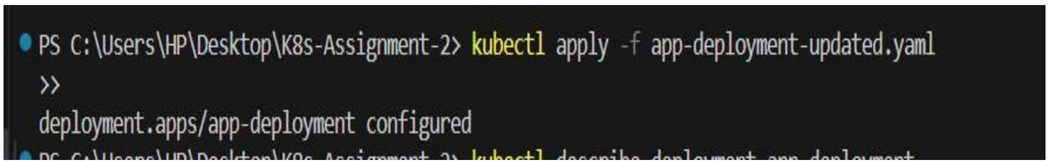


```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: app-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-nginx-app
  template:
    metadata:
      labels:
        app: my-nginx-app
    spec:
      containers:
      - name: nginx-container
        image: nginx:1.19
        ports:
        - containerPort: 80

```

Apply the updated deployment:

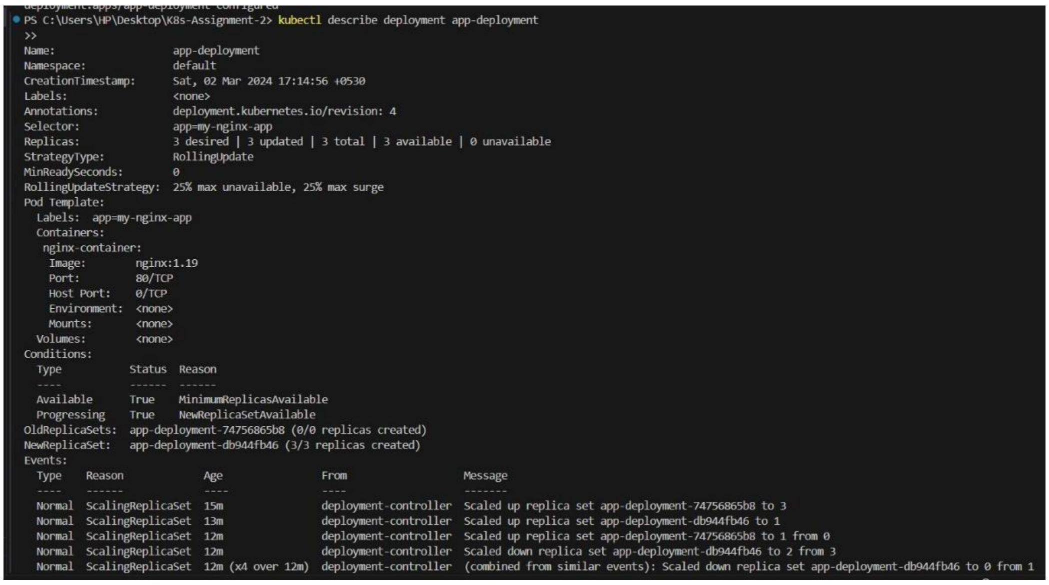
kubectl apply -f app-deployment-updated.yaml



```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl apply -f app-deployment-updated.yaml
>>
deployment.apps/app-deployment configured
```

kubectl describe deployment app-deployment

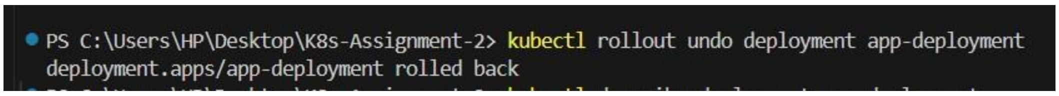
Description of the deployment as we can see that the image version is **nginx:1.19**.



```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl describe deployment app-deployment
>>
Name:                   app-deployment
Namespace:              default
CreationTimestamp:      Sat, 02 Mar 2024 17:14:56 +0530
Labels:                 <none>
Annotations:            deployment.kubernetes.io/revision: 4
Selector:               app=my-nginx-app
Replicas:               3 desired | 3 updated | 3 total | 3 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=my-nginx-app
  Containers:
   nginx-container:
    Image:        nginx:1.19
    Port:         80/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets:  app-deployment-74756865b8 (0/0 replicas created)
NewReplicaSet:   app-deployment-db944fb46 (3/3 replicas created)
Events:
  Type    Reason             Age                From                   Message
  ----    ------             ----               ----                   -------
  Normal  ScalingReplicaSet  15m                deployment-controller  Scaled up replica set app-deployment-74756865b8 to 3
  Normal  ScalingReplicaSet  13m                deployment-controller  Scaled up replica set app-deployment-db944fb46 to 1
  Normal  ScalingReplicaSet  12m                deployment-controller  Scaled up replica set app-deployment-74756865b8 to 1 from 0
  Normal  ScalingReplicaSet  12m                deployment-controller  Scaled down replica set app-deployment-db944fb46 to 2 from 3
  Normal  ScalingReplicaSet  12m (x4 over 12m)  deployment-controller  (combined from similar events): Scaled down replica set app-deployment-db944fb46 to 0 from 1
```

3. Roll Back to a Previous Version:

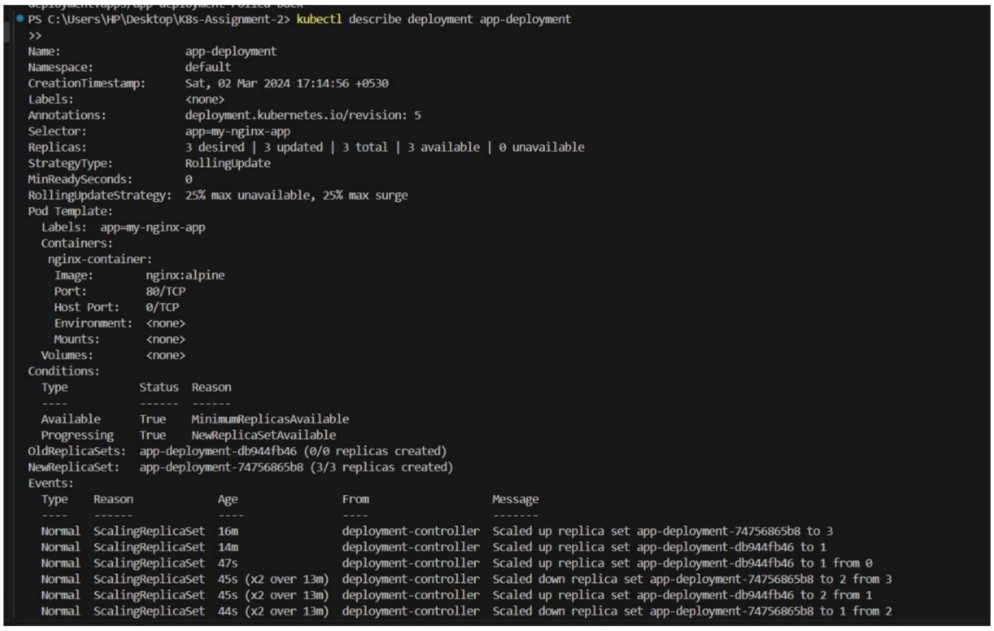
   Run **kubectl rollout undo deployment app-deployment**



```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl rollout undo deployment app-deployment
deployment.apps/app-deployment rolled back
```

Now, let’s see the description of the deployment.

The image version is **nginx:alpine** which was the previous version. So, we have successfully rolled back to the previous version.

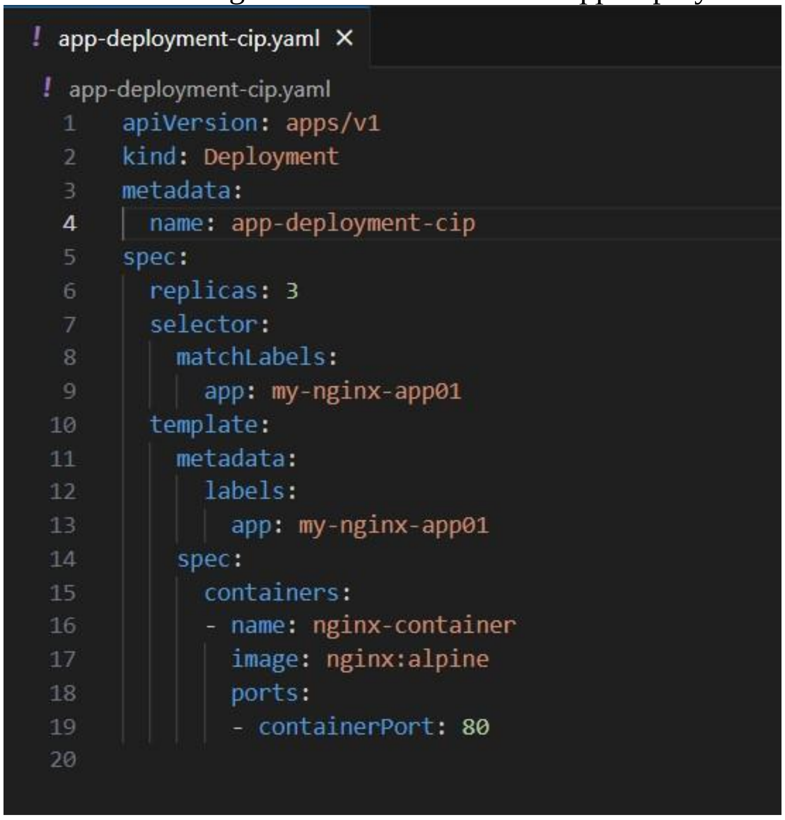


```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl describe deployment app-deployment
>>
Name:                   app-deployment
Namespace:              default
CreationTimestamp:      Sat, 02 Mar 2024 17:14:56 +0530
Labels:                 <none>
Annotations:            deployment.kubernetes.io/revision: 5
Selector:               app=my-nginx-app
Replicas:               3 desired | 3 updated | 3 total | 3 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=my-nginx-app
  Containers:
   nginx-container:
    Image:        nginx:alpine
    Port:         80/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets:  app-deployment-db944fb46 (0/0 replicas created)
NewReplicaSet:   app-deployment-74756865b8 (3/3 replicas created)
Events:
  Type    Reason             Age                From                   Message
  ----    ------             ----               ----                   -------
  Normal  ScalingReplicaSet  16m                deployment-controller  Scaled up replica set app-deployment-74756865b8 to 3
  Normal  ScalingReplicaSet  14m                deployment-controller  Scaled up replica set app-deployment-db944fb46 to 1
  Normal  ScalingReplicaSet  47s                deployment-controller  Scaled up replica set app-deployment-db944fb46 to 1 from 0
  Normal  ScalingReplicaSet  45s (x2 over 13m)  deployment-controller  Scaled down replica set app-deployment-74756865b8 to 2 from 3
  Normal  ScalingReplicaSet  45s (x2 over 13m)  deployment-controller  Scaled up replica set app-deployment-db944fb46 to 2 from 1
  Normal  ScalingReplicaSet  44s (x2 over 13m)  deployment-controller  Scaled down replica set app-deployment-74756865b8 to 1 from 2
```

Task 6: Creating a ClusterIP service and a LoadBalancer service.

1. Create a Deployment:

   Save the following YAML to a file named ‘app-deployment-cip.yaml’:



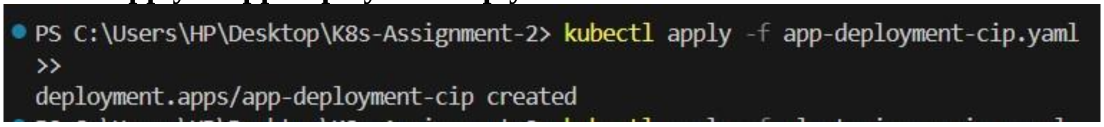
```
! app-deployment-cip.yaml ×
! app-deployment-cip.yaml
```

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: app-deployment-cip
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-nginx-app01
  template:
    metadata:
      labels:
        app: my-nginx-app01
    spec:
      containers:
      - name: nginx-container
        image: nginx:alpine
        ports:
        - containerPort: 80

```

Apply the deployment:

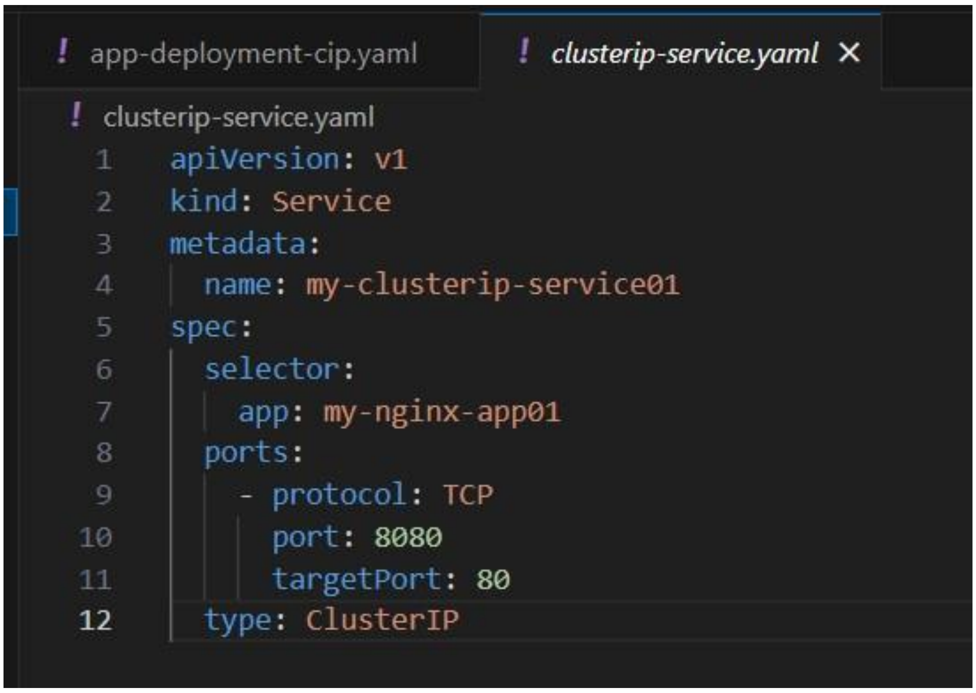
kubectl apply -f app-deployment-cip.yaml



```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl apply -f app-deployment-cip.yaml
>>
deployment.apps/app-deployment-cip created
```

2. Create a ClusterIP Service:

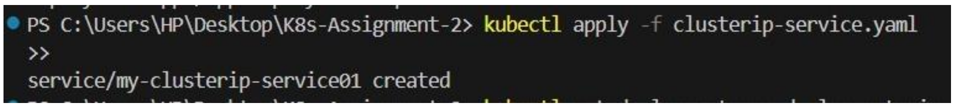
   Save the following YAML to a file named ‘clusterip-service.yaml’:



```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-clusterip-service01
spec:
  selector:
    app: my-nginx-app01
  ports:
    - protocol: TCP
      port: 8080
      targetPort: 80
  type: ClusterIP
```

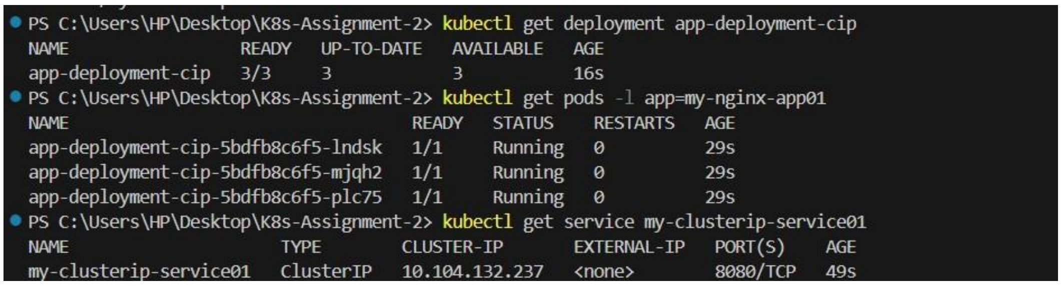
Apply the ClusterIP service:

kubectl apply -f clusterip-service.yaml



```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl apply -f clusterip-service.yaml
>>
service/my-clusterip-service01 created
```

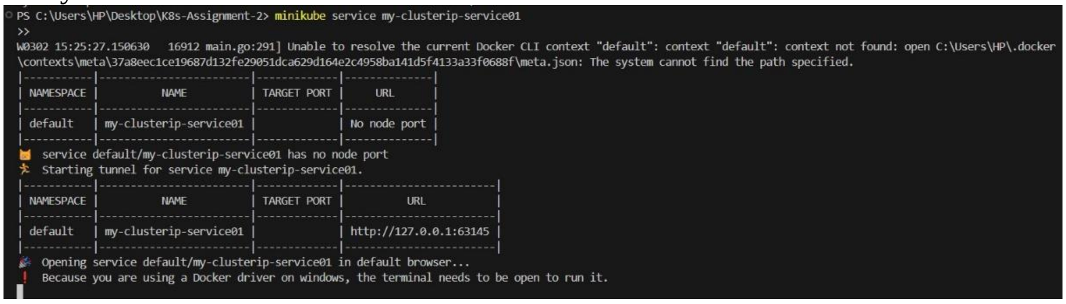
Now, let’s check the status and details of the above deployment and service



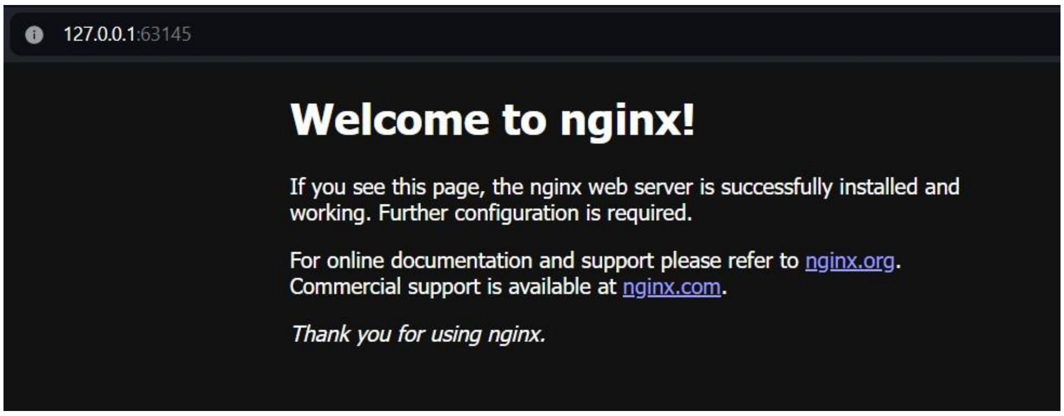
```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl get deployment app-deployment-cip
NAME                 READY   UP-TO-DATE   AVAILABLE   AGE
app-deployment-cip   3/3     3            3           16s
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl get pods -l app=my-nginx-app01
NAME                                  READY   STATUS    RESTARTS   AGE
app-deployment-cip-5bdfb8c6f5-lndsk   1/1     Running   0          29s
app-deployment-cip-5bdfb8c6f5-mjqh2   1/1     Running   0          29s
app-deployment-cip-5bdfb8c6f5-plc75   1/1     Running   0          29s
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl get service my-clusterip-service01
NAME                     TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)    AGE
my-clusterip-service01   ClusterIP   10.104.132.237   <none>        8080/TCP   49s
```

As we can see that there is no external-IP provided.

I can expose the service temporarily by running **minikube service my-clusterip-service01**. This command opens the service in the default browser and returns a URL that you can use to access the service.



```
PS C:\Users\HP\Desktop\K8s-Assignment-2> minikube service my-clusterip-service01
>>
W0302 15:25:27.150630   16912 main.go:291] Unable to resolve the current Docker CLI context "default": context "default": context not found: open C:\Users\HP\.docker
\contexts\meta\37a8eec1ce19687d132fe29051dca629d164e2c4958ba141d5f4133a33f0688f\meta.json: The system cannot find the path specified.
|-----------|-----------------------|-------------|--------------|
| NAMESPACE |         NAME          | TARGET PORT |     URL      |
|-----------|-----------------------|-------------|--------------|
| default   | my-clusterip-service01 |            | No node port |
|-----------|-----------------------|-------------|--------------|
😿  service default/my-clusterip-service01 has no node port
🏃  Starting tunnel for service my-clusterip-service01.
|-----------|-----------------------|-------------|------------------------|
| NAMESPACE |         NAME          | TARGET PORT |          URL           |
|-----------|-----------------------|-------------|------------------------|
| default   | my-clusterip-service01 |            | http://127.0.0.1:63145 |
|-----------|-----------------------|-------------|------------------------|
🎉  Opening service default/my-clusterip-service01 in default browser...
❗  Because you are using a Docker driver on windows, the terminal needs to be open to run it.
```



127.0.0.1:63145

Welcome to nginx!

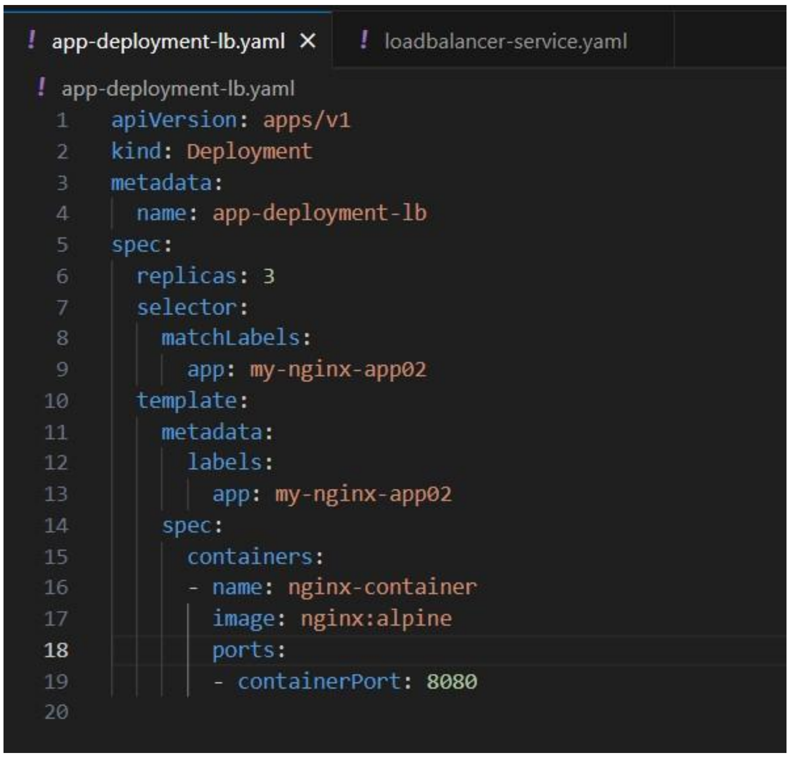
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

3. Create another Deployment:

   Save the following YAML to a file named ‘app-deployment-lb.yaml’:

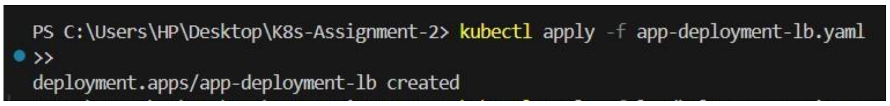


```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: app-deployment-lb
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-nginx-app02
  template:
    metadata:
      labels:
        app: my-nginx-app02
    spec:
      containers:
      - name: nginx-container
        image: nginx:alpine
        ports:
        - containerPort: 8080

```

Apply the deployment:

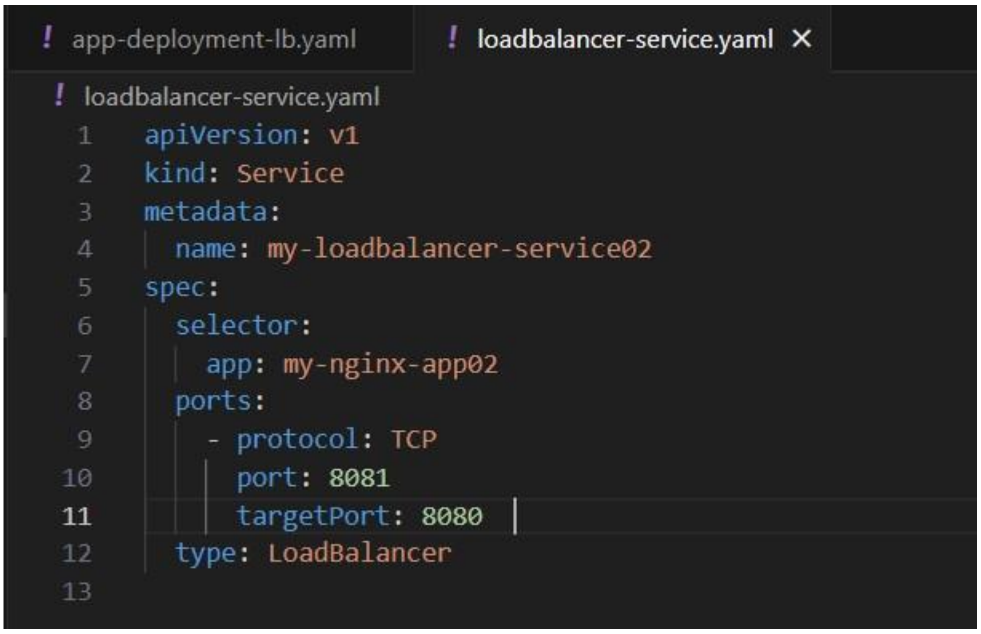
kubectl apply -f app-deployment-lb.yaml



```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl apply -f app-deployment-lb.yaml
>>
deployment.apps/app-deployment-lb created
```

4. Create a LoadBalancer Service:

   Save the following YAML to a file named ‘loadbalancer-service.yaml’:

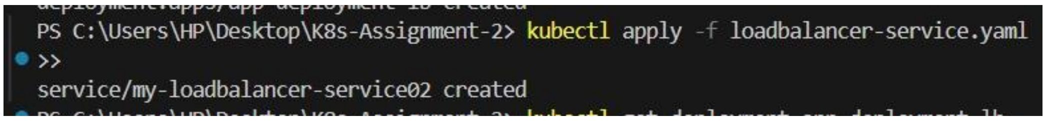


```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-loadbalancer-service02
spec:
  selector:
    app: my-nginx-app02
  ports:
    - protocol: TCP
      port: 8081
      targetPort: 8080
  type: LoadBalancer

```

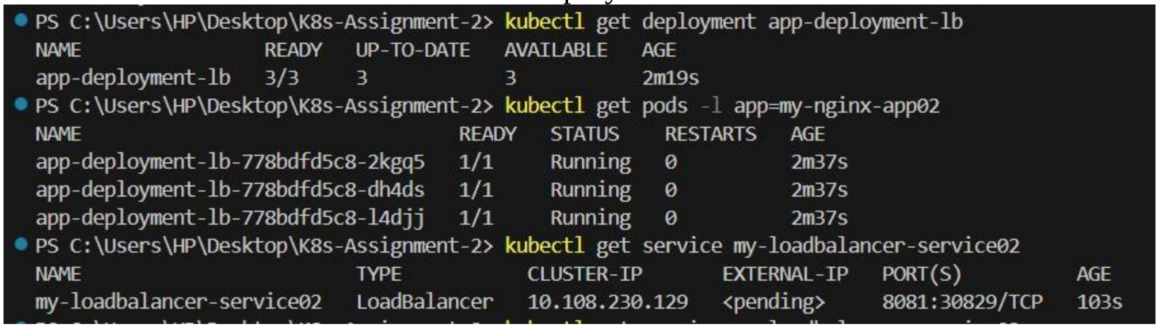
Apply the LoadBalancer service:

kubectl apply -f loadbalancer-service.yaml



```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl apply -f loadbalancer-service.yaml
>>
service/my-loadbalancer-service02 created
```

Check the status and details of the above deployment and service:



```
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl get deployment app-deployment-lb
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
app-deployment-lb   3/3     3            3           2m19s
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl get pods -l app=my-nginx-app02
NAME                                 READY   STATUS    RESTARTS   AGE
app-deployment-lb-778bdfd5c8-2kgq5   1/1     Running   0          2m37s
app-deployment-lb-778bdfd5c8-dh4ds   1/1     Running   0          2m37s
app-deployment-lb-778bdfd5c8-l4djj   1/1     Running   0          2m37s
PS C:\Users\HP\Desktop\K8s-Assignment-2> kubectl get service my-loadbalancer-service02
NAME                        TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)          AGE
my-loadbalancer-service02   LoadBalancer   10.108.230.129   <pending>     8081:30829/TCP   103s
```

It provides external-IP. It is showing pending because LoadBalancer services are typically used in cloud environments, we need to run our Kubernetes cluster in a cloud provider environment like AWS, Azure, GCP, etc.

Summary:

- **ClusterIP:**
    - **Internal communication within the cluster.**
    - **Provides an internal IP address.**
    - **Suitable for microservices talking to each other.**

Whereas,

- **LoadBalancer:**
    - **External access from outside the cluster.**
    - **Provides an external IP address or DNS.**
    - **Suitable for applications that need to be accessed from the internet.**