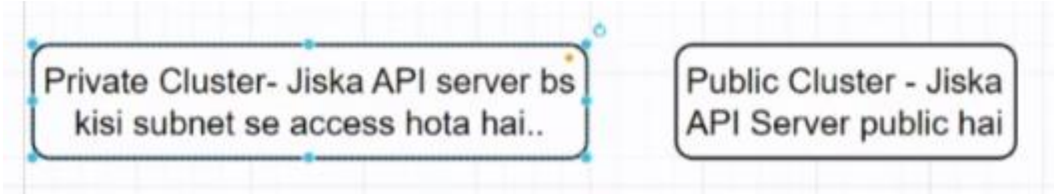
26 Oct 2024

NOTE : 1) Difference between private and public cluster.

2) Difference between kubectl create and kubectl apply command even they almost do the same work = create command creates the new resource as one time creation but apply command creates and updates the already created resources.



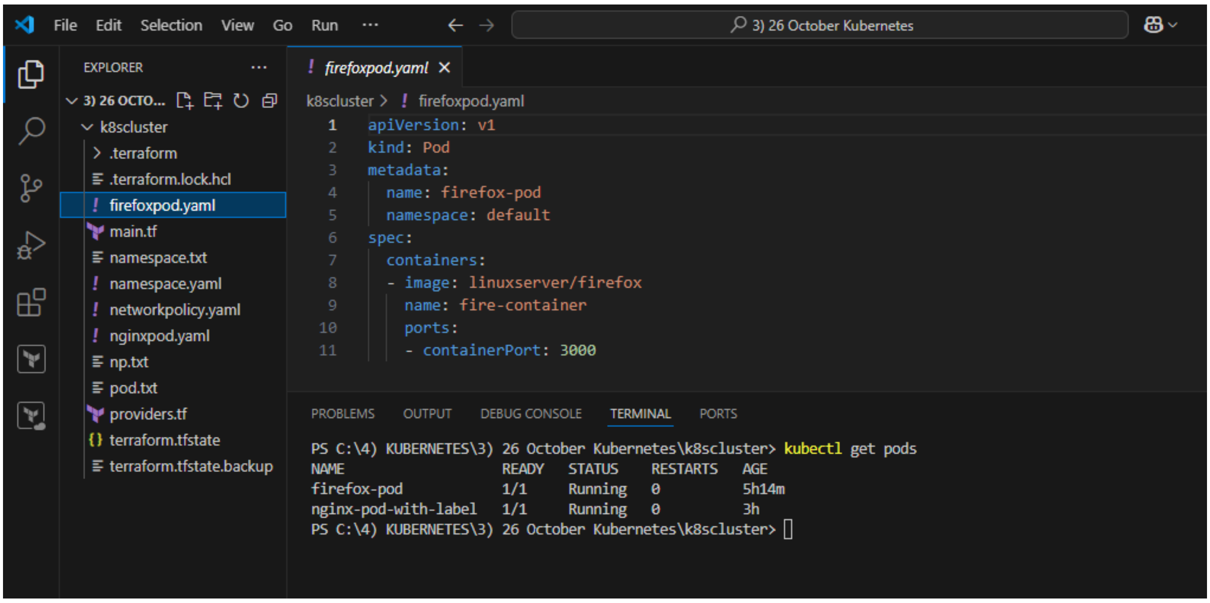
Private Cluster- Jiska API server bs kisi subnet se access hota hai..

Public Cluster - Jiska API Server public hai

1) Create folder “3) 26 October Kubernetes” and copy data of previous folder

AGENDA – Network policy, PV, PVC

1)



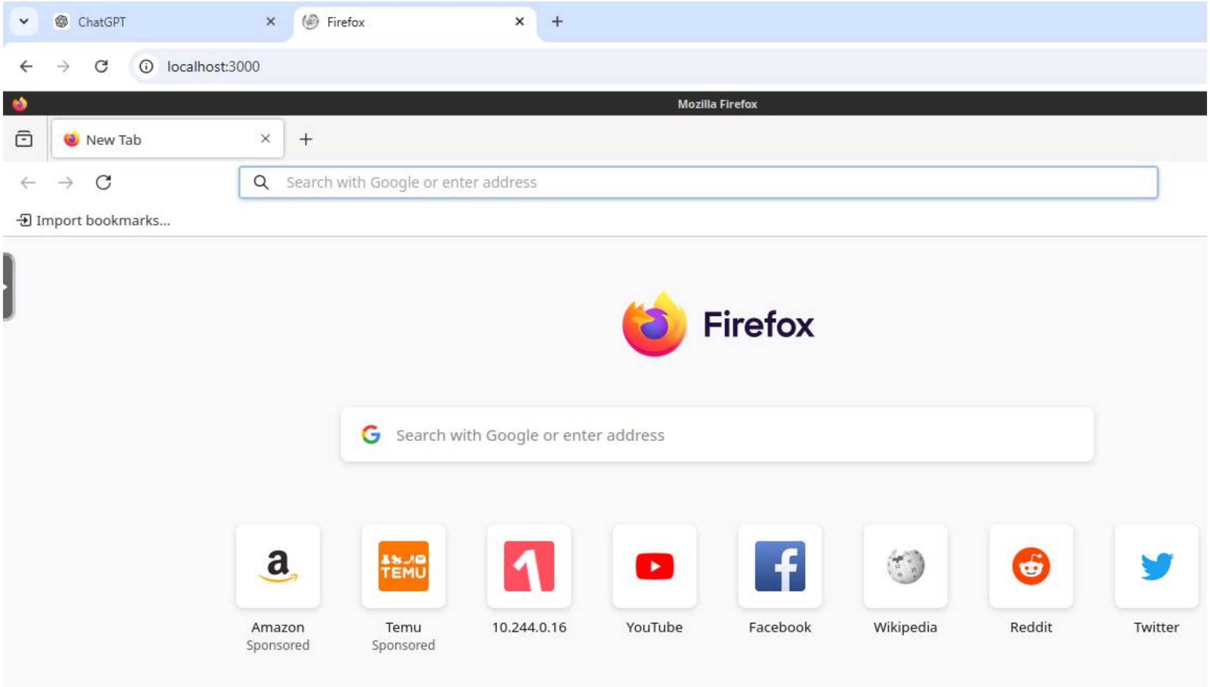
```yaml
apiVersion: v1
kind: Pod
metadata:
  name: firefox-pod
  namespace: default
spec:
  containers:
  - image: linuxserver/firefox
    name: fire-container
    ports:
    - containerPort: 3000
```

```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> kubectl get pods
NAME                  READY   STATUS    RESTARTS   AGE
firefox-pod           1/1     Running   0          5h14m
nginx-pod-with-label  1/1     Running   0          3h
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> 
```

2) **Kubectl get pods**

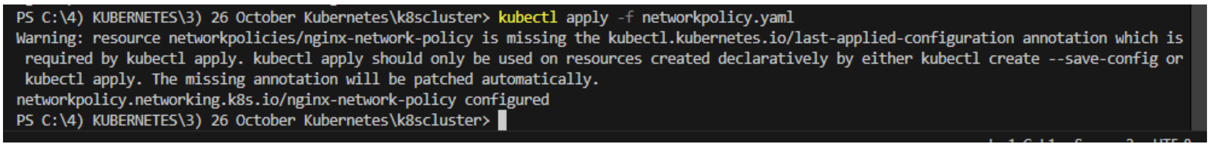
3) **kubectl port-forward firefox-pod 3000:3000 =** It created a new tunnel from our laptop to that firefox pod

4) localhost:3000



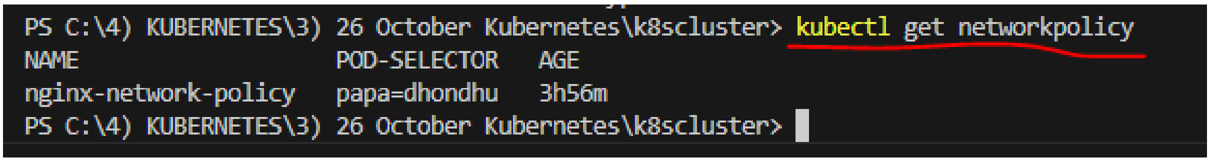
5) Open new terminal in vscode

kubectl apply -f networkpolicy.yaml



```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> kubectl apply -f networkpolicy.yaml
Warning: resource networkpolicies/nginx-network-policy is missing the kubectl.kubernetes.io/last-applied-configuration annotation which is
 required by kubectl apply. kubectl apply should only be used on resources created declaratively by either kubectl create --save-config or
 kubectl apply. The missing annotation will be patched automatically.
networkpolicy.networking.k8s.io/nginx-network-policy configured
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster>
```

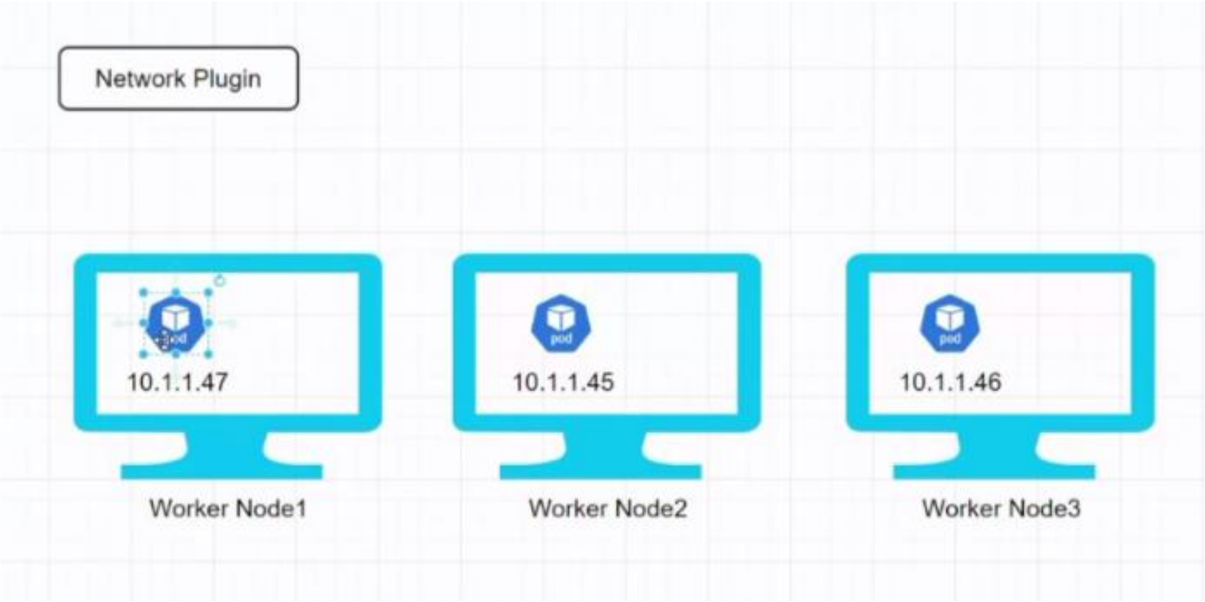
6) **kubectl get networkpolicy**



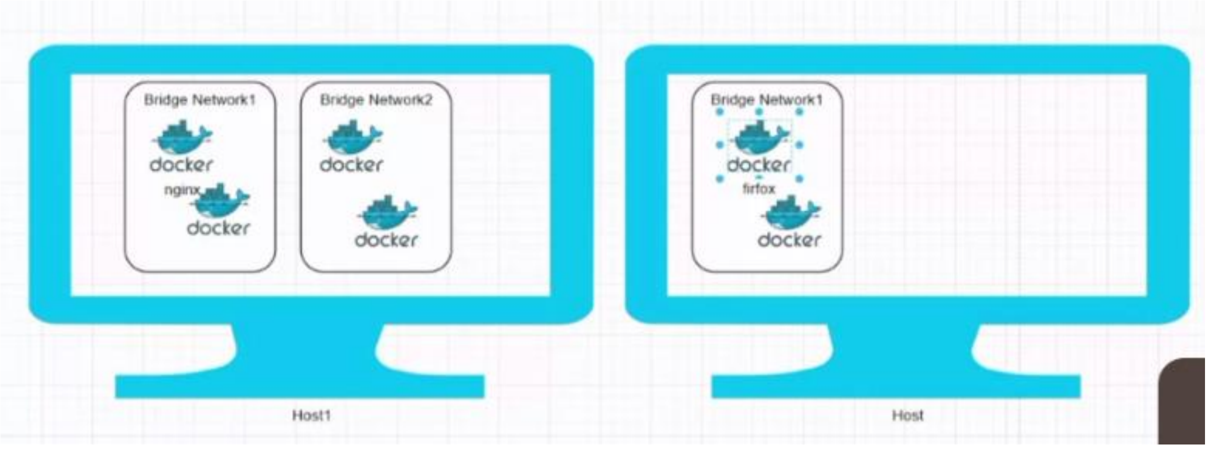
```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> kubectl get networkpolicy
NAME                   POD-SELECTOR    AGE
nginx-network-policy   papa=dhondhu    3h56m
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster>
```

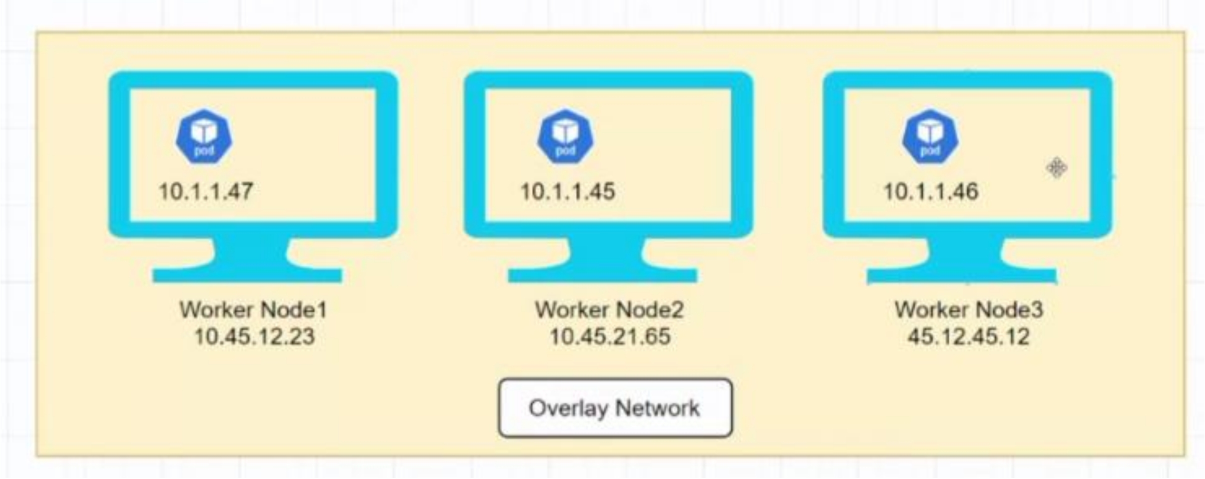
NOTE : 1) One pod in one worker node can communicate to another pod in another worker node by their ips.

2) The responsibility of giving ips in cluster to pods is of network plugin



3) Overlay network in docker





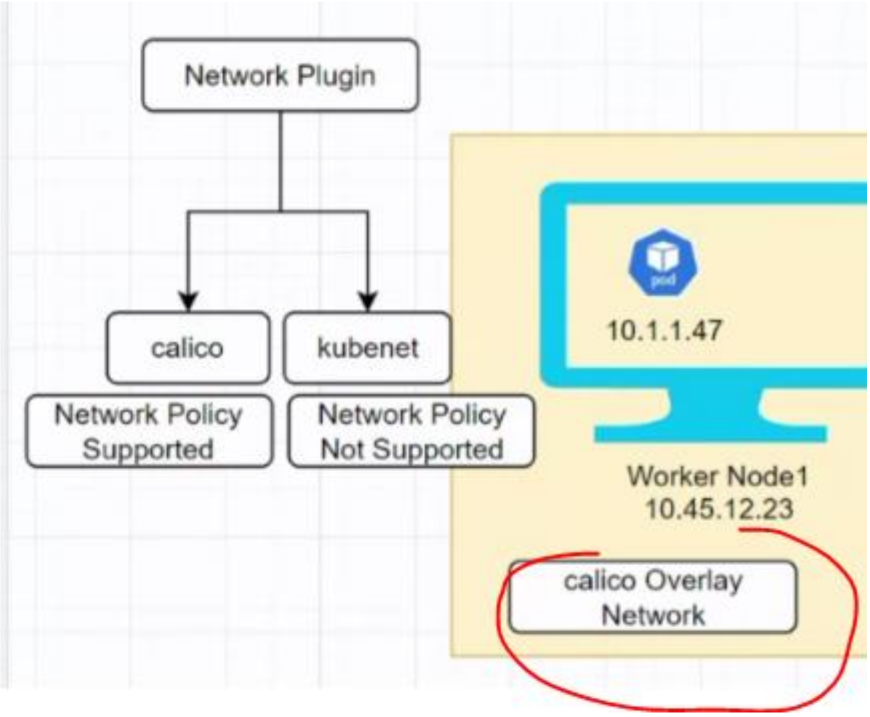
Docker में **overlay network** एक प्रकार का नेटवर्क होता है जो कंटेनरों को अलग-अलग होस्ट्स (machines) पर एक दूसरे से जोड़ता है, जैसे वे एक ही होस्ट पर हों।

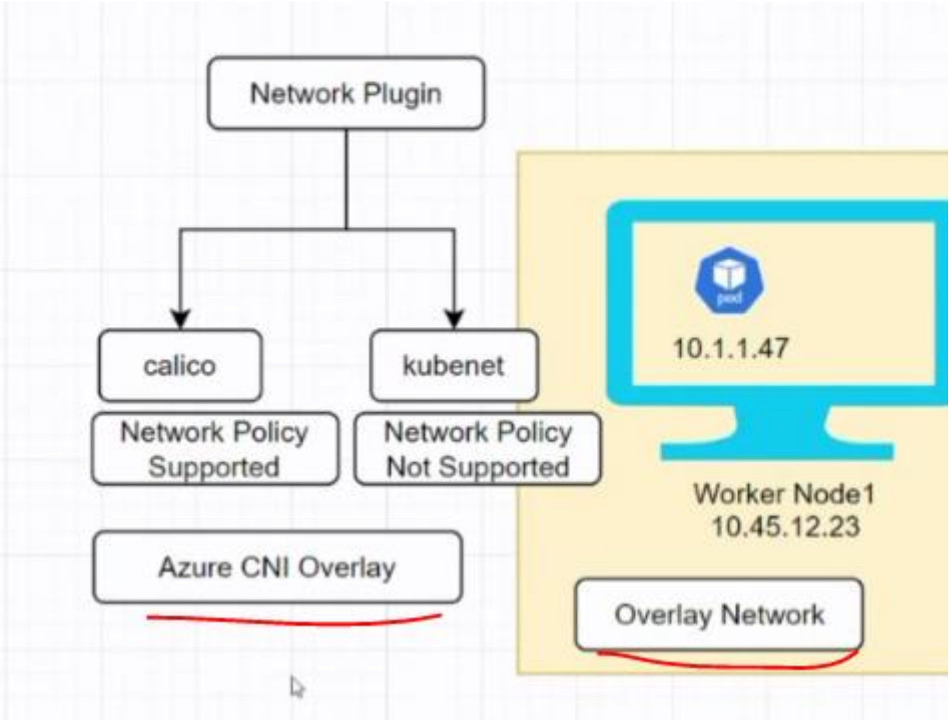
जब हम Docker Swarm या Kubernetes जैसी orchestration tools का उपयोग करते हैं, तो overlay network कंटेनरों को एक दूसरे से securely और efficiently communicate करने की सुविधा देता है, चाहे वे किसी भी physical machine पर हों।

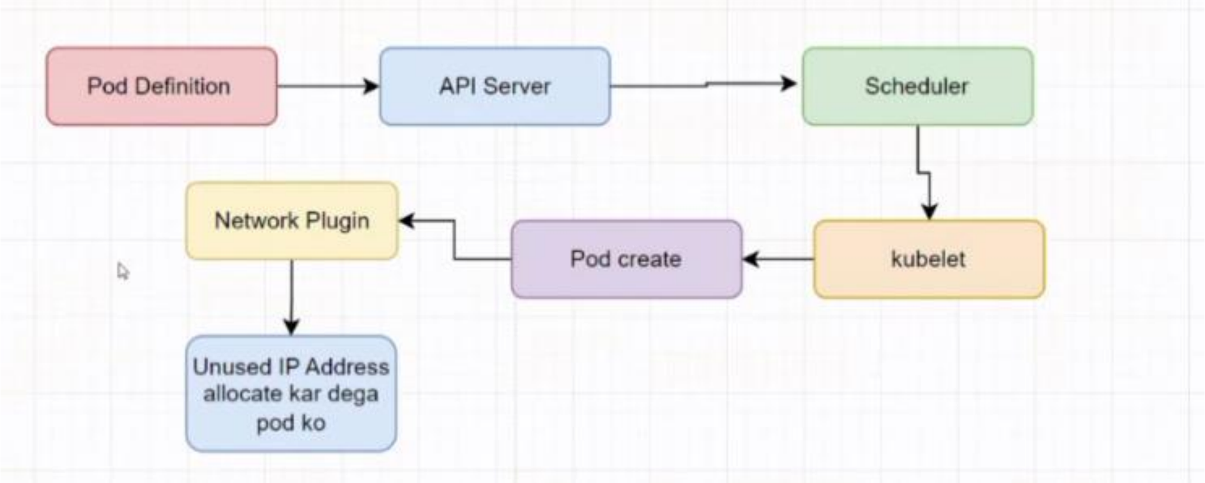
साधारण शब्दों में कहें तो, यह एक virtual नेटवर्क होता है जो विभिन्न मशीनों पर running containers के बीच कनेक्शन बनाता है।

Example: अगर आपके पास दो Docker host हैं और दोनों पर containers चल रहे हैं, तो overlay network इन कंटेनरों को आपस में संवाद करने का तरीका प्रदान करता है, जैसे वे एक ही host पर चल रहे हों।

7)







NOTE : 1) kubenet network do not supports network policy. Donot allow site to site vpn connectivity

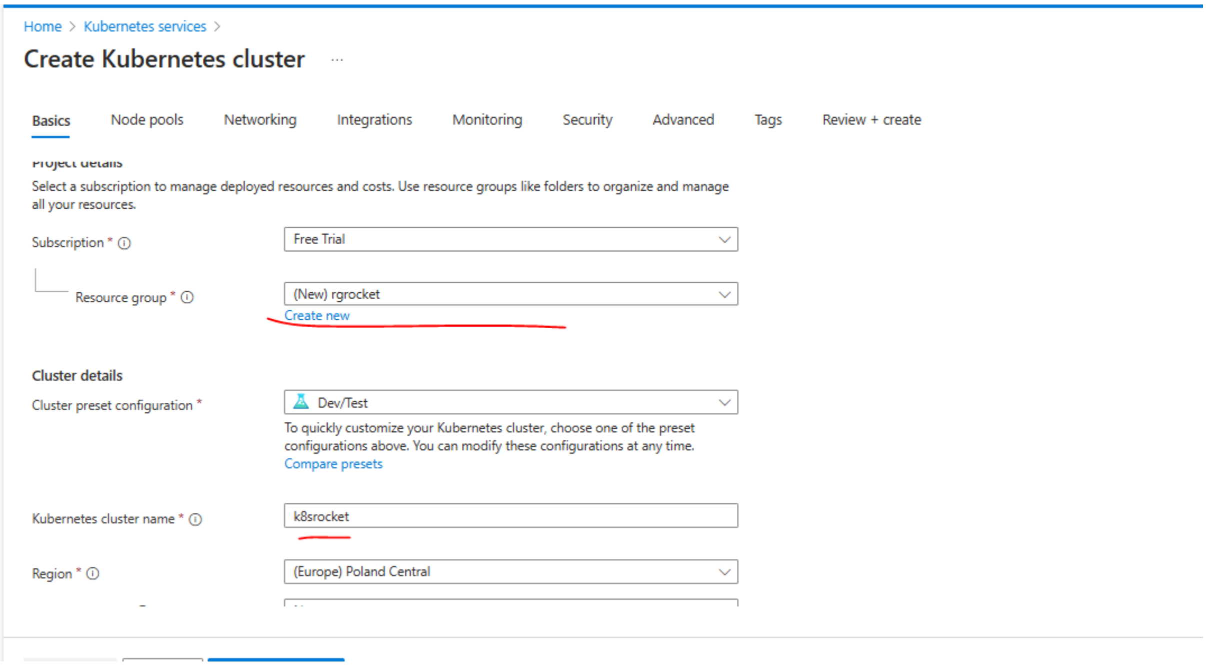
2) AZURE cni with calico allows to set network policy & allow site to site vpn connectivity

3) Difference between kubenet and azure cni (azure container networking interface)

++

AGENDA- Create k8s cluster

1)



Home > Kubernetes services >

Create Kubernetes cluster ...

Project details

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription * ⓘ: Free Trial

Resource group * ⓘ: (New) rgrocket

Create new

Cluster details

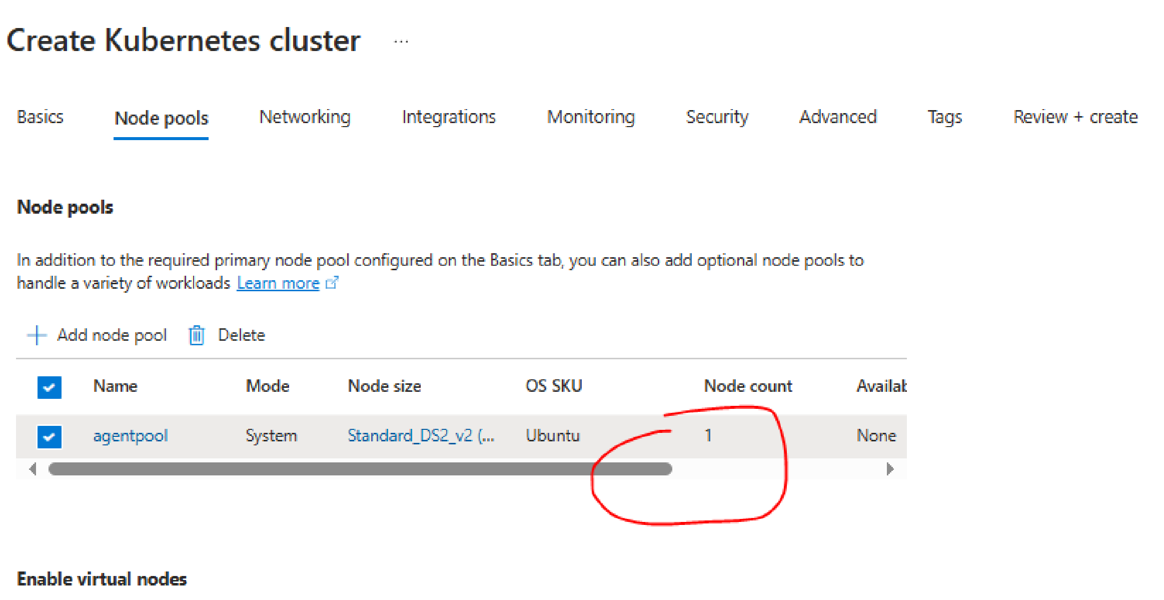
Cluster preset configuration *: Dev/Test

To quickly customize your Kubernetes cluster, choose one of the preset configurations above. You can modify these configurations at any time.

Compare presets

Kubernetes cluster name * ⓘ: k8srocket

Region * ⓘ: (Europe) Poland Central



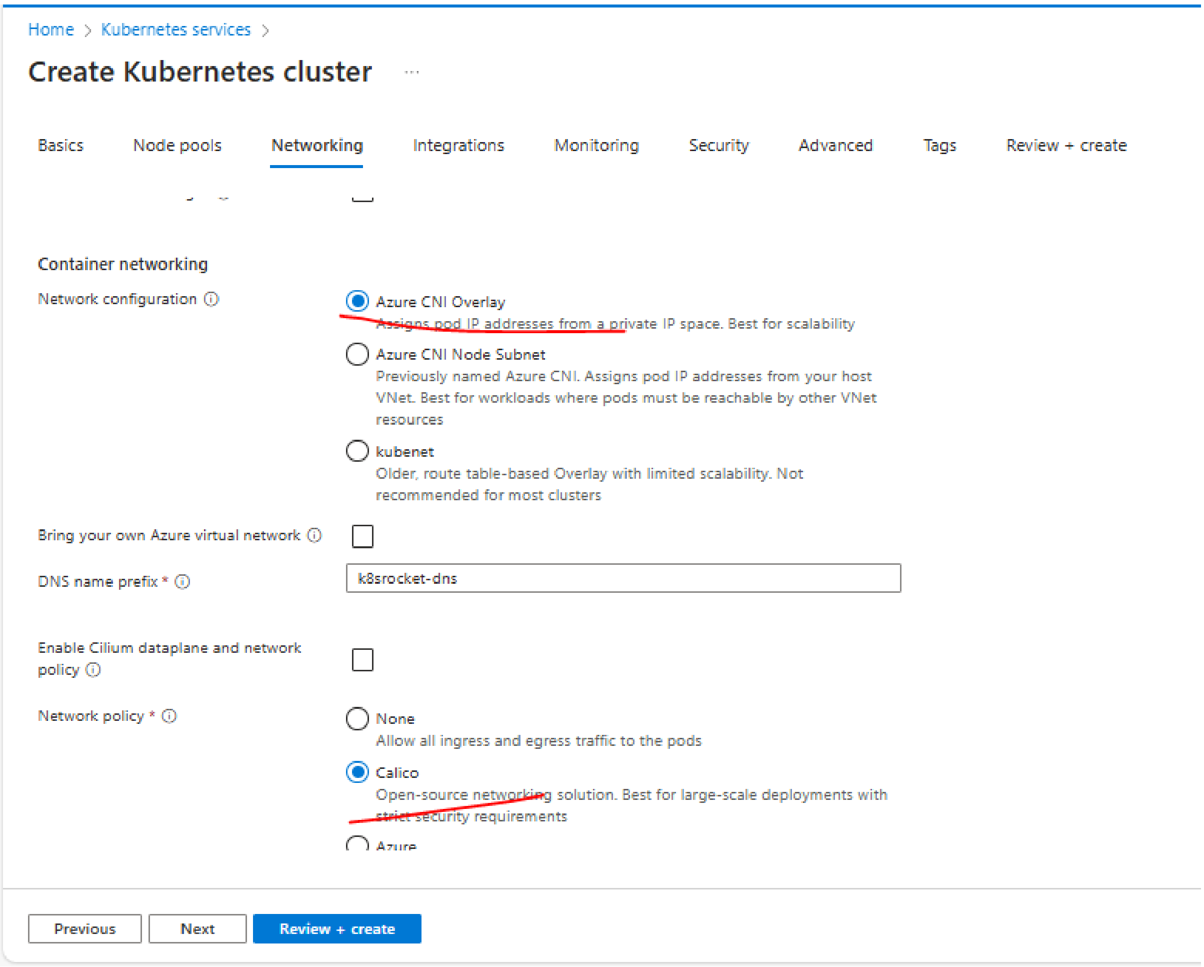
Create Kubernetes cluster ...

Node pools

In addition to the required primary node pool configured on the Basics tab, you can also add optional node pools to handle a variety of workloads Learn more

☑	Name	Mode	Node size	OS SKU	Node count	Availat
☑	agentpool	System	Standard_DS2_v2 (...	Ubuntu	1	None

Enable virtual nodes



Home > Kubernetes services >

Create Kubernetes cluster ...

Basics | Node pools | **Networking** | Integrations | Monitoring | Security | Advanced | Tags | Review + create

Container networking

Network configuration ⓘ

● Azure CNI Overlay
Assigns pod IP addresses from a private IP space. Best for scalability

○ Azure CNI Node Subnet
Previously named Azure CNI. Assigns pod IP addresses from your host VNet. Best for workloads where pods must be reachable by other VNet resources

○ kubenet
Older, route table-based Overlay with limited scalability. Not recommended for most clusters

Bring your own Azure virtual network ⓘ □

DNS name prefix * ⓘ k8srocket-dns

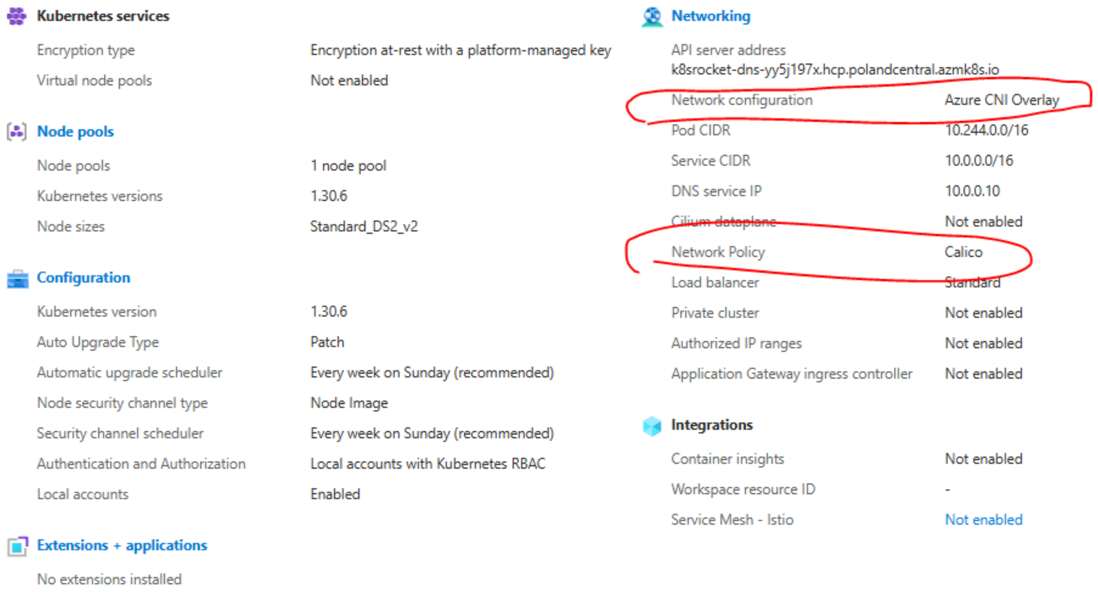
Enable Cilium dataplane and network policy ⓘ □

Network policy * ⓘ

○ None
Allow all ingress and egress traffic to the pods

● Calico
Open-source networking solution. Best for large-scale deployments with strict security requirements

○ Azure



Kubernetes services

Encryption type	Encryption at-rest with a platform-managed key
Virtual node pools	Not enabled

Node pools

Node pools	1 node pool
Kubernetes versions	1.30.6
Node sizes	Standard_DS2_v2

Configuration

Kubernetes version	1.30.6
Auto Upgrade Type	Patch
Automatic upgrade scheduler	Every week on Sunday (recommended)
Node security channel type	Node Image
Security channel scheduler	Every week on Sunday (recommended)
Authentication and Authorization	Local accounts with Kubernetes RBAC
Local accounts	Enabled

Extensions + applications

No extensions installed

Networking

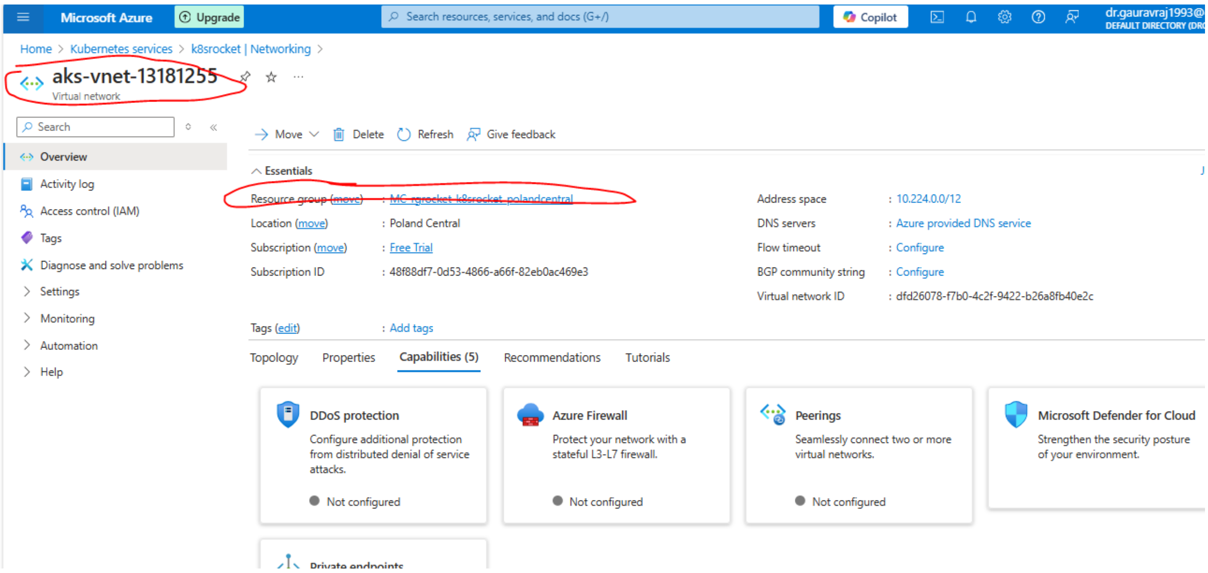
API server address
k8srocket-dns-yy5j197x.hcp.polandcentral.azmk8s.io

Network configuration	Azure CNI Overlay
Pod CIDR	10.244.0.0/16
Service CIDR	10.0.0.0/16
DNS service IP	10.0.0.10
Cilium dataplane	Not enabled
Network Policy	Calico
Load balancer	Standard
Private cluster	Not enabled
Authorized IP ranges	Not enabled
Application Gateway ingress controller	Not enabled

Integrations

Container insights	Not enabled
Workspace resource ID	-
Service Mesh - Istio	Not enabled

2) So above we can see calico network policy which supports network policy and Along with aks cluster a vnet is also created by azure itself in calico.



aks-vnet-13181255

Virtual network

Essentials

Resource group (move)	: MC_rgrocket_k8srocket_polandcentral	Address space	: 10.224.0.0/12
Location (move)	: Poland Central	DNS servers	: Azure provided DNS service
Subscription (move)	: Free Trial	Flow timeout	: Configure
Subscription ID	: 48f88df7-0d53-4866-a66f-82eb0ac469e3	BGP community string	: Configure
		Virtual network ID	: dfd26078-f7b0-4c2f-9422-b26a8fb40e2c
Tags (edit)	: Add tags		

DDoS protection
Configure additional protection from distributed denial of service attacks.
● Not configured

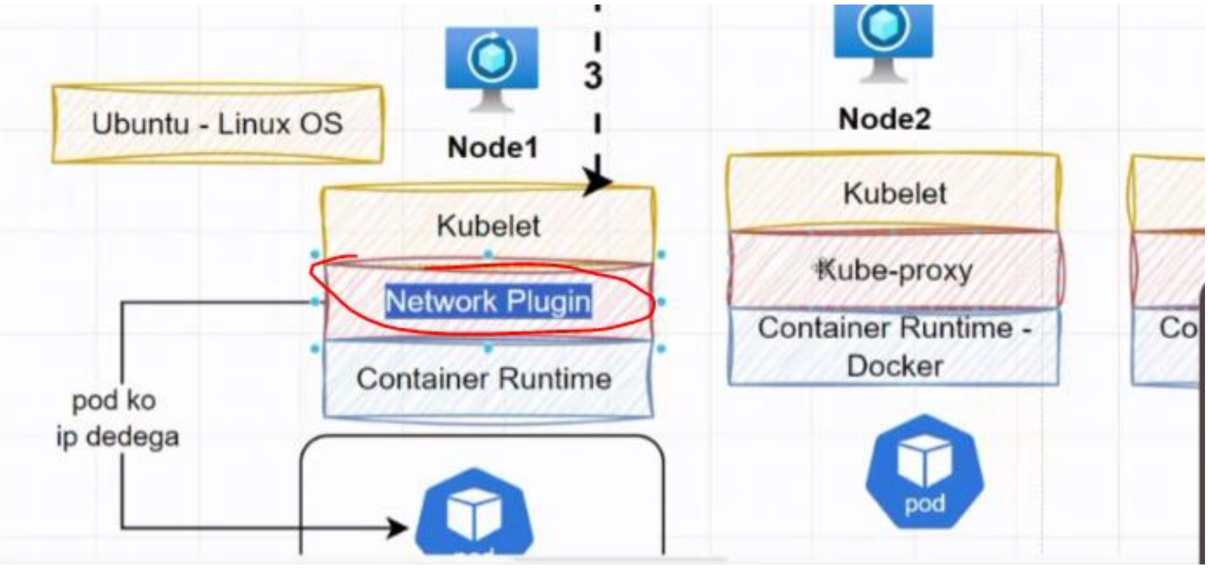
Azure Firewall
Protect your network with a stateful L3-L7 firewall.
● Not configured

Peerings
Seamlessly connect two or more virtual networks.
● Not configured

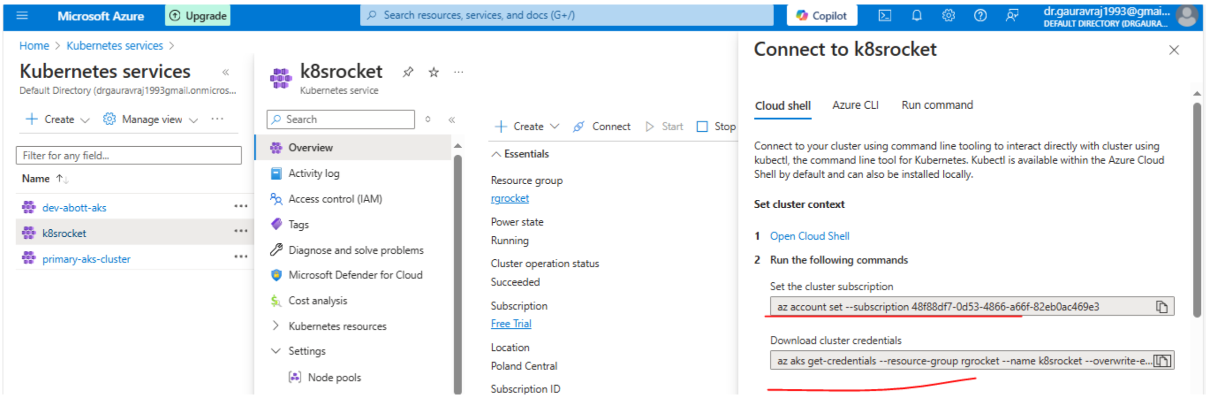
Microsoft Defender for Cloud
Strengthen the security posture of your environment.

Private endpoints

3) Now in diagram mention cni plugin or network plugin instead of kubeproxy which assigns ip to pod



4) Now since cluster is created so connecting it on our computer



Kubernetes services

Default Directory (drgauravraj1993gmail.onmicros...

k8srocket

Kubernetes service

Essentials

Resource group
rgrocket

Power state
Running

Cluster operation status
Succeeded

Subscription
Free Trial

Location
Poland Central

Subscription ID

Connect to k8srocket

Connect to your cluster using command line tooling to interact directly with cluster using kubectl, the command line tool for Kubernetes. Kubectl is available within the Azure Cloud Shell by default and can also be installed locally.

Set cluster context

1 Open Cloud Shell

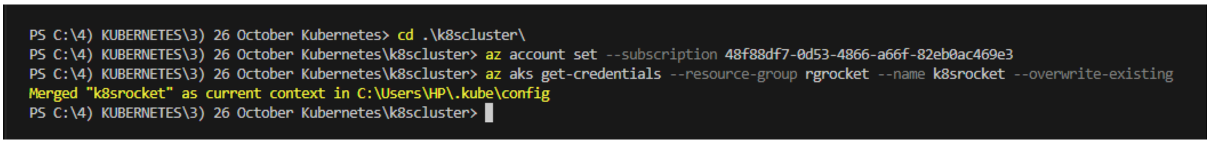
2 Run the following commands

Set the cluster subscription

```
az account set --subscription 48f88df7-0d53-4866-a66f-82eb0ac469e3
```

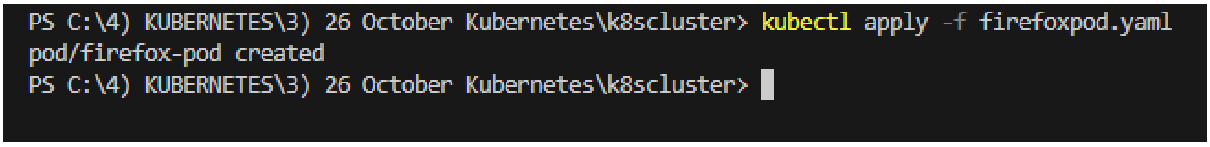
Download cluster credentials

```
az aks get-credentials --resource-group rgrocket --name k8srocket --overwrite-e...
```



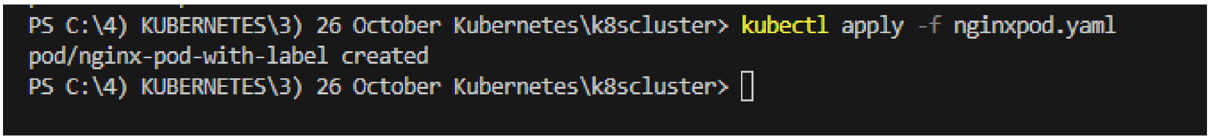
```
PS C:\4) KUBERNETES\3) 26 October Kubernetes> cd .\k8scluster\
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> az account set --subscription 48f88df7-0d53-4866-a66f-82eb0ac469e3
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> az aks get-credentials --resource-group rgrocket --name k8srocket --overwrite-existing
Merged "k8srocket" as current context in C:\Users\HP\.kube\config
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster>
```

5) **kubectl apply -f firefoxpod.yaml** = create firefox pod



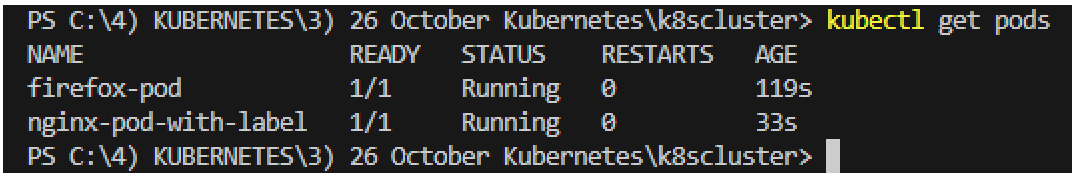
```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> kubectl apply -f firefoxpod.yaml
pod/firefox-pod created
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster>
```

6) **kubectl apply -f nginxpod.yaml** = create nginx pod



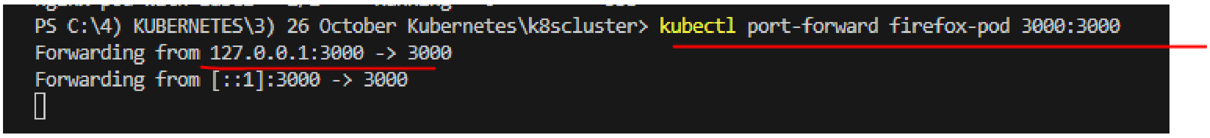
```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> kubectl apply -f nginxpod.yaml
pod/nginx-pod-with-label created
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster>
```

7) **kubectl get pods**



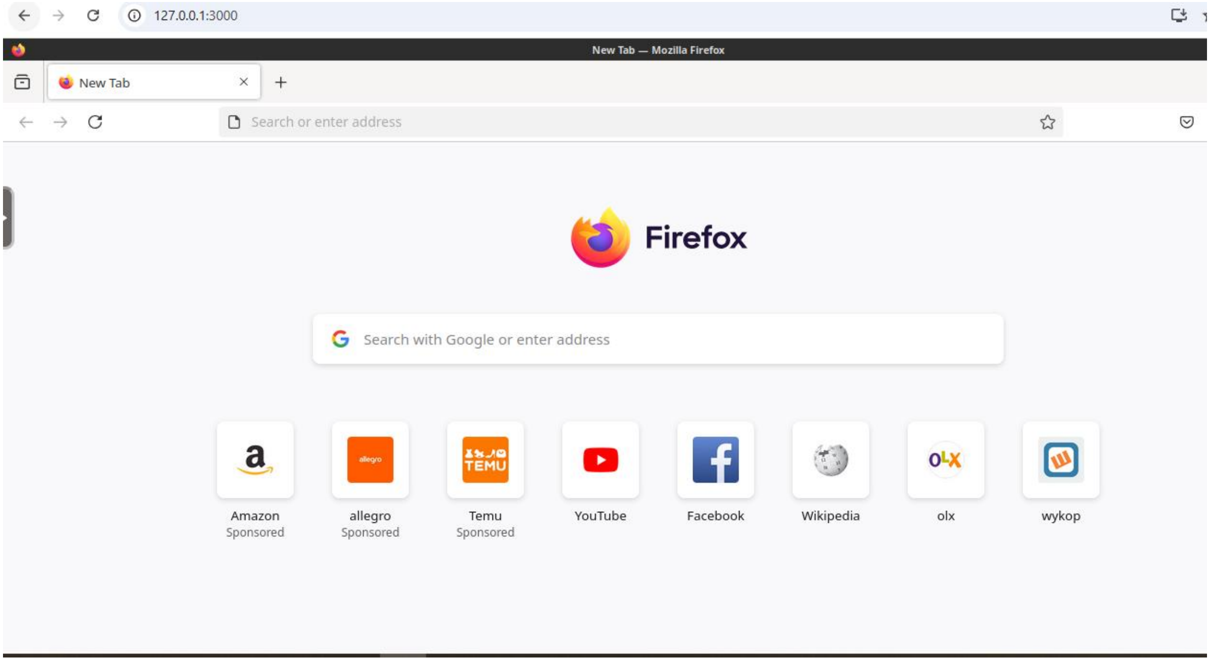
```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> kubectl get pods
NAME                   READY   STATUS    RESTARTS   AGE
firefox-pod            1/1     Running   0          119s
nginx-pod-with-label   1/1     Running   0          33s
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster>
```

8) **kubectl port-forward firefox-pod 3000:3000 =** port forwarding firefox from our local till pod or creating tunnel



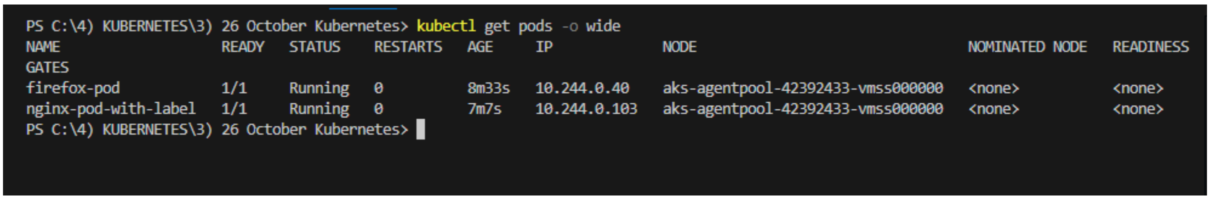
```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> kubectl port-forward firefox-pod 3000:3000
Forwarding from 127.0.0.1:3000 -> 3000
Forwarding from [::1]:3000 -> 3000
```

9) **127.0.0.1:3000** = Run in browser



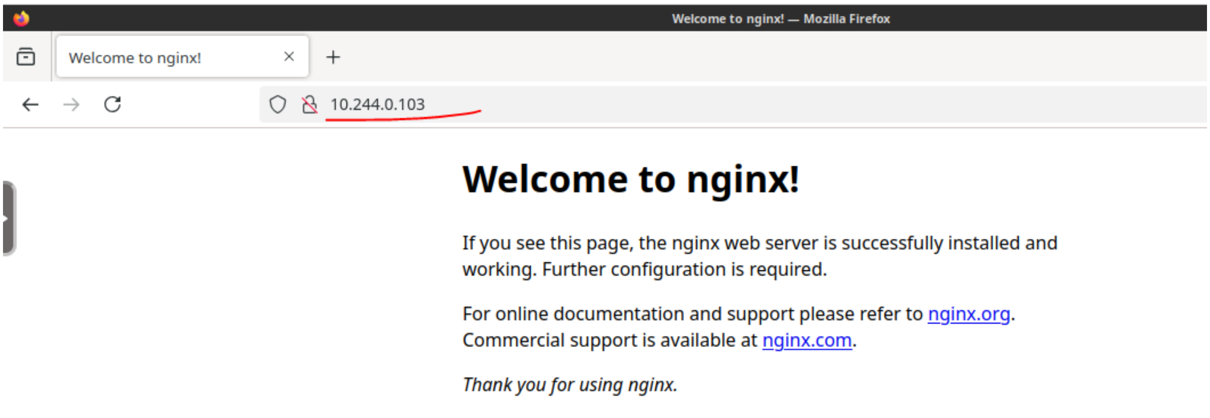
10) Change to another terminal in vscode

kubectl get pods -o wide



```
PS C:\4) KUBERNETES\3) 26 October Kubernetes> kubectl get pods -o wide
NAME                  READY   STATUS    RESTARTS   AGE    IP            NODE                                NOMINATED NODE   READINESS
GATES
firefox-pod           1/1     Running   0          8m33s  10.244.0.40   aks-agentpool-42392433-vmss000000   <none>           <none>
nginx-pod-with-label  1/1     Running   0          7m7s   10.244.0.103  aks-agentpool-42392433-vmss000000   <none>           <none>
PS C:\4) KUBERNETES\3) 26 October Kubernetes> 
```

11) 10.244.0.103= run nginx pod ip in firefox



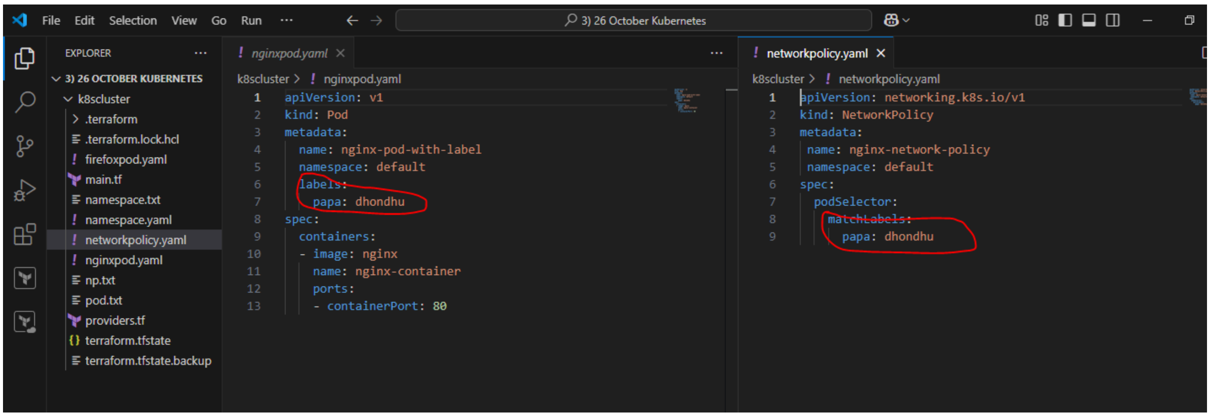
Welcome to nginx!

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.

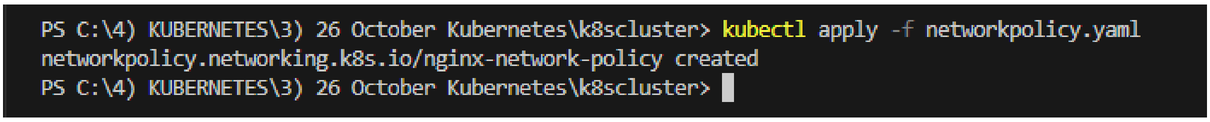
12) Now applying network policy with **labels**



```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod-with-label
  namespace: default
  labels:
    papa: dhondhu
spec:
  containers:
    - image: nginx
      name: nginx-container
      ports:
      - containerPort: 80
```

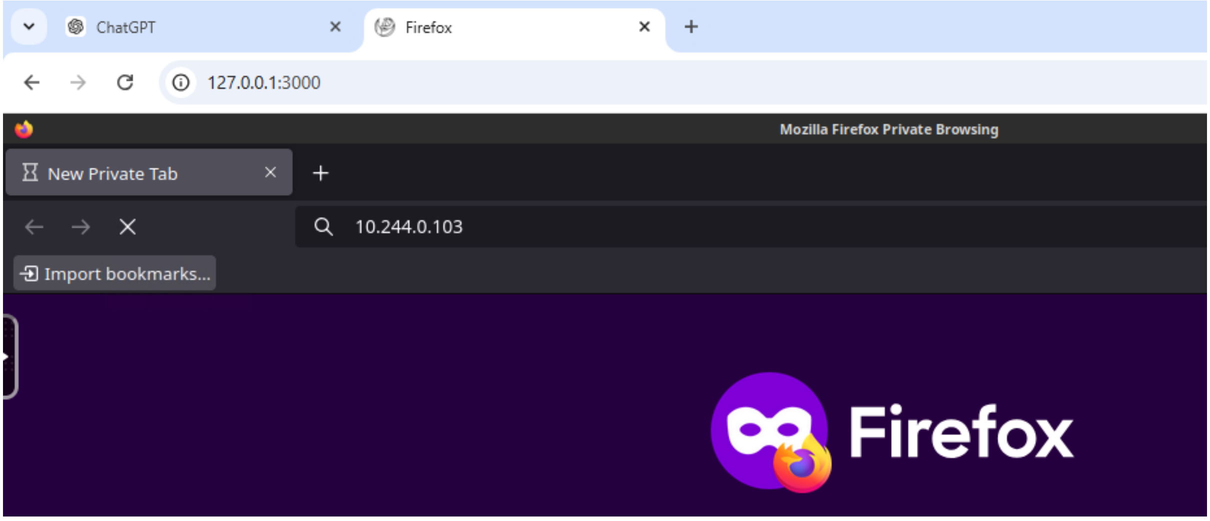
```yaml
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
 name: nginx-network-policy
 namespace: default
spec:
  podSelector:
    matchLabels:
      papa: dhondhu
```

13) **kubectl apply -f networkpolicy.yaml** = apply network policy



```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> kubectl apply -f networkpolicy.yaml
networkpolicy.networking.k8s.io/nginx-network-policy created
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> 
```

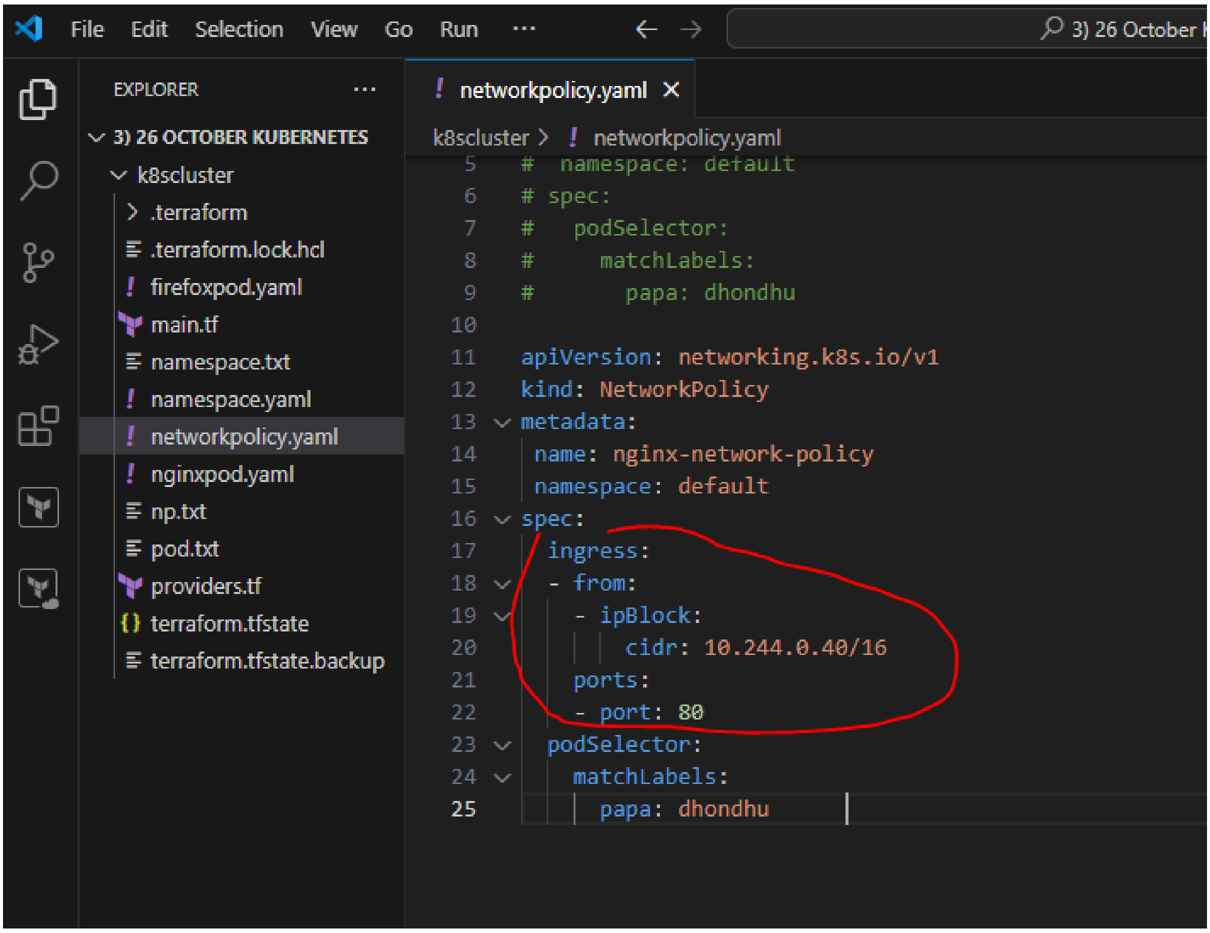
14) **10.244.0.103** = Now run ip of nginx in firefox then it will not run, which shows that network policy has been set, which means all ports on nginx pod has been closed



++

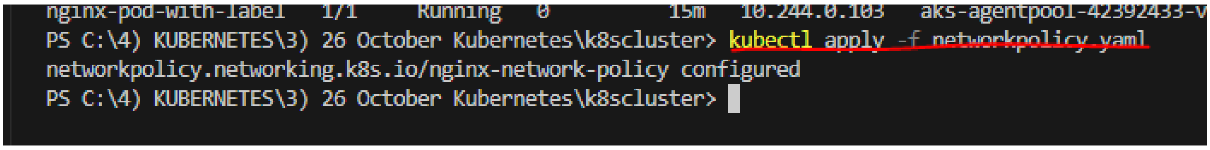
AGENDA – OPENING A SPECIFIC PORT ON NGINX POD

1) Now writing network policy yaml to open port 80



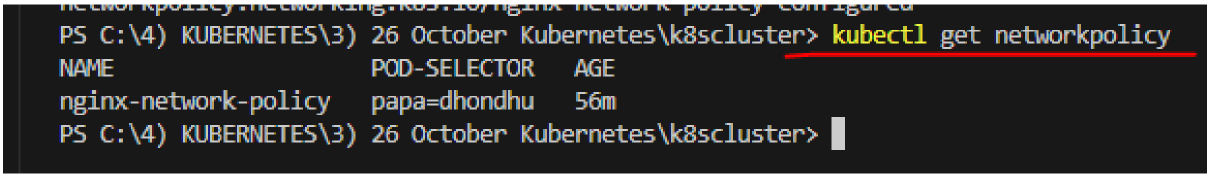
```yaml
#  namespace: default
# spec:
#   podSelector:
#     matchLabels:
#       papa: dhondhu

apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
 name: nginx-network-policy
 namespace: default
spec:
  ingress:
  - from:
    - ipBlock:
        cidr: 10.244.0.40/16
    ports:
    - port: 80
  podSelector:
    matchLabels:
      papa: dhondhu
```

2) **kubectl apply -f networkpolicy.yaml**



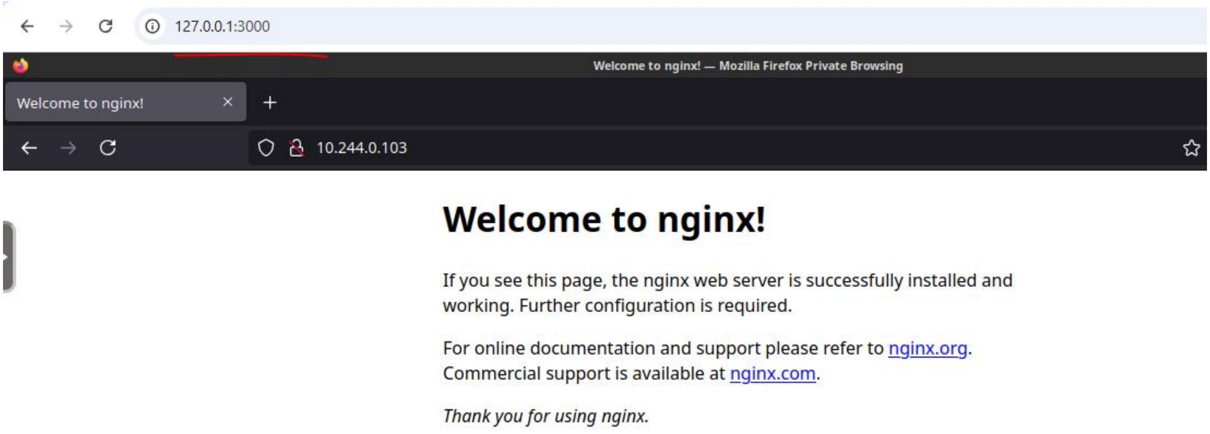
```
nginx-pod-with-label   1/1     Running   0          15m   10.244.0.103   aks-agentpool-42392433-v
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> kubectl apply -f networkpolicy.yaml
networkpolicy.networking.k8s.io/nginx-network-policy configured
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster>
```

3) **kubectl get networkpolicy**



```
networkpolicy.networking.k8s.io/nginx-network-policy configured
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> kubectl get networkpolicy
NAME                   POD-SELECTOR   AGE
nginx-network-policy   papa=dhondhu   56m
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster>
```

4) http://10.244.0.103/ = Again run ip of nginx which shows that 80 port opened to run nginx or access nginx



Welcome to nginx!

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

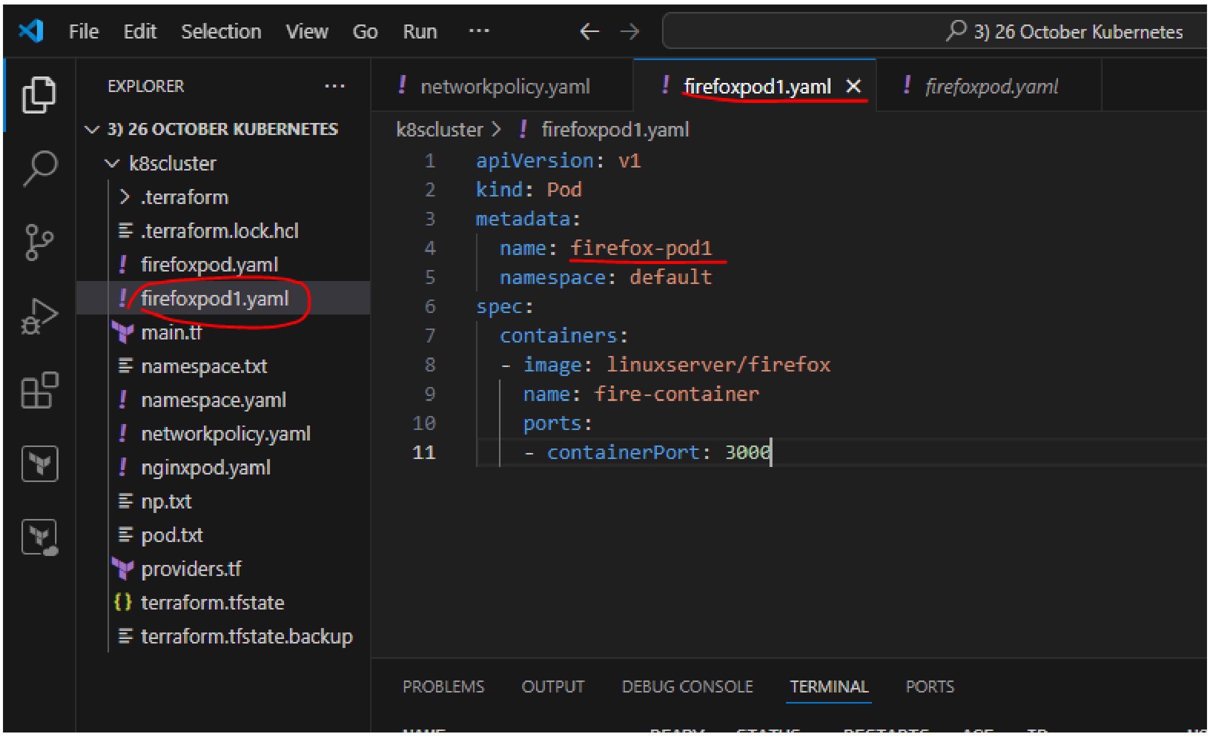
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

++

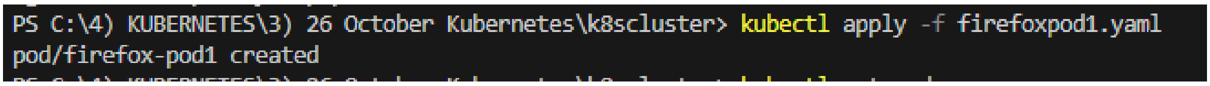
AGENDA – CREATING FIREFOX POD AGAIN

1) Create "firefoxpod1.yaml" file and write code into it.



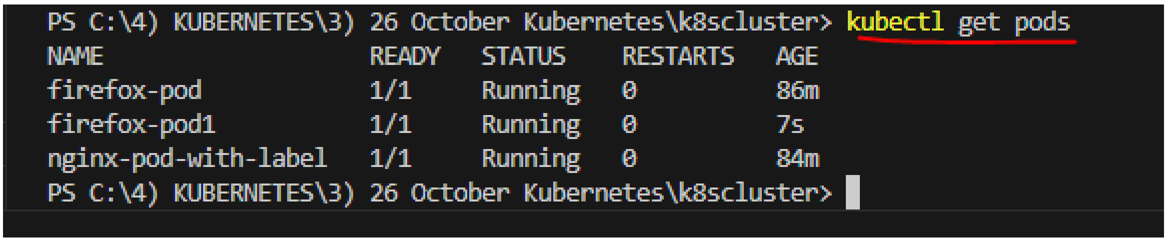
```yaml
apiVersion: v1
kind: Pod
metadata:
  name: firefox-pod1
  namespace: default
spec:
  containers:
  - image: linuxserver/firefox
    name: fire-container
    ports:
    - containerPort: 3000
```

2) **kubectl apply -f firefoxpod1.yaml** = creating new firefox pod



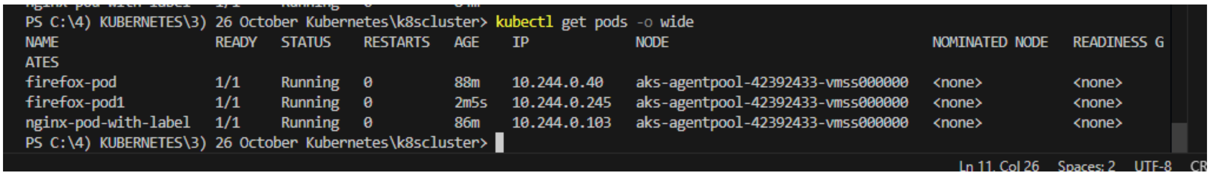
```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> kubectl apply -f firefoxpod1.yaml
pod/firefox-pod1 created
```

3) **kubectl get pods**



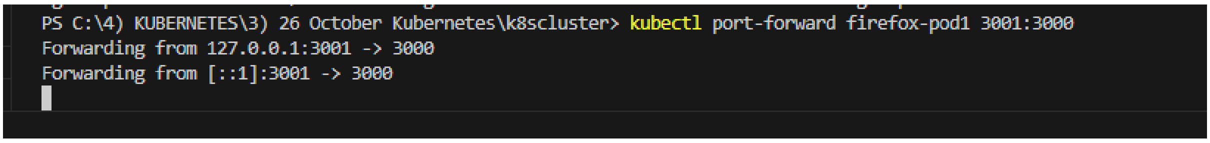
```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> kubectl get pods
NAME                  READY   STATUS    RESTARTS   AGE
firefox-pod           1/1     Running   0          86m
firefox-pod1          1/1     Running   0          7s
nginx-pod-with-label  1/1     Running   0          84m
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster>
```

4) **kubectl get pods -o wide**



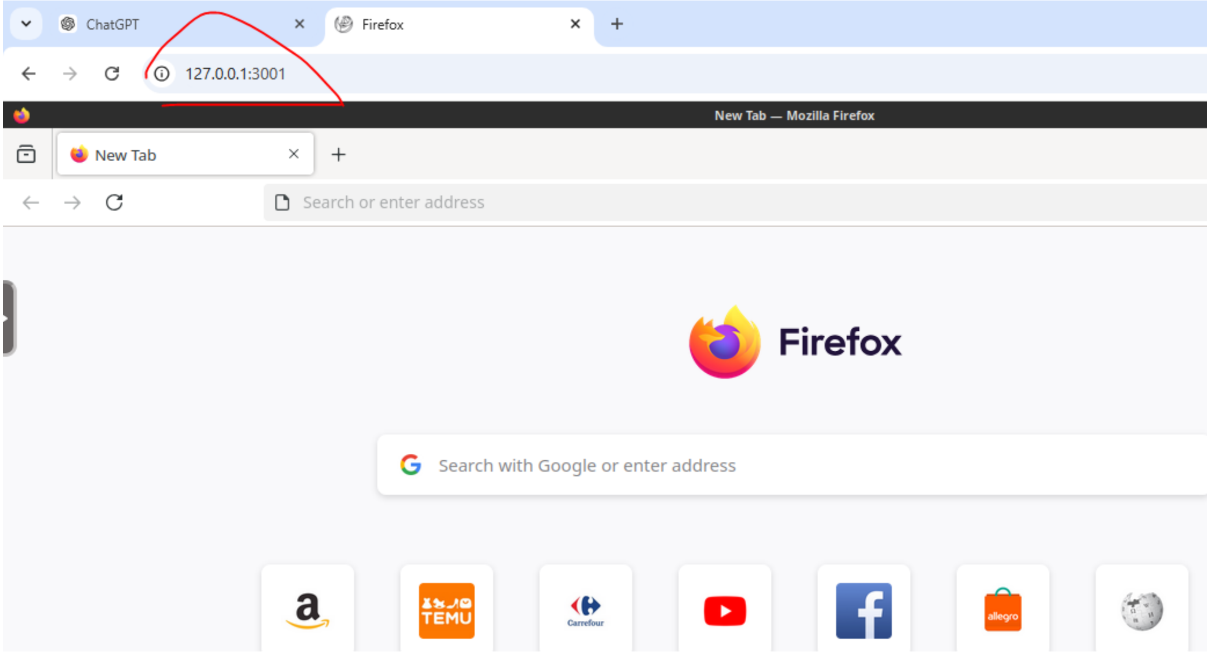
```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> kubectl get pods -o wide
NAME                  READY   STATUS    RESTARTS   AGE    IP             NODE                                NOMINATED NODE   READINESS G
ATES
firefox-pod           1/1     Running   0          88m    10.244.0.40    aks-agentpool-42392433-vmss000000   <none>           <none>
firefox-pod1          1/1     Running   0          2m5s   10.244.0.245   aks-agentpool-42392433-vmss000000   <none>           <none>
nginx-pod-with-label  1/1     Running   0          86m    10.244.0.103   aks-agentpool-42392433-vmss000000   <none>           <none>
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster>
```

5) **kubectl port-forward firefox-pod1 3001:3000 =** creating tunnel between our laptop and firefox pod

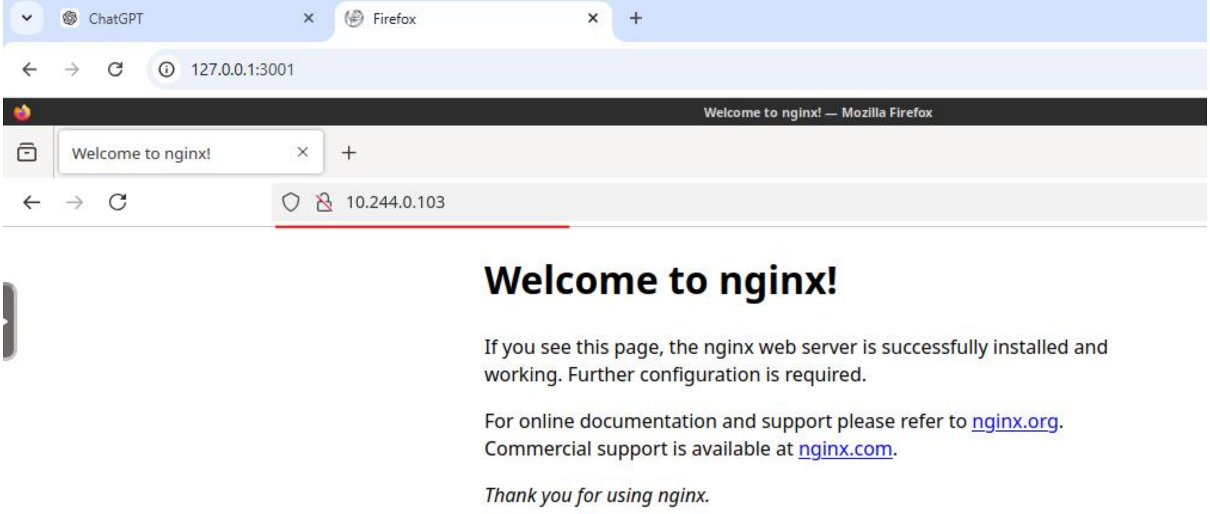


```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\k8scluster> kubectl port-forward firefox-pod1 3001:3000
Forwarding from 127.0.0.1:3001 -> 3000
Forwarding from [::1]:3001 -> 3000
```

6) **127.0.0.1:3001** = run new firefox in browser



7) 10.244.0.103 = run nginx ip in new firefox and its running because we have opened cidr of all ranges in networkpolicy.yaml file



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.

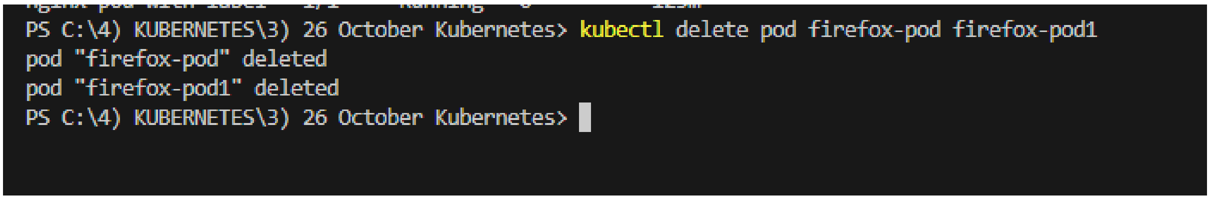
NOTE : 1) Where is network policy applied in k8s ? = On pod

2) Can we apply network policy on node? = no

3) If network policy is not working after applying, then what we will check? = Network plugin

8) Now deleting both firefox pods

kubectl delete pod firefox-pod firefox-pod1

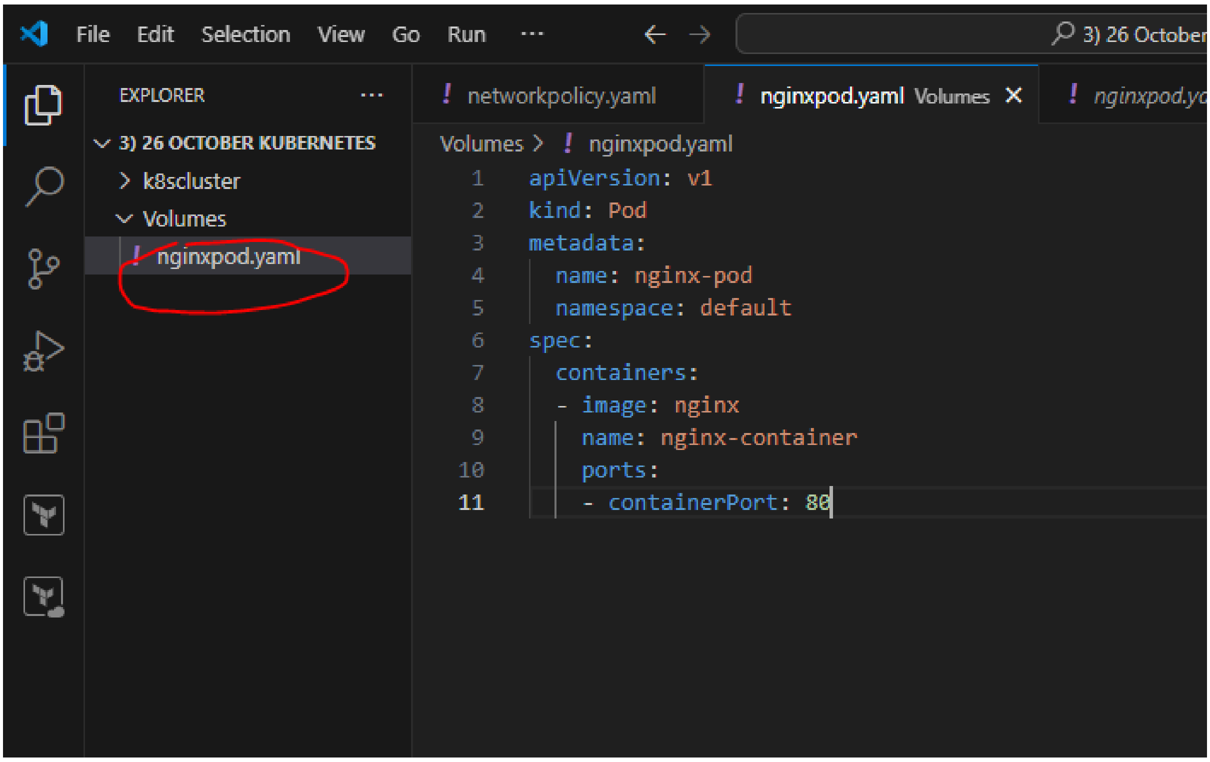


```
nginx-pod-with-label   1/1     Running   0          123m
PS C:\4) KUBERNETES\3) 26 October Kubernetes> kubectl delete pod firefox-pod firefox-pod1
pod "firefox-pod" deleted
pod "firefox-pod1" deleted
PS C:\4) KUBERNETES\3) 26 October Kubernetes> 
```

++

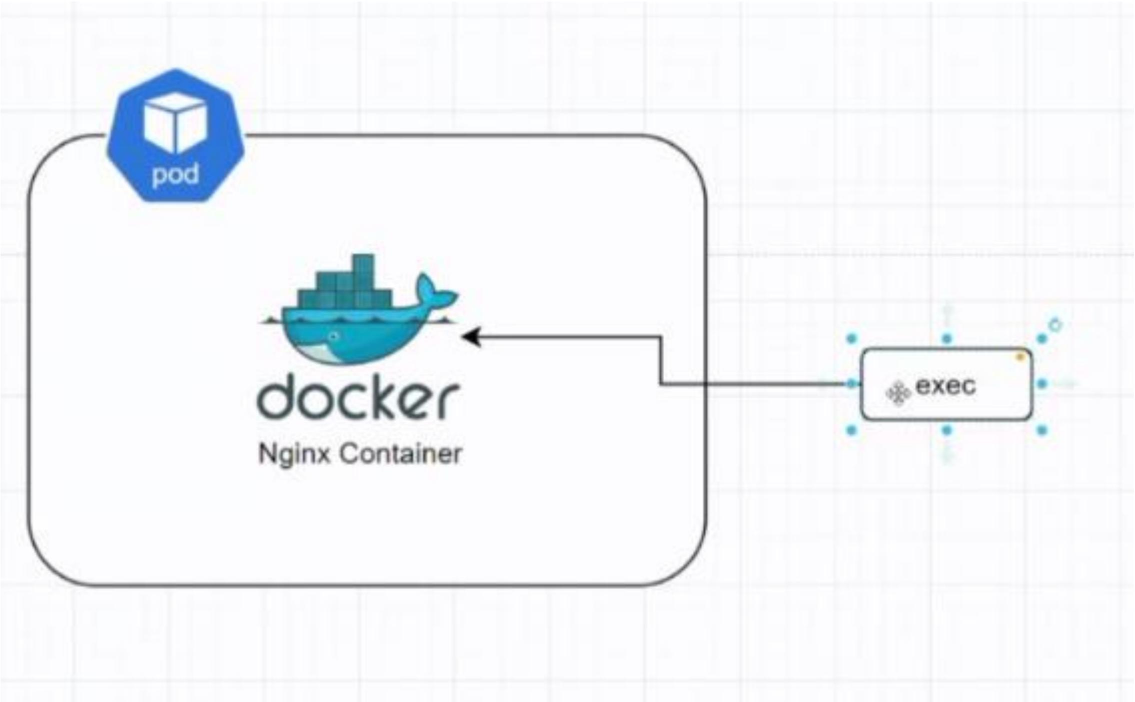
AGENDA – PV and PVC

1) Create folder Volumes and write nginx pod yaml file

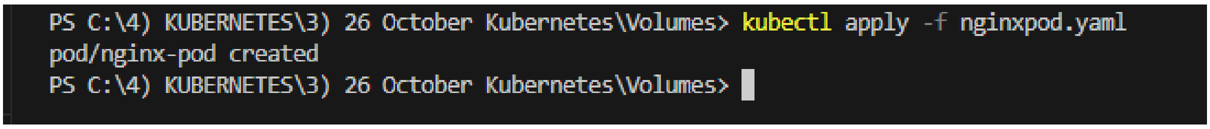


```yaml
apiVersion: v1
kind: Pod
metadata:
  name: nginx-pod
  namespace: default
spec:
  containers:
  - image: nginx
    name: nginx-container
    ports:
    - containerPort: 80
```

2)

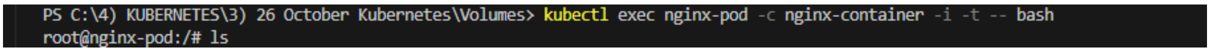


3) **kubectl apply -f nginxpod.yaml** = create pod



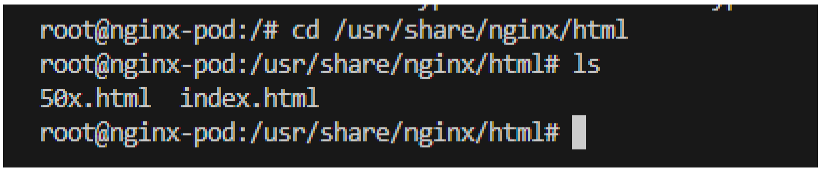
```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\Volumes> kubectl apply -f nginxpod.yaml
pod/nginx-pod created
PS C:\4) KUBERNETES\3) 26 October Kubernetes\Volumes> 
```

4) **kubectl exec nginx-pod -c nginx-container -i -t – bash** = exec commad is used to enter in container inside pod



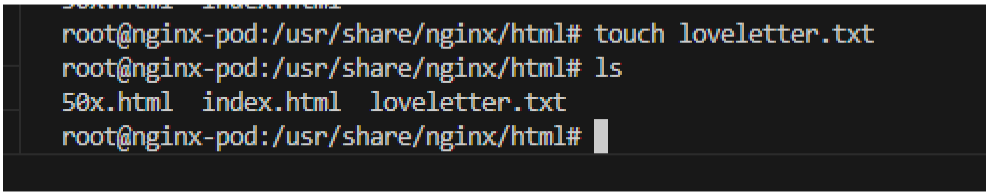
```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\Volumes> kubectl exec nginx-pod -c nginx-container -i -t -- bash
root@nginx-pod:/# ls
```

cd /usr/share/nginx/html



```
root@nginx-pod:/# cd /usr/share/nginx/html
root@nginx-pod:/usr/share/nginx/html# ls
50x.html  index.html
root@nginx-pod:/usr/share/nginx/html# 
```

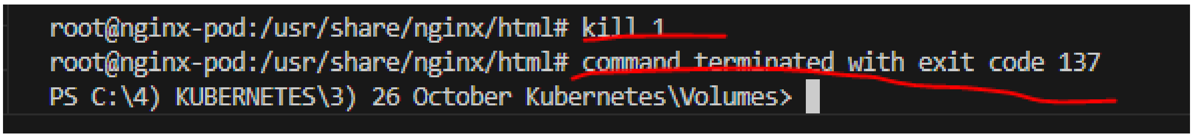
5) **touch loveletter.txt = create file**



```
root@nginx-pod:/usr/share/nginx/html# touch loveletter.txt
root@nginx-pod:/usr/share/nginx/html# ls
50x.html  index.html  loveletter.txt
root@nginx-pod:/usr/share/nginx/html# 
```

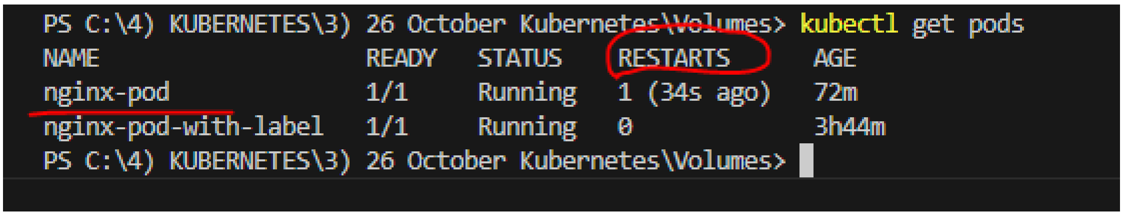
NOTE : 1) 1 container = 1 process means 1 service runs in 1 container

6) KILL 1 = 1 is like process id of container to kill it



```
root@nginx-pod:/usr/share/nginx/html# kill 1
root@nginx-pod:/usr/share/nginx/html# command terminated with exit code 137
PS C:\4) KUBERNETES\3) 26 October Kubernetes\Volumes>
```

7) **kubectl get pods**



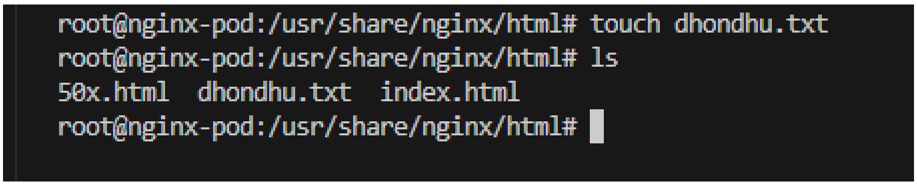
```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\Volumes> kubectl get pods
NAME                    READY   STATUS    RESTARTS       AGE
nginx-pod               1/1     Running   1 (34s ago)    72m
nginx-pod-with-label    1/1     Running   0              3h44m
PS C:\4) KUBERNETES\3) 26 October Kubernetes\Volumes>
```

8) So above restart shows that when we killed container in pod then pod had the responsibility to restart the container on its own, which we can see it has done it.

9) **kubectl exec nginx-pod -c nginx-container -i -t – bash**

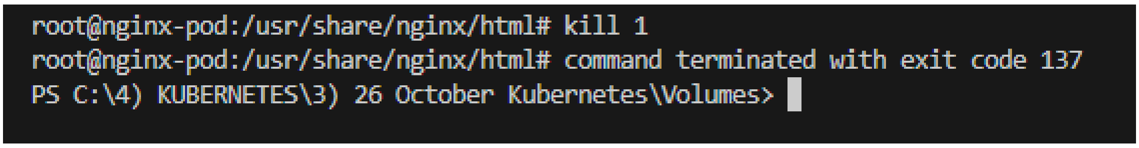
cd /usr/share/nginx/html

touch dhondhu.txt



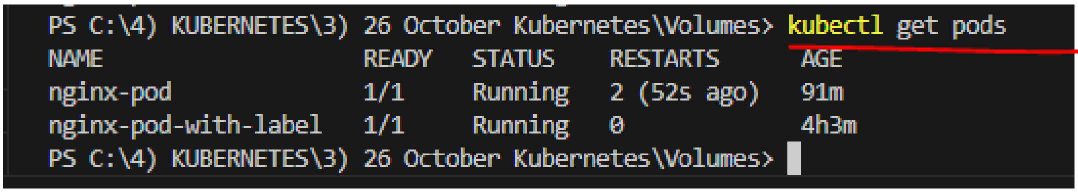
```
root@nginx-pod:/usr/share/nginx/html# touch dhondhu.txt
root@nginx-pod:/usr/share/nginx/html# ls
50x.html  dhondhu.txt  index.html
root@nginx-pod:/usr/share/nginx/html#
```

Kill 1 = container ko maar diya ya band ho gaya



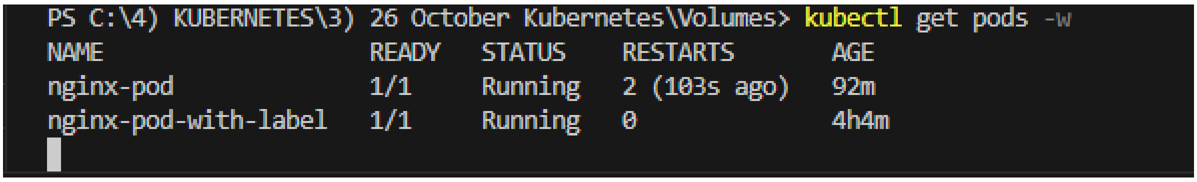
```
root@nginx-pod:/usr/share/nginx/html# kill 1
root@nginx-pod:/usr/share/nginx/html# command terminated with exit code 137
PS C:\4) KUBERNETES\3) 26 October Kubernetes\Volumes>
```

kubectl get pods



```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\Volumes> kubectl get pods
NAME                    READY   STATUS    RESTARTS      AGE
nginx-pod               1/1     Running   2 (52s ago)   91m
nginx-pod-with-label    1/1     Running   0             4h3m
PS C:\4) KUBERNETES\3) 26 October Kubernetes\Volumes>
```

kubectl get pods –w



```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\Volumes> kubectl get pods -w
NAME                    READY   STATUS    RESTARTS       AGE
nginx-pod               1/1     Running   2 (103s ago)   92m
nginx-pod-with-label    1/1     Running   0              4h4m
```

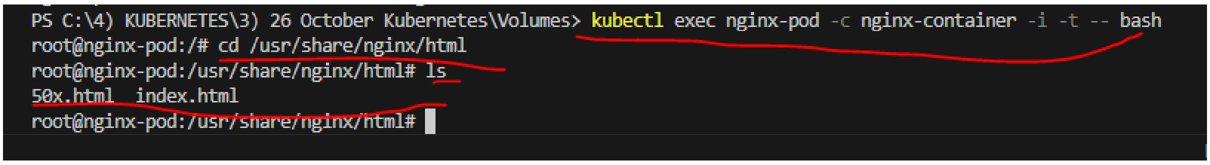
10) container mara to k8s ab use restart krega

11) So after restarting we can see now the container which is created by pod do not have dhondhu.txt file that we had created in previous container

kubectl exec nginx-pod -c nginx-container -i -t -- bash

cd /usr/share/nginx/html

ls



```
PS C:\4) KUBERNETES\3) 26 October Kubernetes\Volumes> kubectl exec nginx-pod -c nginx-container -i -t -- bash
root@nginx-pod:/# cd /usr/share/nginx/html
root@nginx-pod:/usr/share/nginx/html# ls
50x.html  index.html
root@nginx-pod:/usr/share/nginx/html#
```

12) Now it's a great pain that file is deleted or lost in new container so basically data is lost so we will find solution for the same.