

Week 2 Day 2 – Deployments

DAVID HIUHU

101509799

Deployment Demo

- Minikube with two nodes
 - minikube start --nodes 2 -p multinode-demo

```
Select dhiuhu@DESKTOP-ATLGL31: ~
dhiuhu@DESKTOP-ATLGL31:~$ minikube start --nodes 2 -p multinode-demo
[multinode-demo] minikube v1.28.0 on Ubuntu 22.04 (amd64)
* Automatically selected the docker driver
* Using Docker driver with root privileges
* Starting control plane node multinode-demo in cluster multinode-demo
* Pulling base image ...
* Creating docker container (CPUs=2, Memory=2200MB) ...
* Preparing Kubernetes v1.28.3 on Docker 24.0.7 ...
  * Generating certificates and keys ...
  * Booting up control plane ...
  * Configuring RBAC rules ...
  * Configuring CNI (Container Networking Interface) ...
  * Using image gcr.io/k8s-minikube/storage-provisioner:v5
  * Verifying Kubernetes components...
  * Inhabited addons: storage provisioner, default storageclass
* Starting worker node multinode-demo-m02 in cluster multinode-demo
* Pulling base image ...
* Creating docker container (CPUs=2, Memory=2200MB) ...
* Found network options:
  * NO_PROXY=192.168.58.2
* Preparing Kubernetes v1.28.3 on Docker 24.0.7 ...
  * env NO_PROXY=192.168.58.2
* Verifying Kubernetes components...
* Done! kubectl is now configured to use "multinode-demo" cluster and "default" namespace by default
dhiuhu@DESKTOP-ATLGL31:~$ kubectl get nodes
NAME                STATUS    ROLES    AGE   VERSION
multinode-demo      Ready     control-plane   4m18s   v1.28.3
multinode-demo-m02  Ready     <none>         3m40s   v1.28.3
dhiuhu@DESKTOP-ATLGL31:~$ minikube status -p multinode-demo
multinode-demo
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured

multinode-demo-m02
type: Worker
host: Running
```

The screenshot displays the Kubernetes Dashboard in a web browser at the URL `127.0.0.1:33281/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/#/node?namespace=default`. The dashboard shows the 'Nodes' tab with two nodes listed: `multinode-demo-m02` (Control Plane) and `multinode-demo` (Worker). Both nodes are in a 'Ready' state. A terminal window is overlaid on the dashboard, showing the command `minikube dashboard -p multinode-demo` being executed. The terminal output indicates that the dashboard is being enabled, the proxy is being launched, and the dashboard is accessible at `http://127.0.0.1:33281/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/`.

```
Select dhiuhu@DESKTOP-ATLGL31: ~
Some dashboard features require the metrics-server addon. To enable all features please run:

    minikube -p multinode-demo addons enable metrics-server

Verifying dashboard health ...
Launching proxy ...
Verifying proxy health ...
Opening http://127.0.0.1:33281/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
http://127.0.0.1:33281/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/

^C
dhiuhu@DESKTOP-ATLGL31:~$ minikube stop -p multinode-demo
Stopping node "multinode-demo" ...
Powering off "multinode-demo" via SSH ...
Stopping node "multinode-demo-m02" ...
Powering off "multinode-demo-m02" via SSH ...
2 nodes stopped.
dhiuhu@DESKTOP-ATLGL31:~$ minikube delete --all
Deleting "minikube" in docker ...
Removing /home/dhiuhu/.minikube/machines/minikube ...
Removed all traces of the "minikube" cluster.
Deleting "multinode-demo" in docker ...
Removing /home/dhiuhu/.minikube/machines/multinode-demo ...
Removing /home/dhiuhu/.minikube/machines/multinode-demo-m02 ...
Removed all traces of the "multinode-demo" cluster.
Successfully deleted all profiles
dhiuhu@DESKTOP-ATLGL31:~$ kubectl get nodes
E0119 18:24:06.588508 45632 memcache.go:265] couldn't get current server API group list: the server could not find the requested resource
E0119 18:24:06.591674 45632 memcache.go:265] couldn't get current server API group list: the server could not find the requested resource
E0119 18:24:06.595026 45632 memcache.go:265] couldn't get current server API group list: the server could not find the requested resource
E0119 18:24:06.597728 45632 memcache.go:265] couldn't get current server API group list: the server could not find the requested resource
E0119 18:24:06.599950 45632 memcache.go:265] couldn't get current server API group list: the server could not find the requested resource
Error from server (NotFound): the server could not find the requested resource
dhiuhu@DESKTOP-ATLGL31:~$
```

```
dhiuhu@DESKTOP-ATLGL31:~$ minikube delete --all
Deleting "minikube" in docker ...
Removing /home/dhiuhu/.minikube/machines/minikube ...
Removed all traces of the "minikube" cluster.
Deleting "multinode-demo" in docker ...
Removing /home/dhiuhu/.minikube/machines/multinode-demo ...
Removing /home/dhiuhu/.minikube/machines/multinode-demo-m02 ...
Removed all traces of the "multinode-demo" cluster.
Successfully deleted all profiles
dhiuhu@DESKTOP-ATLGL31:~$ kubectl get nodes
E0119 18:24:06.588508 45632 memcache.go:265] couldn't get current server API group list: the server could not find the requested resource
E0119 18:24:06.591674 45632 memcache.go:265] couldn't get current server API group list: the server could not find the requested resource
E0119 18:24:06.595026 45632 memcache.go:265] couldn't get current server API group list: the server could not find the requested resource
E0119 18:24:06.597728 45632 memcache.go:265] couldn't get current server API group list: the server could not find the requested resource
E0119 18:24:06.599950 45632 memcache.go:265] couldn't get current server API group list: the server could not find the requested resource
Error from server (NotFound): the server could not find the requested resource
dhiuhu@DESKTOP-ATLGL31:~$ clear
```

ReplicaSet Demo

- Replica Set YAML.

```
dhiuhu@DESKTOP-ATLGL31: ~
GNU nano 6.2 replicaset.yaml
apiVersion: apps/v1
kind: ReplicaSet
metadata:
  name: nginx
  labels:
    app: nginx
    tier: lb
spec:
  replicas: 3
  selector:
    matchLabels:
      tier: lb
  template:
    metadata:
      labels:
        tier: lb
  spec:
    containers:
      - name: nginx-replicaset
        image: nginx
```

- **Apply ReplicaSet YAML.**

kubectl apply -f replicaset.yaml

Delete a pod in the ReplicaSet

kubectl delete pod nginx-<code>

kubectl get pods

kubectl get replicaset

```
dhiuhu@DESKTOP-ATLGL31:~$ nano replicaset.yaml
dhiuhu@DESKTOP-ATLGL31:~$ kubectl apply -f replicaset.yaml
replicaset.apps/nginx unchanged
dhiuhu@DESKTOP-ATLGL31:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
nginx-54w2c    1/1     Running   0           107s
nginx-fk4tg    1/1     Running   0           4m12s
nginx-sw6n8    1/1     Running   0           4m22s
dhiuhu@DESKTOP-ATLGL31:~$ kubectl delete pod nginx-54w2c
pod "nginx-54w2c" deleted
dhiuhu@DESKTOP-ATLGL31:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
nginx-fk4tg    1/1     Running   0           4m44s
nginx-h97ce    1/1     Running   0           6s
nginx-sw6n8    1/1     Running   0           4m44s
dhiuhu@DESKTOP-ATLGL31:~$ kubectl delete pod nginx-sw6n8
pod "nginx-sw6n8" deleted
dhiuhu@DESKTOP-ATLGL31:~$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
nginx-cgpc7    1/1     Running   0           3s
nginx-fk4tg    1/1     Running   0           5m
nginx-h97ce    1/1     Running   0           22s
dhiuhu@DESKTOP-ATLGL31:~$ kubectl get replicaset
NAME          DESIRED   CURRENT   READY   AGE
nginx         3          3          3       5m33s
dhiuhu@DESKTOP-ATLGL31:~$
```

Deployment YAML with ReplicaSet

```
dhiuhu@DESKTOP-ATLGL31:~$ nano deployment.yaml
GNU nano 6.2
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:1.14.2
          ports:
            - containerPort: 80
```

kubectl get deployments

kubectl rollout status deployment nginx

```
dhiuhu@DESKTOP-ATLGL31: ~  
dhiuhu@DESKTOP-ATLGL31:~$ nano deployment.yml  
dhiuhu@DESKTOP-ATLGL31:~$ kubectl apply -f deployment.yml  
deployment.apps/nginx-deployment unchanged  
dhiuhu@DESKTOP-ATLGL31:~$ kubectl get deployments  
NAME                READY   UP-TO-DATE   AVAILABLE   AGE  
nginx-deployment    3/3     3            3           2m29s  
dhiuhu@DESKTOP-ATLGL31:~$ kubectl rollout status deployment nginx-deployment  
deployment "nginx-deployment" successfully rolled out  
dhiuhu@DESKTOP-ATLGL31:~$
```

StatefulSet YAML

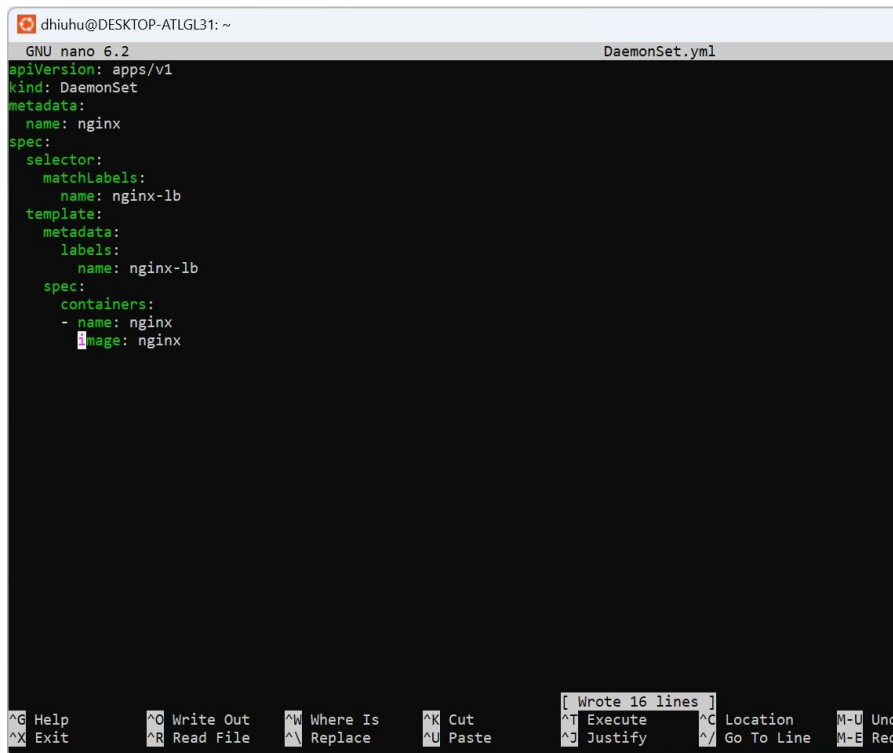
```
Select dhiuhu@DESKTOP-ATLGL31: ~  
GNU nano 6.2 StatefulSet.yml  
apiVersion: v1  
kind: Service  
metadata:  
  name: nginx  
  labels:  
    app: nginx  
spec:  
  ports:  
  - port: 80  
    name: web  
    clusterIP: None  
  selector:  
    app: nginx  
---  
apiVersion: apps/v1  
kind: StatefulSet  
metadata:  
  name: web  
spec:  
  selector:  
    matchLabels:  
      app: nginx # has to match .spec.template.metadata.labels  
  serviceName: "nginx"  
  replicas: 3 # by default is 1  
  template:  
    metadata:  
      labels:  
        app: nginx # has to match .spec.selector.matchLabels  
    spec:  
      terminationGracePeriodSeconds: 10  
      containers:  
      - name: nginx  
        image: k8s.gcr.io/nginx-slim:0.8  
        ports:  
        - containerPort: 80  
          name: web  
Wrote 48 lines
```

```
dhiuhu@DESKTOP-ATLGL31: ~
GNU nano 6.2 StatefulSet.yml
apiVersion: v1
kind: Service
metadata:
  name: nginx
  labels:
    app: nginx
spec:
  ports:
    - port: 80
      name: web
  clusterIP: None
  selector:
    app: nginx
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: web
spec:
  selector:
    matchLabels:
      app: nginx # has to match .spec.template.metadata.labels
  serviceName: "nginx"
  replicas: 3 # by default is 1
  template:
    metadata:
      labels:
        app: nginx # has to match .spec.selector.matchLabels
    spec:
      terminationGracePeriodSeconds: 10
      containers:
        - name: nginx
          image: k8s.gcr.io/nginx-slim:0.8
```

kubectl get statefulsets

```
dhiuhu@DESKTOP-ATLGL31: ~
dhiuhu@DESKTOP-ATLGL31:~$ nano StatefulSet.yml
dhiuhu@DESKTOP-ATLGL31:~$ kubectl apply -f StatefulSet.yml
service/nginx unchanged
statefulset.apps/web configured
dhiuhu@DESKTOP-ATLGL31:~$ kubectl get statefulsets
NAME      READY   AGE
web       0/3     14m
dhiuhu@DESKTOP-ATLGL31:~$
```

DaemonSet YAML.

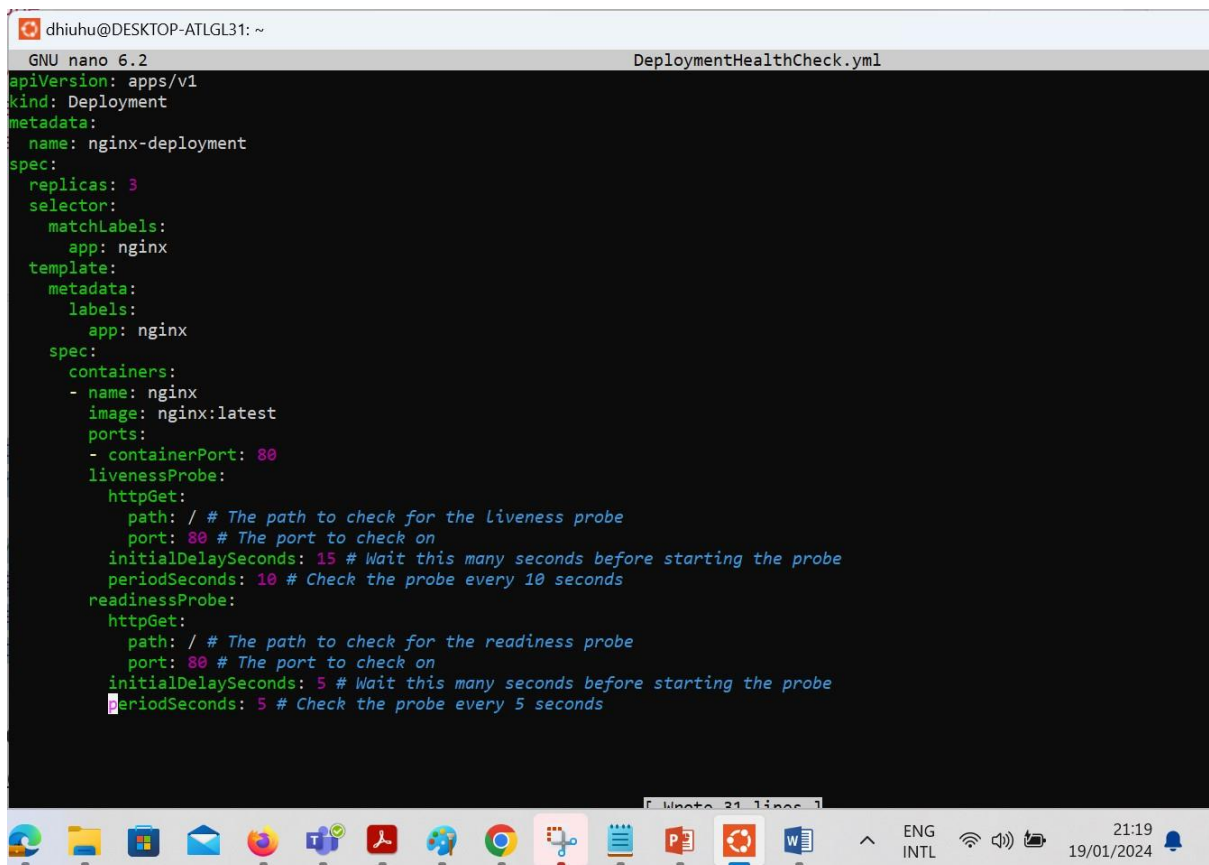


A screenshot of a terminal window with the title bar 'dhiuhu@DESKTOP-ATLGL31: ~'. The window shows the nano 6.2 editor editing a file named 'DaemonSet.yml'. The YAML content is as follows:

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: nginx
spec:
  selector:
    matchLabels:
      name: nginx-lb
  template:
    metadata:
      labels:
        name: nginx-lb
    spec:
      containers:
      - name: nginx
        image: nginx
```

The bottom status bar of the nano editor shows various keyboard shortcuts: ^G Help, ^O Write Out, ^W Where Is, ^K Cut, ^T Execute, ^C Location, M-U Und, ^X Exit, ^R Read File, ^_ Replace, ^U Paste, ^J Justify, ^/ Go To Line, and M-E Red.

Deployment example with resource limits.



A screenshot of a terminal window with the title bar 'dhiuhu@DESKTOP-ATLGL31: ~'. The window shows the nano 6.2 editor editing a file named 'DeploymentHealthCheck.yml'. The YAML content is as follows:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        ports:
        - containerPort: 80
        livenessProbe:
          httpGet:
            path: / # The path to check for the liveness probe
            port: 80 # The port to check on
            initialDelaySeconds: 15 # Wait this many seconds before starting the probe
            periodSeconds: 10 # Check the probe every 10 seconds
        readinessProbe:
          httpGet:
            path: / # The path to check for the readiness probe
            port: 80 # The port to check on
            initialDelaySeconds: 5 # Wait this many seconds before starting the probe
            periodSeconds: 5 # Check the probe every 5 seconds
```

The bottom status bar of the nano editor shows '[Wrote 31 lines]'. The Windows taskbar is visible at the bottom of the screen, showing various application icons and the system clock indicating 21:19 on 19/01/2024.

Deployment with health checks.

```
dhiuhu@DESKTOP-ATLGL31: ~
GNU nano 6.2 DeploymentHealthCheck.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:latest
          ports:
            - containerPort: 80
          livenessProbe:
            httpGet:
              path: / # The path to check for the liveness probe
              port: 80 # The port to check on
            initialDelaySeconds: 15 # Wait this many seconds before starting the probe
            periodSeconds: 10 # Check the probe every 10 seconds
          readinessProbe:
            httpGet:
              path: / # The path to check for the readiness probe
              port: 80 # The port to check on
            initialDelaySeconds: 5 # Wait this many seconds before starting the probe
            periodSeconds: 5 # Check the probe every 5 seconds
```

```
dhiuhu@DESKTOP-ATLGL31: ~
dhiuhu@DESKTOP-ATLGL31:~$ nano DeploymentHealthCheck.yml
dhiuhu@DESKTOP-ATLGL31:~$ kubectl apply -f DeploymentHealthCheck.yml
deployment.apps/nginx-deployment configured
dhiuhu@DESKTOP-ATLGL31:~$ kubectl get deployments
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
nginx-deployment 3/3      3            3           46m
dhiuhu@DESKTOP-ATLGL31:~$
```

Deploy MERN App To K8s (Minikube)

1. Clone repo on your local machine. Change directory to kubernetes

```
dhiuhu@DESKTOP-ATLGL31: ~/kubernetes-demo-main/kubernetes
dhiuhu@DESKTOP-ATLGL31:~$ ls
DaemonSet.yml      StatefulSet.yml  deployment.yml   flash-app        kubernetes-demo-main
DeploymentHealthCheck.yml  best_mount      example-voting-app  flask-app        replicaset.yml
dhiuhu@DESKTOP-ATLGL31:~$ cd kubernetes-demo-main
dhiuhu@DESKTOP-ATLGL31:~/kubernetes-demo-main$ ls
kubernetes
dhiuhu@DESKTOP-ATLGL31:~/kubernetes-demo-main$ cd to kubernetes
-bash: cd: too many arguments
dhiuhu@DESKTOP-ATLGL31:~/kubernetes-demo-main$ cd kubernetes
dhiuhu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ ls
deployments  secrets  services  stateful-sets
dhiuhu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$
```

2. Make sure you have Minikube installed and running properly.

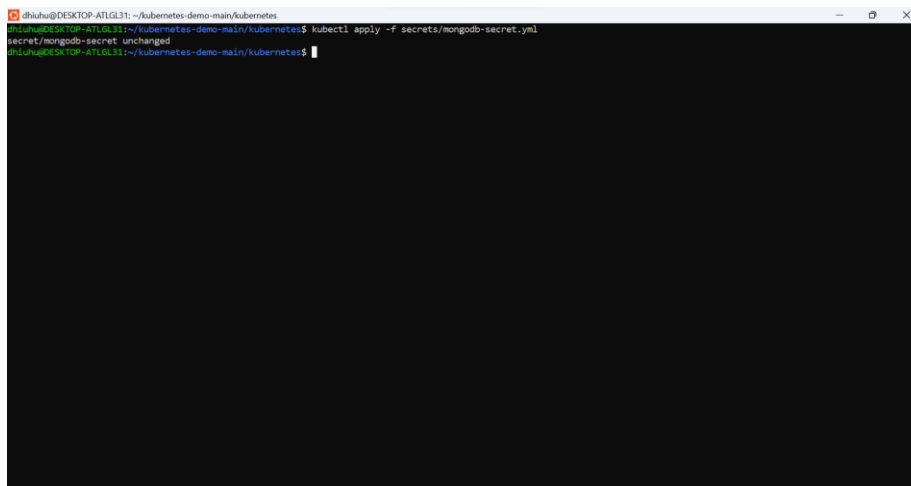
a. Start Minikube

```
Select dhiuhu@DESKTOP-ATLGL31: ~/kubernetes-demo-main/kubernetes
dhiuhu@DESKTOP-ATLGL31:~$ ls
DaemonSet.yml      StatefulSet.yml  deployment.yml   flash-app        kubernetes-demo-main
DeploymentHealthCheck.yml  best_mount      example-voting-app  flask-app        replicaset.yml
dhiuhu@DESKTOP-ATLGL31:~$ cd kubernetes-demo-main
dhiuhu@DESKTOP-ATLGL31:~/kubernetes-demo-main$ ls
kubernetes
dhiuhu@DESKTOP-ATLGL31:~/kubernetes-demo-main$ cd to kubernetes
-bash: cd: too many arguments
dhiuhu@DESKTOP-ATLGL31:~/kubernetes-demo-main$ cd kubernetes
dhiuhu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ ls
deployments  secrets  services  stateful-sets
dhiuhu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ minikube start
🔧 minikube v1.32.0 on Ubuntu 22.04 (amd64)
👉 Using the docker driver based on existing profile
👉 Starting control plane node minikube in cluster minikube
👉 Pulling base image ...
👉 Restarting existing docker container for "minikube" ...
👉 Preparing Kubernetes v1.28.3 on Docker 24.0.7 ...
👉 Configuring bridge CNI (Container Networking Interface) ...
👉 Verifying Kubernetes components...
  * Using image gcr.io/k8s-minikube/storage-provisioner:v5
  * Enabled addons: default-storageclass, storage-provisioner
👉 Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
dhiuhu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$
```

3. You can use a Kubernetes IDE like "Lens" to connect to your Minikube cluster. Makes it easier for monitoring changes. GIT hub for LENSs did not work <https://github.com/lensapp/lens/releases/download/v4.1.3/Lens-4.1.3.AppImage>

4. create the secret. (Holds mongodb admin username and password)

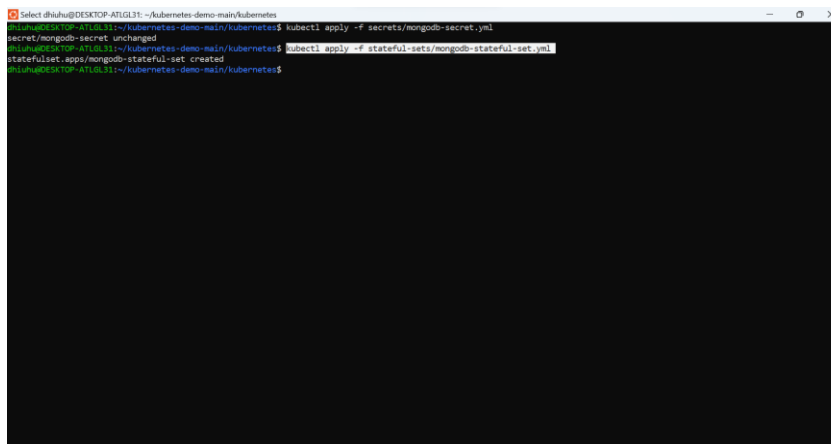
a. kubectl apply -f secrets/mongodb-secret.yml



```
dhiuh@DESKTOP-ATLGL31: ~/kubernetes-demo-main/kubernetes
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f secrets/mongodb-secret.yml
secret/mongodb-secret unchanged
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$
```

5. create the mongodb stateful set. It will create 2 replicas.

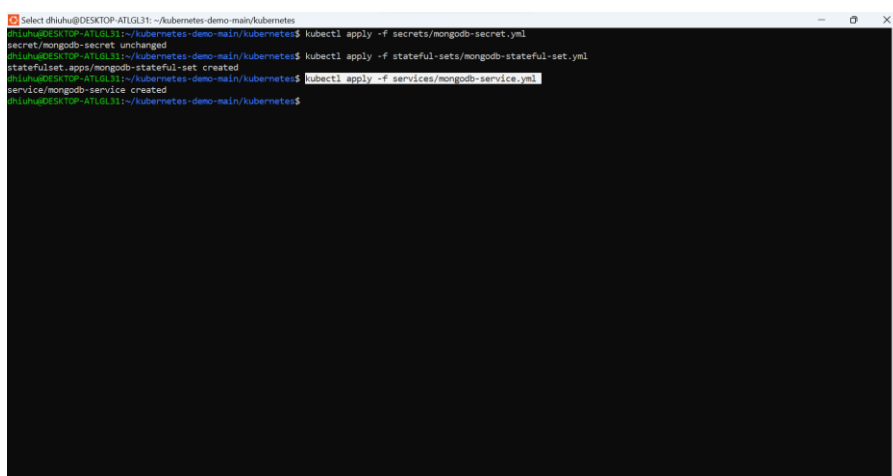
a. kubectl apply -f stateful-sets/mongodb-stateful-set.yml



```
Select dhiuh@DESKTOP-ATLGL31: ~/kubernetes-demo-main/kubernetes
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f secrets/mongodb-secret.yml
secret/mongodb-secret unchanged
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f stateful-sets/mongodb-stateful-set.yml
statefulset.apps/mongodb-stateful-set created
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$
```

6. create the mongodb internal service so that our server can later access the mongodb pods using it.

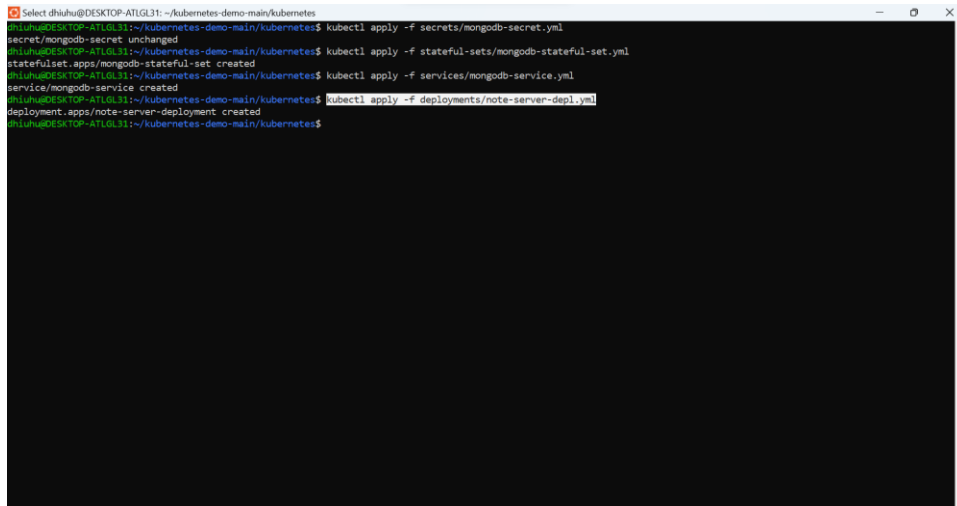
a. kubectl apply -f services/mongodb-service.yml



```
Select dhiuh@DESKTOP-ATLGL31: ~/kubernetes-demo-main/kubernetes
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f secrets/mongodb-secret.yml
secret/mongodb-secret unchanged
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f stateful-sets/mongodb-stateful-set.yml
statefulset.apps/mongodb-stateful-set created
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/mongodb-service.yml
service/mongodb-service created
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$
```

7. deploy the server app by running:

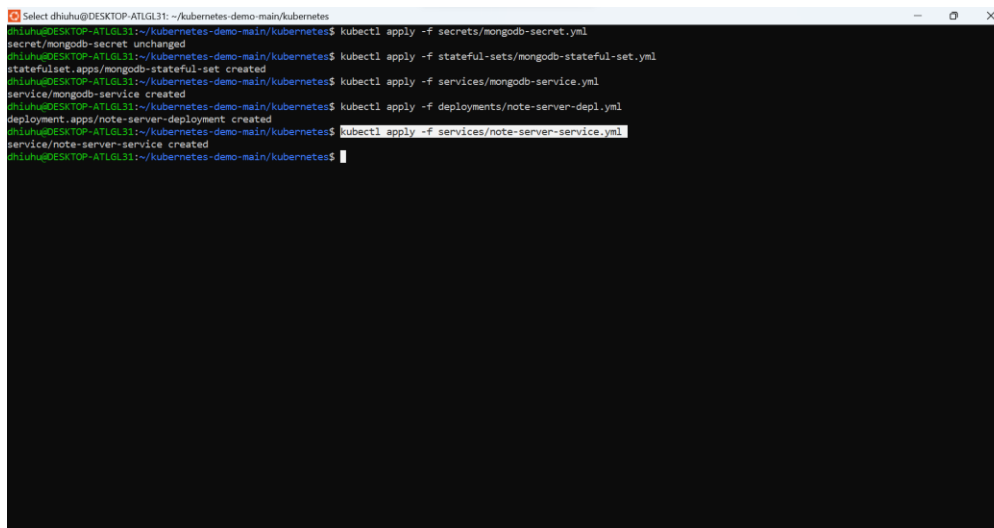
a. `kubectl apply -f deployments/note-server-depl.yml`



```
Select dhiuh@DESKTOP-ATLGL31: ~/kubernetes-demo-main/kubernetes
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f secrets/mongodb-secret.yml
secret/mongodb-secret unchanged
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f stateful-sets/mongodb-stateful-set.yml
statefulset.apps/mongodb-stateful-set created
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/mongodb-service.yml
service/mongodb-service created
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f deployments/note-server-depl.yml
deployment.apps/note-server-deployment created
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$
```

8. create the server internal service for communication between the frontend and the backend.

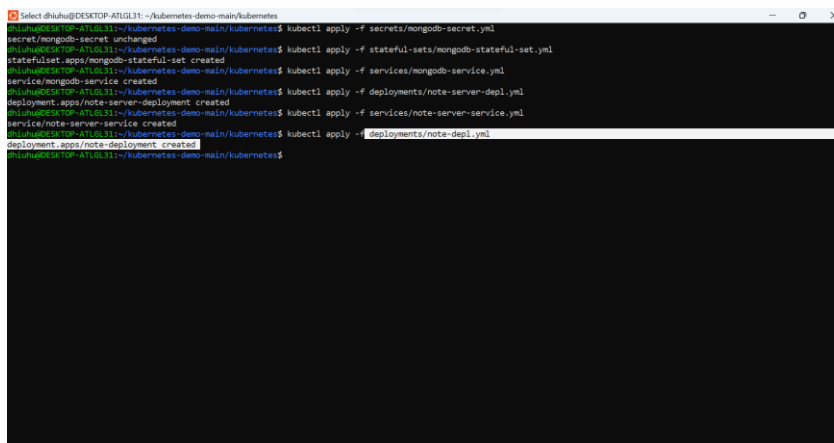
a. `kubectl apply -f services/note-server-service.yml`



```
Select dhiuh@DESKTOP-ATLGL31: ~/kubernetes-demo-main/kubernetes
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f secrets/mongodb-secret.yml
secret/mongodb-secret unchanged
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f stateful-sets/mongodb-stateful-set.yml
statefulset.apps/mongodb-stateful-set created
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/mongodb-service.yml
service/mongodb-service created
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f deployments/note-server-depl.yml
deployment.apps/note-server-deployment created
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/note-server-service.yml
service/note-server-service created
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$
```

9. Deploy frontend:

a. `kubectl apply -f deployments/note-depl.yml`



```
Select dhiuh@DESKTOP-ATLGL31: ~/kubernetes-demo-main/kubernetes
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f secrets/mongodb-secret.yml
secret/mongodb-secret unchanged
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f stateful-sets/mongodb-stateful-set.yml
statefulset.apps/mongodb-stateful-set created
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/mongodb-service.yml
service/mongodb-service created
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f deployments/note-server-depl.yml
deployment.apps/note-server-deployment created
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/note-server-service.yml
service/note-server-service created
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f deployments/note-depl.yml
deployment.apps/note-deployment created
dhiuh@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$
```

10. Create external service for the frontend app (Load Balancer) by running:

a. `kubectl apply -f services/note-service.yml`

```
Select dhihu@DESKTOP-ATLGL31: ~/kubernetes-demo-main/kubernetes$ kubectl apply -f secrets/mongodb-secret.yml
secret/mongodb-secret unchanged
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f stateful-sets/mongodb-stateful-set.yml
statefulset.apps/mongodb-stateful-set created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/mongodb-service.yml
service/mongodb-service created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f deployments/note-server-depl.yml
deployment.apps/note-server-deployment created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/note-server-service.yml
service/note-server-service created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f deployments/note-depl.yml
deployment.apps/note-deployment created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/note-service.yml
service/note-service created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$
```

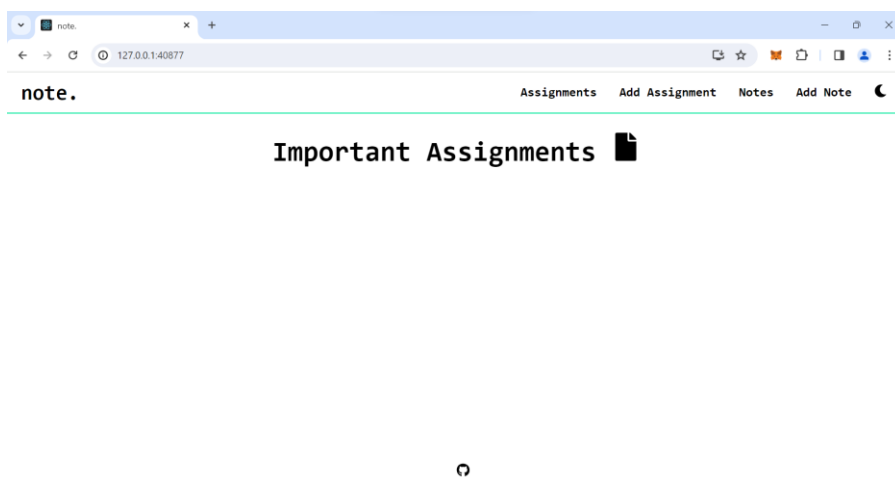
11. To get an external IP assigned to the frontend deployment you can run:

a. `minikube service note-service`

```
dhihu@DESKTOP-ATLGL31: ~/kubernetes-demo-main/kubernetes$ kubectl apply -f secrets/mongodb-secret.yml
secret/mongodb-secret unchanged
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f stateful-sets/mongodb-stateful-set.yml
statefulset.apps/mongodb-stateful-set created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/mongodb-service.yml
service/mongodb-service created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f deployments/note-server-depl.yml
deployment.apps/note-server-deployment created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/note-server-service.yml
service/note-server-service created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f deployments/note-depl.yml
deployment.apps/note-deployment created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/note-service.yml
service/note-service created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ minikube service note-service
-----
NAMESPACE | NAME      | TARGET PORT | URL
-----
default    | note-service | 3080         | http://192.168.49.2:32322
-----
A Starting tunnel for service note-service.
-----
NAMESPACE | NAME      | TARGET PORT | URL
-----
default    | note-service | 3080         | http://127.0.0.1:40877
-----
# Opening service default/note-service in default browser...
# http://127.0.0.1:40877
! Because you are using a Docker driver on linux, the terminal needs to be open to run it.
```

12. Open the IP address in browser and you will be able to see the app running!

a. `default | note-service | http://127.0.0.1:40877`



13.If you want to run a database GUI like mongo-express, you can simply run:

a. kubectl apply -f deployments/mongo-express-depl.yml

```
Select dhihu@DESKTOP-ATLGL31: ~/kubernetes-demo-main/kubernetes
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/note-service.yml
service/note-service created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/note-service.yml
service createdservice/note-service unchanged
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ minikube service note-service
-----
|NAMESPACE|NAME|TARGET PORT|URL|
|-----|-----|-----|-----|
|default|note-service|3000|http://192.168.49.2:32322|
|-----|-----|-----|-----|
A. Starting tunnel for service note-service.
|NAMESPACE|NAME|TARGET PORT|URL|
|-----|-----|-----|-----|
|default|note-service||http://127.0.0.1:40877|
|-----|-----|-----|-----|
# Opening service default/note-service in default browser...
Q http://127.0.0.1:40877
! Because you are using a Docker driver on linux, the terminal needs to be open to run it.
^C@ Stopping tunnel for service note-service.
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f deployments/mongo-express-depl.yml
deployment.apps/mongo-express-deployment created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$
```

14.Later create mongo-express external service for accessing the GUI in browser:

a. kubectl apply -f services/mongo-express-service.yml

b. minikube service mongo-express-service

```
Select dhihu@DESKTOP-ATLGL31: ~/kubernetes-demo-main/kubernetes
|NAMESPACE|NAME|TARGET PORT|URL|
|-----|-----|-----|-----|
|default|note-service|3000|http://192.168.49.2:32322|
|-----|-----|-----|-----|
A. Starting tunnel for service note-service.
|NAMESPACE|NAME|TARGET PORT|URL|
|-----|-----|-----|-----|
|default|note-service||http://127.0.0.1:40877|
|-----|-----|-----|-----|
# Opening service default/note-service in default browser...
Q http://127.0.0.1:40877
! Because you are using a Docker driver on linux, the terminal needs to be open to run it.
^C@ Stopping tunnel for service note-service.
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f deployments/mongo-express-depl.yml
deployment.apps/mongo-express-deployment created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ /mongo-express-depl.yml
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/mongo-express-service.yml
service/mongo-express-service created
dhihu@DESKTOP-ATLGL31:~/kubernetes-demo-main/kubernetes$
```

c. Mongo Express Service

```
Select dhiuh@DESKTOP-ATLG31: ~/kubernetes-demo-main/kubernetes
dhiuh@DESKTOP-ATLG31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f deployments/mongo-express-depl.yml
deployment.apps/mongo-express-deployment created
dhiuh@DESKTOP-ATLG31:~/kubernetes-demo-main/kubernetes$ /mongo-express-depl.yml
nt.apps/mongo-express-deployment created-bash: /mongo-express-depl.yml: No such file or directory
dhiuh@DESKTOP-ATLG31:~/kubernetes-demo-main/kubernetes$ kubectl apply -f services/mongo-express-service.yml
service/mongo-express-service created
dhiuh@DESKTOP-ATLG31:~/kubernetes-demo-main/kubernetes$ minikube service mongo-express-service
```

| NAMESPACE | NAME | TARGET PORT | URL |
|-----------|-----------------------|-------------|---------------------------|
| default | mongo-express-service | 8081 | http://192.168.49.2:30729 |

```

# Starting tunnel for service mongo-express-service.
NAMESPACE   NAME          TARGET PORT  URL
default     mongo-express-service  8081         http://127.0.0.1:41245
# Opening service default/mongo-express-service in default browser...
http://127.0.0.1:41245
Because you are using a Docker driver on linux, the terminal needs to be open to run it.
```

d. the IP address in browser and you will be able to see the app running!

e. default | mongo-express-service | |
http://127.0.0.1:41245

