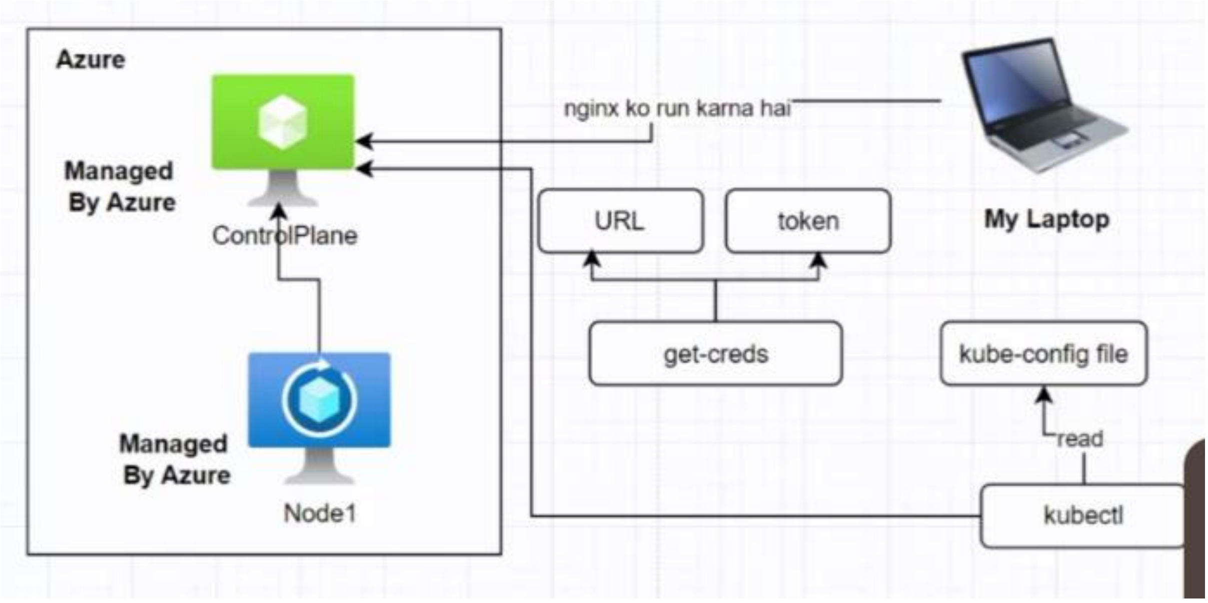
13 October

1) Kubernetes is a paas service

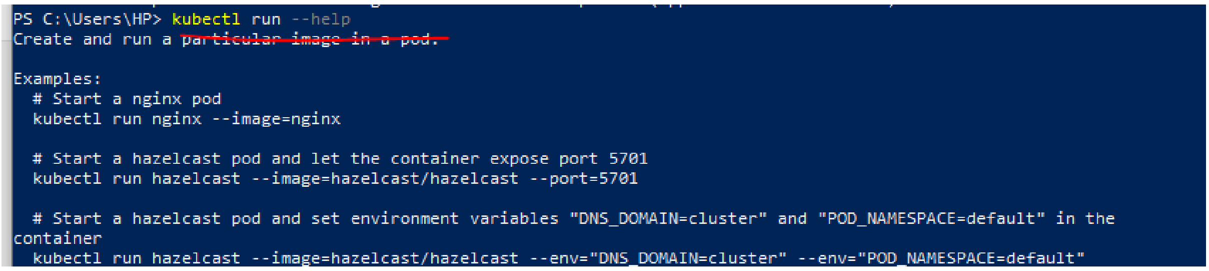
2) Credentials of cluster api address have been stored in kubectl file

3) On my laptop 2 things are stored – kubelet and kubeproxy

4)

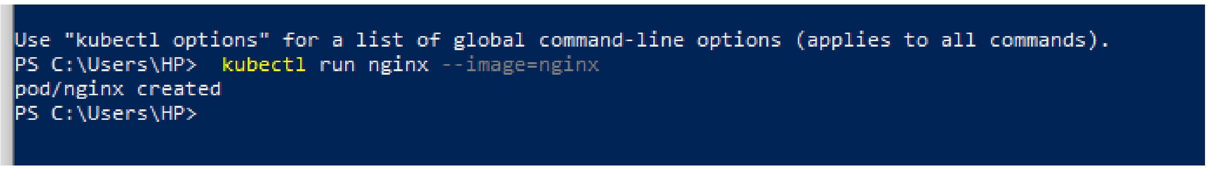


5) **kubectl run --help**



```
PS C:\Users\HP> kubectl run --help
Create and run a ~~particular image in a pod.~~

Examples:
  # Start a nginx pod
  kubectl run nginx --image=nginx

  # Start a hazelcast pod and let the container expose port 5701
  kubectl run hazelcast --image=hazelcast/hazelcast --port=5701

  # Start a hazelcast pod and set environment variables "DNS_DOMAIN=cluster" and "POD_NAMESPACE=default" in the
container
  kubectl run hazelcast --image=hazelcast/hazelcast --env="DNS_DOMAIN=cluster" --env="POD_NAMESPACE=default"
```

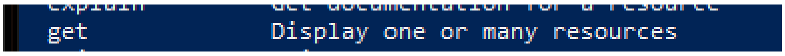
6) **kubectl run nginx --image=nginx**



```
Use "kubectl options" for a list of global command-line options (applies to all commands).
PS C:\Users\HP>  kubectl run nginx --image=nginx
pod/nginx created
PS C:\Users\HP>
```

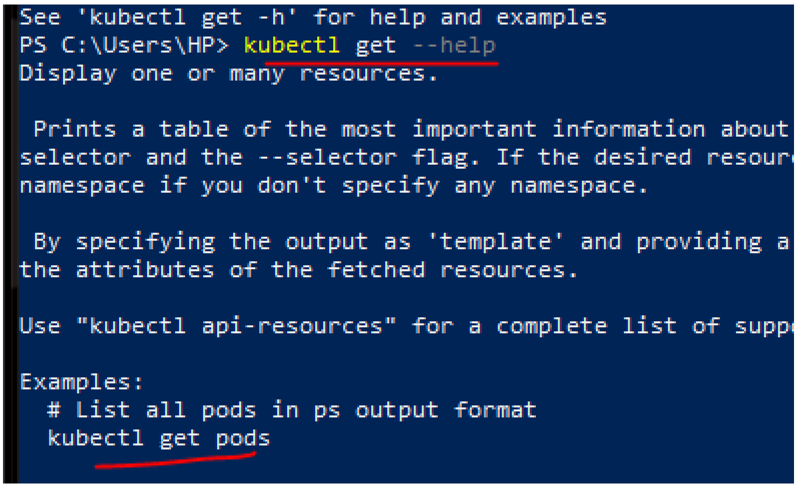
Now firstly kubectl gone to kube-config file and read the api server url and token and kubectl send the information about installing of nginx on api server (in control plane). Then api server go to scheduler and ask which node is free then after getting free node the kubelet in node will be asked to run nginx pod.

7) Now we have to check what is made in our cluster then how we will see

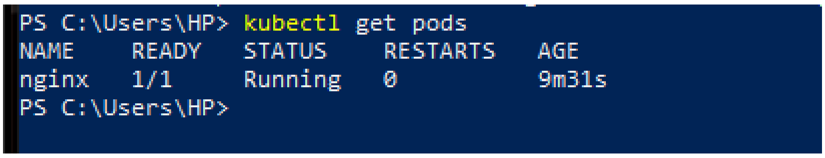


```
  explain         Get documentation for a resource
  get             Display one or many resources
```

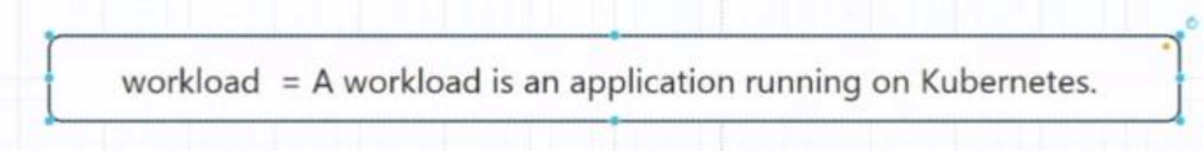
8) **kubectl get pods**



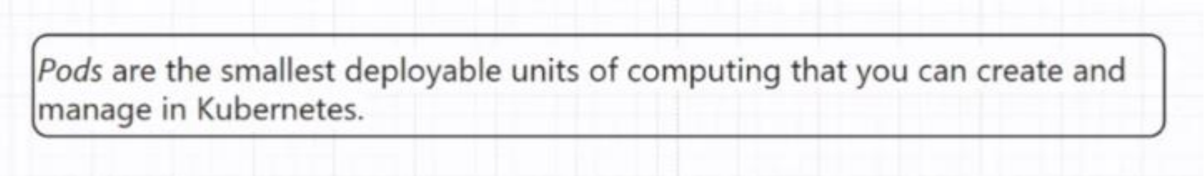
```
See 'kubectl get -h' for help and examples
PS C:\Users\HP> kubectl get --help
Display one or many resources.

 Prints a table of the most important information about
selector and the --selector flag. If the desired resour
namespace if you don't specify any namespace.

 By specifying the output as 'template' and providing a
the attributes of the fetched resources.

Use "kubectl api-resources" for a complete list of supp

Examples:
  # List all pods in ps output format
  kubectl get pods
```



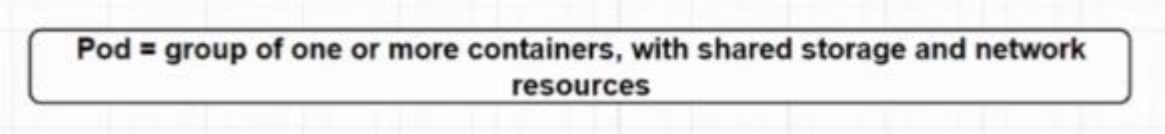
```
PS C:\Users\HP> kubectl get pods
NAME    READY   STATUS    RESTARTS   AGE
nginx   1/1     Running   0          9m31s
PS C:\Users\HP>
```



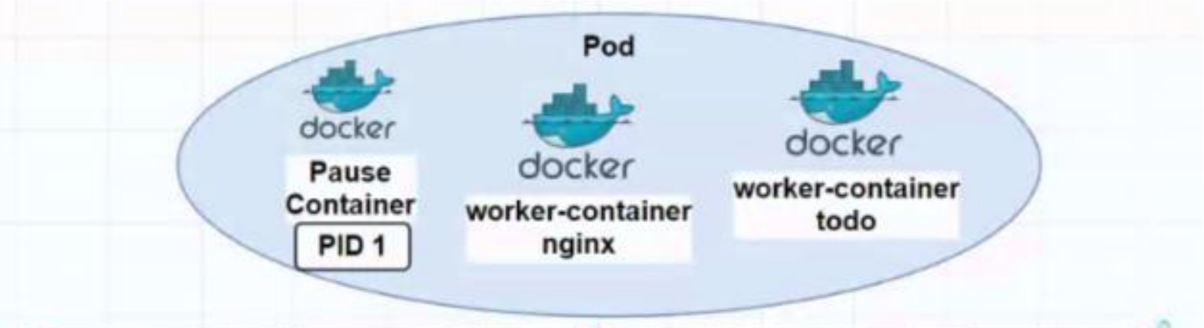
workload = A workload is an application running on Kubernetes.

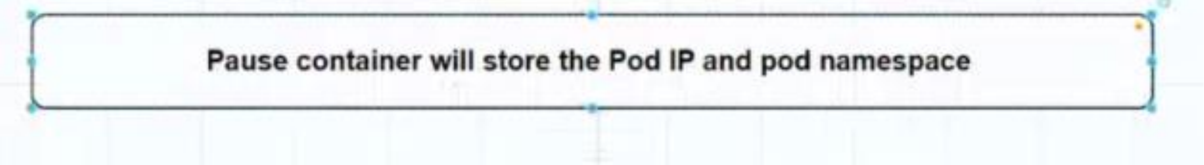


Pods are the smallest deployable units of computing that you can create and manage in Kubernetes.

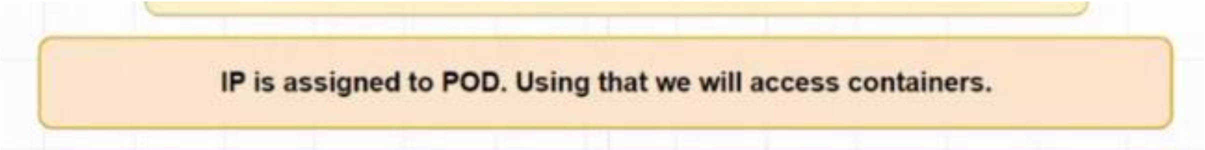


Pod = group of one or more containers, with shared storage and network resources



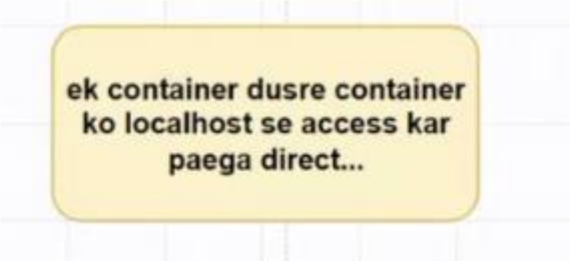


Pause container will store the Pod IP and pod namespace

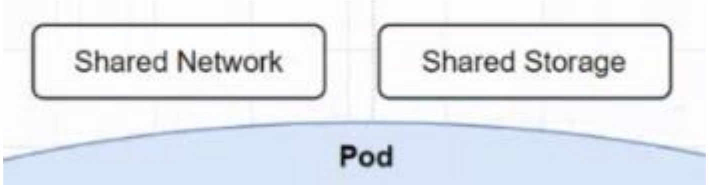


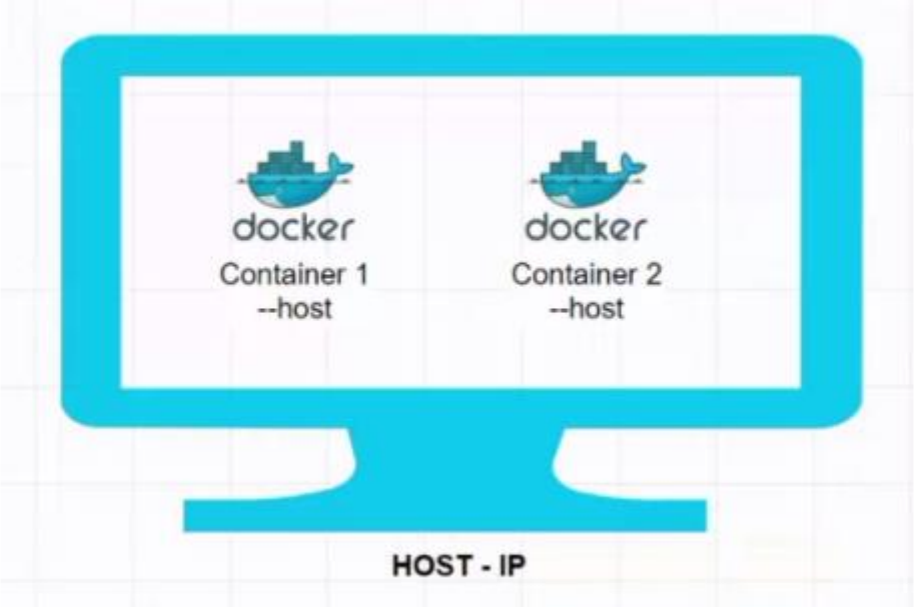
IP is assigned to POD. Using that we will access containers.

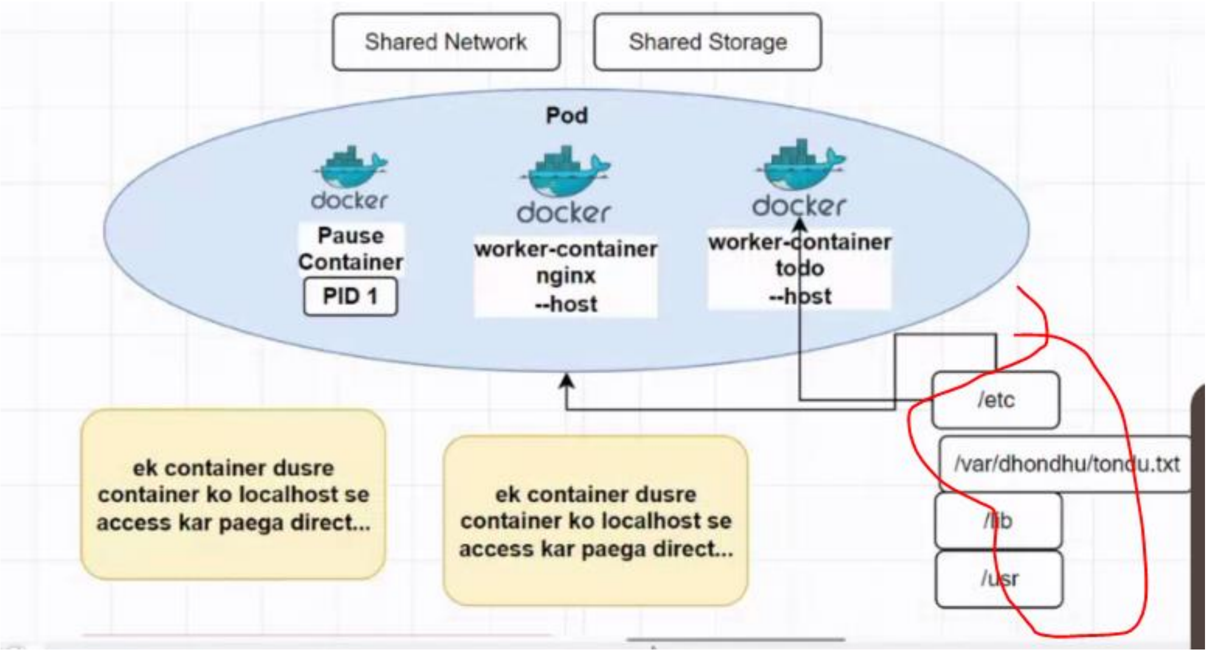
9) Inside pod all containers will use shared networking concept as they will use same ip address and network. And will also share the same storage also



ek container dusre container ko localhost se access kar paega direct...







Shared Network

Shared Storage

Pod

docker Pause Container PID 1

docker worker-container nginx --host

docker worker-container todo --host

/etc

/var/dhondhu/tondu.txt

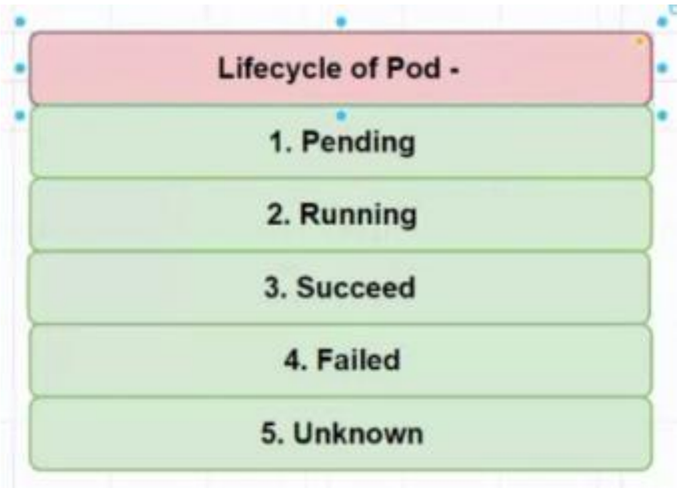
/lib

/usr

ek container dusre container ko localhost se access kar paega direct...

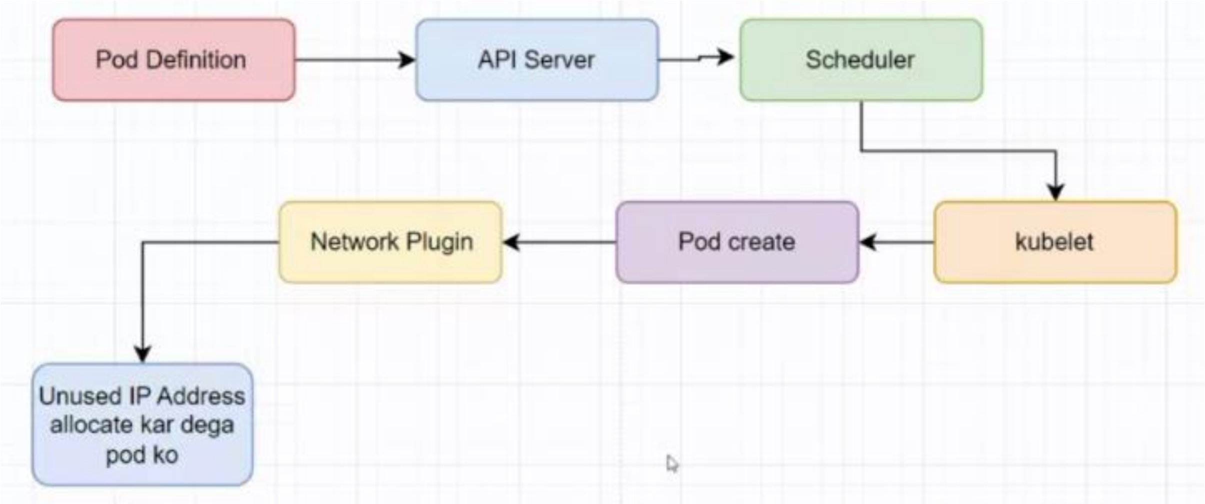
ek container dusre container ko localhost se access kar paega direct...

10) pod lifecycle



Lifecycle of Pod -

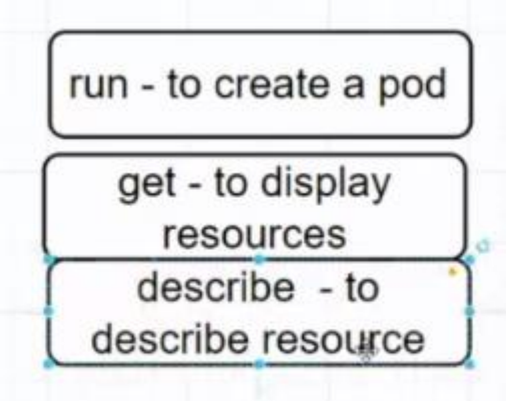
1. Pending
2. Running
3. Succeed
4. Failed
5. Unknown



Pod Definition → API Server → Scheduler → kubelet → Pod create → Network Plugin → Unused IP Address allocate kar dega pod ko

*) How ip is assigned to plugin = by cni plugin (Container Network Interface (CNI))

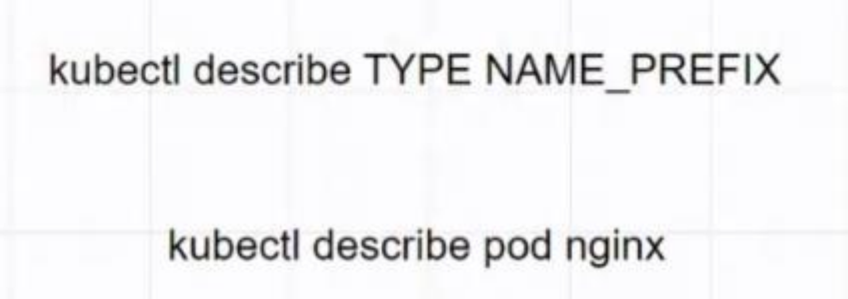
11)



run - to create a pod

get - to display resources

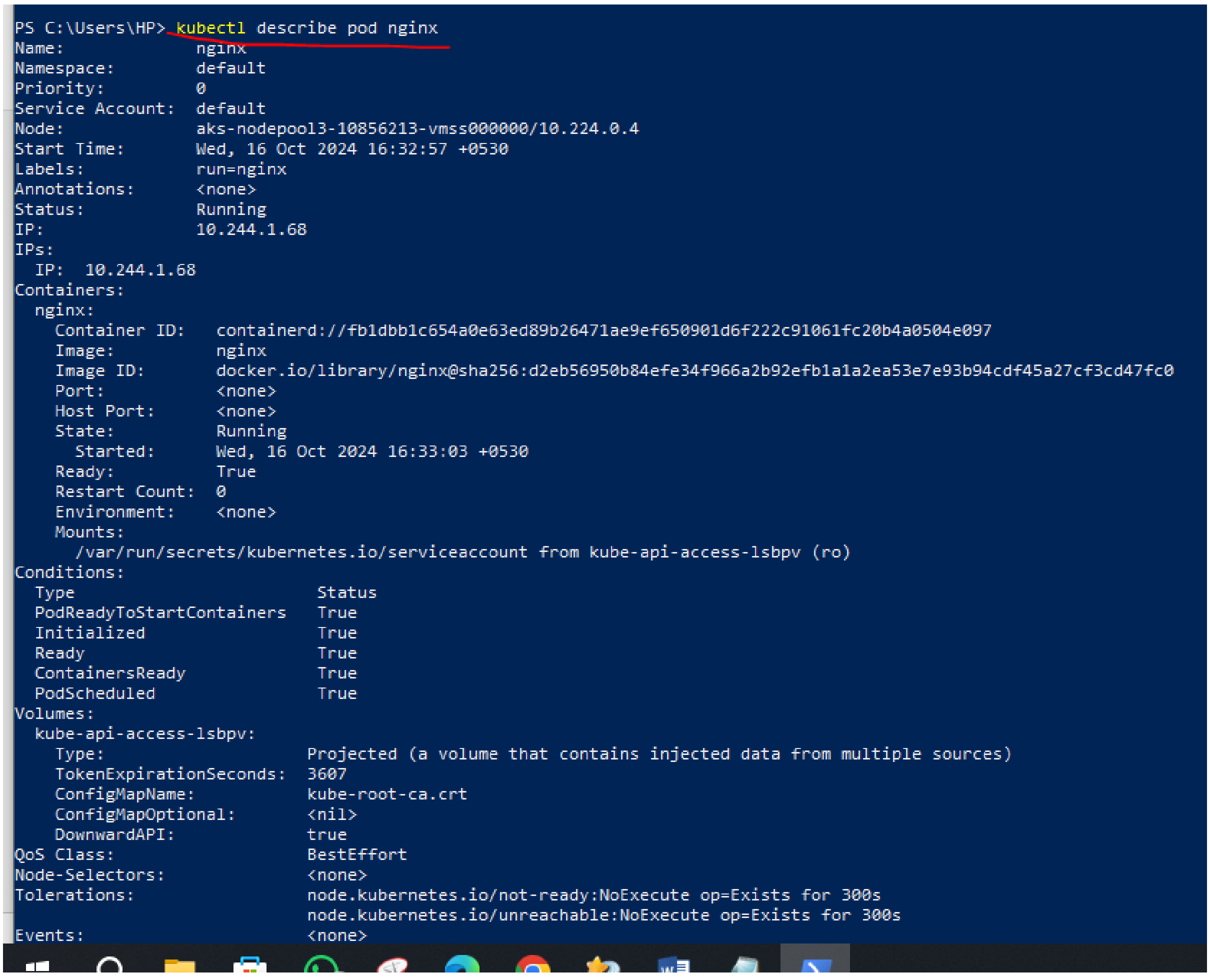
describe - to describe resource



kubectl describe TYPE NAME_PREFIX

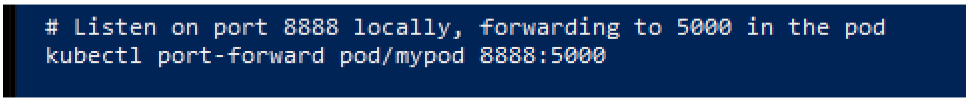
kubectl describe pod nginx

12) **kubectl describe pod nginx**



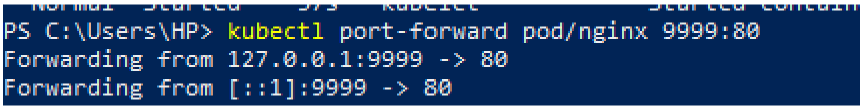
```
PS C:\Users\HP> kubectl describe pod nginx
Name:             nginx
Namespace:        default
Priority:         0
Service Account:  default
Node:             aks-nodepool3-10856213-vmss000000/10.224.0.4
Start Time:       Wed, 16 Oct 2024 16:32:57 +0530
Labels:           run=nginx
Annotations:      <none>
Status:           Running
IP:               10.244.1.68
IPs:
  IP:  10.244.1.68
Containers:
  nginx:
    Container ID:   containerd://fb1dbb1c654a0e63ed89b26471ae9ef650901d6f222c91061fc20b4a0504e097
    Image:          nginx
    Image ID:       docker.io/library/nginx@sha256:d2eb56950b84efe34f966a2b92efb1a1a2ea53e7e93b94cdf45a27cf3cd47fc0
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Wed, 16 Oct 2024 16:33:03 +0530
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-lsbpv (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-lsbpv:
    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:                      <none>
```

13) Now since we need to run nginx so we will do port forwarding



```
# Listen on port 8888 locally, forwarding to 5000 in the pod
kubectl port-forward pod/mypod 8888:5000
```

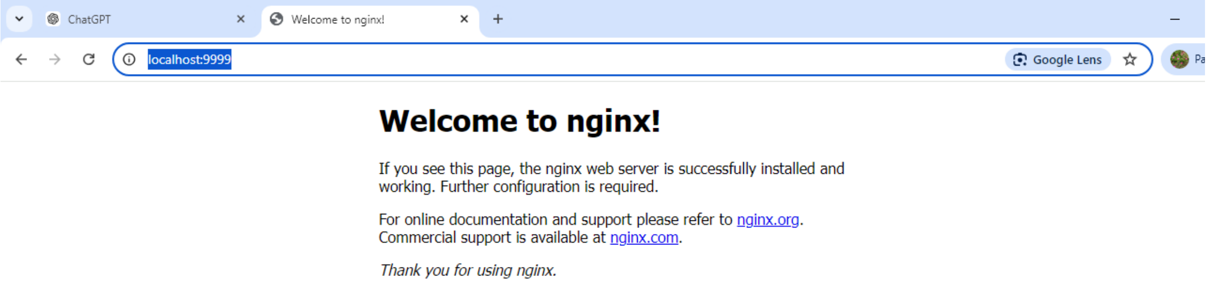
14) **kubectl port-forward pod/nginx 9999:80**



```
  Normal   Started    37s   kubelet            Started contain
PS C:\Users\HP> kubectl port-forward pod/nginx 9999:80
Forwarding from 127.0.0.1:9999 -> 80
Forwarding from [::1]:9999 -> 80
```

15) Run in browser – localhost:9999

http://localhost:9999/



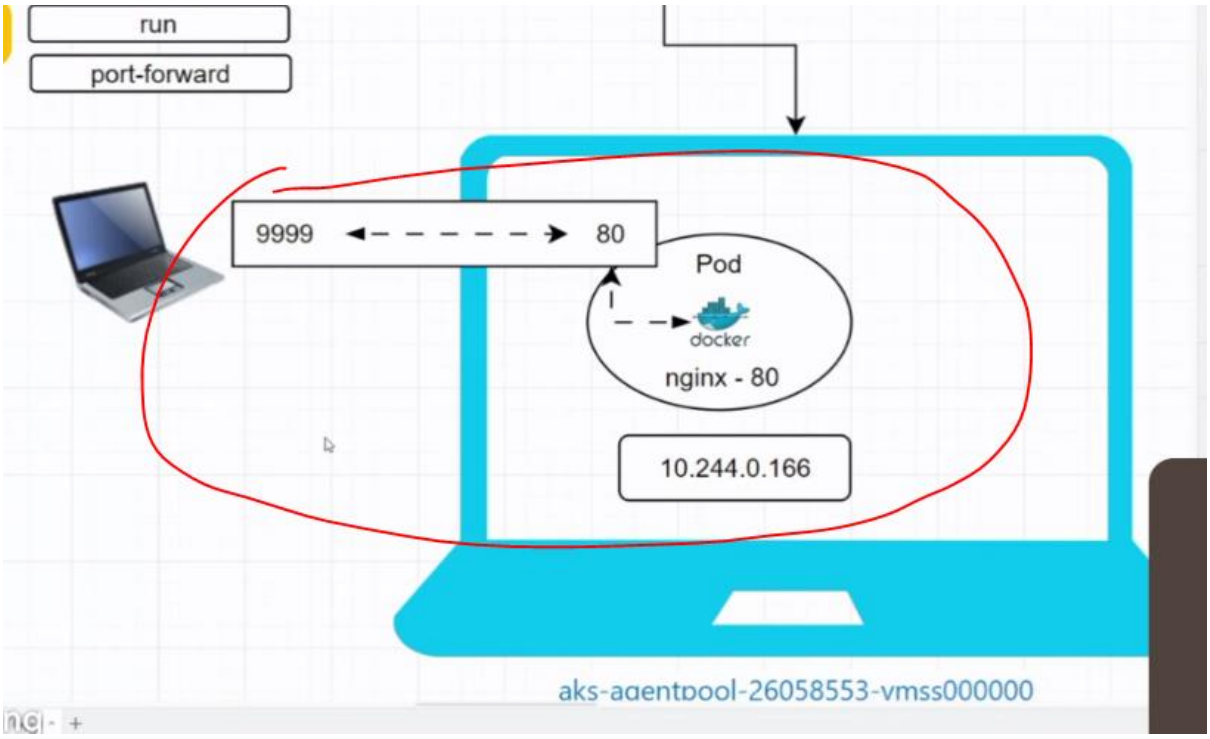
Welcome to nginx!

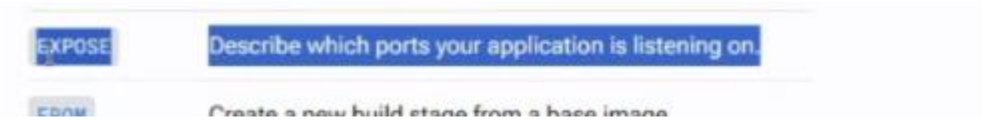
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

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

Thank you for using nginx.

16) Since tunnel is created from host to pod and then pod to container. So in pod port syncing between pod and container is already done by **expose command** in docker file. Or the tunnel between pod and container is already made by expose command in docker file.

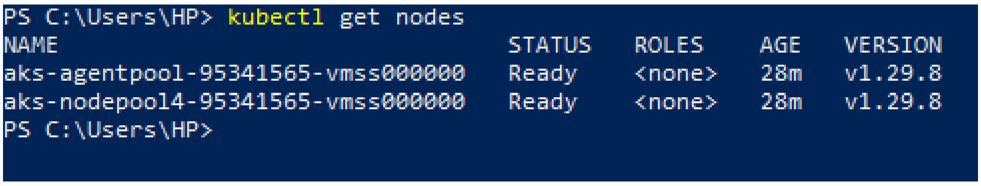




EXPOSE   Describe which ports your application is listening on.

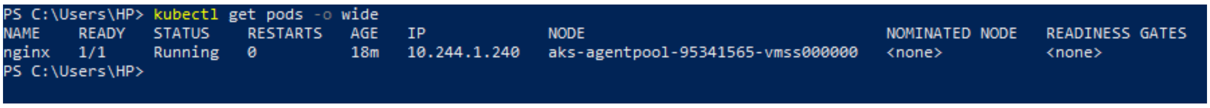
FROM   Create a new build stage from a base image

17) **kubectl get nodes** – how many nodes are running



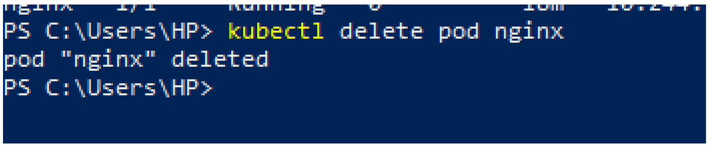
```
PS C:\Users\HP> kubectl get nodes
NAME                                STATUS   ROLES    AGE   VERSION
aks-agentpool-95341565-vmss000000   Ready    <none>   28m   v1.29.8
aks-nodepool4-95341565-vmss000000   Ready    <none>   28m   v1.29.8
PS C:\Users\HP>
```

18) **kubectl get pods -o wide** = more info about pod

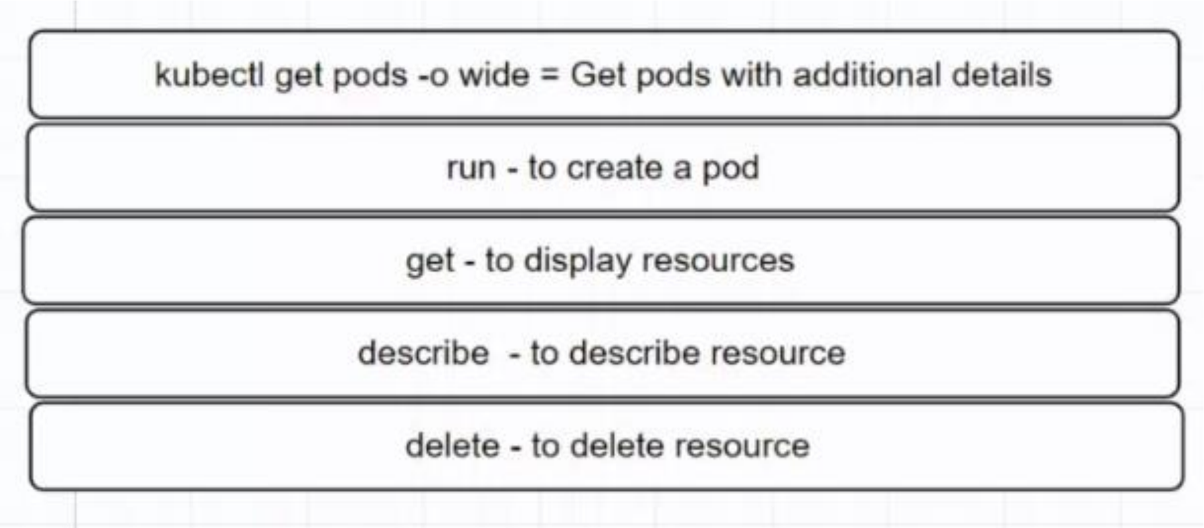


```
PS C:\Users\HP> kubectl get pods -o wide
NAME    READY   STATUS    RESTARTS   AGE   IP             NODE                                NOMINATED NODE   READINESS GATES
nginx   1/1     Running   0          18m   10.244.1.240   aks-agentpool-95341565-vmss000000   <none>           <none>
PS C:\Users\HP>
```

19) **kubectl delete pod nginx** - to delete the pod



```
nginx   1/1   Running   0   18m   10.244.
PS C:\Users\HP> kubectl delete pod nginx
pod "nginx" deleted
PS C:\Users\HP>
```



kubectl get pods -o wide = Get pods with additional details

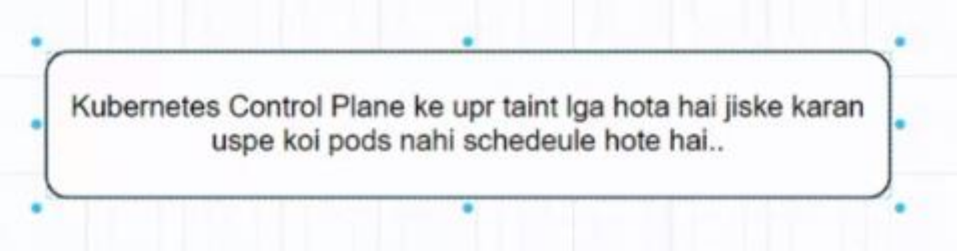
run - to create a pod

get - to display resources

describe - to describe resource

delete - to delete resource

20) Why workload run on worker nodes not on master computer? Taint laga hota hai

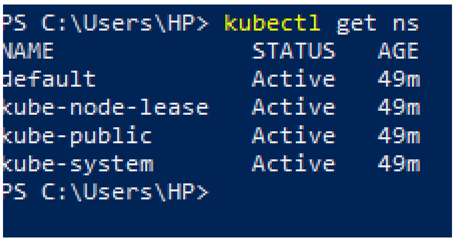


Kubernetes Control Plane ke upr taint lga hota hai jiske karan uspe koi pods nahi schedeule hote hai..

+++

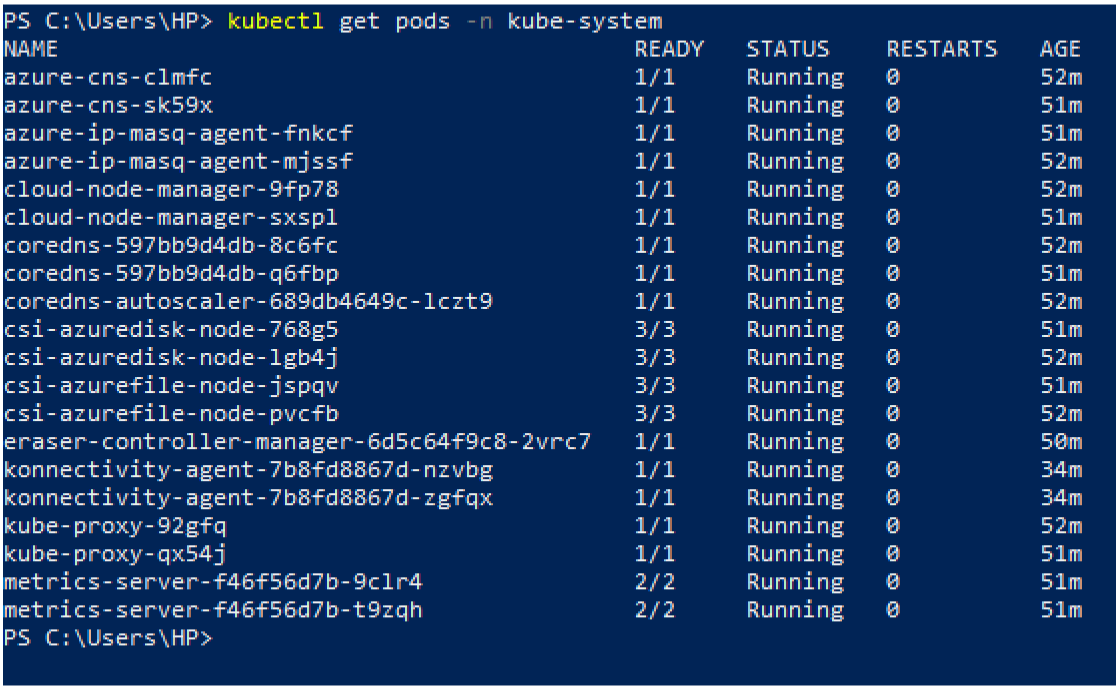
AGENDA – CREATE A NAMESPACE

1) **kubectl get ns** – namespace



```
PS C:\Users\HP> kubectl get ns
NAME              STATUS   AGE
default           Active   49m
kube-node-lease   Active   49m
kube-public       Active   49m
kube-system       Active   49m
PS C:\Users\HP>
```

2) **kubectl get pods -n kube-system**

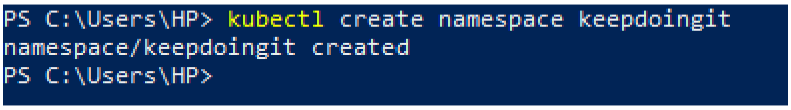


```
PS C:\Users\HP> kubectl get pods -n kube-system
NAME                                         READY   STATUS    RESTARTS   AGE
azure-cns-clmfc                              1/1     Running   0          52m
azure-cns-sk59x                              1/1     Running   0          51m
azure-ip-masq-agent-fnkcf                    1/1     Running   0          51m
azure-ip-masq-agent-mjssf                    1/1     Running   0          52m
cloud-node-manager-9fp78                     1/1     Running   0          52m
cloud-node-manager-sxspl                     1/1     Running   0          51m
coredns-597bb9d4db-8c6fc                     1/1     Running   0          52m
coredns-597bb9d4db-q6fbp                     1/1     Running   0          51m
coredns-autoscaler-689db4649c-lczt9          1/1     Running   0          52m
csi-azuredisk-node-768g5                     3/3     Running   0          51m
csi-azuredisk-node-lgb4j                     3/3     Running   0          52m
csi-azurefile-node-jspqv                     3/3     Running   0          51m
csi-azurefile-node-pvcfb                     3/3     Running   0          52m
eraser-controller-manager-6d5c64f9c8-2vrc7   1/1     Running   0          50m
konnectivity-agent-7b8fd8867d-nzvbg          1/1     Running   0          34m
konnectivity-agent-7b8fd8867d-zgfqx          1/1     Running   0          34m
kube-proxy-92gfq                             1/1     Running   0          52m
kube-proxy-qx54j                             1/1     Running   0          51m
metrics-server-f46f56d7b-9clr4               2/2     Running   0          51m
metrics-server-f46f56d7b-t9zqh               2/2     Running   0          51m
PS C:\Users\HP>
```

3) Similarly run commands for other namespaces

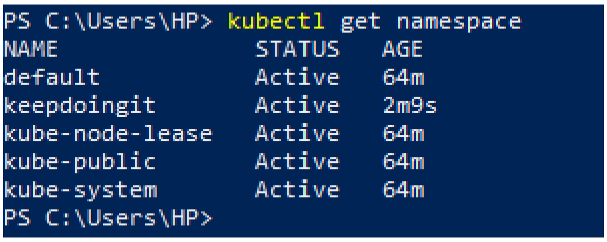
4) Now for creating namespace

kubectl create namespace keepdoingit - create namespace



```
PS C:\Users\HP> kubectl create namespace keepdoingit
namespace/keepdoingit created
PS C:\Users\HP>
```

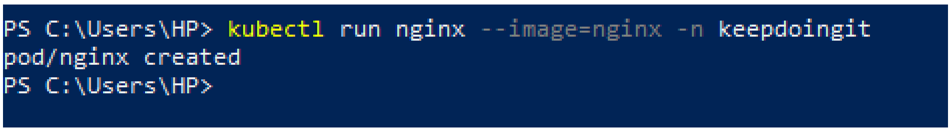
5) **kubectl get namespace**



```
PS C:\Users\HP> kubectl get namespace
NAME              STATUS   AGE
default           Active   64m
keepdoingit       Active   2m9s
kube-node-lease   Active   64m
kube-public       Active   64m
kube-system       Active   64m
PS C:\Users\HP>
```

6) Now run pod in this namespace – so run command helps us to create pod

kubectl run nginx --image=nginx -n keepdoingit



```
PS C:\Users\HP> kubectl run nginx --image=nginx -n keepdoingit
pod/nginx created
PS C:\Users\HP>
```

The command `**kubectl run nginx --image=nginx -n keepdoingit**` creates a new Kubernetes pod named `**nginx**` in the namespace `**keepdoingit**`, using the `nginx` image.

Here's a breakdown:

- `kubectl run`: Command to create a new pod.

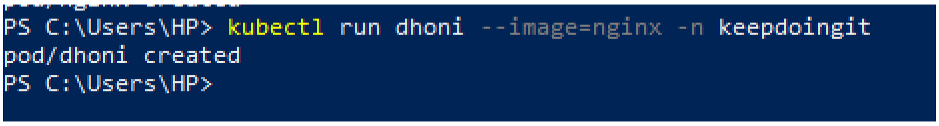
- `nginx`: Name of the pod.

- `--image=nginx`: Specifies the container image to use (in this case, the official Nginx image).

- `-n keepdoingit`: Indicates the namespace where the pod will be created, which is `keepdoingit`.

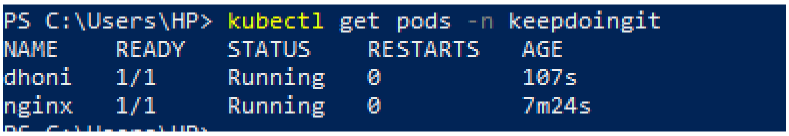
This command essentially sets up an **Nginx web server** in the specified namespace.

kubectl run dhoni --image=nginx -n keepdoingit



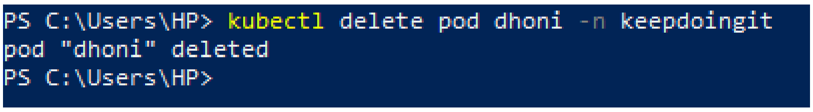
```
PS C:\Users\HP> kubectl run dhoni --image=nginx -n keepdoingit
pod/dhoni created
PS C:\Users\HP>
```

7) **kubectl get pods -n keepdoingit** = displays a list of all pods in the Kubernetes namespace "keepdoingit"



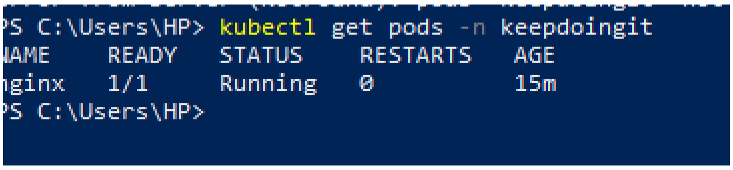
```
PS C:\Users\HP> kubectl get pods -n keepdoingit
NAME     READY   STATUS    RESTARTS   AGE
dhoni    1/1     Running   0          107s
nginx    1/1     Running   0          7m24s
```

8) **kubectl delete pod dhoni -n keepdoingit** – deleted pod dhoni



```
PS C:\Users\HP> kubectl delete pod dhoni -n keepdoingit
pod "dhoni" deleted
PS C:\Users\HP>
```

9) **kubectl get pods -n keepdoingit** - check running pods in namespace keepdoingit



```
S C:\Users\HP> kubectl get pods -n keepdoingit
AME     READY   STATUS    RESTARTS   AGE
ginx    1/1     Running   0          15m
S C:\Users\HP>
```

10) Now we can see after deleting this dhoni pod, controller should auto create the new pod but it has not so for this **pain** we will see how controller will create automatically new pods after pods die