Kubernetes

Day3

Lab 1

Requirements

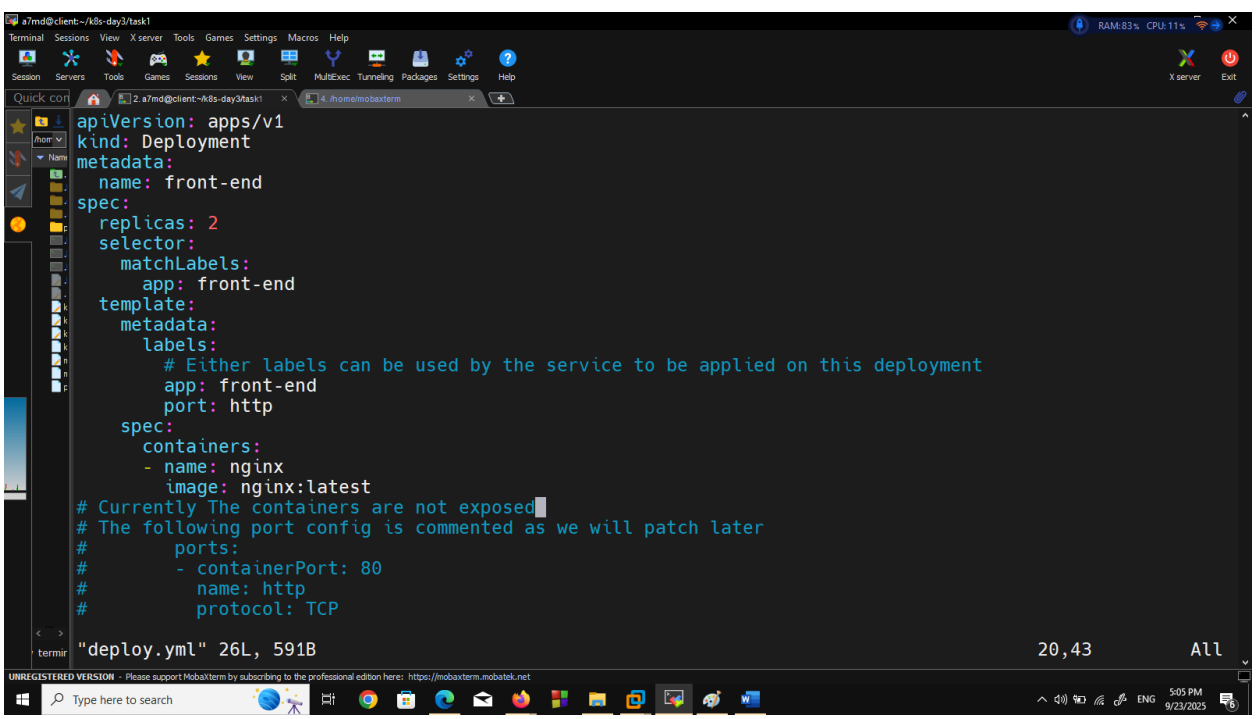
Reconfigure the existing deployment front-end and add a port specification named http exposing port 80/tcp of the existing container nginx.

Create a new service named front-end-svc exposing the container port http.

Configure the new service to also expose the individual Pods via a NodePort on the nodes on which they are scheduled.

Solutions

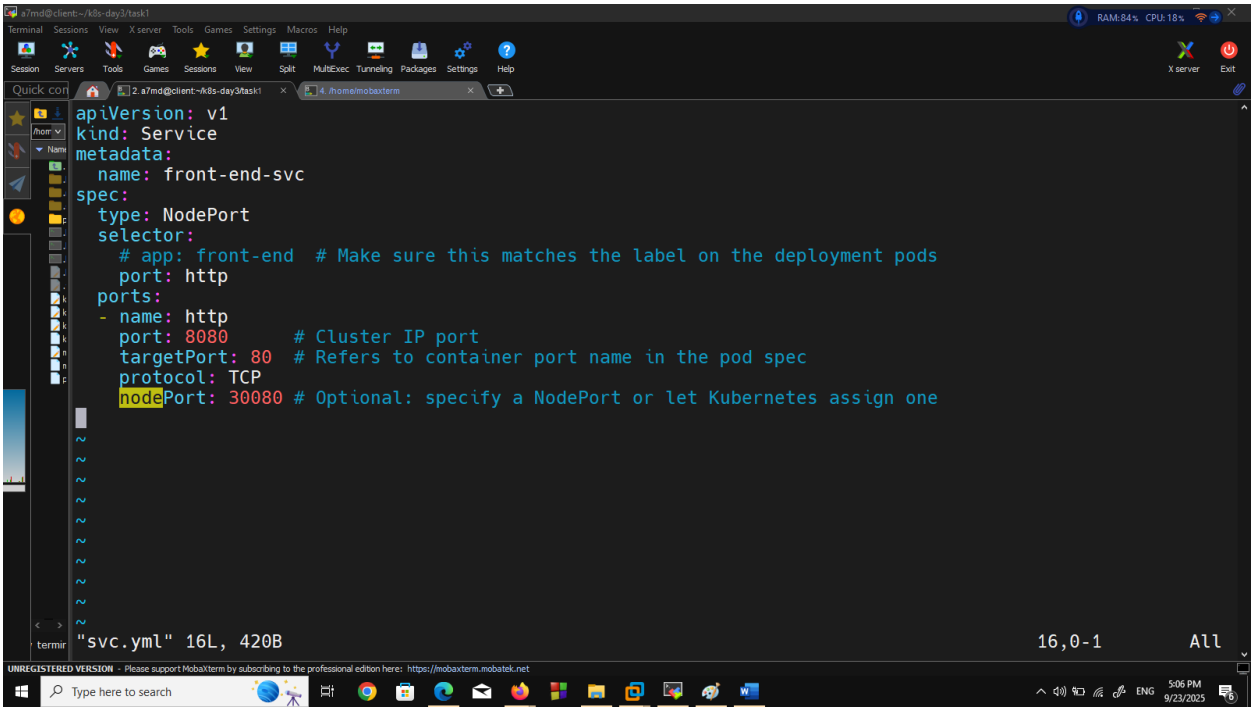
1- Yaml file for the deployment:



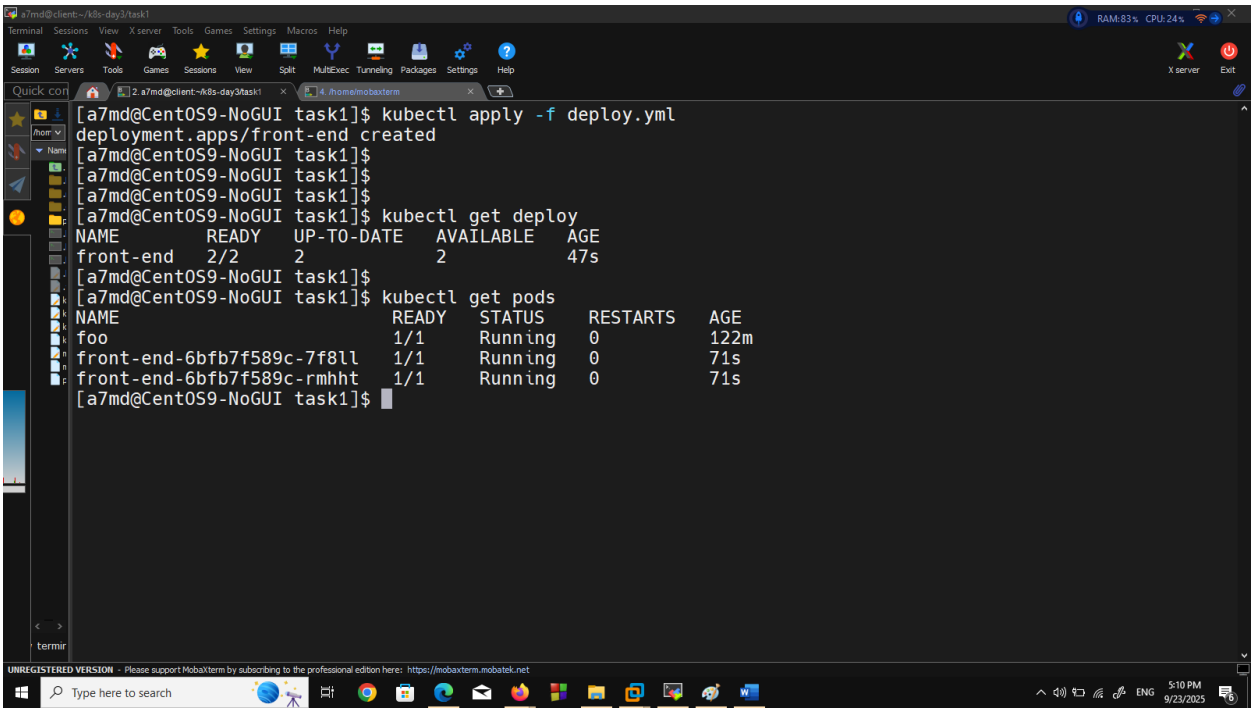
```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: front-end
spec:
  replicas: 2
  selector:
    matchLabels:
      app: front-end
  template:
    metadata:
      labels:
        # Either labels can be used by the service to be applied on this deployment
        app: front-end
        port: http
    spec:
      containers:
      - name: nginx
        image: nginx:latest
# Currently The containers are not exposed
# The following port config is commented as we will patch later
#        ports:
#        - containerPort: 80
#          name: http
#          protocol: TCP
```

"deploy.yml" 26L, 591B 20,43 All

2- Yaml file for the service that exposes the containers:

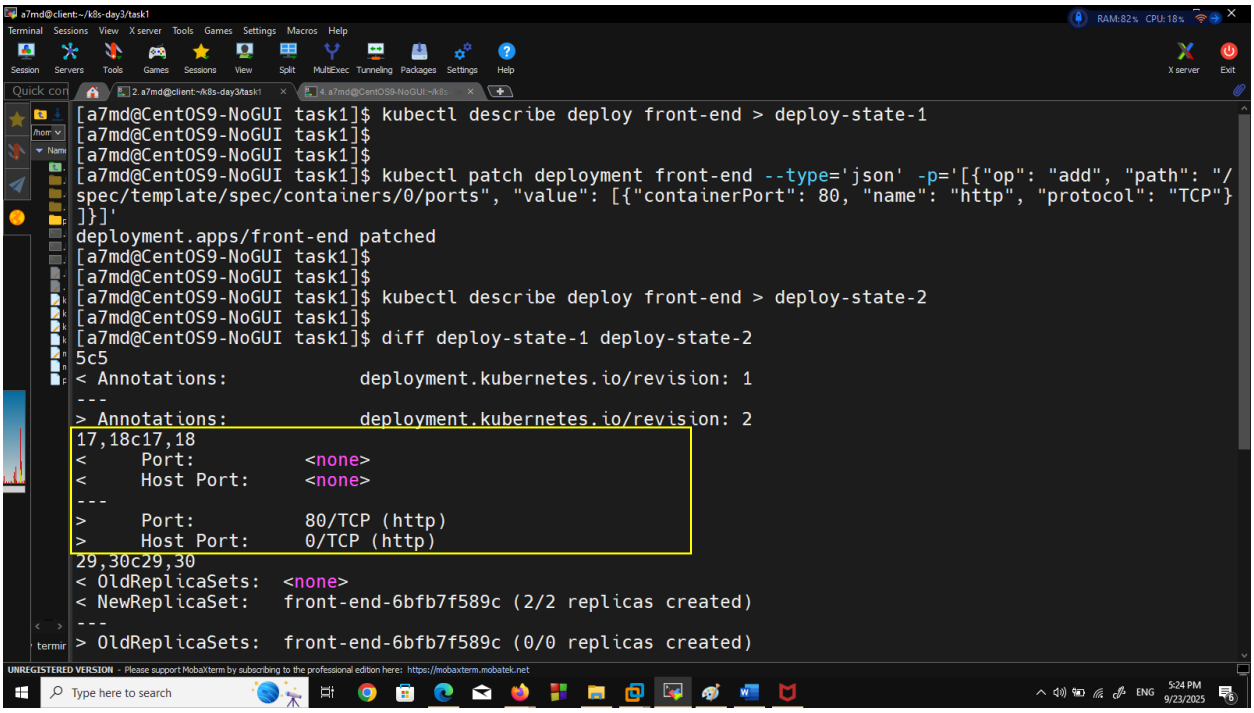


3- Patching the port configuration to the container's configuration:



```
[a7md@CentOS9-NoGUI task1]$ kubectl apply -f deploy.yml
deployment.apps/front-end created
[a7md@CentOS9-NoGUI task1]$
[a7md@CentOS9-NoGUI task1]$
[a7md@CentOS9-NoGUI task1]$
[a7md@CentOS9-NoGUI task1]$ kubectl get deploy
NAME        READY   UP-TO-DATE   AVAILABLE   AGE
front-end   2/2     2            2           47s
[a7md@CentOS9-NoGUI task1]$
[a7md@CentOS9-NoGUI task1]$ kubectl get pods
NAME                          READY   STATUS    RESTARTS   AGE
foo                           1/1     Running   0          122m
front-end-6bfb7f589c-7f8ll    1/1     Running   0          71s
front-end-6bfb7f589c-rmhht    1/1     Running   0          71s
[a7md@CentOS9-NoGUI task1]$
```

Figure 1: Start the deploy



```
[a7md@CentOS9-NoGUI task1]$ kubectl describe deploy front-end > deploy-state-1
[a7md@CentOS9-NoGUI task1]$
[a7md@CentOS9-NoGUI task1]$
[a7md@CentOS9-NoGUI task1]$ kubectl patch deployment front-end --type='json' -p='[{"op": "add", "path": "/
spec/template/spec/containers/0/ports", "value": [{"containerPort": 80, "name": "http", "protocol": "TCP"}
]}]'
deployment.apps/front-end patched
[a7md@CentOS9-NoGUI task1]$
[a7md@CentOS9-NoGUI task1]$
[a7md@CentOS9-NoGUI task1]$ kubectl describe deploy front-end > deploy-state-2
[a7md@CentOS9-NoGUI task1]$
[a7md@CentOS9-NoGUI task1]$ diff deploy-state-1 deploy-state-2
5c5
< Annotations:            deployment.kubernetes.io/revision: 1
---
> Annotations:            deployment.kubernetes.io/revision: 2
17,18c17,18
<     Port:          <none>
<     Host Port:     <none>
---
>     Port:          80/TCP (http)
>     Host Port:     0/TCP (http)
29,30c29,30
< OldReplicaSets:  <none>
< NewReplicaSet:   front-end-6bfb7f589c (2/2 replicas created)
---
> OldReplicaSets:  front-end-6bfb7f589c (0/0 replicas created)
```

Figure 2: Inspecting the change in the port configurations of the deployment before and after patching

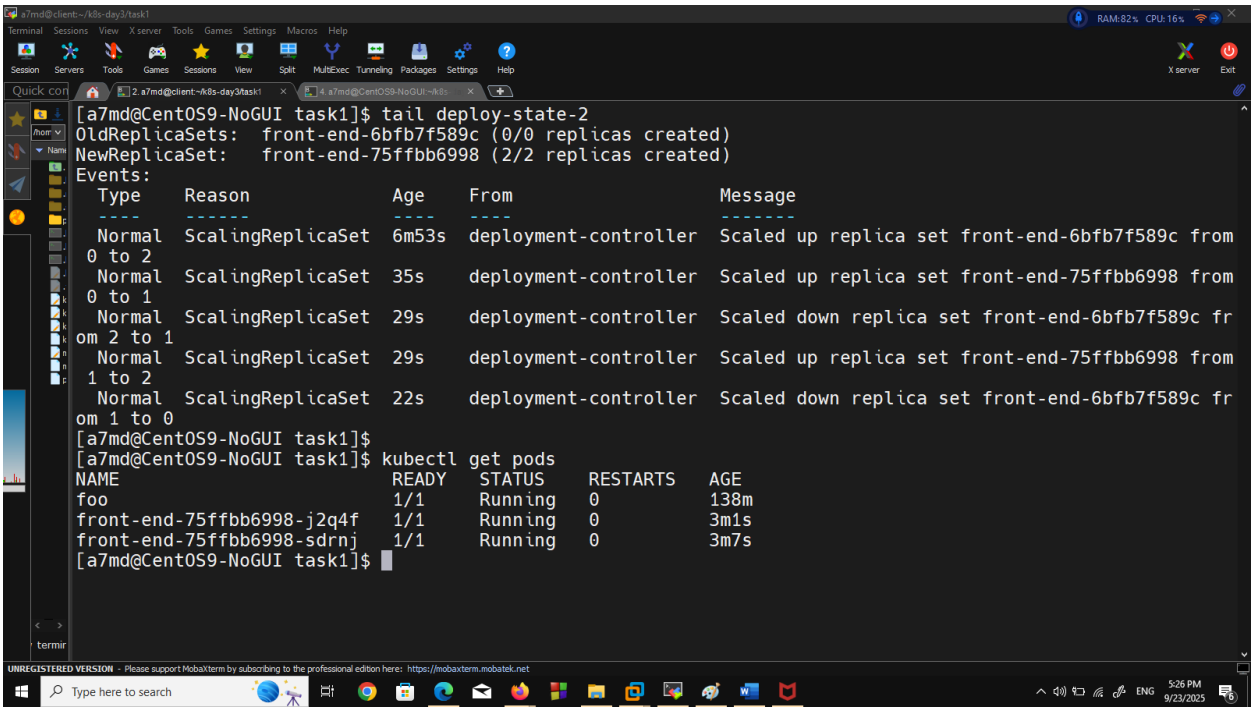
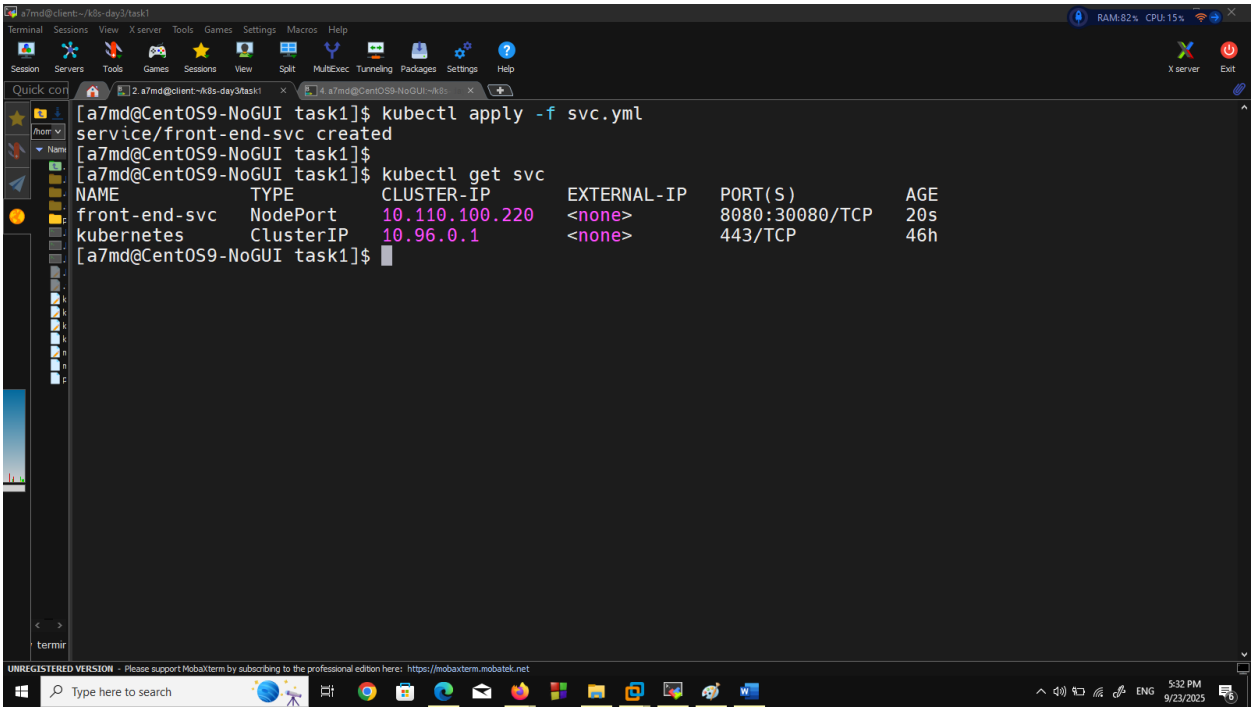


Figure 3: Notice also that the deployment has created new pods with the new port configurations, instead of the old ones

4- Applying and running the service:



5- Accessing the web service from the cluster:

```
[a7md@CentOS9-NoGUI task1]$ minikube ssh
docker@minikube:~$
docker@minikube:~$ curl http://10.110.100.220:8080
<!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>
</html>
docker@minikube:~$
```

6- Accessing the web service from the host machine:

```
[a7md@CentOS9-NoGUI task1]$ docker inspect minikube | grep -i ipaddress
            "SecondaryIPAddresses": null,
            "IPAddress": "",
                    "IPAddress": "192.168.49.2",
[a7md@CentOS9-NoGUI task1]$
[a7md@CentOS9-NoGUI task1]$ minikube ip
192.168.49.2
[a7md@CentOS9-NoGUI task1]$
[a7md@CentOS9-NoGUI task1]$ curl http://$(minikube ip):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>
```

7- Which web server relies to the service ?

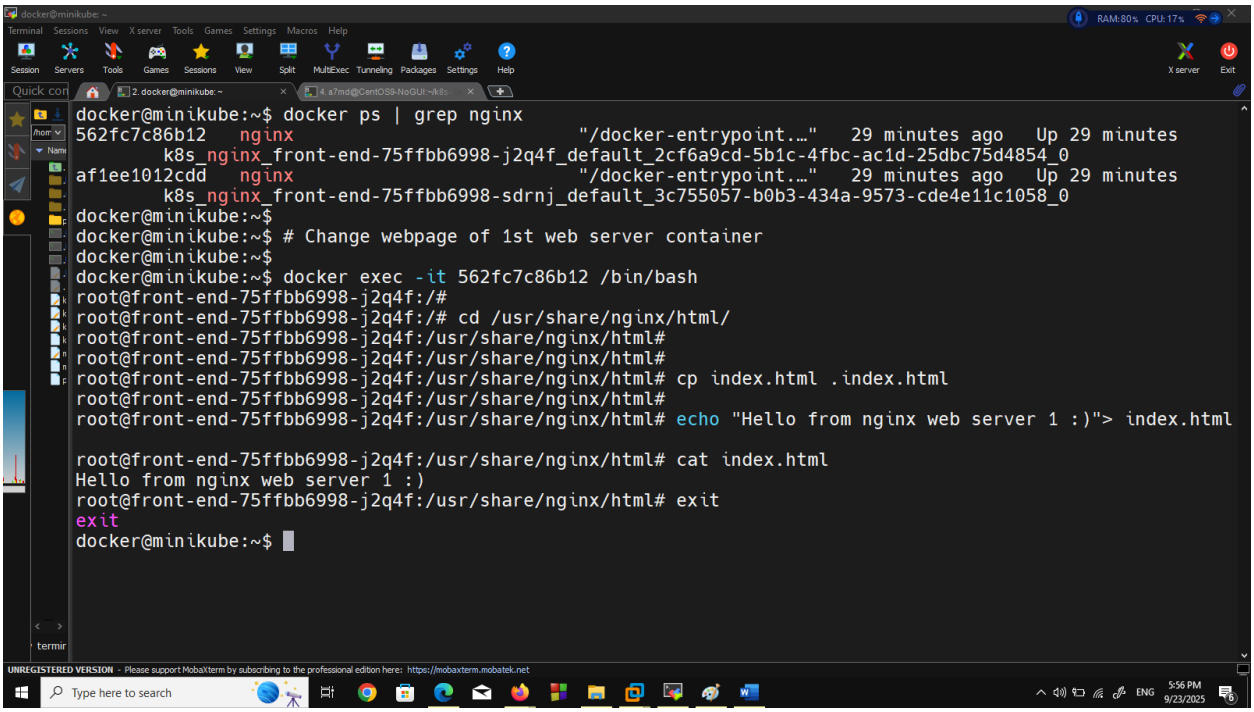


Figure 4: Change the webpage on 1st nginx container

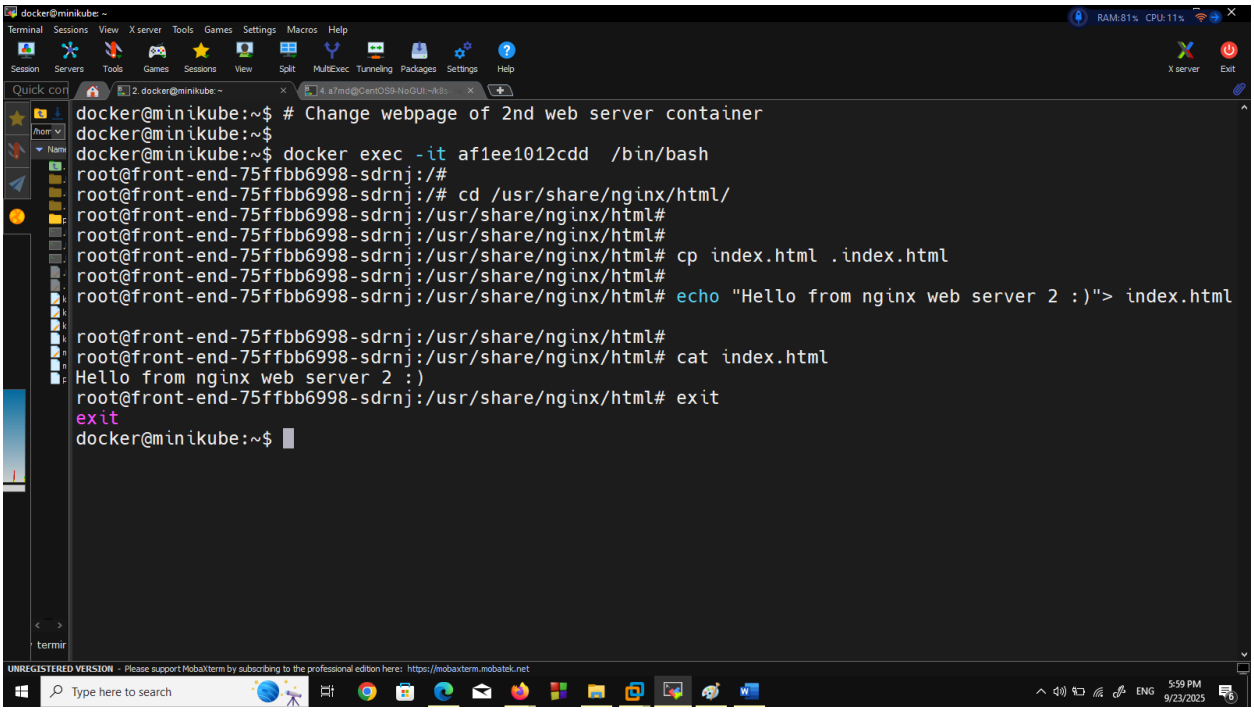


Figure 5: Change the webpage on 2nd nginx container

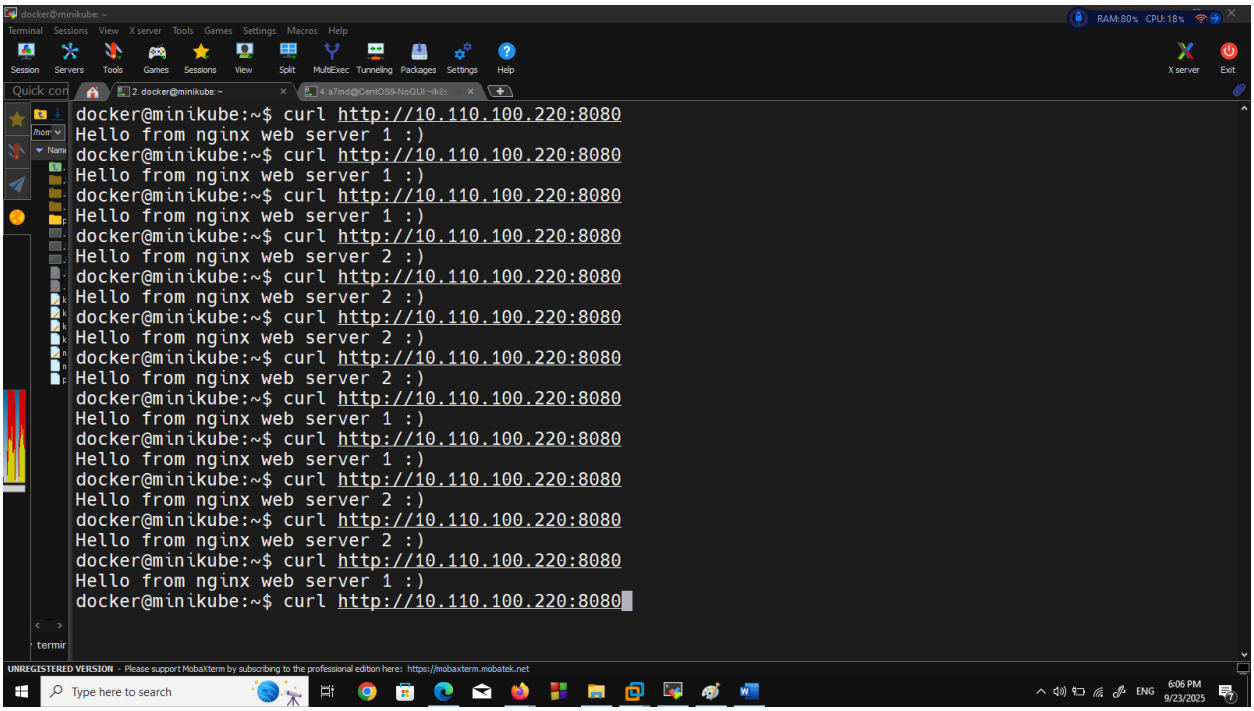


Figure 6: Notice how load-balancing is performed on accessing the deployment pods