Kubernetes

Day 2

Lab 2

<u>Simulating a Faulty Node</u>

# ---CKA Example Task -

# A Kubernetes worker node, named node01 is in state NotReady.

# Investigate why this is the case.

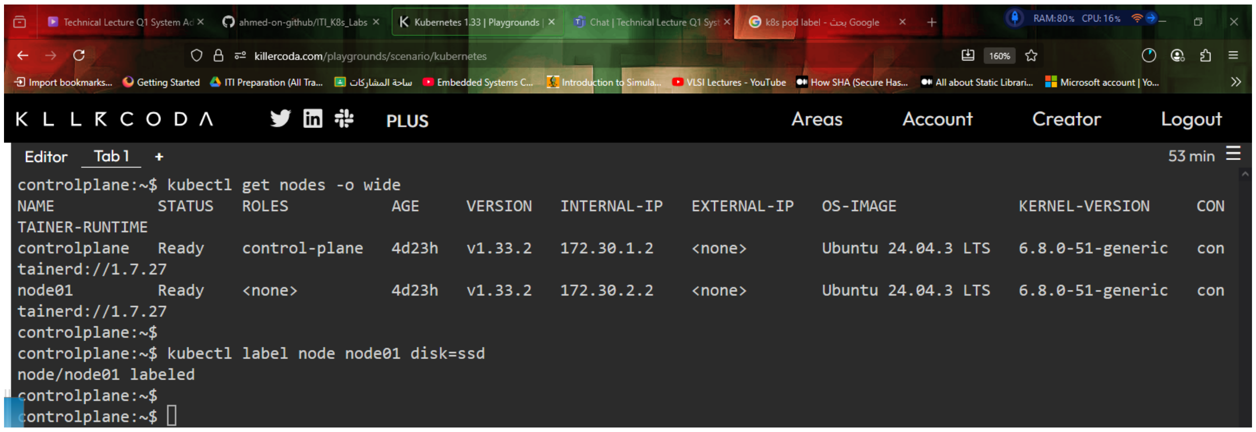
# perform any appropriate steps to bring the node to a Ready state,

#ensuring that any changes are made permanent.

Solution:

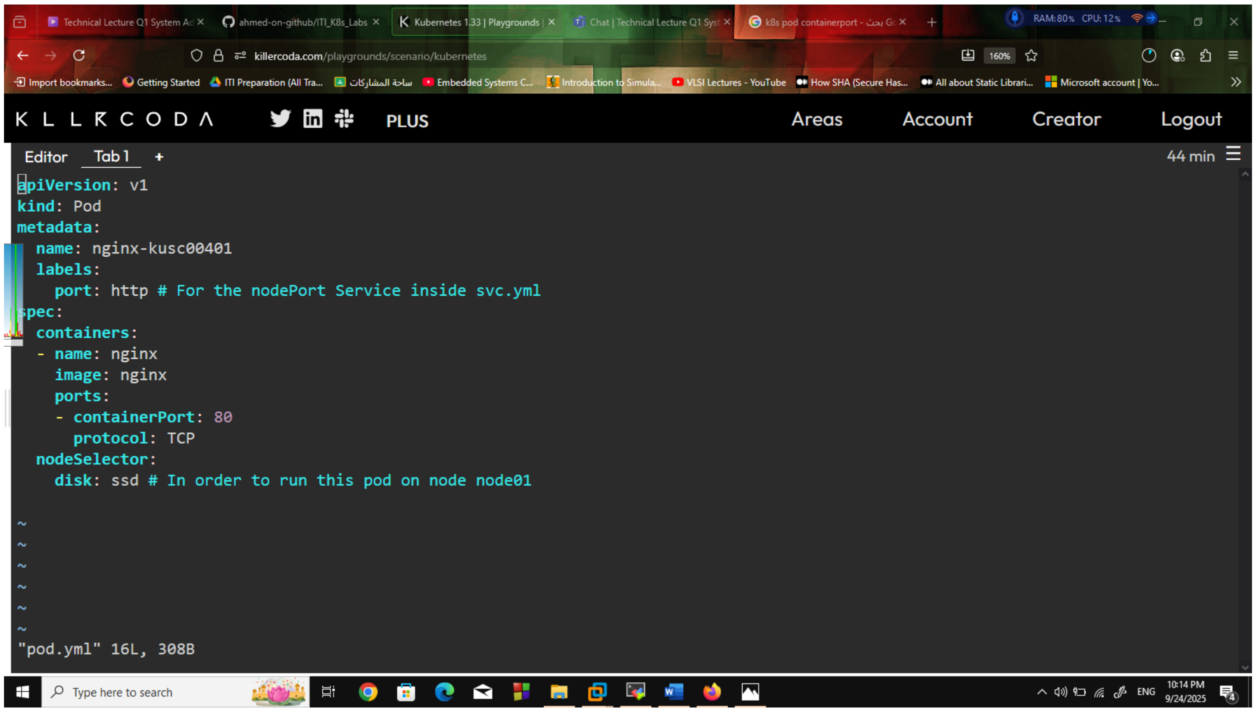
1- Simulating the situation:

We consider a running nginx pod on node01 before it become “NotReady”.

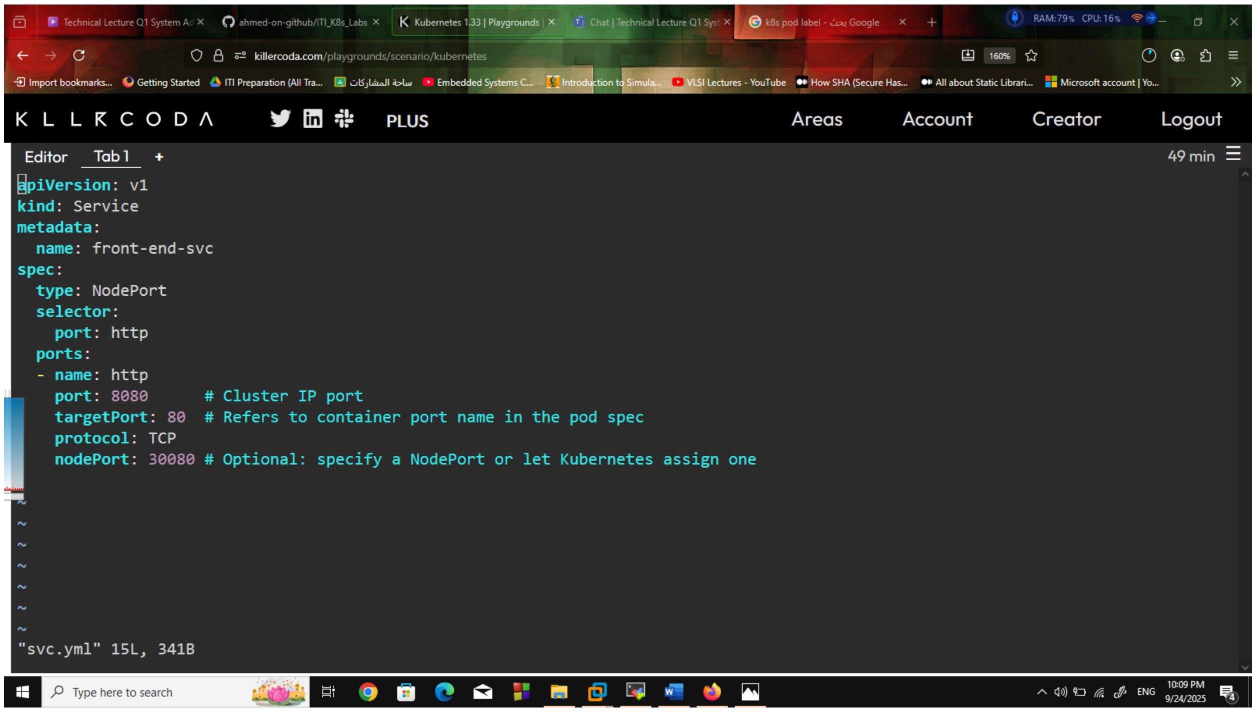


```
controlplane:~$ kubectl get nodes -o wide
NAME           STATUS   ROLES           AGE     VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION     CON
TAINER-RUNTIME
controlplane   Ready    control-plane   4d23h   v1.33.2   172.30.1.2    <none>        Ubuntu 24.04.3 LTS   6.8.0-51-generic   con
tainerd://1.7.27
node01         Ready    <none>          4d23h   v1.33.2   172.30.2.2    <none>        Ubuntu 24.04.3 LTS   6.8.0-51-generic   con
tainerd://1.7.27
controlplane:~$
controlplane:~$ kubectl label node node01 disk=ssd
node/node01 labeled
controlplane:~$
controlplane:~$
```

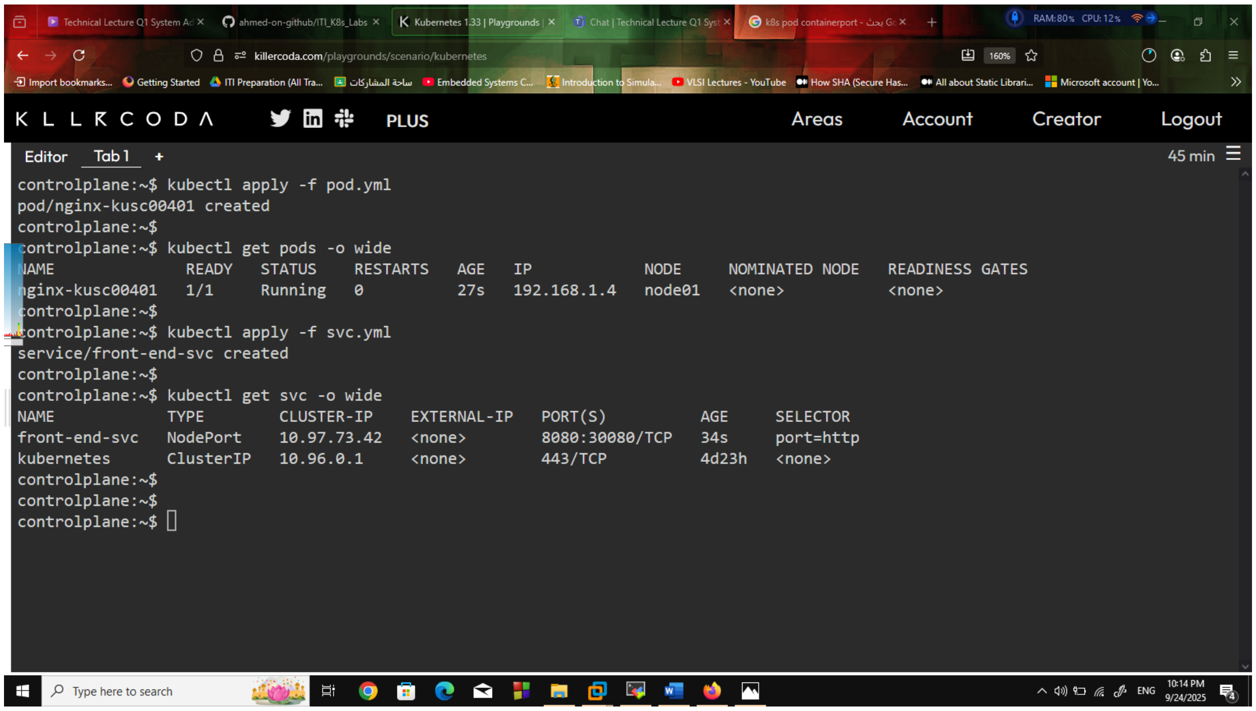
Mark the node “node01” with a label so that an nginx pod would select it to run on it.



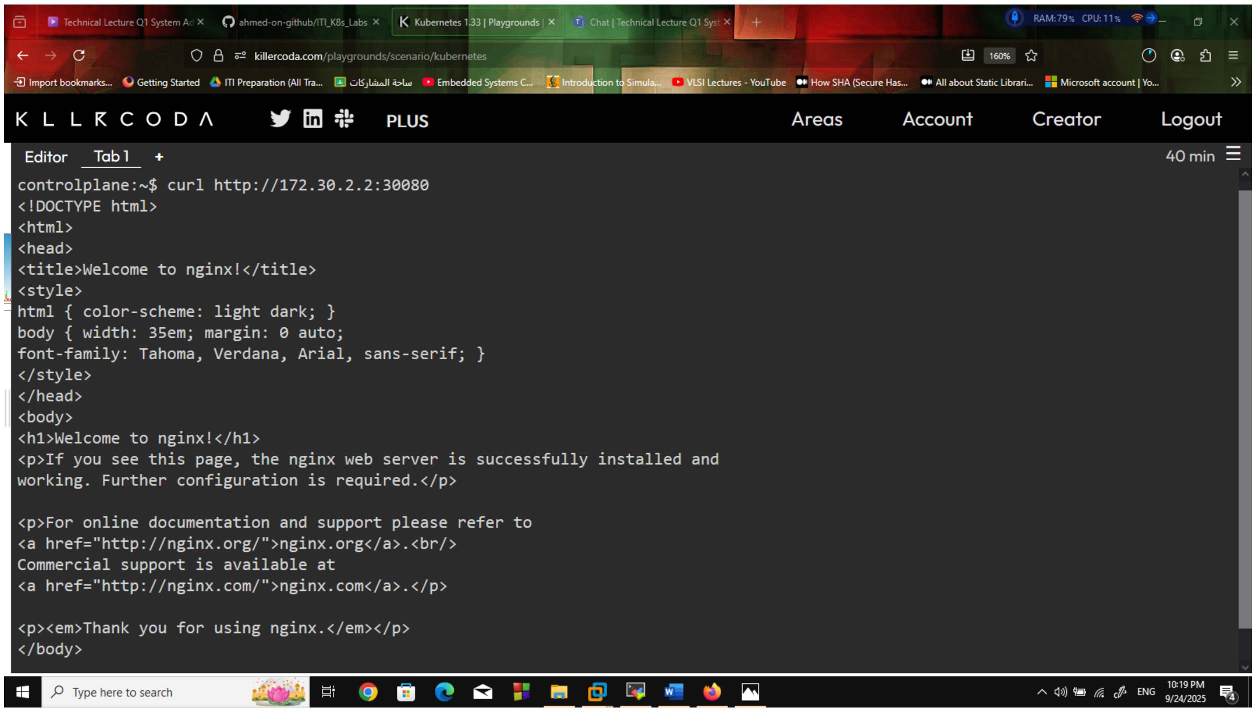
```
apiVersion: v1
kind: Pod
metadata:
  name: nginx-kusc00401
  labels:
    port: http # For the nodePort Service inside svc.yml
spec:
  containers:
  - name: nginx
    image: nginx
    ports:
    - containerPort: 80
      protocol: TCP
  nodeSelector:
    disk: ssd # In order to run this pod on node node01

~
~
~
~
~
~
"pod.yml" 16L, 308B
```



```
apiVersion: v1
kind: Service
metadata:
  name: front-end-svc
spec:
  type: NodePort
  selector:
    port: http
  ports:
  - name: http
    port: 8080      # Cluster IP port
    targetPort: 80  # Refers to container port name in the pod spec
    protocol: TCP
    nodePort: 30080 # Optional: specify a NodePort or let Kubernetes assign one
~
~
~
~
~
~
~
"svc.yml" 15L, 341B
```



```
controlplane:~$ kubectl apply -f pod.yml
pod/nginx-kusc00401 created
controlplane:~$
controlplane:~$ kubectl get pods -o wide
NAME              READY   STATUS    RESTARTS   AGE   IP            NODE     NOMINATED NODE   READINESS GATES
nginx-kusc00401   1/1     Running   0          27s   192.168.1.4   node01   <none>           <none>
controlplane:~$
controlplane:~$ kubectl apply -f svc.yml
service/front-end-svc created
controlplane:~$
controlplane:~$ kubectl get svc -o wide
NAME            TYPE        CLUSTER-IP    EXTERNAL-IP   PORT(S)          AGE     SELECTOR
front-end-svc   NodePort    10.97.73.42   <none>        8080:30080/TCP   34s     port=http
kubernetes      ClusterIP   10.96.0.1     <none>        443/TCP          4d23h   <none>
controlplane:~$
controlplane:~$
controlplane:~$
```

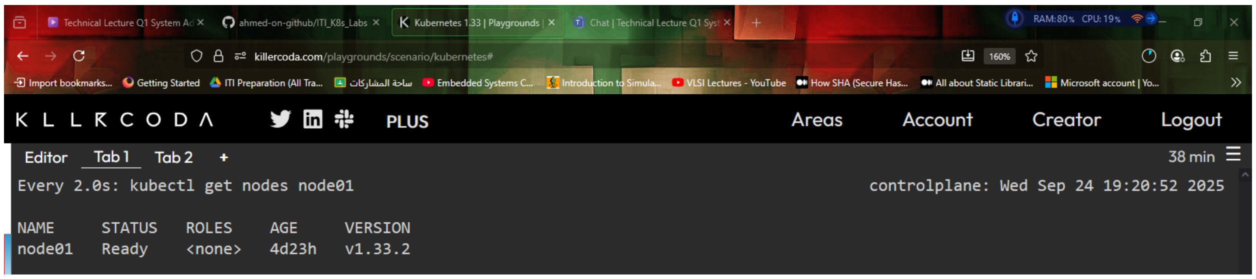


```
controlplane:~$ curl http://172.30.2.2:30080
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
```

The website is reachable and working.

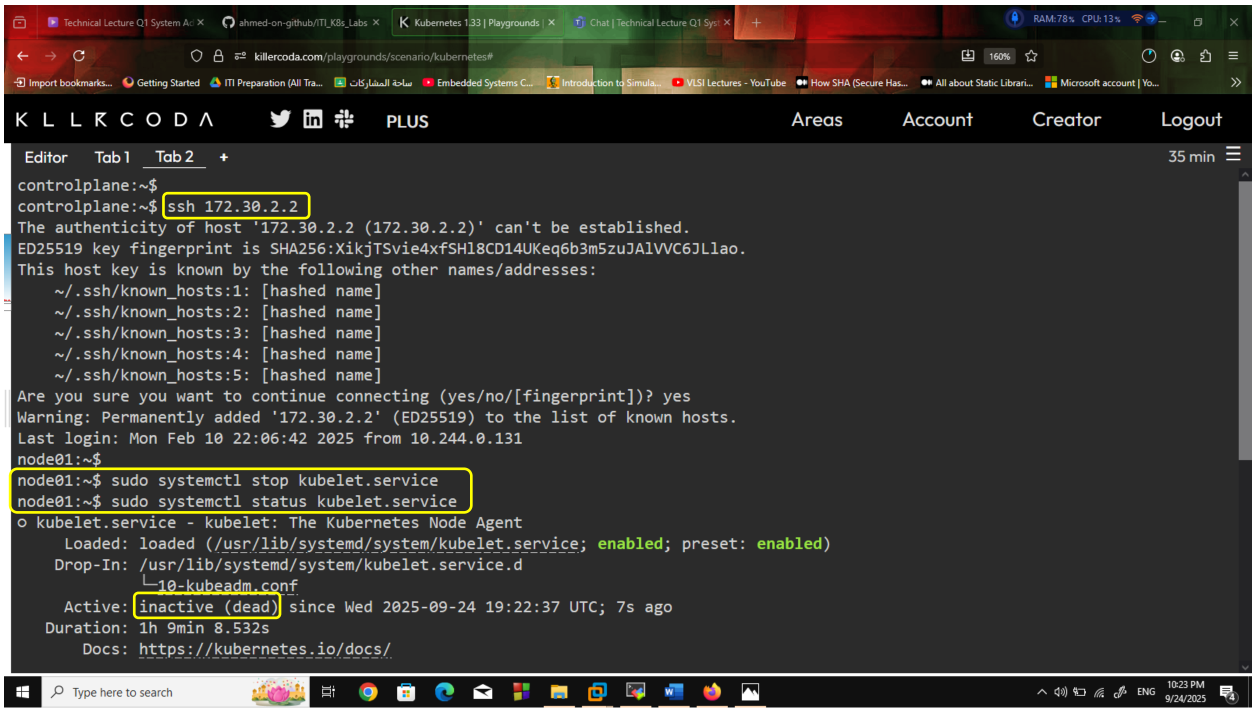
2- Getting the node NotReady:

Running “watch kubectl get nodes node01” to get updated state for that node every 2 seconds.

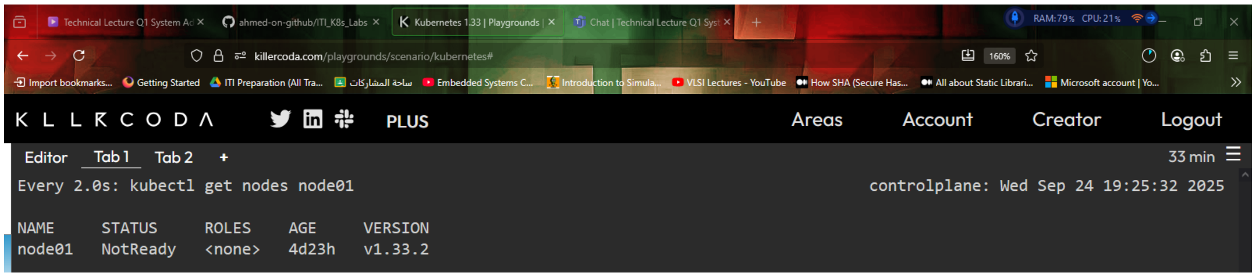


```
Every 2.0s: kubectl get nodes node01                          controlplane: Wed Sep 24 19:20:52 2025

NAME     STATUS   ROLES    AGE    VERSION
node01   Ready    <none>   4d23h  v1.33.2
```

The node can become NotReady If the kubelet.service is stopped inside its container.

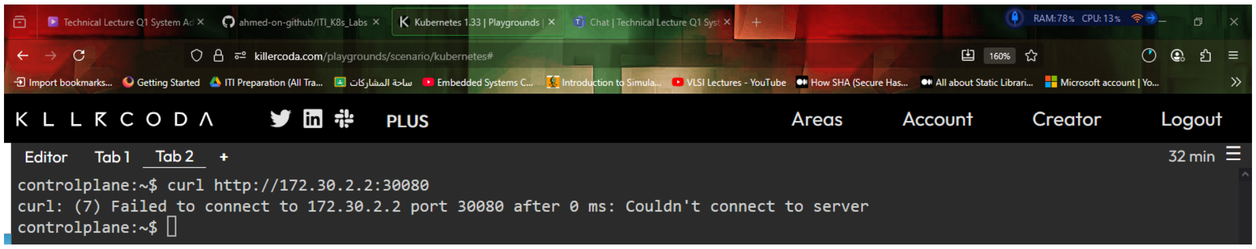


```
controlplane:~$
controlplane:~$ ssh 172.30.2.2
The authenticity of host '172.30.2.2 (172.30.2.2)' can't be established.
ED25519 key fingerprint is SHA256:XikjTSvie4xfSHl8CD14UKeq6b3m5zuJAlVVC6JLlao.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:1: [hashed name]
    ~/.ssh/known_hosts:2: [hashed name]
    ~/.ssh/known_hosts:3: [hashed name]
    ~/.ssh/known_hosts:4: [hashed name]
    ~/.ssh/known_hosts:5: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.30.2.2' (ED25519) to the list of known hosts.
Last login: Mon Feb 10 22:06:42 2025 from 10.244.0.131
node01:~$
node01:~$ sudo systemctl stop kubelet.service
node01:~$ sudo systemctl status kubelet.service
○ kubelet.service - kubelet: The Kubernetes Node Agent
     Loaded: loaded (/usr/lib/systemd/system/kubelet.service; enabled; preset: enabled)
    Drop-In: /usr/lib/systemd/system/kubelet.service.d
             └─10-kubeadm.conf
     Active: inactive (dead) since Wed 2025-09-24 19:22:37 UTC; 7s ago
   Duration: 1h 9min 8.532s
       Docs: https://kubernetes.io/docs/
```



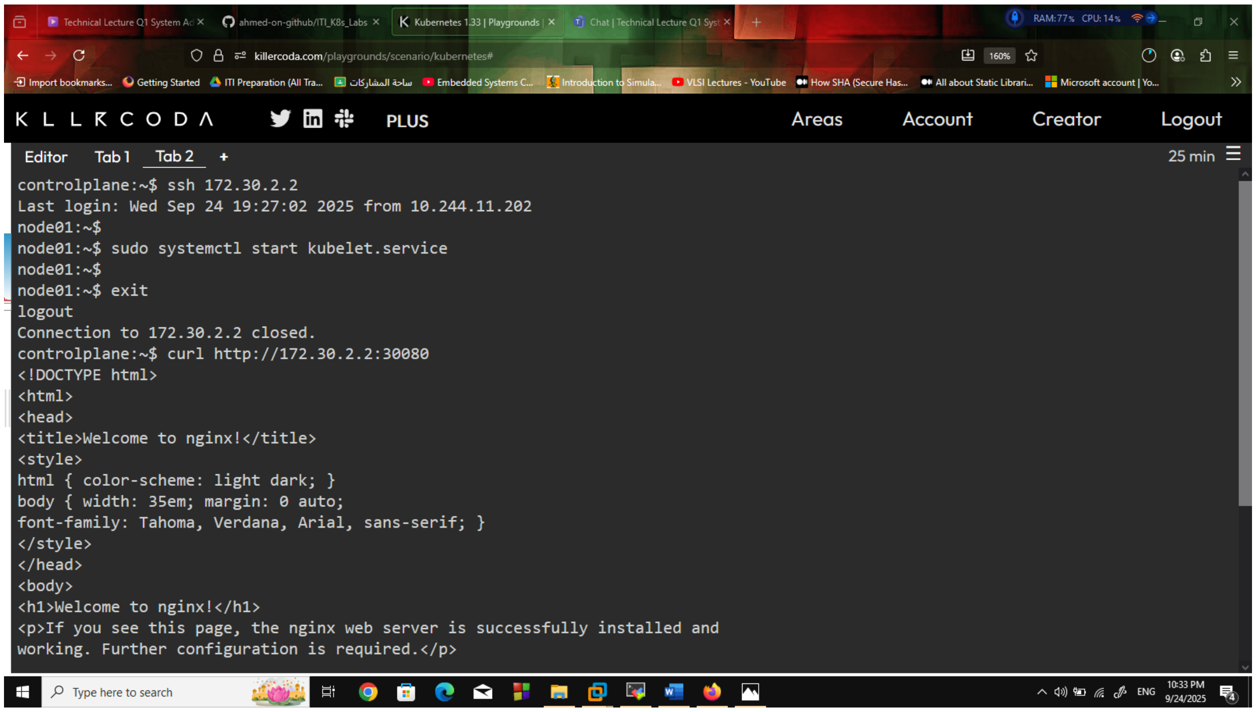
```
Every 2.0s: kubectl get nodes node01                                controlplane: Wed Sep 24 19:25:32 2025

NAME     STATUS     ROLES    AGE    VERSION
node01   NotReady   <none>   4d23h  v1.33.2
```

The webpage becomes unreachable as the master’s API server can’t communicate the with the node’s kubelet.

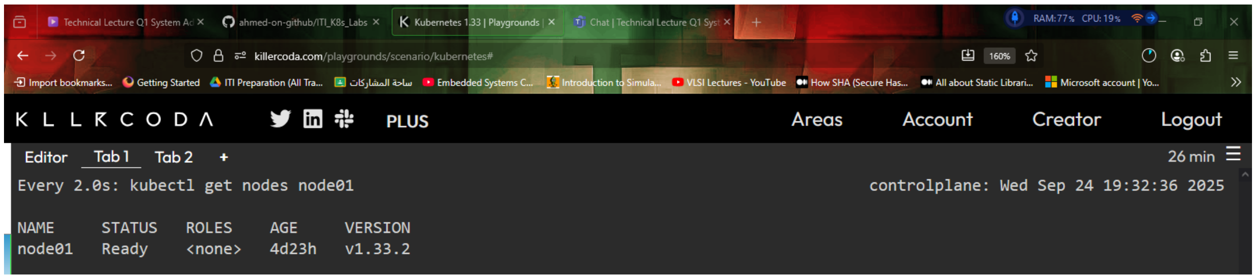


```
controlplane:~$ curl http://172.30.2.2:30080
curl: (7) Failed to connect to 172.30.2.2 port 30080 after 0 ms: Couldn't connect to server
controlplane:~$ 
```

After restarting the service on the node, the web server is reachable again.

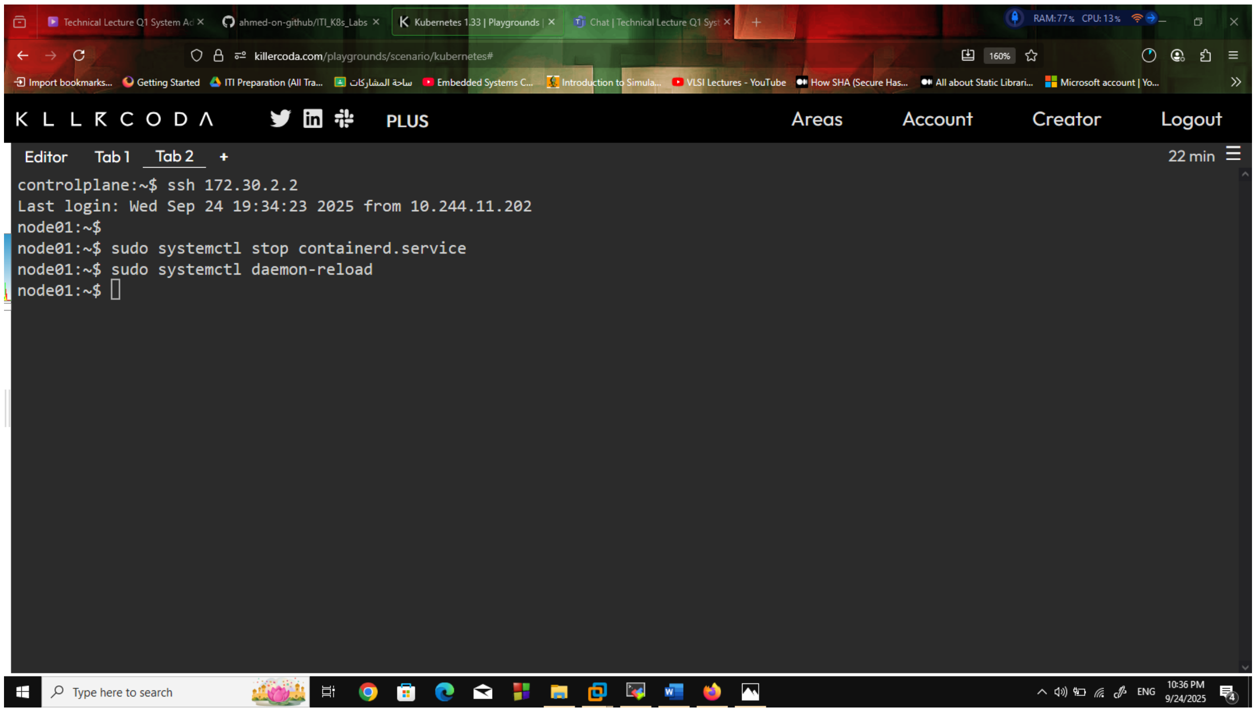


```
controlplane:~$ ssh 172.30.2.2
Last login: Wed Sep 24 19:27:02 2025 from 10.244.11.202
node01:~$
node01:~$ sudo systemctl start kubelet.service
node01:~$
node01:~$ exit
logout
Connection to 172.30.2.2 closed.
controlplane:~$ curl http://172.30.2.2:30080
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>
```

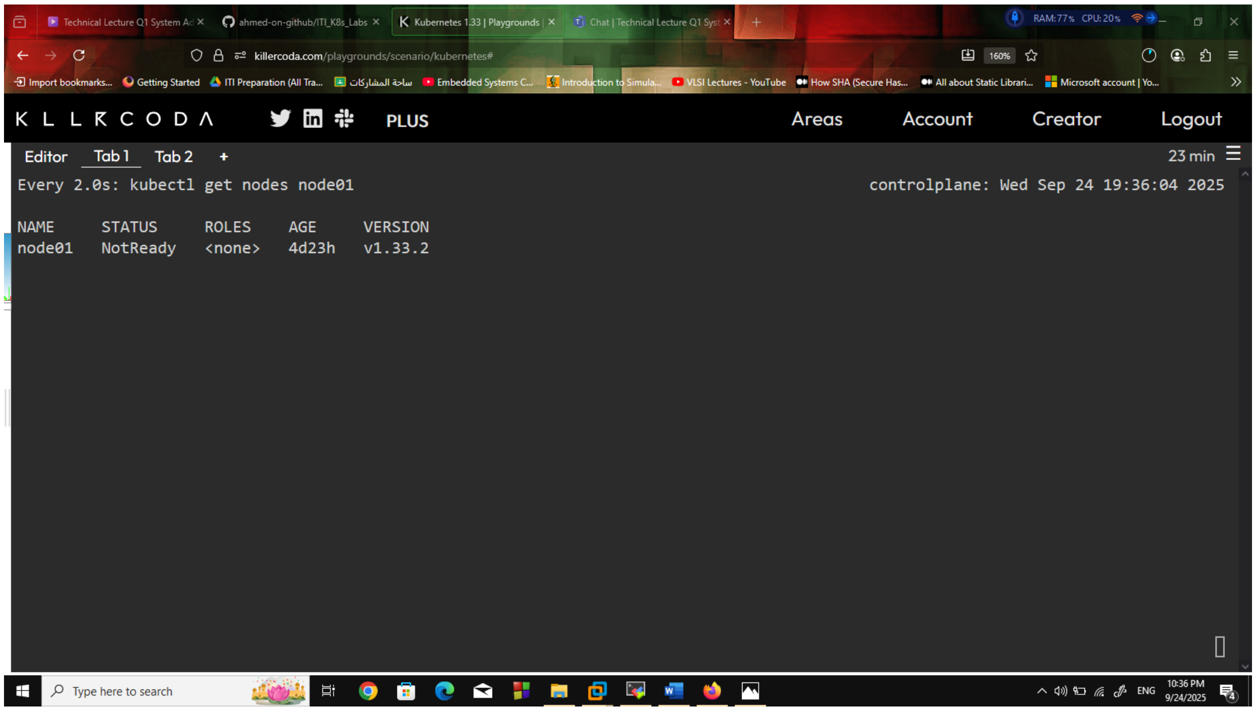


```
Every 2.0s: kubectl get nodes node01                         controlplane: Wed Sep 24 19:32:36 2025

NAME     STATUS   ROLES    AGE    VERSION
node01   Ready    <none>   4d23h  v1.33.2
```

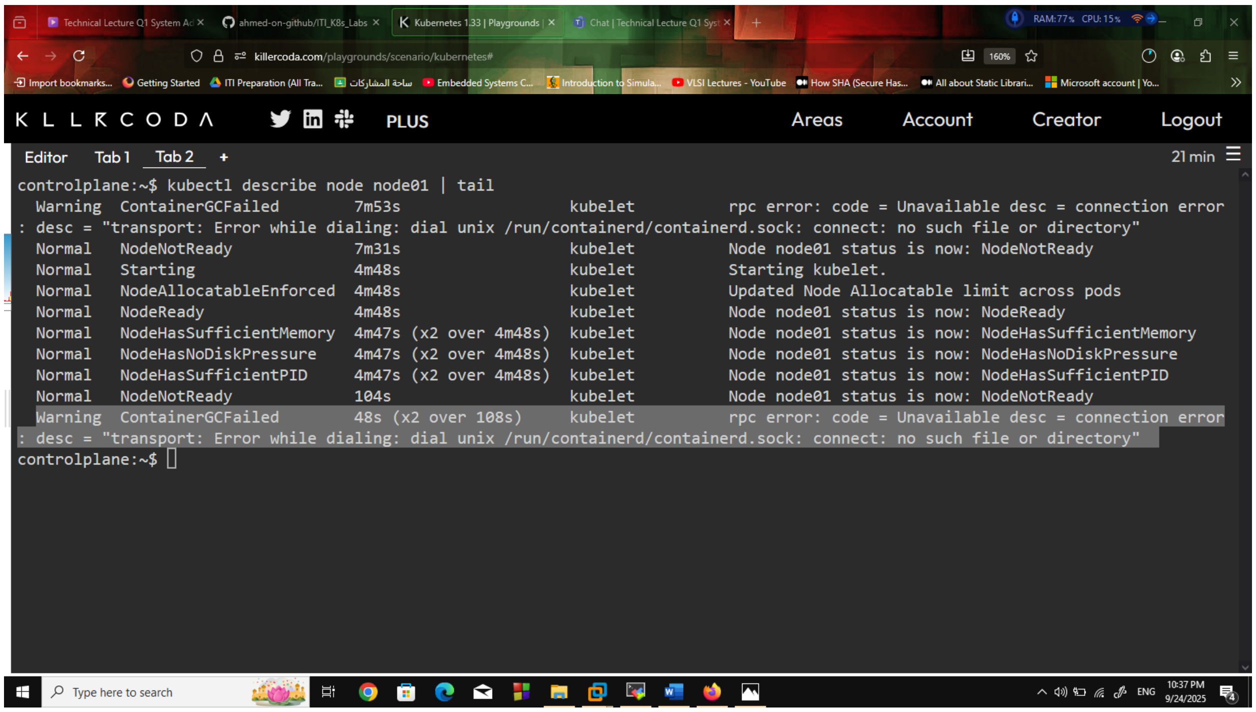
Another way is when the containerd.service is stopped:



```
controlplane:~$ ssh 172.30.2.2
Last login: Wed Sep 24 19:34:23 2025 from 10.244.11.202
node01:~$ 
node01:~$ sudo systemctl stop containerd.service
node01:~$ sudo systemctl daemon-reload
node01:~$ 
```



```
Every 2.0s: kubectl get nodes node01                       controlplane: Wed Sep 24 19:36:04 2025

NAME     STATUS     ROLES    AGE     VERSION
node01   NotReady   <none>   4d23h   v1.33.2
```



```
controlplane:~$ kubectl describe node node01 | tail
  Warning  ContainerGCFailed          7m53s                   kubelet          rpc error: code = Unavailable desc = connection error
: desc = "transport: Error while dialing: dial unix /run/containerd/containerd.sock: connect: no such file or directory"
  Normal   NodeNotReady               7m31s                   kubelet          Node node01 status is now: NodeNotReady
  Normal   Starting                   4m48s                   kubelet          Starting kubelet.
  Normal   NodeAllocatableEnforced    4m48s                   kubelet          Updated Node Allocatable limit across pods
  Normal   NodeReady                  4m48s                   kubelet          Node node01 status is now: NodeReady
  Normal   NodeHasSufficientMemory    4m47s (x2 over 4m48s)   kubelet          Node node01 status is now: NodeHasSufficientMemory
  Normal   NodeHasNoDiskPressure      4m47s (x2 over 4m48s)   kubelet          Node node01 status is now: NodeHasNoDiskPressure
  Normal   NodeHasSufficientPID       4m47s (x2 over 4m48s)   kubelet          Node node01 status is now: NodeHasSufficientPID
  Normal   NodeNotReady               104s                    kubelet          Node node01 status is now: NodeNotReady
  Warning  ContainerGCFailed          48s (x2 over 108s)      kubelet          rpc error: code = Unavailable desc = connection error
: desc = "transport: Error while dialing: dial unix /run/containerd/containerd.sock: connect: no such file or directory"
controlplane:~$ 
```

Although in this case, the webpage is still reachable. However, it’s still a case that get’s the node in the “NotReady” state.