- Check if Kubernetes is running and nodes are ready.

```bash

kubectl cluster-info  ## kubernetes sistemi kurulu olup olmama kontrolu yapilir

kubectl get no             ## sistemde kac tane node kurulmus ona bakilir default olarak

- Create `yaml` file named `web-flask.yaml` and explain fields of it.

```yaml

apiVersion: apps/v1

kind: Deployment

metadata:

  name: web-flask-deploy

spec:

  replicas: 3

  selector:

    matchLabels:

      app: web-flask # deployment bu etiketi taşıyan podlarla eşleşir

  minReadySeconds: 10 #hazır olduğunu anlamamız için geçen süre

  strategy: # pod ekleme çıkarma güncelleme sırasında izlenecek yol

    type: RollingUpdate # sırayla güncelle ya da recreate ile aynı anda güncelle

    rollingUpdate:

      maxUnavailable: 1

      maxSurge: 1

  template: # bu kısımdan itibaren pod ve container detayları tanımlanır

    metadata:

      labels: # pod etikeleri burada

        app: web-flask

        env: front-end

    spec:

      containers:

      - name: web-flask-cont

        image: mefekadocker/web-flask:0.2 # docker hub repodan çekilen imajın adı

        ports:

        - containerPort: 8000 # container içindeki uygulamanın portu

```

- Create the web-flask Deployment.

```bash

kubectl apply -f web-flask.yaml

- Show the Pods detailed information and learn their IP addresses:

```bash

kubectl get pods -o wide   ##podlarin ayrintili bilgisi verilir

We now check we can ping a Pod inside the cluster.

- Create a `test.yaml` file to create a Pod that pings a Pod inside the cluster.

```yaml

apiVersion: v1

kind: Pod

metadata:

  name: busybox-test

spec:

  containers:

  - name: busybox

    image: radial/busyboxplus:curl

    command: ['sh','-c','while true;do sleep 3600;done'] # container oluştururken şu komutu çalıştırmasını sağla, böylelikle çalışıp stop etmeyecek, sürekli çalışacak

```

- Create a `busybox` pod and log into the container.

```bash

kubectl get pods

kubectl apply -f test.yaml

- Show the Pods detailed information and learn their IP addresses again.

```bash

kubectl get pods -o wide

- Scale the deployment down to zero.

```bash

kubectl scale deploy web-flask-deploy --replicas=0

- Scale the deployment up to three replicas.

```bash

kubectl scale deploy web-flask-deploy --replicas=3

- Get the documentation of `Services` and its fields.

```bash

kubectl explain svc

- Create a `web-svc.yaml` file with following content and explain fields of it.

```yaml

apiVersion: v1

kind: Service

metadata:

  name: web-flask-svc

  labels:

    app: web-flask

spec:

  type: ClusterIP

  ports:

  - port: 3000  # clusterIP servis portu

    targetPort: 8000 # container yayınını yakalamak için hedef port

  selector:

    env: front-end # label env: front-end olan podlar eşlenecek

```

```bash

kubectl apply -f web-svc.yaml

- List the services.

```bash

kubectl get svc -o wide

burdaki service modeli olan cluster ip ile private ip gibi davranir ve bu cluster ip ye disardan erişim olmaz sadece ayni worker node icerisnde bulunan podlar ve sevicesler birbiri ile iletişime geçebilirler ping atabilirler.

Yukarıdakı uygulama da bir tane Deployment olusturmak için “web-flask.yaml “ adinda dosya oluşturuldu.BU kisimda “Replicast “ . “podlar “olusuturldu selector kisminda olan tagler yani label lar ile diğer oluşturulacak dosyalardaki labellar =in ayni olmasi gerekmektedir dosyalarin birbirlarii taniyabilmesi için.

Worker node da oluşan podlarinin public ip si alinip Google da yeni sayfada public ip yapistirlip bakilir site gorulup gorulmedidigine .ancak bu pod ulaşılamaz ulasilmasi için piblic ip sonuna iki nokta eklenip yanina cluster ip port numasi yazilidiginda worker node içindeki worker noda ulasilir ve sayfa gorulur.

- Display information about the `web-flask-svc` Service.

```bash

kubectl describe svc web-flask-svc

bu komut ile service içine ayrintili bakilir .

[03:42:48]ubuntu[~]:) $ kubectl describe svc web-flask-svc

Name: web-flask-svc

Namespace: default

Labels: app=web-flask

Annotations: <none>

Selector: env=front-end

Type: ClusterIP

IP Family Policy: SingleStack

IP Families: IPv4

IP: 10.101.138.134

IPs: 10.101.138.134

Port: <unset> 3000/TCP

TargetPort: 8000/TCP

Endpoints: 10.244.1.2:8000,10.244.1.3:8000,10.244.1.4:8000

Session Affinity: None

Events: <none>

Service içindeki bilgiler yukaridaki gibidir. Service cluster ip türündedir lendo ip si gösterir ve podlar icrinde pulunan konteynerlera baglanabilgini g,sterir endpoint oluşurmuştur. Endpoints: 10.244.1.2:8000,10.244.1.3:8000,10.244.1.4:8000

Bununla birlikte podlara baglanti olgunu gostetir burdaki 8000 portu oluşan podlarin icnideki containerler erişim yapan port numarasidir. 10.244.1.2 bu ve di[er 2 si deployment replikasi ile oluşan podlarin içlerindeki kontaynerlarin ip numaralaridir .Targetgroup container portlaridir.Port cluster servis portudur.

kubectl exec -it busybox-test -- sh

/ # curl <IP of service web-flask-svc>:3000

/ # ping web-flask-svc

/ # curl web-flask-svc:3000

Yukarıdakı kubectl exec -it busybox-test – sh komutu ile busybox-test adli dosyanin yani diğer deployment portu sinida olusturf]dugumuz pod olusturlmak için yapilan test yaml dosyadsir bu pod dosyasinin içine girilir bu kod ile ve depolymen içinde oluşan podlara yani containerlara ping atilip ulasildigi g,rulur yani cluster ip li service ayni worker node içinde olan ba;ka bir pod ulaşabilir ancak diasrdan herhangi bir ulasim cluster ip servisne ulasilam olmaz /

Masternode içinde oluşturulan yukaridaki objelrin tumu worker node da caliri o nedenle service objesinin nodeport turu olan objenin port numasi 3000 veril =mis yularidaki ornekte dis portu verilmemiş kubernetes kendi 30000 ile 32757 arasinda bir port numarasi atar.bu pordlar ile disridan nodepord icinnde olan podlaria ulasilmasi saglanir bunu worker node da olusmus olan yukaridaki deployment yaml objesi ile replicasetler ile oluşturulan podlari görebilmek için worker node podlarina public ip si yazilir ve sonuna iki nokta ile30000 ile 32000 li arasinda sistemin atadigi yada kendi atadigimiz port nosu yazilir ve Google yeni saydada denenir .ve boylece pod içindeki conteyner da yazili olan mesaj vs bilgilere vs ulasilir.

Bunu masternode terminalinde de Curl service ip nosu yada service adi ile sonuna 3000 yani service in pord nosu girlidiginde bu srvice içindeki bilgilara yazilara erişim olur ve yazilar gorulur bu curl kommutu ile .