MASTER COMPONENTS

kube-apiserver

The Kubernetes cluster is exposed via API and made available outside the cluster
through the kube-apiserver component.
The kube-apiserver is the only component that all other master and
worker components can directly communicate directly with.
Because of this, it serves as the interface for all cluster communications

ETCD

Kubernetes uses etcd to store all its data – its configuration data, its state, and its metadata.
Kubernetes is a distributed system, so it needs a distributed data store like etcd.
etcd lets any of the nodes in the Kubernetes cluster read and write data

kube-controller

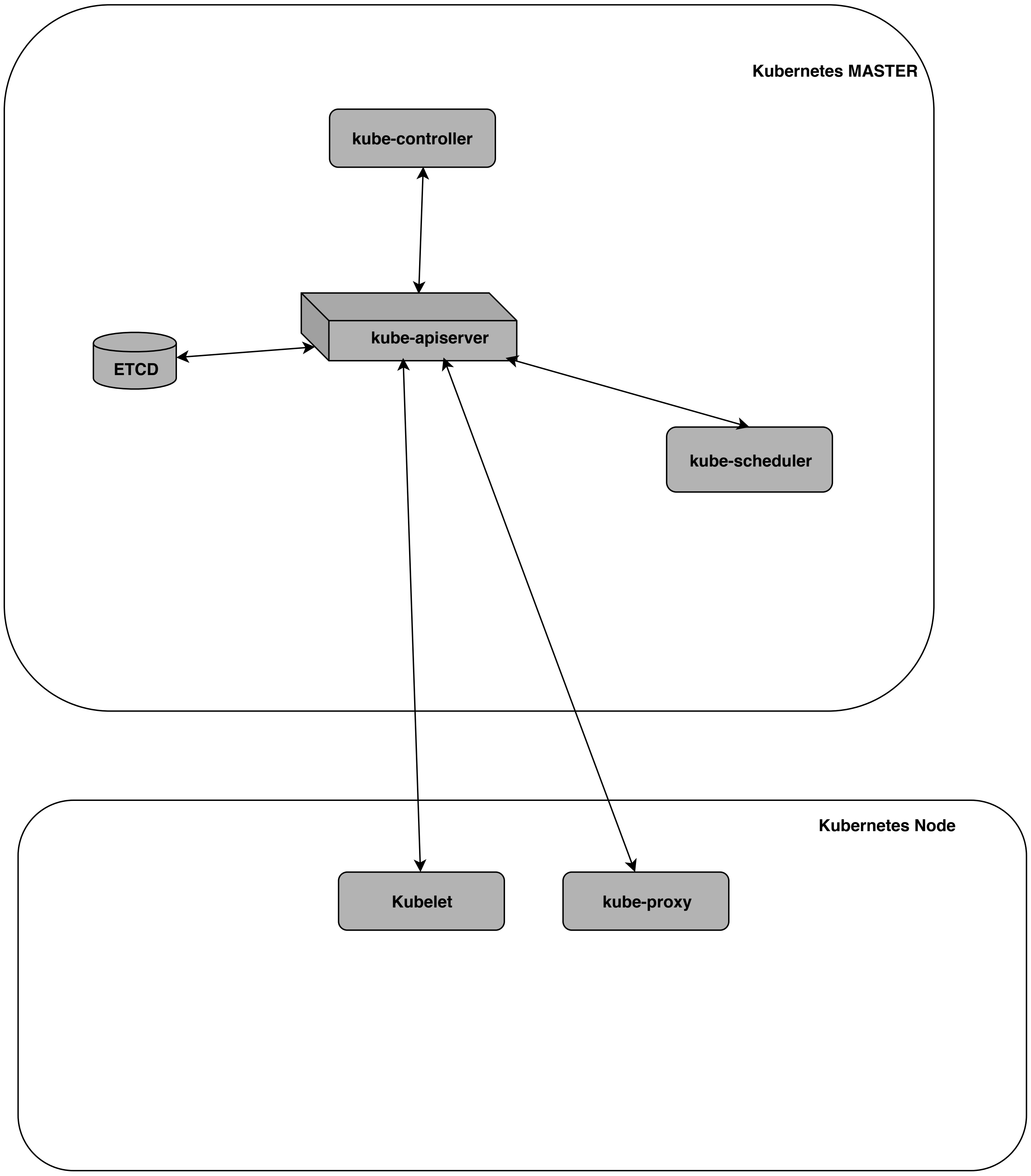
The Kubernetes controller talks to API Server to create, delete and update the resources they manage so that
the cluster gets back to desired state.

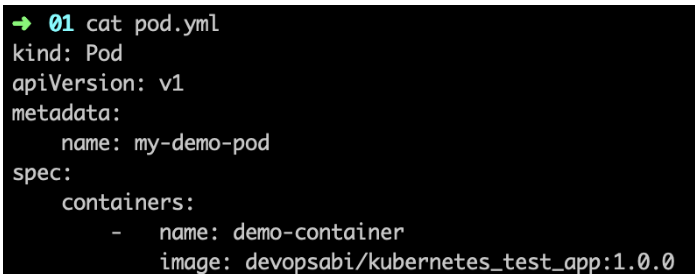
kube-scheduler

This is a component on the master that watches newly created pods that have no node assigned,
and selects a node for them to run on.

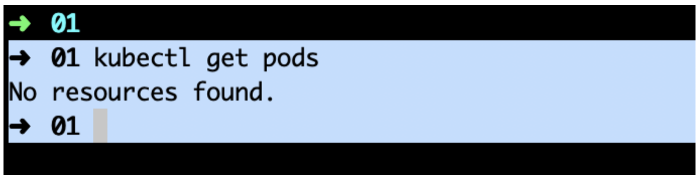
NODE COMPONENTS

Kubelet	The kubelet is responsible for maintaining a set of pods, which are composed of one or more containers, on a local system. Within a Kubernetes cluster, the kubelet functions as a local agent that watches for pod specs via the Kubernetes API server.
kube-proxy	kube-proxy is a network proxy that runs on each node in your cluster, implementing part of the Kubernetes Service. concept. kube-proxy maintains network rules on nodes. These network rules allow network communication to your Pods from network sessions inside or outside of your cluster.

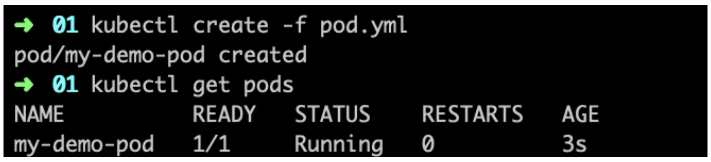




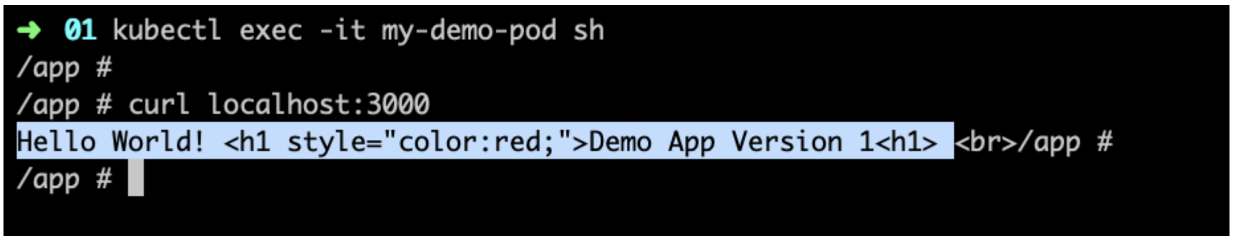
```
➜ 01 cat pod.yml
kind: Pod
apiVersion: v1
metadata:
    name: my-demo-pod
spec:
    containers:
        -   name: demo-container
            image: devopsabi/kubernetes_test_app:1.0.0
```



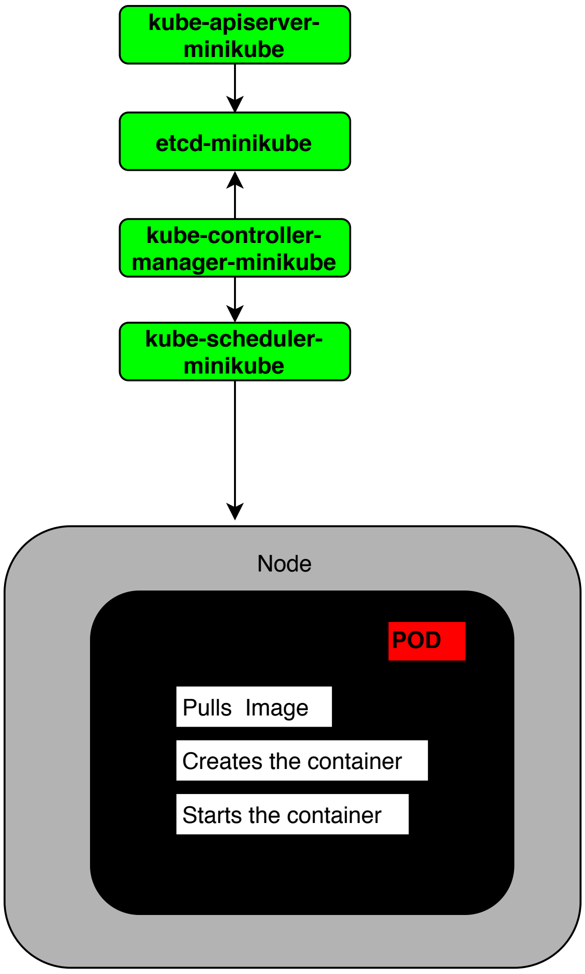
```
➜ 01
➜ 01 kubectl get pods
No resources found.
➜ 01
```

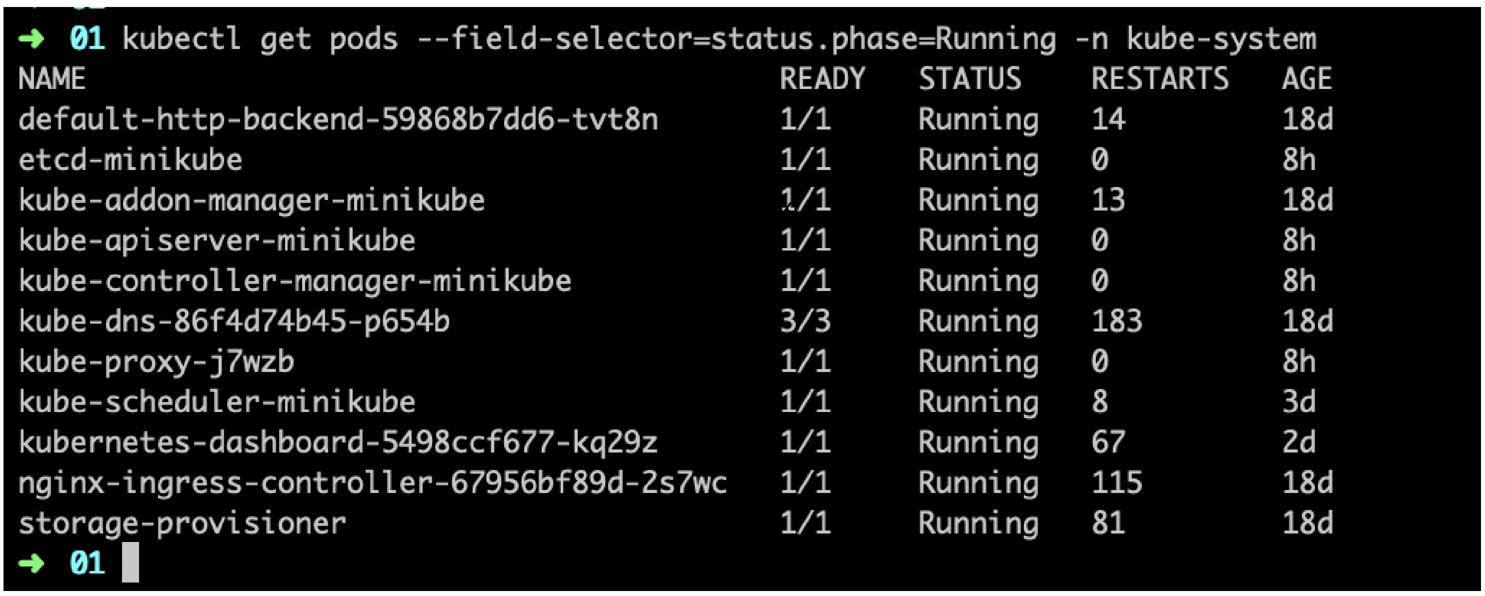


```
➜ 01 kubectl create -f pod.yml
pod/my-demo-pod created
➜ 01 kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
my-demo-pod   1/1     Running   0          3s
```

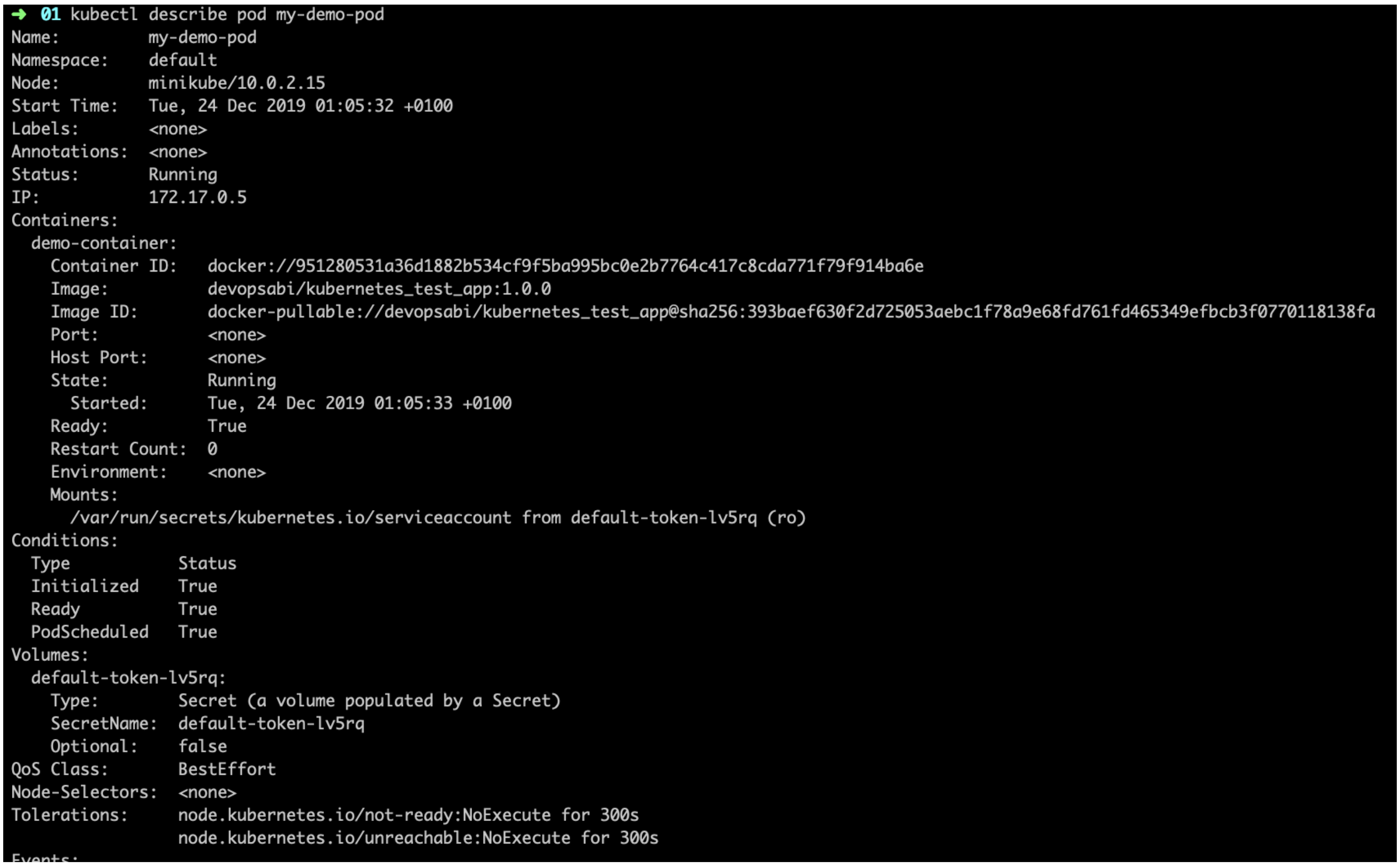


```
➜ 01 kubectl exec -it my-demo-pod sh
/app #
/app # curl localhost:3000
Hello World! <h1 style="color:red;">Demo App Version 1<h1> <br>/app #
/app #
```

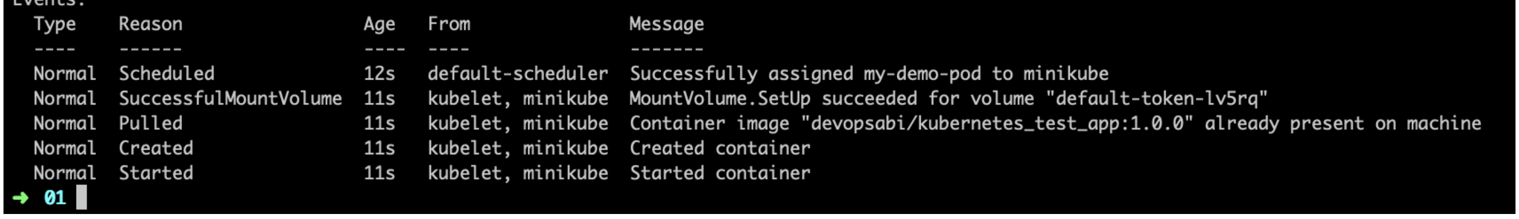




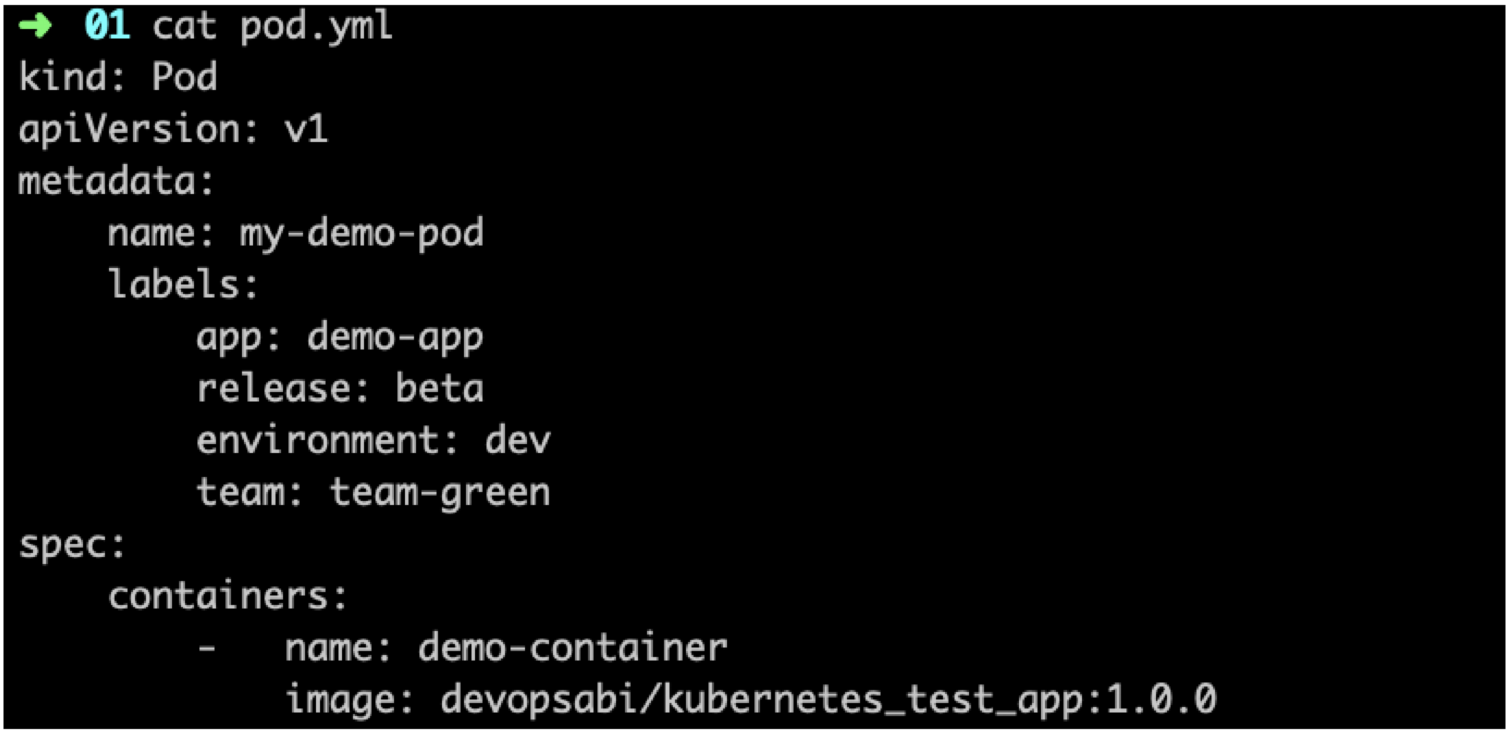
```
➜ 01 kubectl get pods --field-selector=status.phase=Running -n kube-system
NAME                                        READY   STATUS    RESTARTS   AGE
default-http-backend-59868b7dd6-tvt8n       1/1     Running   14         18d
etcd-minikube                               1/1     Running   0          8h
kube-addon-manager-minikube                 1/1     Running   13         18d
kube-apiserver-minikube                     1/1     Running   0          8h
kube-controller-manager-minikube            1/1     Running   0          8h
kube-dns-86f4d74b45-p654b                   3/3     Running   183        18d
kube-proxy-j7wzb                            1/1     Running   0          8h
kube-scheduler-minikube                     1/1     Running   8          3d
kubernetes-dashboard-5498ccf677-kq29z       1/1     Running   67         2d
nginx-ingress-controller-67956bf89d-2s7wc   1/1     Running   115        18d
storage-provisioner                         1/1     Running   81         18d
➜ 01
```



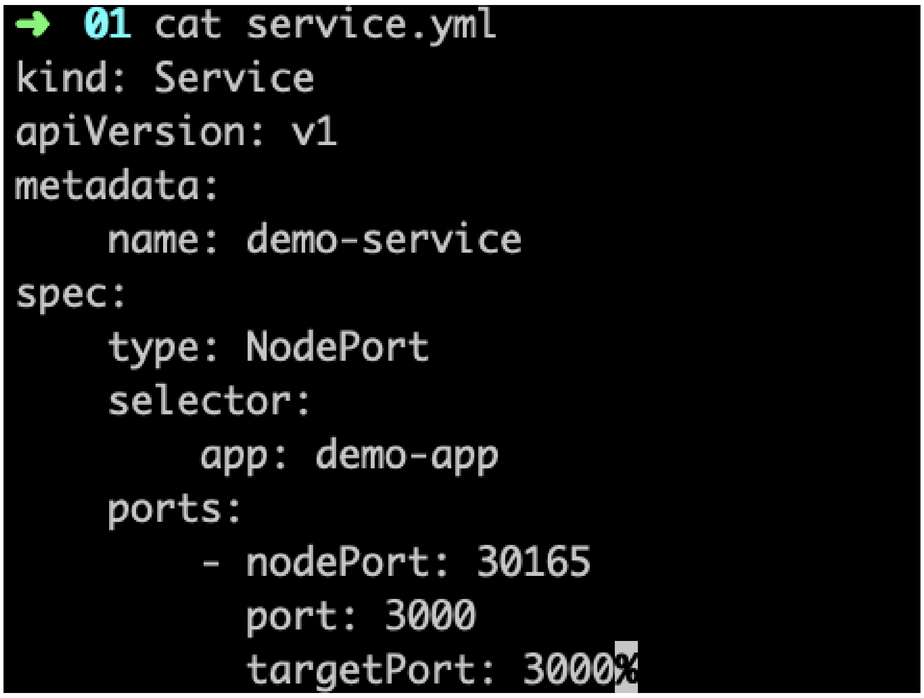
```
➜ 01 kubectl describe pod my-demo-pod
Name:           my-demo-pod
Namespace:      default
Node:           minikube/10.0.2.15
Start Time:     Tue, 24 Dec 2019 01:05:32 +0100
Labels:         <none>
Annotations:    <none>
Status:         Running
IP:             172.17.0.5
Containers:
  demo-container:
    Container ID:   docker://951280531a36d1882b534cf9f5ba995bc0e2b7764c417c8cda771f79f914ba6e
    Image:          devopsabi/kubernetes_test_app:1.0.0
    Image ID:       docker-pullable://devopsabi/kubernetes_test_app@sha256:393baef630f2d725053aebc1f78a9e68fd761fd465349efbcb3f0770118138fa
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Tue, 24 Dec 2019 01:05:33 +0100
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-lv5rq (ro)
Conditions:
  Type           Status
  Initialized    True
  Ready          True
  PodScheduled   True
Volumes:
  default-token-lv5rq:
    Type:        Secret (a volume populated by a Secret)
    SecretName:  default-token-lv5rq
    Optional:    false
QoS Class:       BestEffort
Node-Selectors:  <none>
Tolerations:     node.kubernetes.io/not-ready:NoExecute for 300s
                 node.kubernetes.io/unreachable:NoExecute for 300s
Events:
```



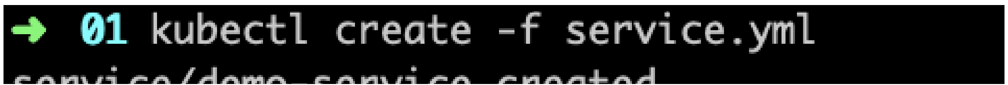
```
Events:
  Type    Reason                 Age   From                Message
  ----    ------                 ----  ----                -------
  Normal  Scheduled              12s   default-scheduler   Successfully assigned my-demo-pod to minikube
  Normal  SuccessfulMountVolume  11s   kubelet, minikube   MountVolume.SetUp succeeded for volume "default-token-lv5rq"
  Normal  Pulled                 11s   kubelet, minikube   Container image "devopsabi/kubernetes_test_app:1.0.0" already present on machine
  Normal  Created                11s   kubelet, minikube   Created container
  Normal  Started                11s   kubelet, minikube   Started container
➜  01
```



```
➜  01 cat pod.yml
kind: Pod
apiVersion: v1
metadata:
    name: my-demo-pod
    labels:
        app: demo-app
        release: beta
        environment: dev
        team: team-green
spec:
    containers:
        -   name: demo-container
            image: devopsabi/kubernetes_test_app:1.0.0
```

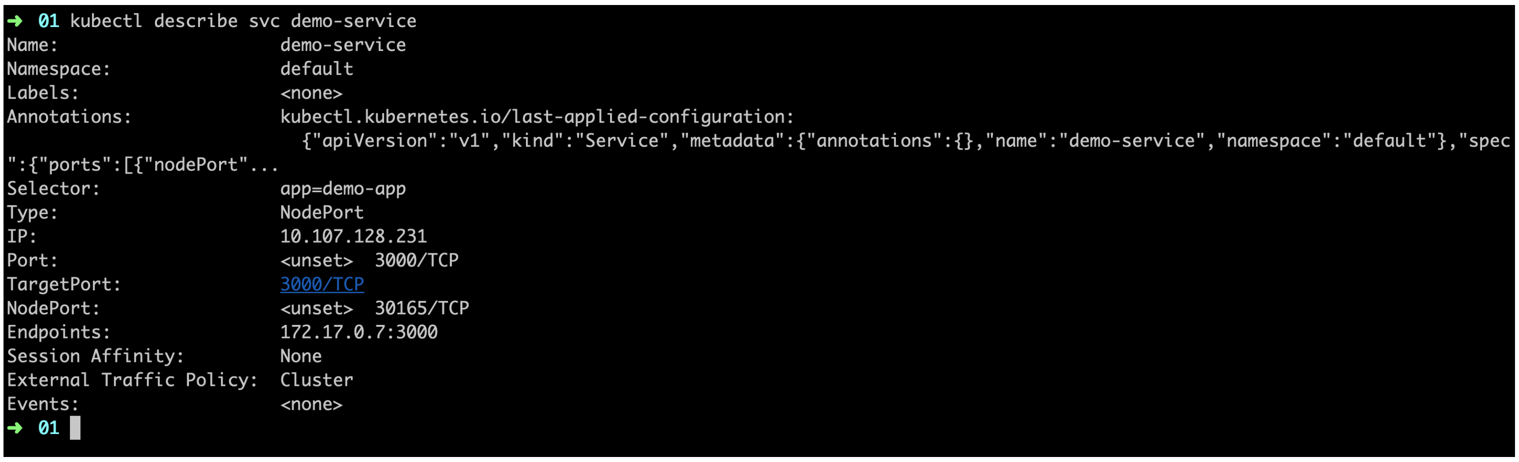


```
➜  01 cat service.yml
kind: Service
apiVersion: v1
metadata:
    name: demo-service
spec:
    type: NodePort
    selector:
        app: demo-app
    ports:
        - nodePort: 30165
          port: 3000
          targetPort: 3000%
```

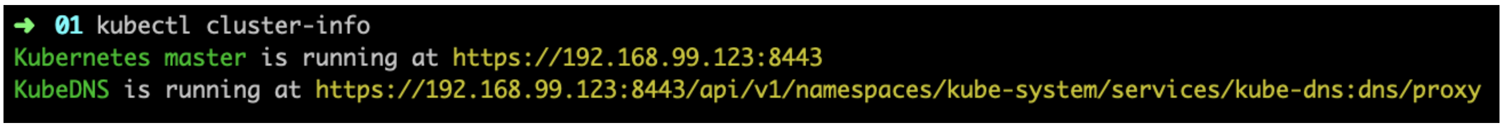


```
➜  01 kubectl create -f service.yml
service/demo-service created
```

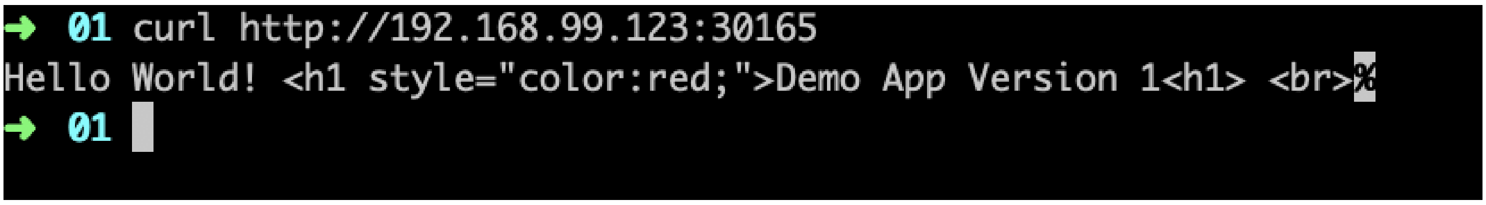
```
service/demo-service created
```



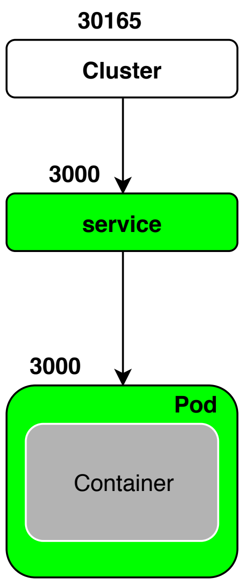
```
➜ 01 kubectl describe svc demo-service
Name:                     demo-service
Namespace:                default
Labels:                   <none>
Annotations:              kubectl.kubernetes.io/last-applied-configuration:
                            {"apiVersion":"v1","kind":"Service","metadata":{"annotations":{},"name":"demo-service","namespace":"default"},"spec
":{"ports":[{"nodePort"...
Selector:                 app=demo-app
Type:                     NodePort
IP:                       10.107.128.231
Port:                     <unset>  3000/TCP
TargetPort:               3000/TCP
NodePort:                 <unset>  30165/TCP
Endpoints:                172.17.0.7:3000
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
➜ 01
```

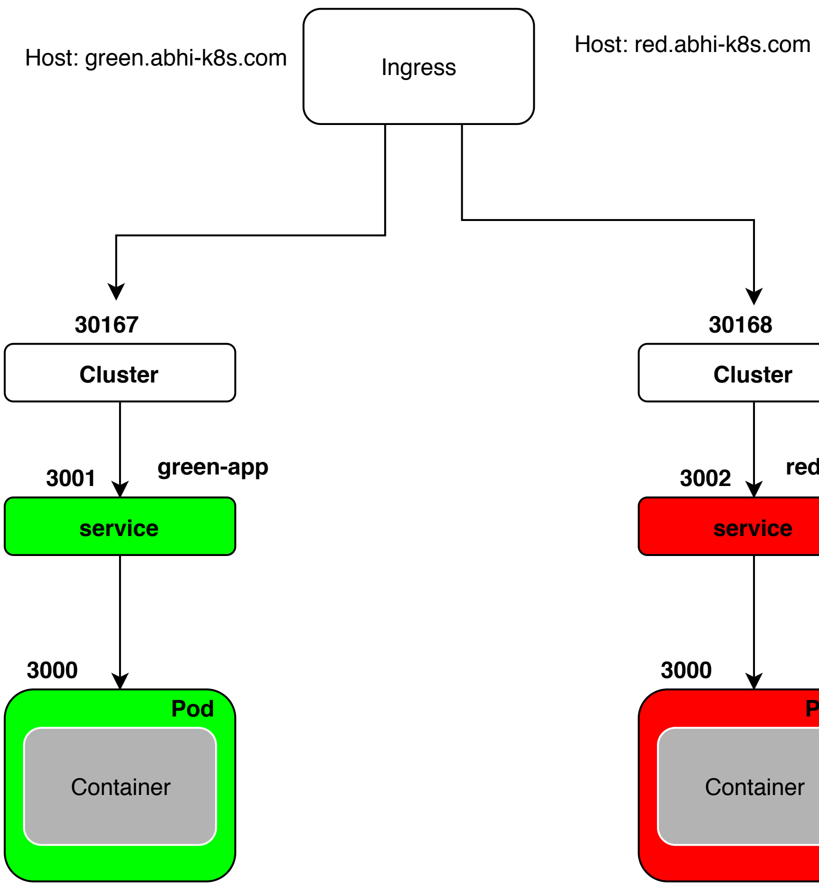


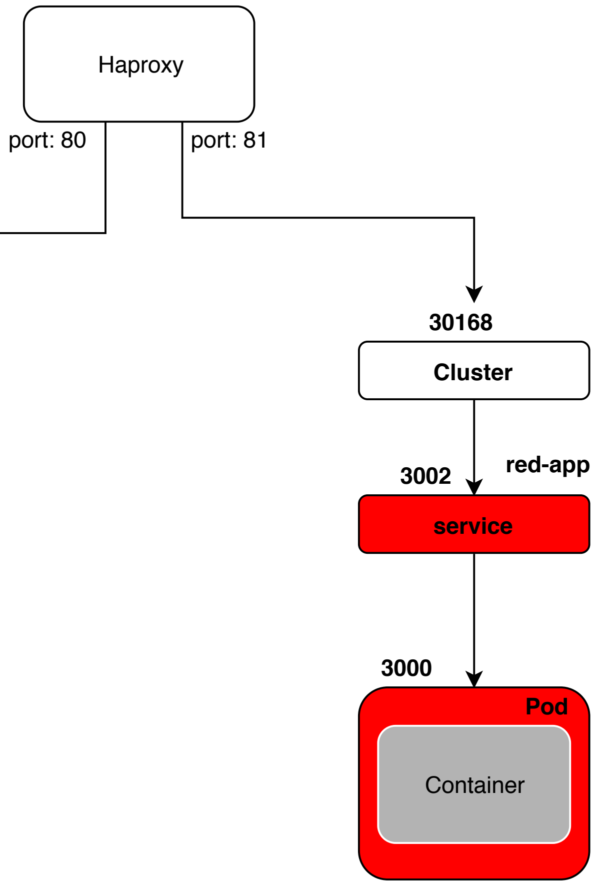
```
➜ 01 kubectl cluster-info
Kubernetes master is running at https://192.168.99.123:8443
KubeDNS is running at https://192.168.99.123:8443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
```



```
➜ 01 curl http://192.168.99.123:30165
Hello World! <h1 style="color:red;">Demo App Version 1<h1> <br>%
➜ 01
```




Host: green.abhi-k8s.com
Ingress
Host: red.abhi-k8s.com
30167
Cluster
3001
green-app
service
3000
Pod
Container
30168
Cluster
3002
red-app
service
3000
Pod
Container


Haproxy
port: 80
port: 81
30167
Cluster
3001
green-app
service
3000
Pod
Container
30168
Cluster
3002
red-app
service
3000
Pod
Container