CSE 322

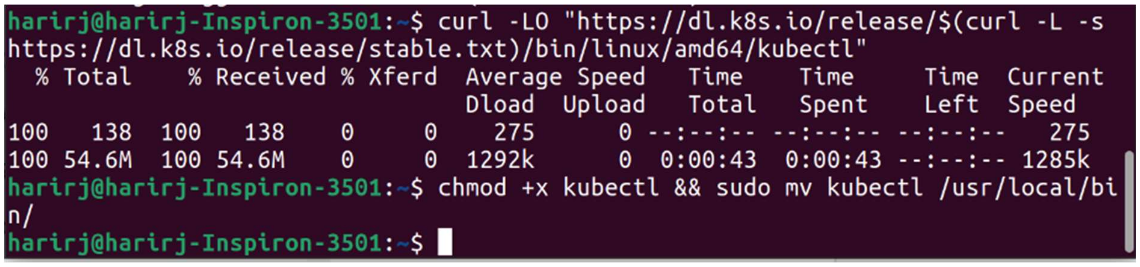
CLOUD COMPUTING

LAB 8

R J HARI **2022BCS0125**

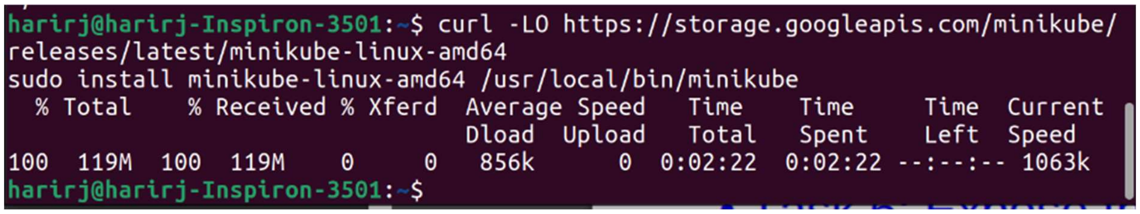
Task 1: Installation of a Kubernetes Cluster

Step 1 : Install kubectl



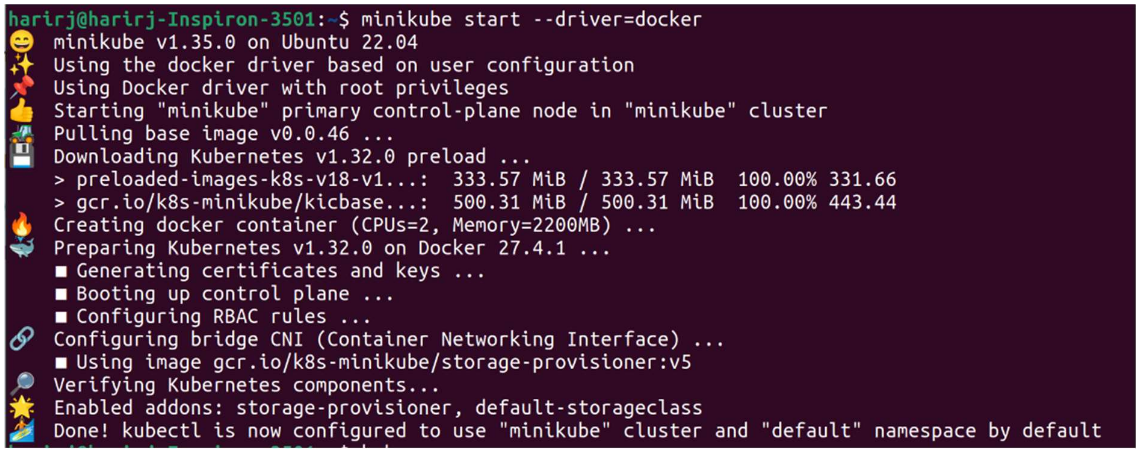
```
harirj@harirj-Inspiron-3501:~$ curl -LO "https://dl.k8s.io/release/$(curl -L -s
https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   138  100   138    0     0    275      0 --:--:-- --:--:-- --:--:--   275
100 54.6M  100 54.6M    0     0  1292k      0  0:00:43  0:00:43 --:--:-- 1285k
harirj@harirj-Inspiron-3501:~$ chmod +x kubectl && sudo mv kubectl /usr/local/bi
n/
harirj@harirj-Inspiron-3501:~$ 
```

Step 2:Install minikube



```
harirj@harirj-Inspiron-3501:~$ curl -LO https://storage.googleapis.com/minikube/
releases/latest/minikube-linux-amd64
sudo install minikube-linux-amd64 /usr/local/bin/minikube
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  119M  100  119M    0     0   856k      0  0:02:22  0:02:22 --:--:-- 1063k
harirj@harirj-Inspiron-3501:~$
```

Step 3 : Start the Kubernetes cluster

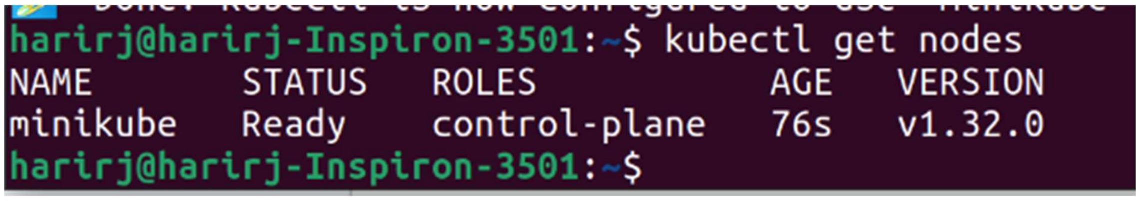


```
harirj@harirj-Inspiron-3501:~$ minikube start --driver=docker
😄  minikube v1.35.0 on Ubuntu 22.04
✨  Using the docker driver based on user configuration
📌  Using Docker driver with root privileges
👍  Starting "minikube" primary control-plane node in "minikube" cluster
🚜  Pulling base image v0.0.46 ...
💾  Downloading Kubernetes v1.32.0 preload ...
    > preloaded-images-k8s-v18-v1...:  333.57 MiB / 333.57 MiB  100.00% 331.66
    > gcr.io/k8s-minikube/kicbase...:  500.31 MiB / 500.31 MiB  100.00% 443.44
🔥  Creating docker container (CPUs=2, Memory=2200MB) ...
🐳  Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
    ▪ Generating certificates and keys ...
    ▪ Booting up control plane ...
    ▪ Configuring RBAC rules ...
🔗  Configuring bridge CNI (Container Networking Interface) ...
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🔎  Verifying Kubernetes components...
🌟  Enabled addons: storage-provisioner, default-storageclass
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

Task 2: Check the Cluster Details

Check cluster nodes

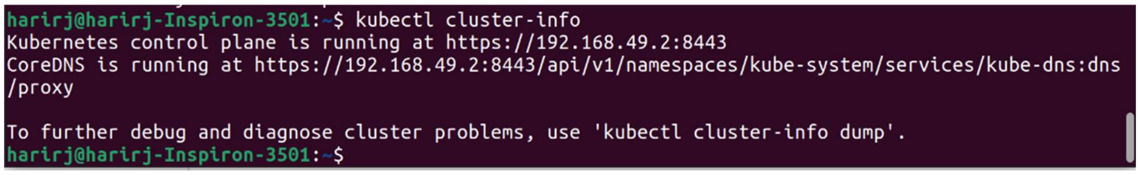
kubectl get nodes



```
harirj@harirj-Inspiron-3501:~$ kubectl get nodes
NAME       STATUS   ROLES           AGE   VERSION
minikube   Ready    control-plane   76s   v1.32.0
harirj@harirj-Inspiron-3501:~$
```

Check cluster information

kubectl cluster-info

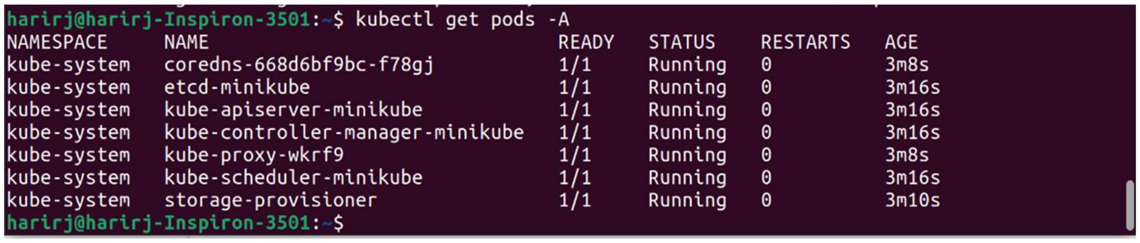


```
harirj@harirj-Inspiron-3501:~$ kubectl cluster-info
Kubernetes control plane is running at https://192.168.49.2:8443
CoreDNS is running at https://192.168.49.2:8443/api/v1/namespaces/kube-system/services/kube-dns:dns
/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
harirj@harirj-Inspiron-3501:~$
```

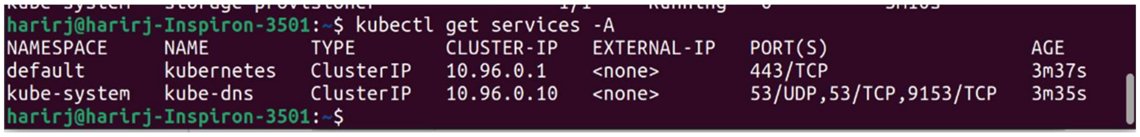
Check running pods and services

kubectl get pods -A



```
harirj@harirj-Inspiron-3501:~$ kubectl get pods -A
NAMESPACE     NAME                               READY   STATUS    RESTARTS   AGE
kube-system   coredns-668d6bf9bc-f78gj           1/1     Running   0          3m8s
kube-system   etcd-minikube                      1/1     Running   0          3m16s
kube-system   kube-apiserver-minikube            1/1     Running   0          3m16s
kube-system   kube-controller-manager-minikube   1/1     Running   0          3m16s
kube-system   kube-proxy-wkrf9                   1/1     Running   0          3m8s
kube-system   kube-scheduler-minikube            1/1     Running   0          3m16s
kube-system   storage-provisioner                1/1     Running   0          3m10s
harirj@harirj-Inspiron-3501:~$
```

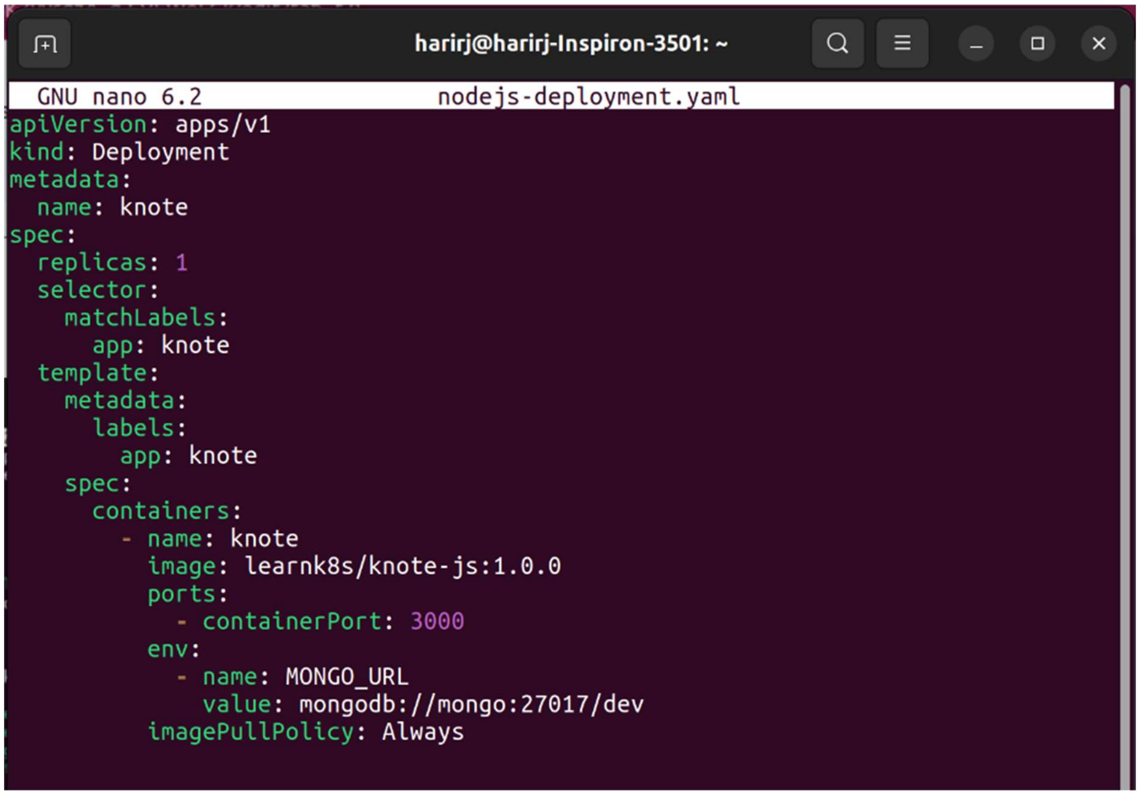
kubectl get services -A



```
harirj@harirj-Inspiron-3501:~$ kubectl get services -A
NAMESPACE     NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)                  AGE
default       kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP                  3m37s
kube-system   kube-dns     ClusterIP   10.96.0.10   <none>        53/UDP,53/TCP,9153/TCP   3m35s
harirj@harirj-Inspiron-3501:~$
```

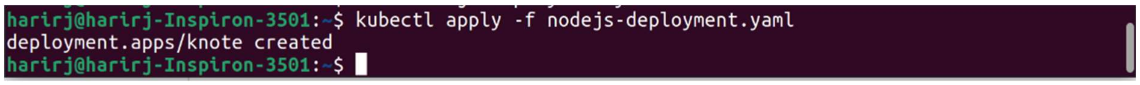
Task 3: Creating Deployments and Running a Node.js Application

Step 1 : Create a Deployment YAML file (nodejs-deployment.yaml)



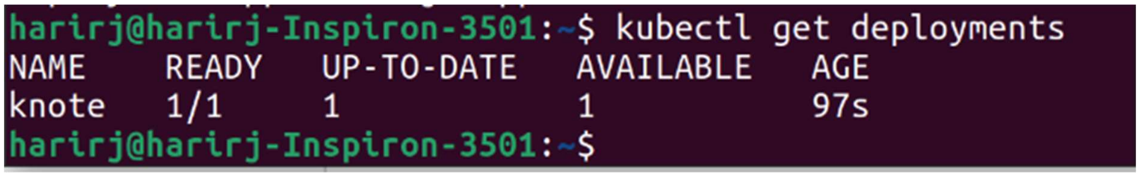
```
  GNU nano 6.2                    nodejs-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: knote
spec:
  replicas: 1
  selector:
    matchLabels:
      app: knote
  template:
    metadata:
      labels:
        app: knote
    spec:
      containers:
        - name: knote
          image: learnk8s/knote-js:1.0.0
          ports:
            - containerPort: 3000
          env:
            - name: MONGO_URL
              value: mongodb://mongo:27017/dev
          imagePullPolicy: Always
```

Step 2 : Apply the Deployment

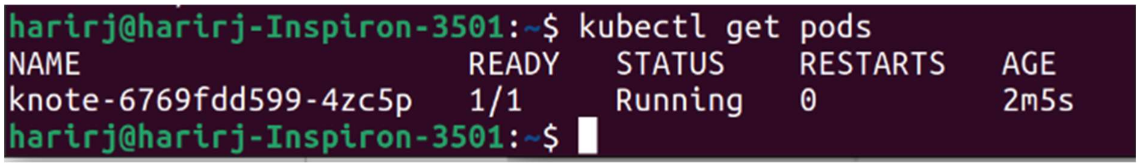


```
harirj@harirj-Inspiron-3501:~$ kubectl apply -f nodejs-deployment.yaml
deployment.apps/knote created
harirj@harirj-Inspiron-3501:~$
```

Step 3 : Verify Deployment

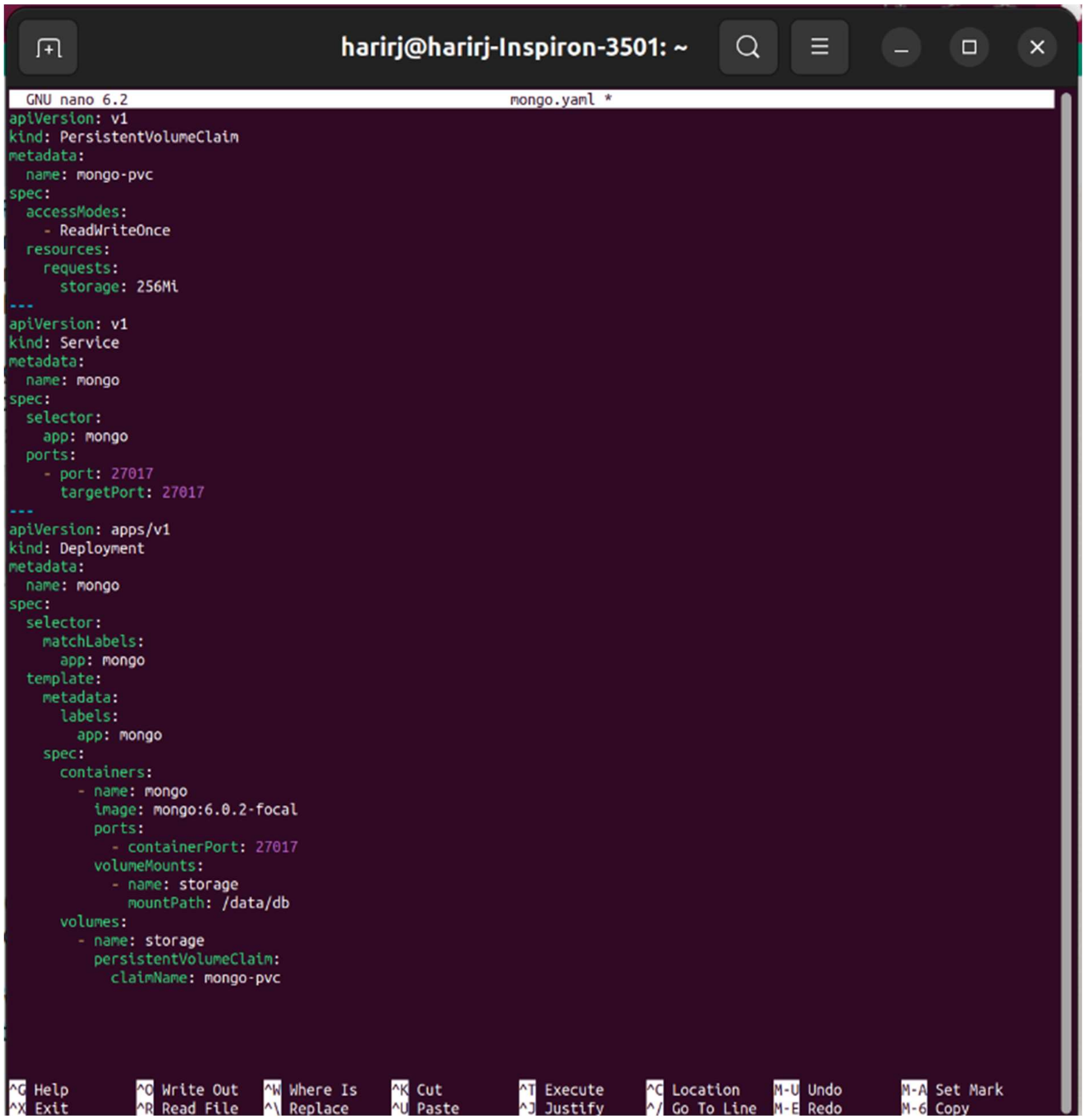


```
harirj@harirj-Inspiron-3501:~$ kubectl get deployments
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
knote   1/1     1            1           97s
harirj@harirj-Inspiron-3501:~$
```



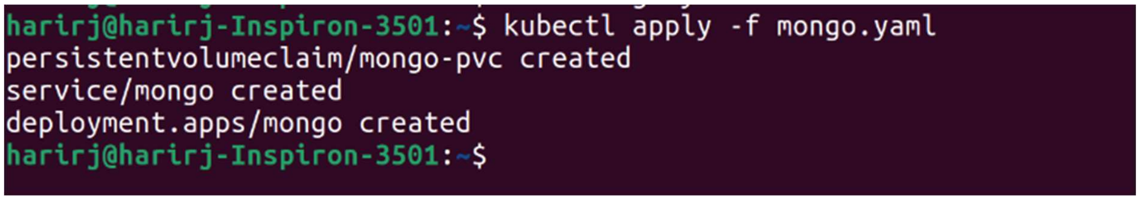
```
harirj@harirj-Inspiron-3501:~$ kubectl get pods
NAME                      READY   STATUS    RESTARTS   AGE
knote-6769fdd599-4zc5p   1/1     Running   0          2m5s
harirj@harirj-Inspiron-3501:~$
```

Deployment Service and Persistent Volume Clain YAML file for mongo DB Database



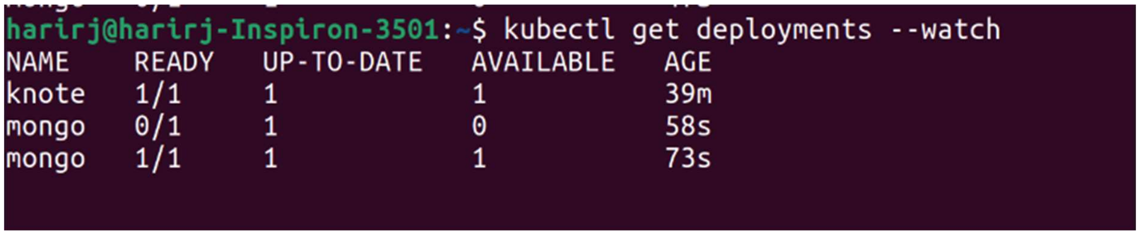
```
  GNU nano 6.2                                        mongo.yaml *
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mongo-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 256Mi
---
apiVersion: v1
kind: Service
metadata:
  name: mongo
spec:
  selector:
    app: mongo
  ports:
    - port: 27017
      targetPort: 27017
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongo
spec:
  selector:
    matchLabels:
      app: mongo
  template:
    metadata:
      labels:
        app: mongo
    spec:
      containers:
        - name: mongo
          image: mongo:6.0.2-focal
          ports:
            - containerPort: 27017
          volumeMounts:
            - name: storage
              mountPath: /data/db
      volumes:
        - name: storage
          persistentVolumeClaim:
            claimName: mongo-pvc
```

Applying mongo.yaml

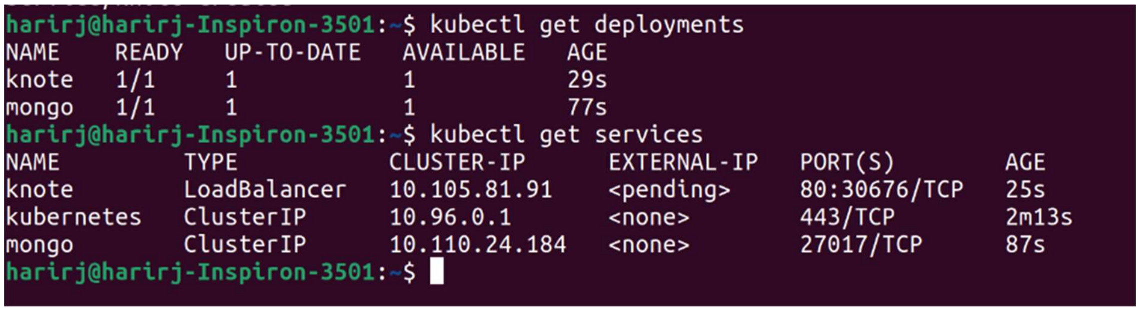


```
harirj@harirj-Inspiron-3501:~$ kubectl apply -f mongo.yaml
persistentvolumeclaim/mongo-pvc created
service/mongo created
deployment.apps/mongo created
harirj@harirj-Inspiron-3501:~$
```

Verifying deployment and Services



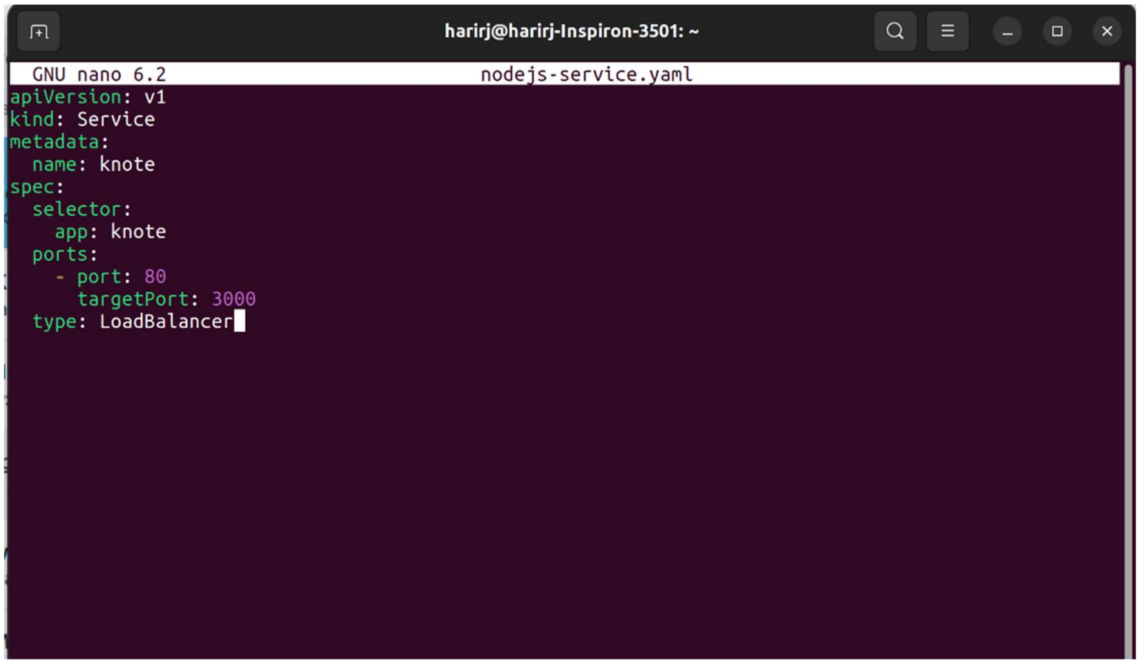
```
harirj@harirj-Inspiron-3501:~$ kubectl get deployments --watch
NAME     READY   UP-TO-DATE   AVAILABLE   AGE
knote    1/1     1            1           39m
mongo    0/1     1            0           58s
mongo    1/1     1            1           73s
```



```
harirj@harirj-Inspiron-3501:~$ kubectl get deployments
NAME    READY   UP-TO-DATE   AVAILABLE   AGE
knote   1/1     1            1           29s
mongo   1/1     1            1           77s
harirj@harirj-Inspiron-3501:~$ kubectl get services
NAME         TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
knote        LoadBalancer   10.105.81.91    <pending>     80:30676/TCP   25s
kubernetes   ClusterIP      10.96.0.1       <none>        443/TCP        2m13s
mongo        ClusterIP      10.110.24.184   <none>        27017/TCP      87s
harirj@harirj-Inspiron-3501:~$
```

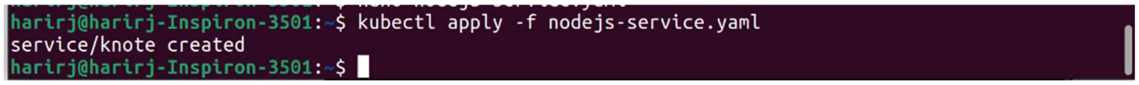
Task 4: Expose the Application Results to the Outside World

Step 1 : Create a Service YAML file (nodejs-service.yaml)



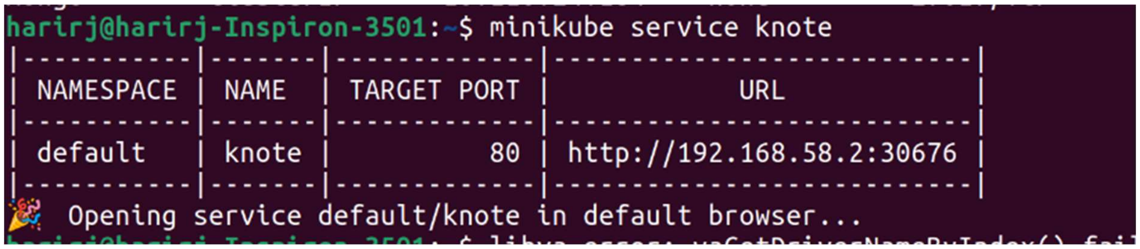
```
  GNU nano 6.2                    nodejs-service.yaml
apiVersion: v1
kind: Service
metadata:
  name: knote
spec:
  selector:
    app: knote
  ports:
    - port: 80
      targetPort: 3000
  type: LoadBalancer
```

Step 2 : Apply the Service

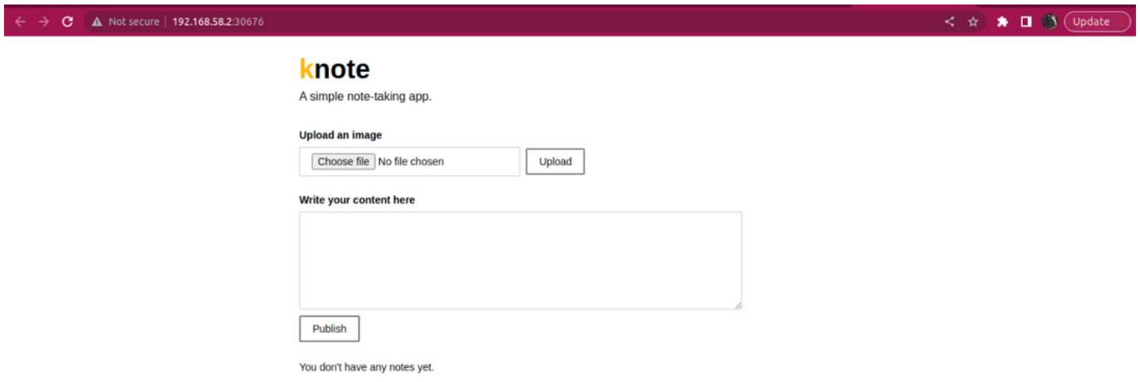


```
harirj@harirj-Inspiron-3501:~$ kubectl apply -f nodejs-service.yaml
service/knote created
harirj@harirj-Inspiron-3501:~$
```

Step 3 : Access the application



```
harirj@harirj-Inspiron-3501:~$ minikube service knote
|-----------|-------|-------------|---------------------------|
| NAMESPACE | NAME  | TARGET PORT |            URL            |
|-----------|-------|-------------|---------------------------|
| default   | knote |          80 | http://192.168.58.2:30676 |
|-----------|-------|-------------|---------------------------|
🎉  Opening service default/knote in default browser...
```



knote

A simple note-taking app.

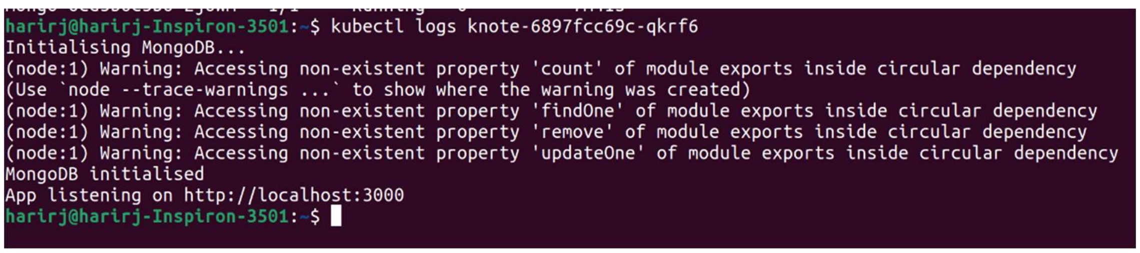
Upload an image

Write your content here

You don't have any notes yet.

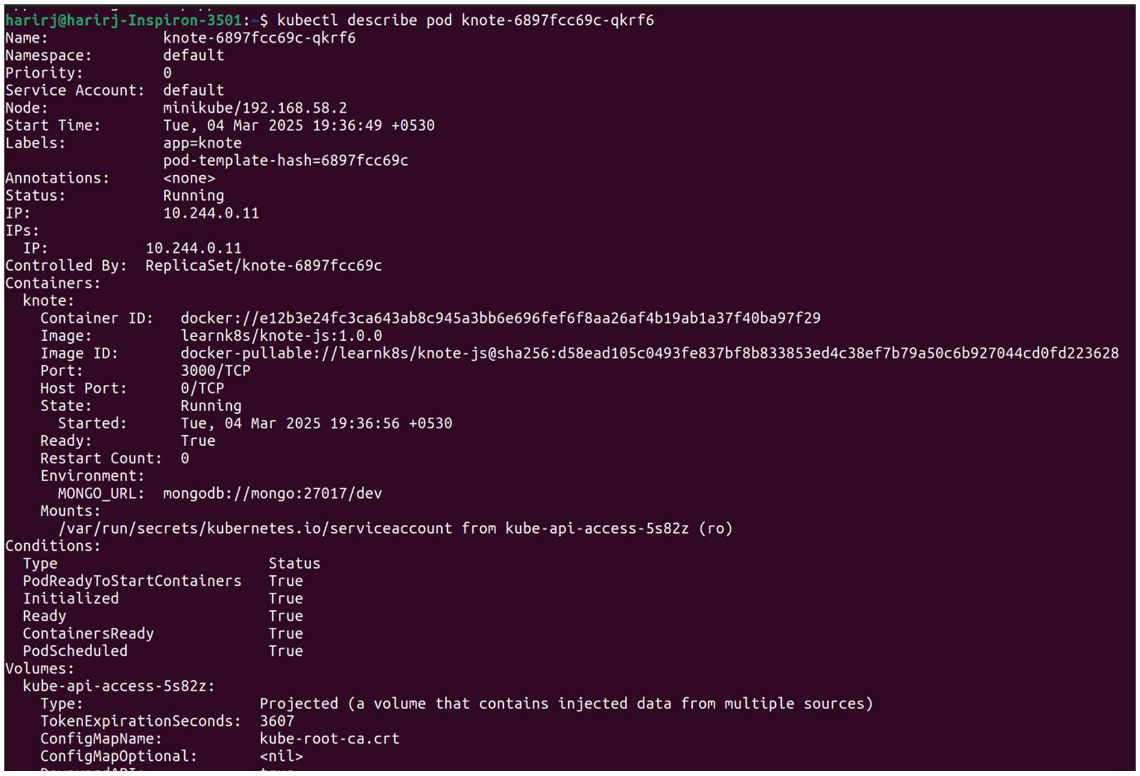
Task 5: Monitoring or Analyzing the Pods

Step 1 : Check logs of a pod

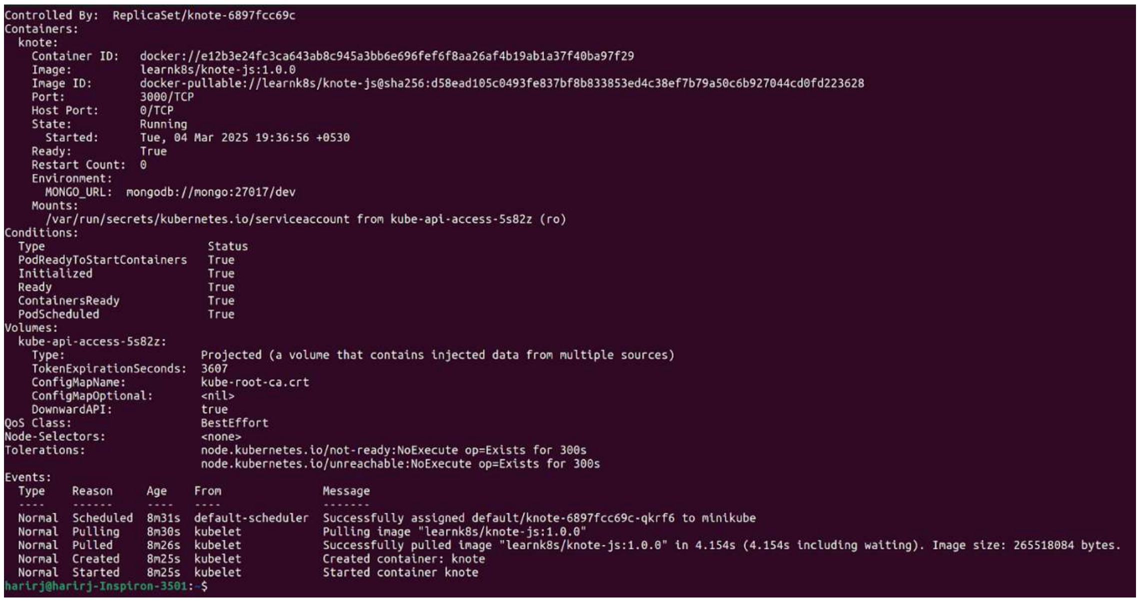


```
harirj@harirj-Inspiron-3501:~$ kubectl logs knote-6897fcc69c-qkrf6
Initialising MongoDB...
(node:1) Warning: Accessing non-existent property 'count' of module exports inside circular dependency
(Use `node --trace-warnings ...` to show where the warning was created)
(node:1) Warning: Accessing non-existent property 'findOne' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'remove' of module exports inside circular dependency
(node:1) Warning: Accessing non-existent property 'updateOne' of module exports inside circular dependency
MongoDB initialised
App listening on http://localhost:3000
harirj@harirj-Inspiron-3501:~$ 
```

Step 2 : Describe a pod for more details

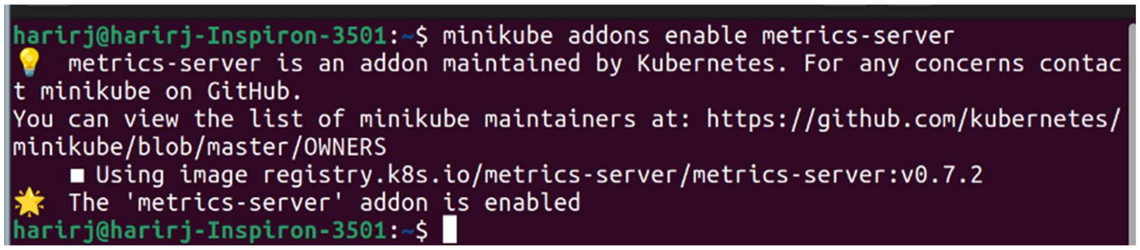


```
harirj@harirj-Inspiron-3501:~$ kubectl describe pod knote-6897fcc69c-qkrf6
Name:             knote-6897fcc69c-qkrf6
Namespace:        default
Priority:         0
Service Account:  default
Node:             minikube/192.168.58.2
Start Time:       Tue, 04 Mar 2025 19:36:49 +0530
Labels:           app=knote
                  pod-template-hash=6897fcc69c
Annotations:      <none>
Status:           Running
IP:               10.244.0.11
IPs:
  IP:           10.244.0.11
Controlled By:  ReplicaSet/knote-6897fcc69c
Containers:
  knote:
    Container ID:   docker://e12b3e24fc3ca643ab8c945a3bb6e696fef6f8aa26af4b19ab1a37f40ba97f29
    Image:          learnk8s/knote-js:1.0.0
    Image ID:       docker-pullable://learnk8s/knote-js@sha256:d58ead105c0493fe837bf8b833853ed4c38ef7b79a50c6b927044cd0fd223628
    Port:           3000/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Tue, 04 Mar 2025 19:36:56 +0530
    Ready:          True
    Restart Count:  0
    Environment:
      MONGO_URL:  mongodb://mongo:27017/dev
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-5s82z (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-5s82z:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
```

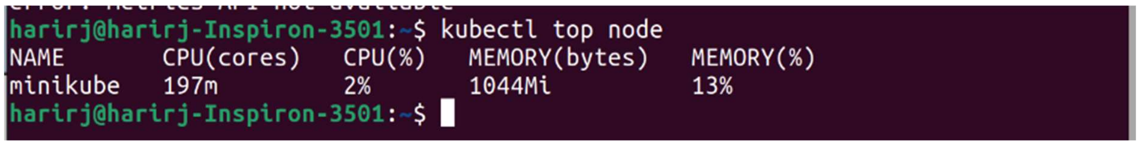


```
Controlled By:  ReplicaSet/knote-6897fcc69c
Containers:
  knote:
    Container ID:   docker://e12b3e24fc3ca643ab8c945a3bb6e696fef6f8aa26af4b19ab1a37f40ba97f29
    Image:          learnk8s/knote-js:1.0.0
    Image ID:       docker-pullable://learnk8s/knote-js@sha256:d58ead105c0493fe837bf8b833853ed4c38ef7b79a50c6b927044cd0fd223628
    Port:           3000/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Tue, 04 Mar 2025 19:36:56 +0530
    Ready:          True
    Restart Count:  0
    Environment:
      MONGO_URL:  mongodb://mongo:27017/dev
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-5s82z (ro)
Conditions:
  Type                        Status
  PodReadyToStartContainers   True
  Initialized                 True
  Ready                       True
  ContainersReady             True
  PodScheduled                True
Volumes:
  kube-api-access-5s82z:
    Type:                    Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:  3607
    ConfigMapName:           kube-root-ca.crt
    ConfigMapOptional:       <nil>
    DownwardAPI:             true
QoS Class:                   BestEffort
Node-Selectors:              <none>
Tolerations:                 node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                             node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type    Reason     Age    From               Message
  ----    ------     ----   ----               -------
  Normal  Scheduled  8m31s  default-scheduler  Successfully assigned default/knote-6897fcc69c-qkrf6 to minikube
  Normal  Pulling    8m30s  kubelet            Pulling image "learnk8s/knote-js:1.0.0"
  Normal  Pulled     8m26s  kubelet            Successfully pulled image "learnk8s/knote-js:1.0.0" in 4.154s (4.154s including waiting). Image size: 265518084 bytes.
  Normal  Created    8m25s  kubelet            Created container: knote
  Normal  Started    8m25s  kubelet            Started container knote
harirj@harirj-Inspiron-3501:~$
```

Step 3 : Enable metric-server addon

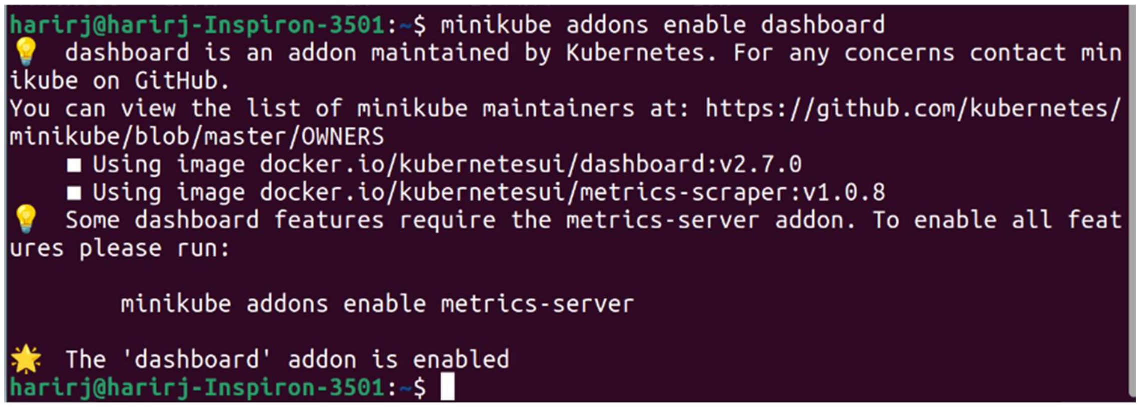


```
harirj@harirj-Inspiron-3501:~$ minikube addons enable metrics-server
💡  metrics-server is an addon maintained by Kubernetes. For any concerns contac
t minikube on GitHub.
You can view the list of minikube maintainers at: https://github.com/kubernetes/
minikube/blob/master/OWNERS
    ▪ Using image registry.k8s.io/metrics-server/metrics-server:v0.7.2
🌟  The 'metrics-server' addon is enabled
harirj@harirj-Inspiron-3501:~$ 
```



```
haririj@haririj-Inspiron-3501:~$ kubectl top node
NAME       CPU(cores)   CPU(%)   MEMORY(bytes)   MEMORY(%)
minikube   197m         2%       1044Mi          13%
haririj@haririj-Inspiron-3501:~$
```

Step 4: Enable dashboard addon

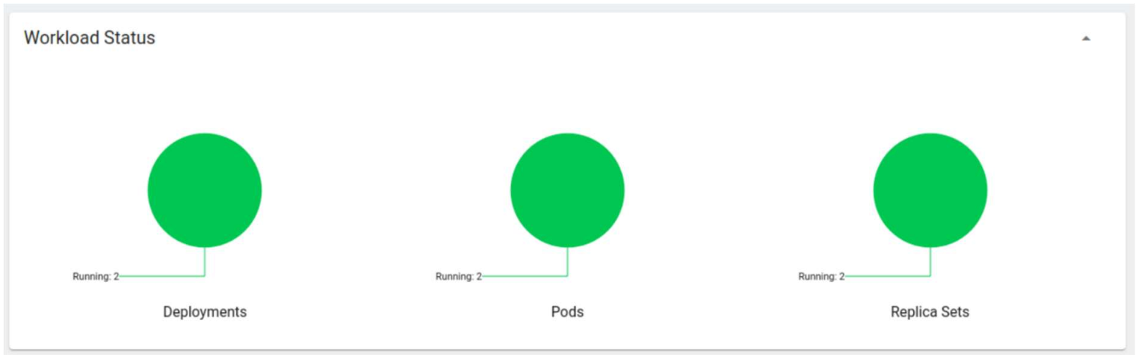


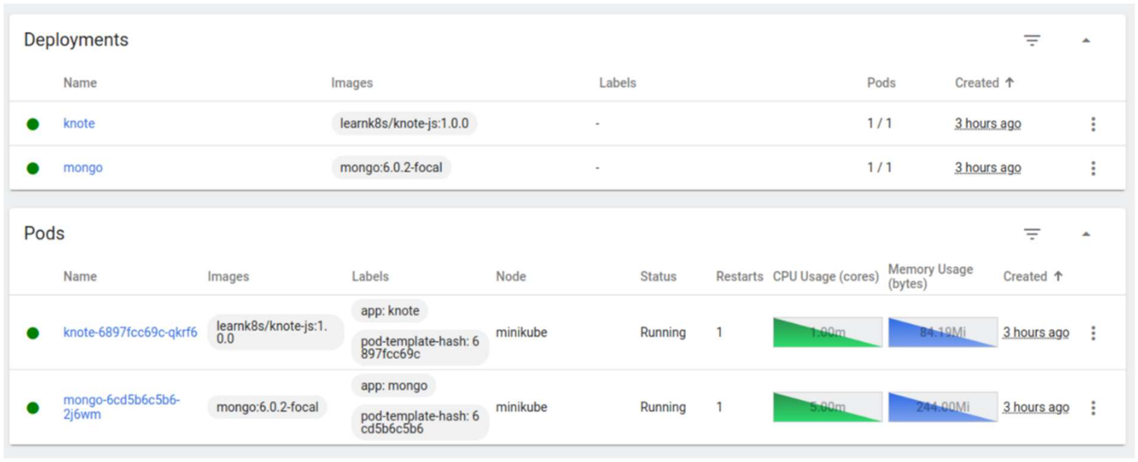
```
haririj@haririj-Inspiron-3501:~$ minikube addons enable dashboard
💡  dashboard is an addon maintained by Kubernetes. For any concerns contact min
ikube on GitHub.
You can view the list of minikube maintainers at: https://github.com/kubernetes/
minikube/blob/master/OWNERS
    ▪ Using image docker.io/kubernetesui/dashboard:v2.7.0
    ▪ Using image docker.io/kubernetesui/metrics-scraper:v1.0.8
💡  Some dashboard features require the metrics-server addon. To enable all feat
ures please run:

        minikube addons enable metrics-server

🌟  The 'dashboard' addon is enabled
haririj@haririj-Inspiron-3501:~$
```

<u>Dashboard</u>



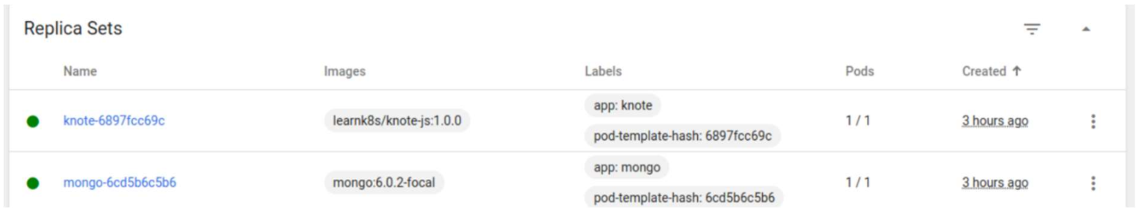


Deployments

	Name	Images	Labels	Pods	Created
●	knote	learnk8s/knote-js:1.0.0	-	1 / 1	3 hours ago
●	mongo	mongo:6.0.2-focal	-	1 / 1	3 hours ago

Pods

	Name	Images	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created
●	knote-6897fcc69c-qkrf6	learnk8s/knote-js:1.0.0	app: knote pod-template-hash: 6897fcc69c	minikube	Running	1	1.00m	84.19Mi	3 hours ago
●	mongo-6cd5b6c5b6-2j6wm	mongo:6.0.2-focal	app: mongo pod-template-hash: 6cd5b6c5b6	minikube	Running	1	5.00m	244.00Mi	3 hours ago

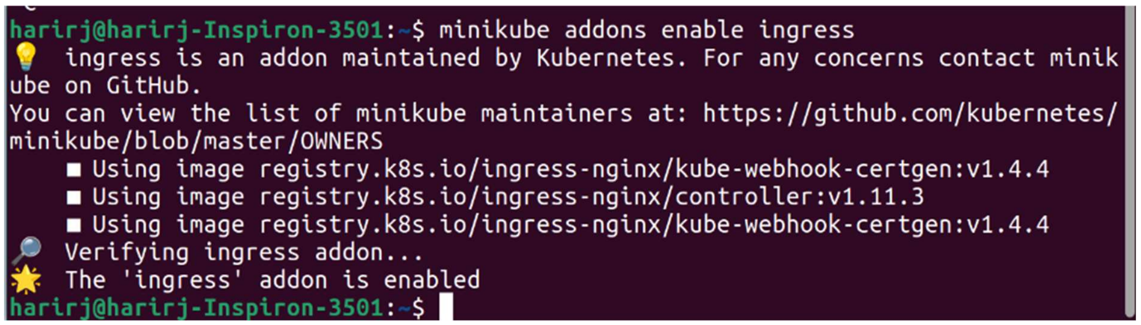


Replica Sets

	Name	Images	Labels	Pods	Created ↑
●	knote-6897fcc69c	learnk8s/knote-js:1.0.0	app: knote pod-template-hash: 6897fcc69c	1 / 1	3 hours ago
●	mongo-6cd5b6c5b6	mongo:6.0.2-focal	app: mongo pod-template-hash: 6cd5b6c5b6	1 / 1	3 hours ago

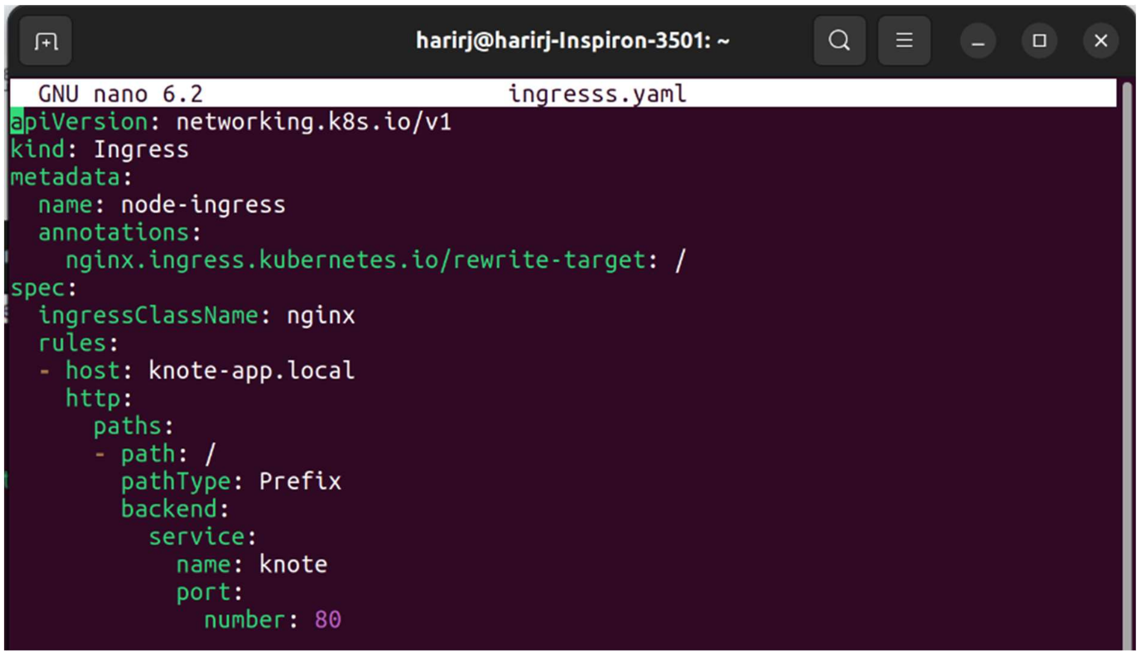
Task 6: Expose application to the external world

Step 1: Enable ingress addon in Minikube



```
harirj@harirj-Inspiron-3501:~$ minikube addons enable ingress
💡  ingress is an addon maintained by Kubernetes. For any concerns contact minik
ube on GitHub.
You can view the list of minikube maintainers at: https://github.com/kubernetes/
minikube/blob/master/OWNERS
    ▪ Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.4.4
    ▪ Using image registry.k8s.io/ingress-nginx/controller:v1.11.3
    ▪ Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.4.4
🔎  Verifying ingress addon...
🌟  The 'ingress' addon is enabled
harirj@harirj-Inspiron-3501:~$
```

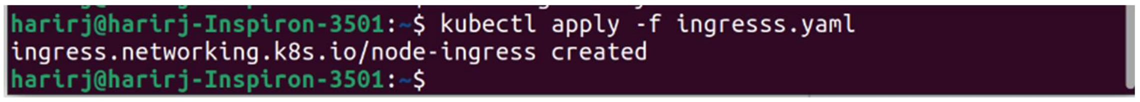
Step 2: configure Ingress YAML File (ingresss.yaml)



harirj@harirj-Inspiron-3501: ~

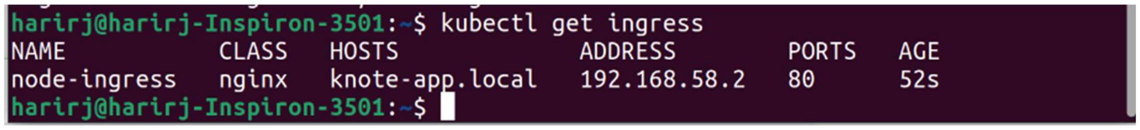
```
  GNU nano 6.2                    ingresss.yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: node-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  ingressClassName: nginx
  rules:
  - host: knote-app.local
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: knote
            port:
              number: 80
```

Step 3 : Apply the ingresss.yaml



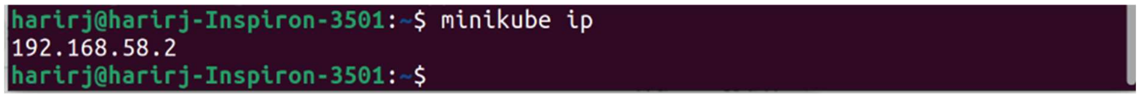
```
harirj@harirj-Inspiron-3501:~$ kubectl apply -f ingresss.yaml
ingress.networking.k8s.io/node-ingress created
harirj@harirj-Inspiron-3501:~$
```

Step 4 : Check if the Ingress is created properly



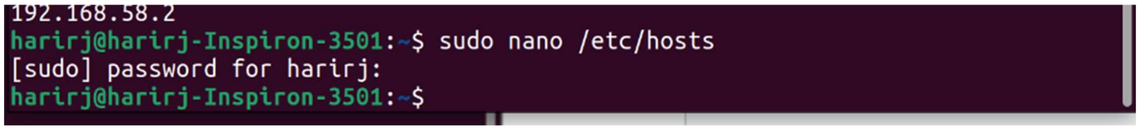
```
harirj@harirj-Inspiron-3501:~$ kubectl get ingress
NAME           CLASS   HOSTS            ADDRESS        PORTS   AGE
node-ingress   nginx   knote-app.local  192.168.58.2   80      52s
harirj@harirj-Inspiron-3501:~$
```

Step 5 : Get Minikube's IP



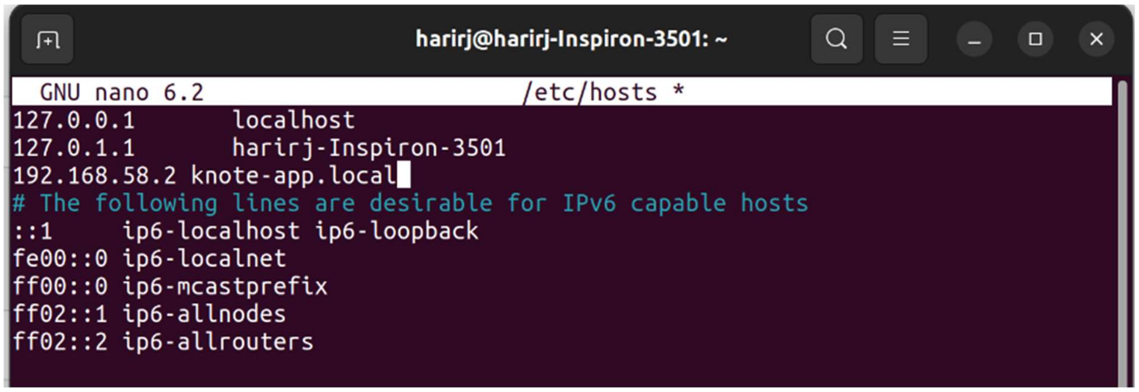
```
harirj@harirj-Inspiron-3501:~$ minikube ip
192.168.58.2
harirj@harirj-Inspiron-3501:~$
```

Step 6 : Update /etc/hosts



```
192.168.58.2
harirj@harirj-Inspiron-3501:~$ sudo nano /etc/hosts
[sudo] password for harirj:
harirj@harirj-Inspiron-3501:~$
```

Add the line 192.168.58.2 knote-app.local

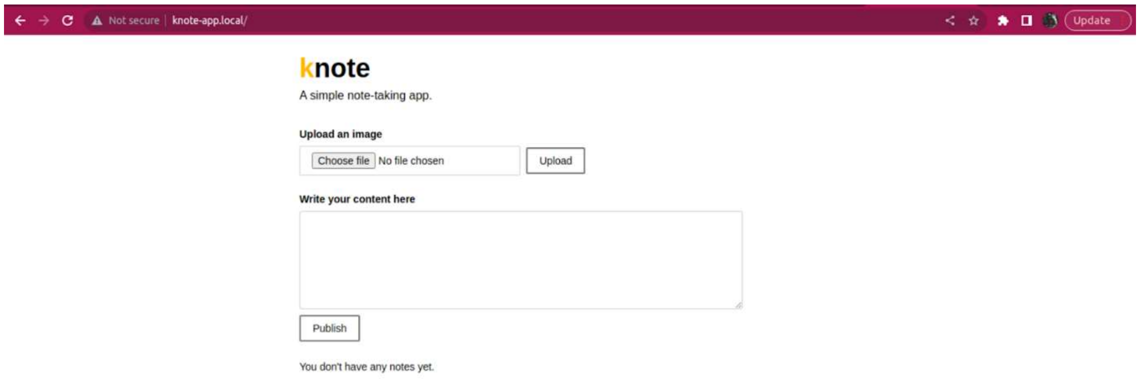


harirj@harirj-Inspiron-3501: ~

```
  GNU nano 6.2                          /etc/hosts *
127.0.0.1       localhost
127.0.1.1       harirj-Inspiron-3501
192.168.58.2 knote-app.local
# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Step 7 : Test the app in a browser

http://knote-app.local



Not secure | knote-app.local/

knote

A simple note-taking app.

Upload an image

Choose file No file chosen Upload

Write your content here

Publish

You don't have any notes yet.