**scenario (edit existing deployment)**

**Pre-requisite**

Let us deploy a nginx deployment with ports

vim deploy.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

name: front-end

spec:

replicas: 3

selector:

matchLabels:

app: nginx

template:

metadata:

labels:

app: nginx

spec:

containers:

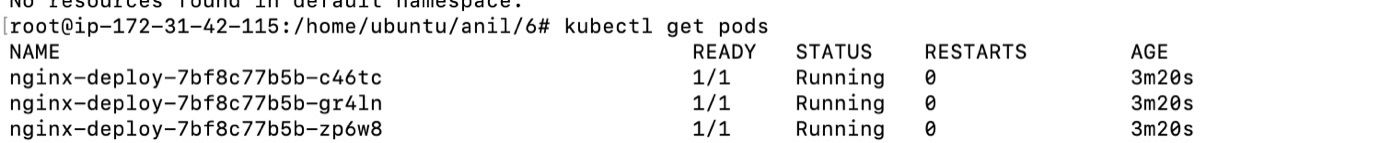
- name: nginx

image: nginx:latest

kubectl create -f deploy.yaml

**verify pods are created and running**

kubectl get pods

****

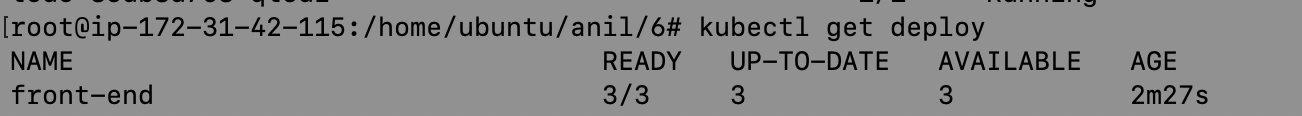
**Task –**

1. Reconfigure the existing deployment front-end and add a port specification named http exposing port 80/tcp of the existing container nginx.
2. Create a new service named front-end-svc exposing the container port http.
3. Congure the new service to also expose the individual Pods via a NodePort on the nodes on which they are scheduled.

Task 1 : solution : Reconfigure the existing deployment front-end and add a port specification named http exposing port 80/tcp of the existing container nginx.

**verify pods are created and running**

kubectl get deploy

****

kubectl edit deployment front-end

Add below 3 lines code in container section

ports:

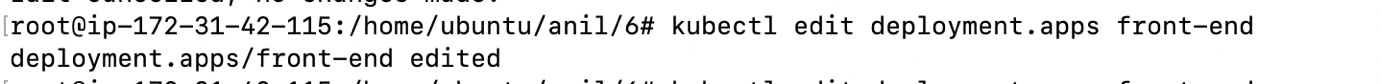
- name: http

containerPort: 80

**A screenshot of a computer

Description automatically generated**

**Exit the file**

****

Task 2 : Create a new service named front-end-svc exposing the container port http.

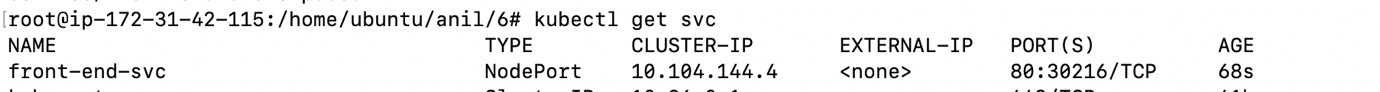
Task 3 : Congure the new service to also expose the individual Pods via a NodePort on the nodes on which they are scheduled.

Single below command will complete both tasks

kubectl expose deployment front-end --name=front-end-svc --port=80 --type=NodePort --protocol=TCP

verify its created

kubectl get svc



====================Lab completion =====================