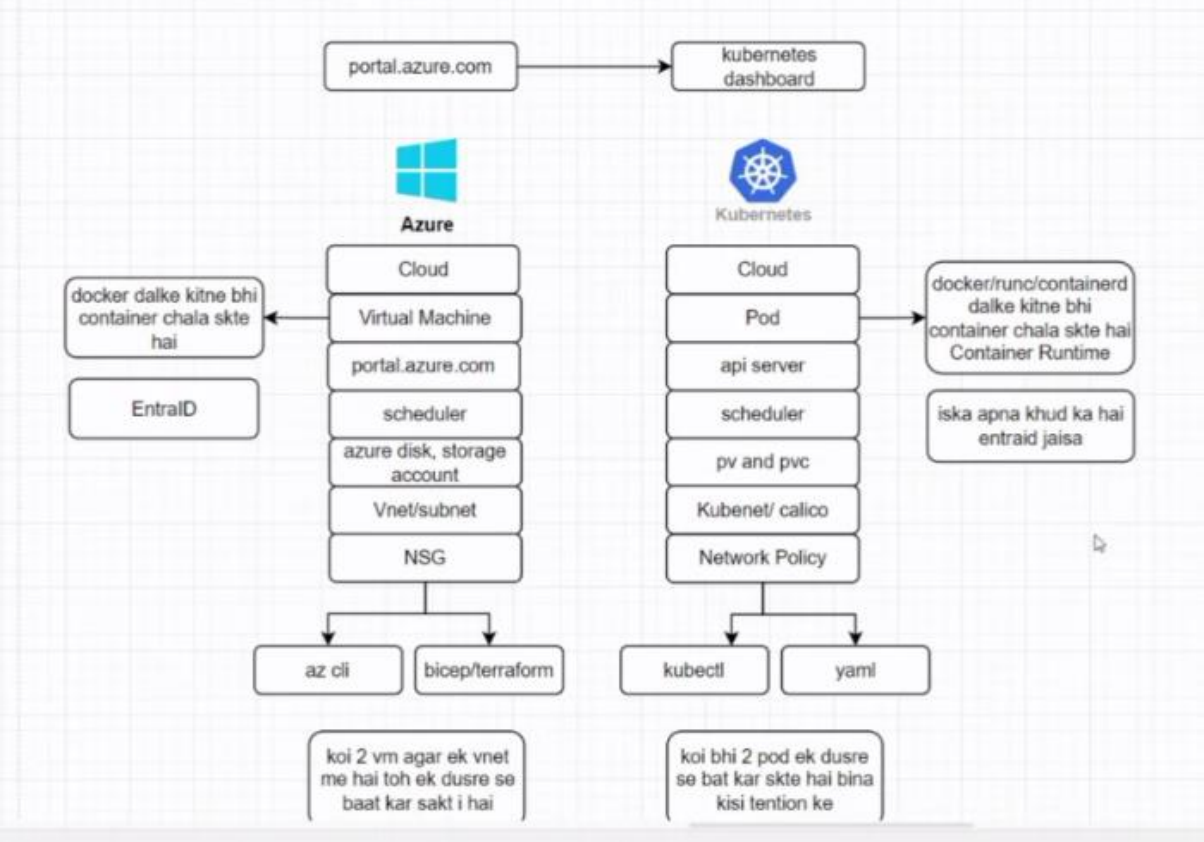
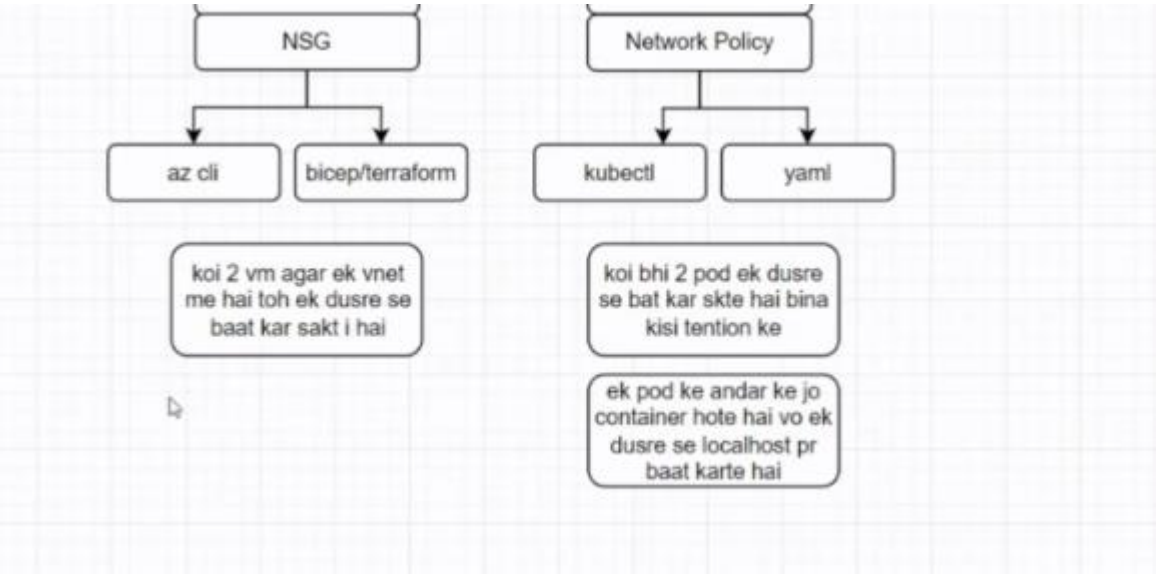
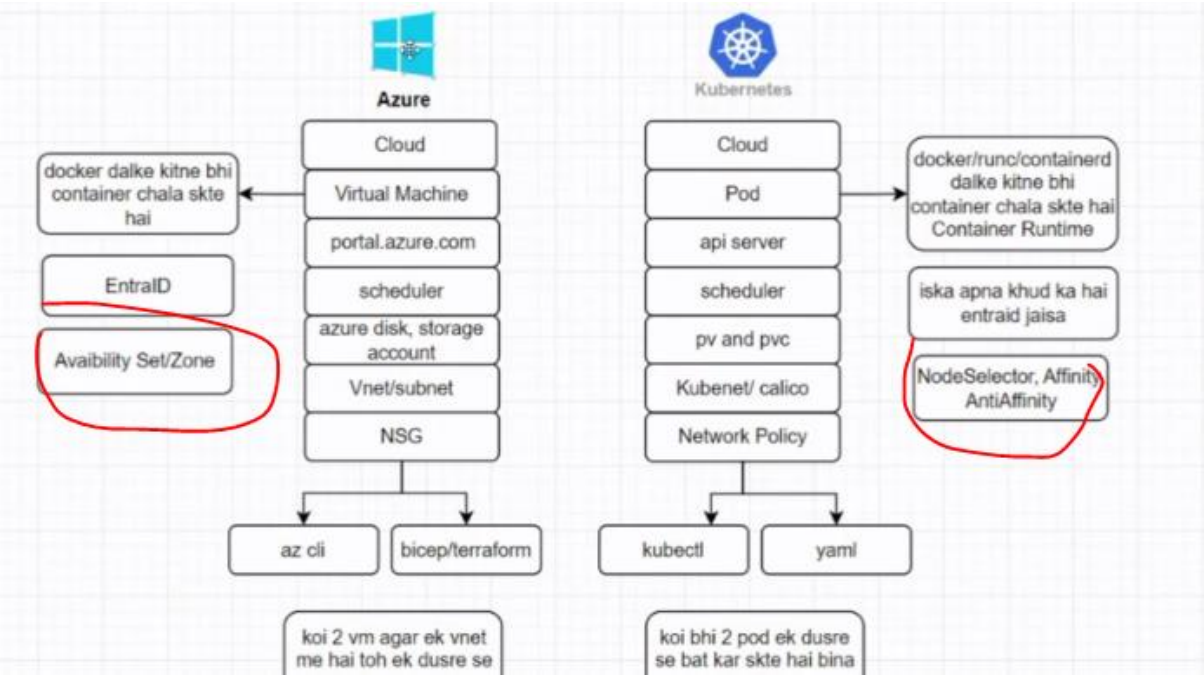
3 Nov

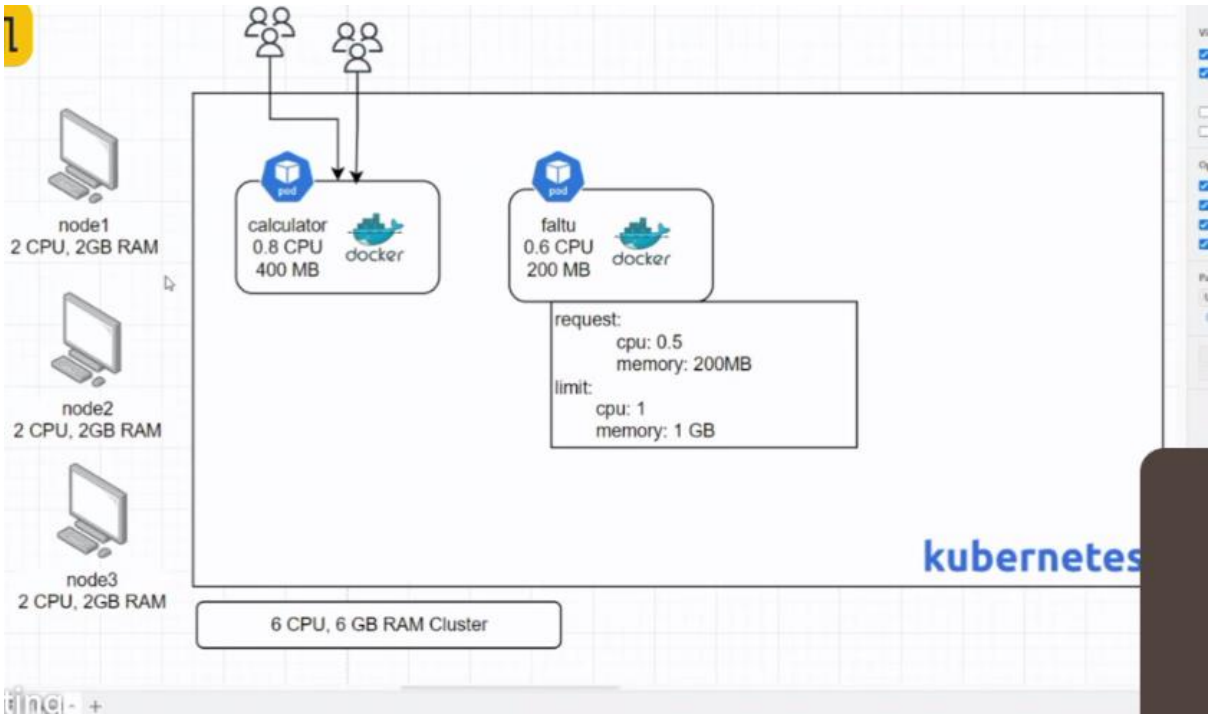
1)


portal.azure.com
kubernetes dashboard
Azure
Kubernetes
Cloud
Virtual Machine
portal.azure.com
scheduler
azure disk, storage account
Vnet/subnet
NSG
docker dalke kitne bhi container chala skte hai
EntraID
Cloud
Pod
api server
scheduler
pv and pvc
Kubenet/ calico
Network Policy
docker/runc/containerd dalke kitne bhi container chala skte hai Container Runtime
iska apna khud ka hai entraid jaisa
az cli
bicep/terraform
kubectl
yaml
koi 2 vm agar ek vnet me hai toh ek dusre se baat kar sakt i hai
koi bhi 2 pod ek dusre se bat kar skte hai bina kisi tention ke

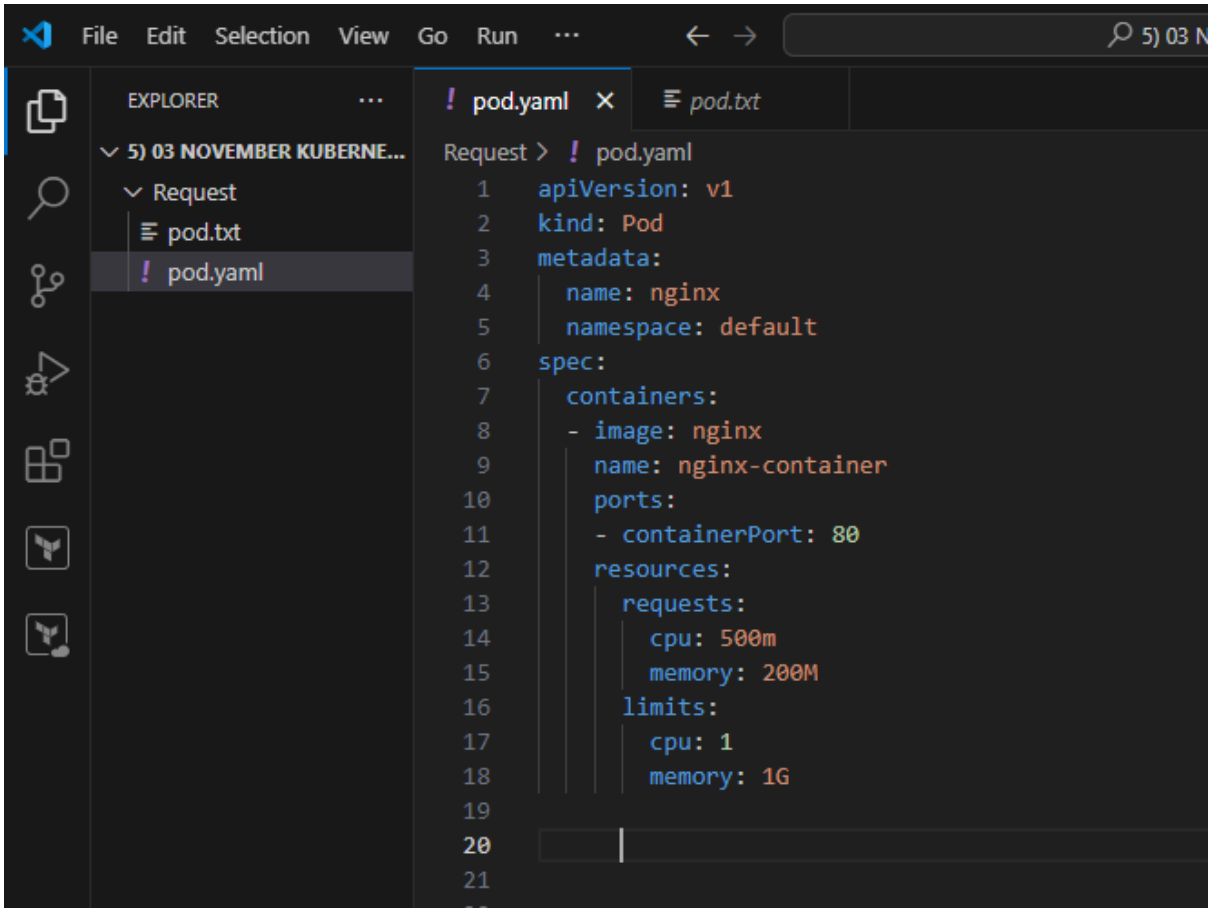

NSG
Network Policy
az cli
bicep/terraform
kubectl
yaml
koi 2 vm agar ek vnet me hai toh ek dusre se baat kar sakt i hai
koi bhi 2 pod ek dusre se bat kar skte hai bina kisi tention ke
ek pod ke andar ke jo container hote hai vo ek dusre se localhost pr baat karte hai



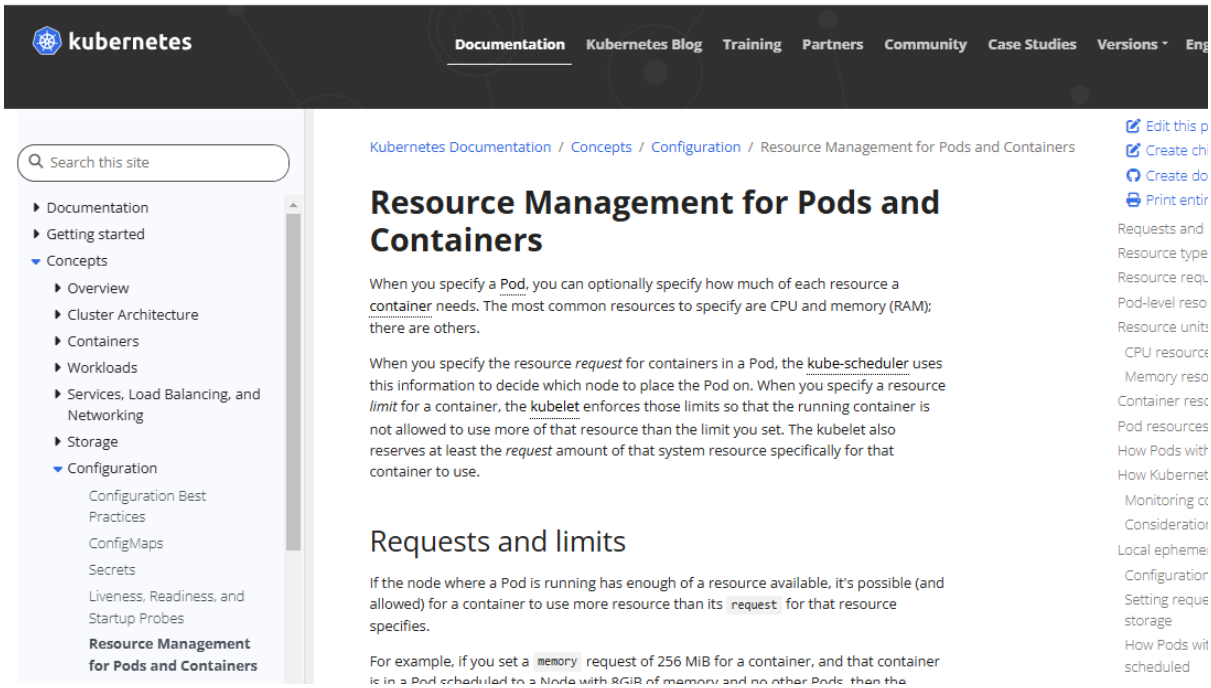
2) Difference between request and limit



2) Create folder “5) 03 November Kubernetes” and create folder “Request” and then file “pod.yaml”.

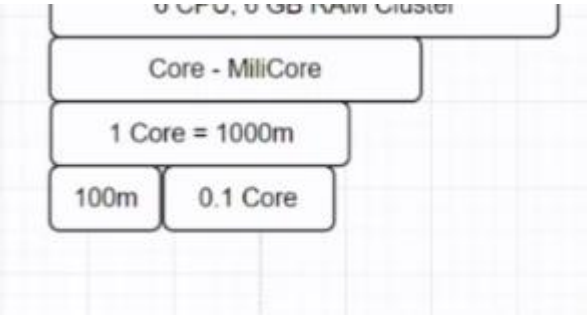


3) search = https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/



- `spec.containers[].resources.limits.cpu`
- `spec.containers[].resources.limits.memory`
- `spec.containers[].resources.limits.hugepages-<size>`
- `spec.containers[].resources.requests.cpu`
- `spec.containers[].resources.requests.memory`
- `spec.containers[].resources.requests.hugepages-<size>`

Kubernetes, 1 CPU unit is equivalent to **1 physical CPU core**, or **1 virtual core**, depending on whether the node is a physical host or a virtual machine



4) **kubectl apply -f pod.yaml** = create pod

```
PS C:\4) KUBERNETES\5) 03 November Kubernetes\Request> kubectl apply -f pod.yaml
pod/nginx created
PS C:\4) KUBERNETES\5) 03 November Kubernetes\Request>
```

5) **kubectl get pods**

```
PS C:\4) KUBERNETES\5) 03 November Kubernetes\Request> kubectl get pods
NAME    READY   STATUS    RESTARTS   AGE
nginx   1/1     Running   0          2m11s
PS C:\4) KUBERNETES\5) 03 November Kubernetes\Request>
```

6) **kubectl top pod** = pod kitni memory kha raha hai

```
PS C:\4) KUBERNETES\5) 03 November Kubernetes\Request> kubectl top pod
NAME    CPU(cores)   MEMORY(bytes)
nginx   0m           3Mi
PS C:\4) KUBERNETES\5) 03 November Kubernetes\Request>
```

7) SEARCH = stress utility linux

https://www.geeksforgeeks.org/linux-stress-command-with-examples/

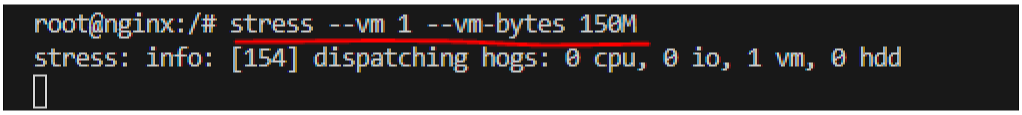
8) **kubectl exec nginx -c nginx-container -i -t – bash** = go inside container

```
PS C:\4) KUBERNETES\5) 03 November Kubernetes\Request> kubectl exec nginx -c nginx-container -i -t -- bash
root@nginx:/#
```

9) **apt update**

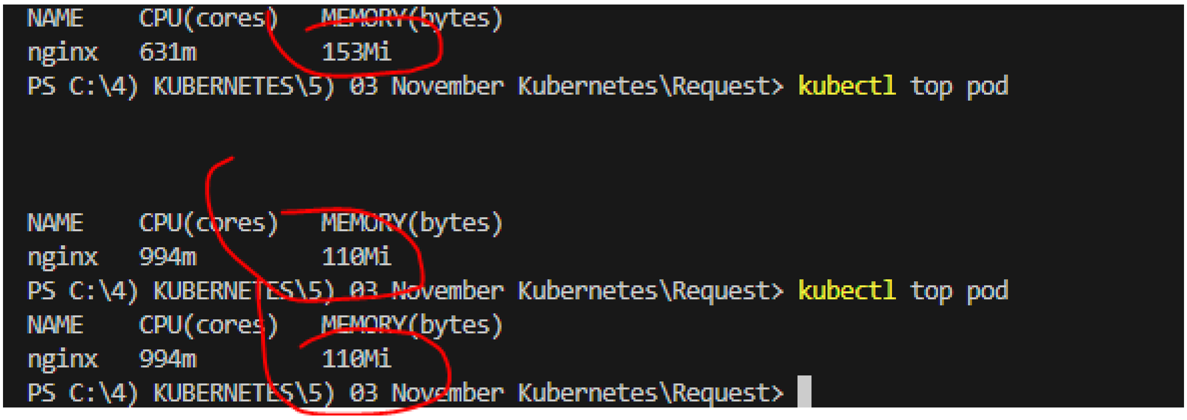
apt install stress

10) **stress --vm 1 --vm-bytes 150M**



```
root@nginx:/# stress --vm 1 --vm-bytes 150M
stress: info: [154] dispatching hogs: 0 cpu, 0 io, 1 vm, 0 hdd
```

11) **kubectl top pod**

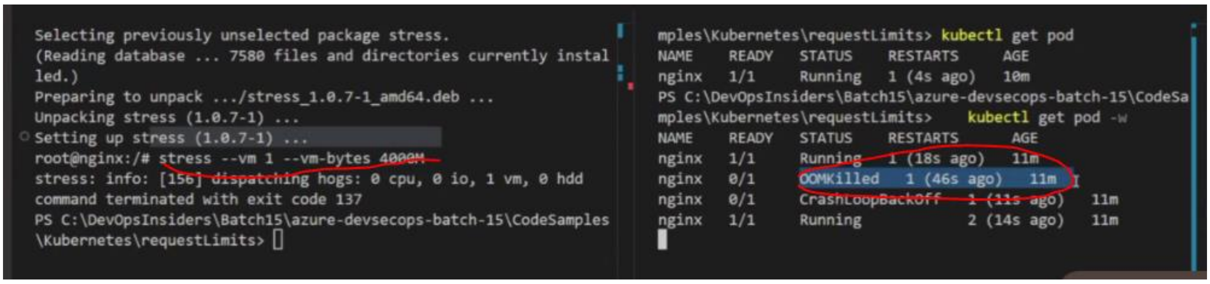


```
NAME    CPU(cores)   MEMORY(bytes)
nginx   631m         153Mi
PS C:\4) KUBERNETES\5) 03 November Kubernetes\Request> kubectl top pod



NAME    CPU(cores)   MEMORY(bytes)
nginx   994m         110Mi
PS C:\4) KUBERNETES\5) 03 November Kubernetes\Request> kubectl top pod
NAME    CPU(cores)   MEMORY(bytes)
nginx   994m         110Mi
PS C:\4) KUBERNETES\5) 03 November Kubernetes\Request>
```

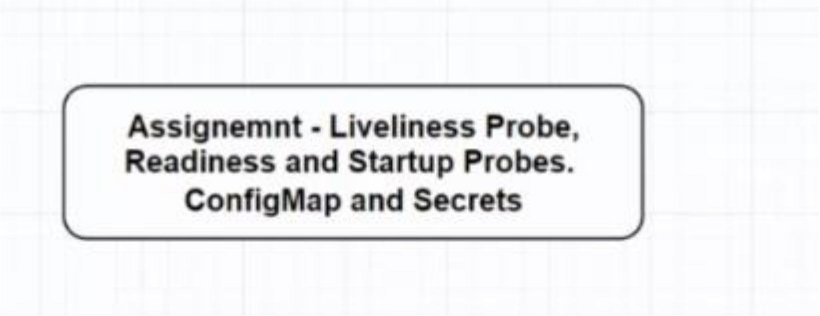
12) jyada stess se container mar gaya tha. OOM = out of memory



```
Selecting previously unselected package stress.
(Reading database ... 7580 files and directories currently instal
led.)
Preparing to unpack .../stress_1.0.7-1_amd64.deb ...
Unpacking stress (1.0.7-1) ...
Setting up stress (1.0.7-1) ...
root@nginx:/# stress --vm 1 --vm-bytes 4000M
stress: info: [156] dispatching hogs: 0 cpu, 0 io, 1 vm, 0 hdd
command terminated with exit code 137
PS C:\DevOpsInsiders\Batch15\azure-devsecops-batch-15\CodeSamples
\Kubernetes\requestLimits> 
```

```
mples\Kubernetes\requestLimits> kubectl get pod
NAME    READY   STATUS    RESTARTS     AGE
nginx   1/1     Running   1 (4s ago)   10m
PS C:\DevOpsInsiders\Batch15\azure-devsecops-batch-15\CodeSa
mples\Kubernetes\requestLimits>    kubectl get pod -w
NAME    READY   STATUS    RESTARTS      AGE
nginx   1/1     Running   1 (18s ago)   11m
nginx   0/1     OOMKilled   1 (46s ago)   11m
nginx   0/1     CrashLoopBackOff   1 (11s ago)   11m
nginx   1/1     Running            2 (14s ago)   11m
```

NOTE: 1) What is CrashLoopBackoff error

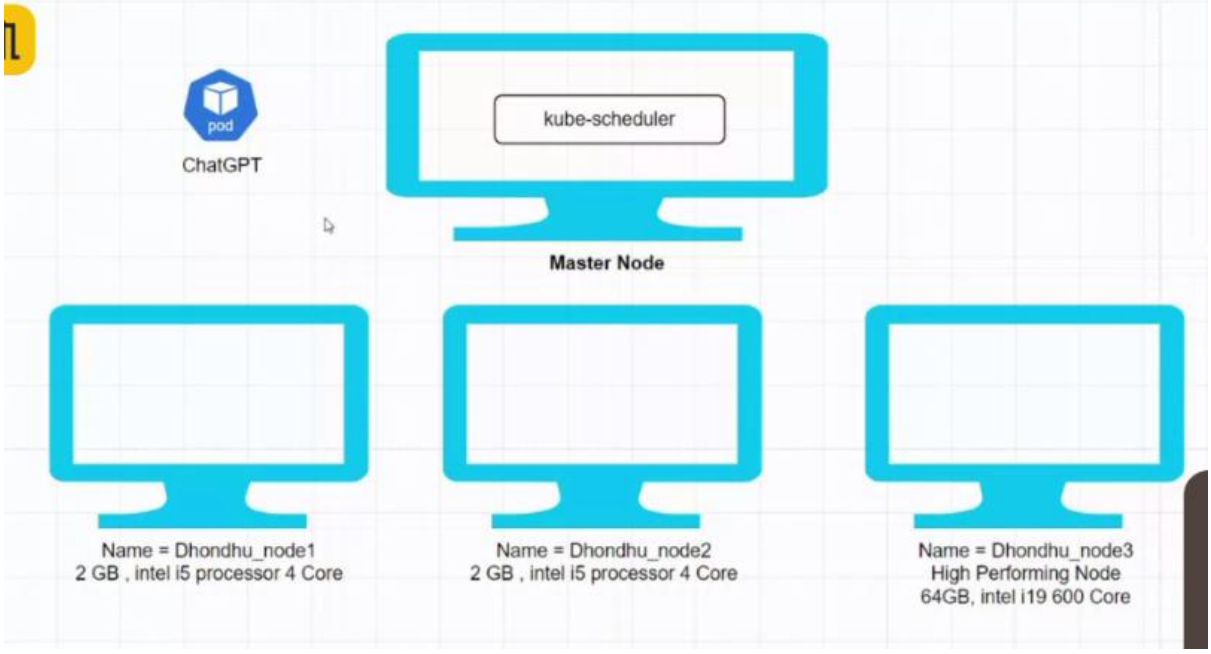


Assignemnt - Liveliness Probe,
Readiness and Startup Probes.
ConfigMap and Secrets

++

AGENDA – SCHEDULING

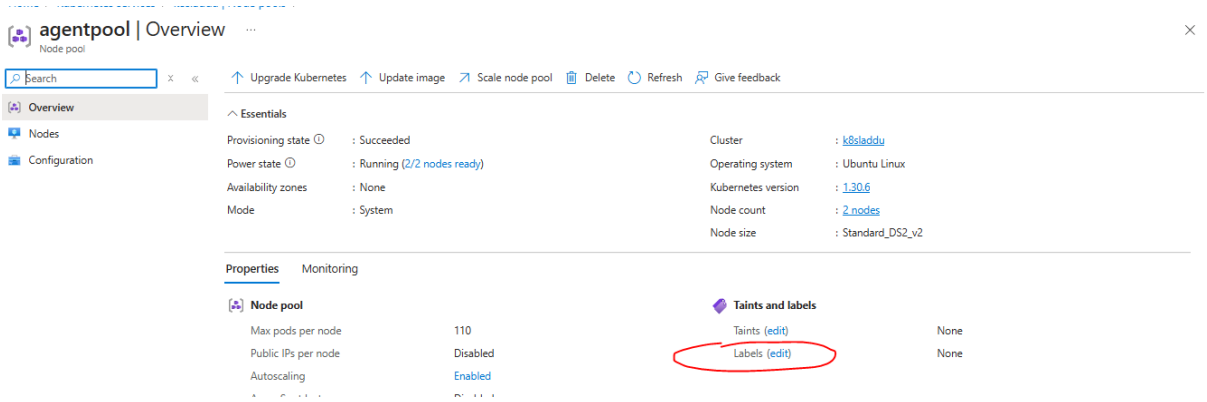
1)

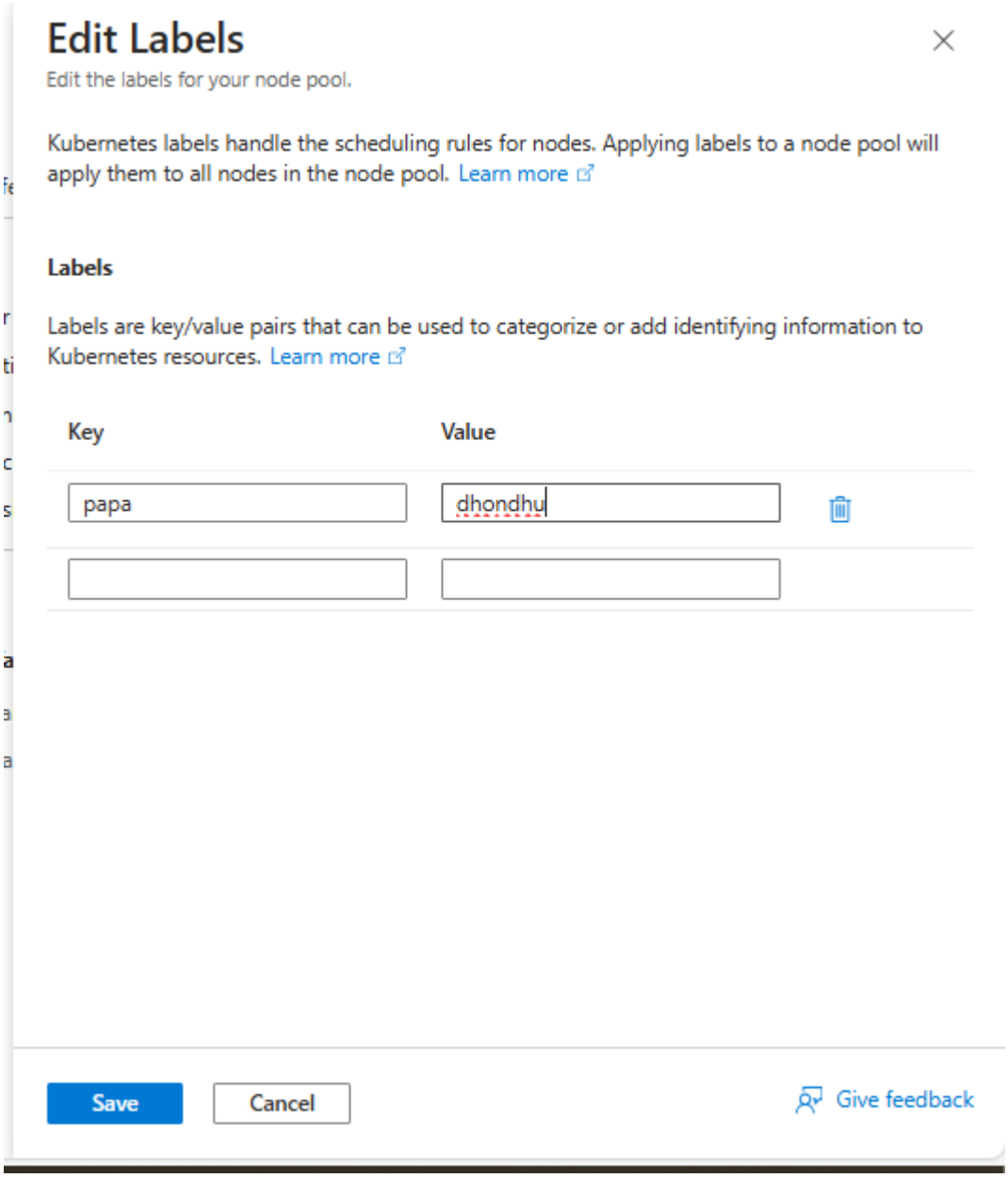


2) nodeSelector: put label papa: dhondhu

```
spec:
  nodeSelector:
  papa: dhondhu
```

3) in cluster go to nodepools to set matching label





Edit Labels

Edit the labels for your node pool.

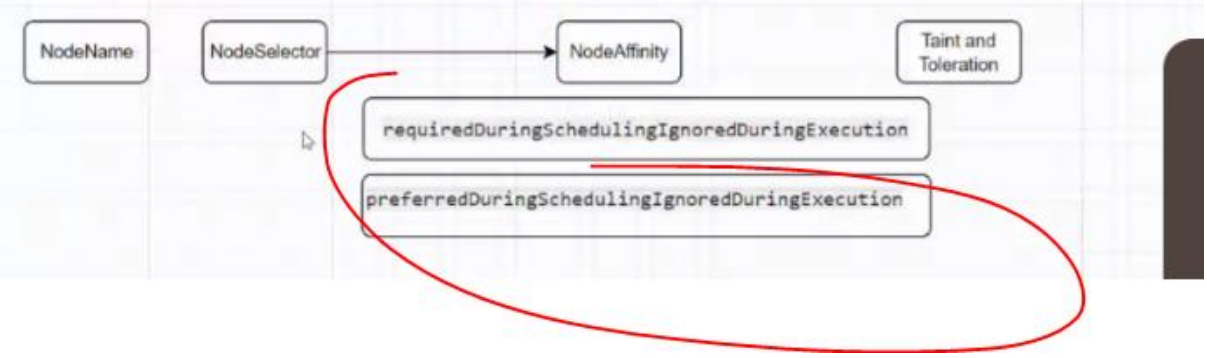
Kubernetes labels handle the scheduling rules for nodes. Applying labels to a node pool will apply them to all nodes in the node pool. Learn more

Labels

Labels are key/value pairs that can be used to categorize or add identifying information to Kubernetes resources. Learn more

Key	Value
papa	dhondhu

4) NODE AFFINITY = Advanced version of node selector



++

AGENDA – TAINT & TOLERATION

Home > Kubernetes services > k8sladdu | Node pools >

agentpool | Overview
Node pool

Search

- Overview
- Nodes
- Configuration

Upgrade Kubernetes | Update image | Scale node pool | Delete | Refresh | Give feedback

Essentials

Provisioning state	: Succeeded	Cluster	: k8sladdu
Power state	: Running (2/2 nodes ready)	Operating system	: Ubuntu Linux
Availability zones	: None	Kubernetes version	: 1.30.6
Mode	: System	Node count	: 2 nodes
		Node size	: Standard_DS2_v2

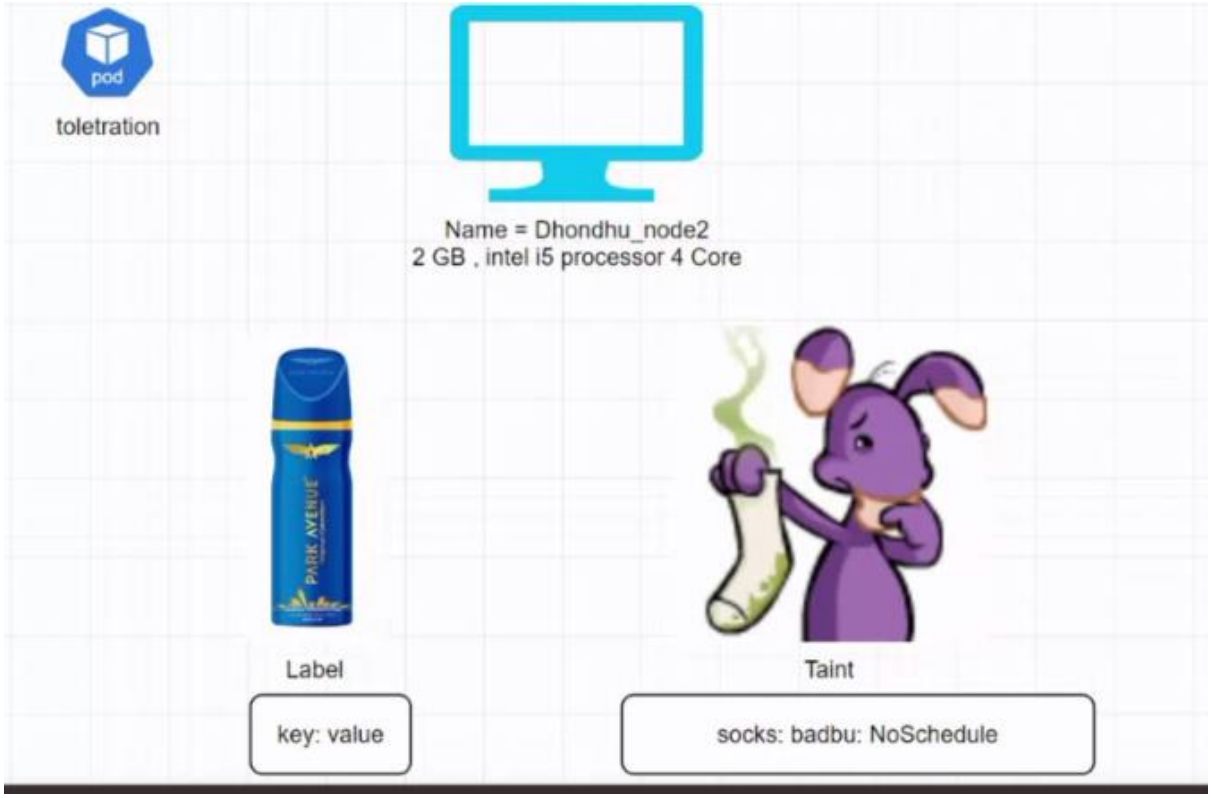
Properties | Monitoring

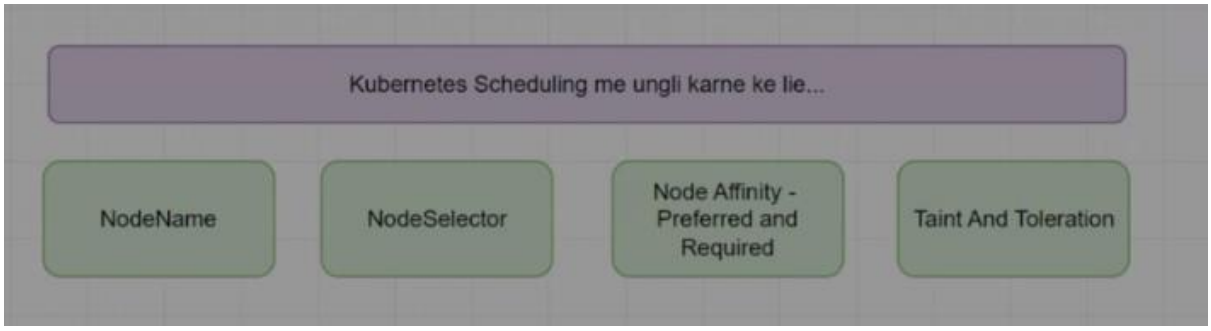
Node pool

Max pods per node	110
Public IPs per node	Disabled
Autoscaling	Enabled
Azure Spot Instance	Disabled
Maximum price	N/A
Scale eviction policy	N/A
Node image version	AKSUbuntu-2204gen2containerd-202412.04.0

Taints and labels

Taints (edit)	None
Labels (edit)	papa : dhondhu





CODE DEKHLO LAST Ke VIDEO SE