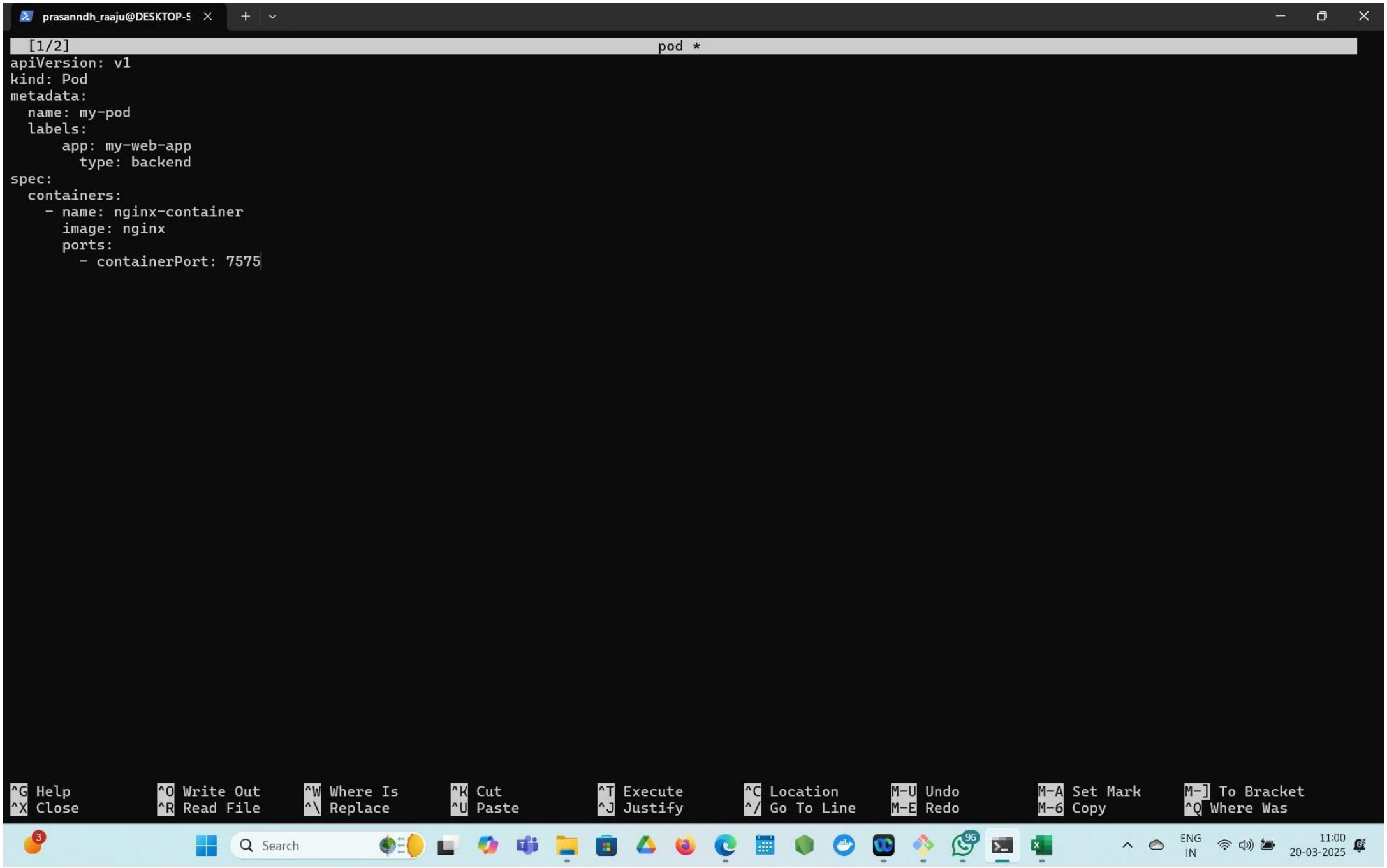
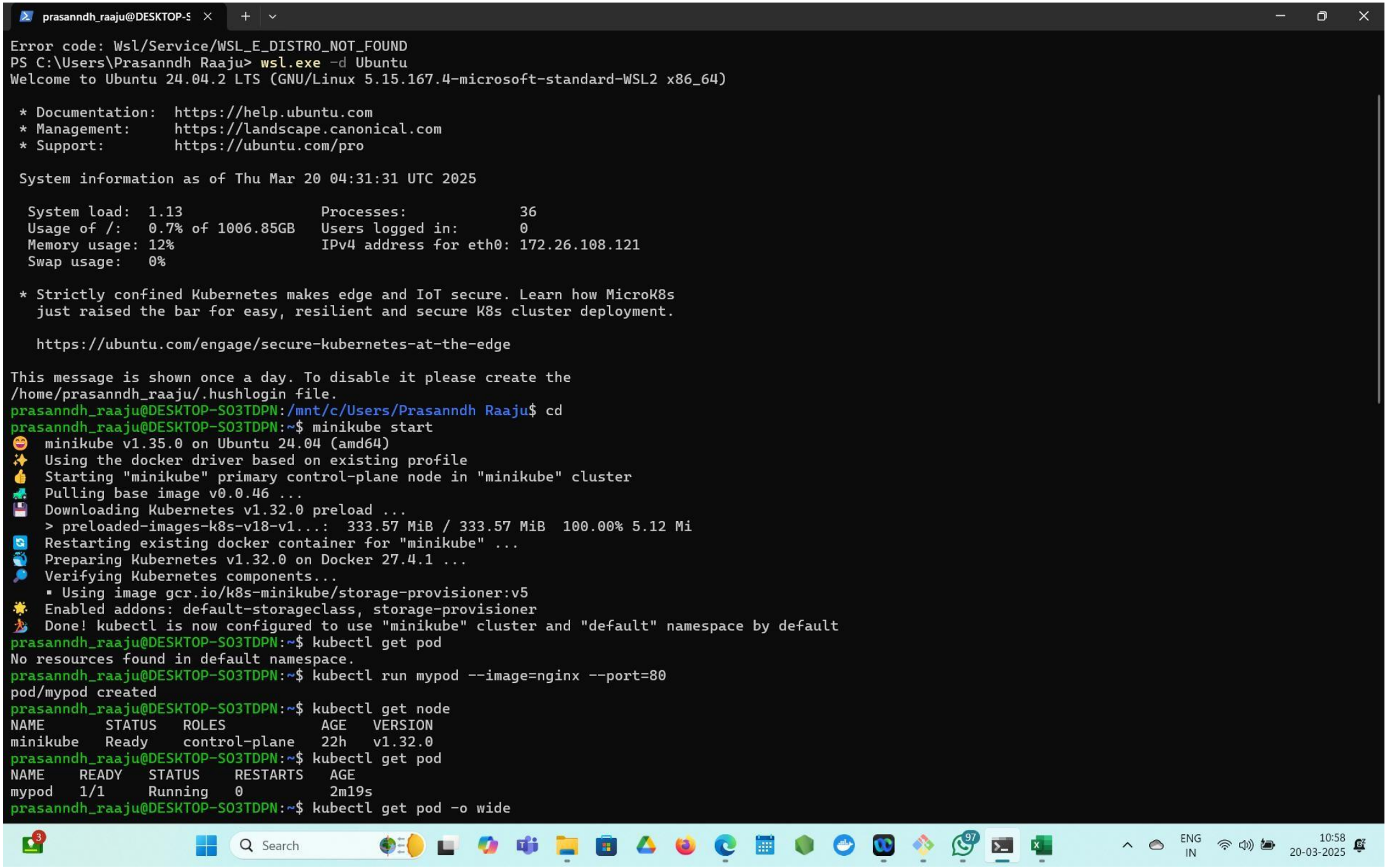
```
root@my-pod: /usr/local
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod yml
[sudo] password for prasanndh_raaju:
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl apply -f pod.yml
error: the path "pod.yml" does not exist
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl apply -f pod.yml
error: the path "pod.yml" does not exist
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl apply -f pod.yml
error: the path "pod.yml" does not exist
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo vi pod yml
2 files to edit
prasanndh_raaju@DESKTOP-SO3TDPN:~$ ls
Jenkinsfile  Jenkinsfile.save  docker-compose.yml  pod  yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod.yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl apply -f pod.yml
error: error parsing pod.yml: error converting YAML to JSON: yaml: line 7: mapping values are not allowed in this context
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod.yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod.yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl apply -f pod.yml
pod/my-pod created
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod
NAME     READY   STATUS              RESTARTS   AGE
my-pod   0/1     ContainerCreating   0          23s
mypod    1/1     Running             0          62m
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod
NAME     READY   STATUS              RESTARTS   AGE
my-pod   0/1     ContainerCreating   0          35s
mypod    1/1     Running             0          62m
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl exec -it my-app -- /bin/bash
Error from server (NotFound): pods "my-app" not found
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl exec -it my-pod -- /bin/bash
root@my-pod:/usr/local/tomcat# l
bin/         conf/           filtered-KEYS  LICENSE  native-jni-lib/  README.md      RUNNING.txt  upstream-KEYS  webapps.dist/
BUILDING.txt  CONTRIBUTING.md  lib/           logs/    NOTICE           RELEASE-NOTES  temp/        webapps/       work/
root@my-pod:/usr/local/tomcat# ls
bin          conf            filtered-KEYS  LICENSE  native-jni-lib  README.md      RUNNING.txt  upstream-KEYS  webapps.dist
BUILDING.txt  CONTRIBUTING.md  lib            logs     NOTICE          RELEASE-NOTES  temp         webapps        work
root@my-pod:/usr/local/tomcat# cd webapps
root@my-pod:/usr/local/tomcat/webapps# ls
maven-web-app  maven-web-app.war
root@my-pod:/usr/local/tomcat/webapps# cd ..
root@my-pod:/usr/local/tomcat# cd ..
root@my-pod:/usr/local#
```

```
coredns-668d6bf9bc-vlmnl           1/1     Running   1 (53m ago)   22h
etcd-minikube                      1/1     Running   1 (53m ago)   22h
kube-apiserver-minikube            1/1     Running   1 (53m ago)   22h
kube-controller-manager-minikube   1/1     Running   1 (53m ago)   22h
kube-proxy-jg2n2                   1/1     Running   1 (53m ago)   22h
kube-scheduler-minikube            1/1     Running   1 (53m ago)   22h
storage-provisioner                1/1     Running   3 (52m ago)   22h
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl logs mypod
kubectl logs mypod: command not found
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod -o wide
NAME    READY   STATUS    RESTARTS   AGE   IP           NODE       NOMINATED NODE   READINESS GATES
mypod   1/1     Running   0          49m   10.244.0.4   minikube   <none>           <none>
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl logs mypod
kubectl logs mypod: command not found
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod yml
[sudo] password for prasanndh_raaju:
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl apply -f pod.yml
error: the path "pod.yml" does not exist
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl apply -f pod.yml
error: the path "pod.yml" does not exist
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl apply -f pod.yml
error: the path "pod.yml" does not exist
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo vi pod yml
2 files to edit
prasanndh_raaju@DESKTOP-SO3TDPN:~$ ls
Jenkinsfile  Jenkinsfile.save  docker-compose.yml  pod  yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod.yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl apply -f pod.yml
error: error parsing pod.yml: error converting YAML to JSON: yaml: line 7: mapping values are not allowed in this context
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod.yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano pod.yml
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl apply -f pod.yml
pod/my-pod created
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod
NAME     READY   STATUS              RESTARTS   AGE
my-pod   0/1     ContainerCreating   0          23s
mypod    1/1     Running             0          62m
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod
NAME     READY   STATUS              RESTARTS   AGE
my-pod   0/1     ContainerCreating   0          35s
mypod    1/1     Running             0          62m
prasanndh_raaju@DESKTOP-SO3TDPN:~$
```


prasanndh_raaju@DESKTOP-S
[1/2] pod *
apiVersion: v1
kind: Pod
metadata:
name: my-pod
labels:
app: my-web-app
type: backend
spec:
containers:
- name: nginx-container
image: nginx
ports:
- containerPort: 7575
^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo M-A Set Mark M-] To Bracket
^X Close ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line M-E Redo M-6 Copy ^Q Where Was
Search
ENG
IN
11:00
20-03-2025


prasanndh_raaju@DESKTOP-S
Error code: Wsl/Service/WSL_E_DISTRO_NOT_FOUND
PS C:\Users\Prasanndh Raaju> wsl.exe -d Ubuntu
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 5.15.167.4-microsoft-standard-WSL2 x86_64)
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/pro
System information as of Thu Mar 20 04:31:31 UTC 2025
System load: 1.13
Usage of /: 0.7% of 1006.85GB
Memory usage: 12%
Swap usage: 0%
Processes: 36
Users logged in: 0
IPv4 address for eth0: 172.26.108.121
* Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
just raised the bar for easy, resilient and secure K8s cluster deployment.
https://ubuntu.com/engage/secure-kubernetes-at-the-edge
This message is shown once a day. To disable it please create the
/home/prasanndh_raaju/.hushlogin file.
prasanndh_raaju@DESKTOP-SO3TDPN:/mnt/c/Users/Prasanndh Raaju$ cd
prasanndh_raaju@DESKTOP-SO3TDPN:~$ minikube start
minikube v1.35.0 on Ubuntu 24.04 (amd64)
Using the docker driver based on existing profile
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% 5.12 Mi
Restarting existing docker container for "minikube" ...
Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
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
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod
No resources found in default namespace.
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl run mypod --image=nginx --port=80
pod/mypod created
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get node
NAME STATUS ROLES AGE VERSION
minikube Ready control-plane 22h v1.32.0
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod
NAME READY STATUS RESTARTS AGE
mypod 1/1 Running 0 2m19s
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod -o wide
Search
ENG
IN
10:58
20-03-2025

```
minikube   Ready    control-plane   22h   v1.32.0
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod
NAME    READY   STATUS    RESTARTS   AGE
mypod   1/1     Running   0          2m19s
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod -o wide
NAME    READY   STATUS    RESTARTS   AGE     IP           NODE       NOMINATED NODE   READINESS GATES
mypod   1/1     Running   0          3m23s   10.244.0.4   minikube   <none>           <none>
prasanndh_raaju@DESKTOP-SO3TDPN:~$ docker images
REPOSITORY                    TAG       IMAGE ID       CREATED        SIZE
prasnndh/simplewebapp         latest    5b4b16fdf6f4   17 hours ago   520MB
<none>                        <none>    a5daec3b5d6b   17 hours ago   520MB
<none>                        <none>    3716e0cc9158   17 hours ago   520MB
<none>                        <none>    9a64a38836ba   18 hours ago   520MB
<none>                        <none>    98613b847b3c   18 hours ago   520MB
tomcat                        9         148e4169be1a   13 days ago    520MB
nginx                         latest    53a18edff809   6 weeks ago    192MB
mysql                         latest    fa262c3a6564   8 weeks ago    797MB
gcr.io/k8s-minikube/kicbase   v0.0.46   e72c4cbe9b29   2 months ago   1.31GB
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pods -n kube-system
NAME                               READY   STATUS    RESTARTS      AGE
coredns-668d6bf9bc-vlmnl           1/1     Running   1 (53m ago)   22h
etcd-minikube                      1/1     Running   1 (53m ago)   22h
kube-apiserver-minikube            1/1     Running   1 (53m ago)   22h
kube-controller-manager-minikube   1/1     Running   1 (53m ago)   22h
kube-proxy-jg2n2                   1/1     Running   1 (53m ago)   22h
kube-scheduler-minikube            1/1     Running   1 (53m ago)   22h
storage-provisioner                1/1     Running   3 (52m ago)   22h
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl logs my-pod
kubectl logs my-pod: command not found
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl logs ndinx
kubectl logs ndinx: command not found
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pods -n kube-system
NAME                               READY   STATUS    RESTARTS      AGE
coredns-668d6bf9bc-vlmnl           1/1     Running   1 (53m ago)   22h
etcd-minikube                      1/1     Running   1 (53m ago)   22h
kube-apiserver-minikube            1/1     Running   1 (53m ago)   22h
kube-controller-manager-minikube   1/1     Running   1 (53m ago)   22h
kube-proxy-jg2n2                   1/1     Running   1 (53m ago)   22h
kube-scheduler-minikube            1/1     Running   1 (53m ago)   22h
storage-provisioner                1/1     Running   3 (52m ago)   22h
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl logs mypod
kubectl logs mypod: command not found
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod -o wide
NAME    READY   STATUS    RESTARTS   AGE   IP           NODE       NOMINATED NODE   READINESS GATES
mypod   1/1     Running   0          49m   10.244.0.4   minikube   <none>           <none>
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl logs mypod
kubectl logs mypod: command not found
```

https://github.com/PrasanndhRaaju-MR/DevOps_War/tree/main

Import favorites | McAfee Security | Dell

PrasanndhRaaju-MR / DevOps_War

Type / to search

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

DevOps_War

Public

Pin | Unwatch 1 | Fork 0 | Star 0

main | 1 Branch | 0 Tags

Go to file t | Add file | Code

PrasanndhRaaju-MR Update Jenkinsfile — c414d34 · 12 minutes ago — 4 Commits

src/main/webapp	first commit	2 days ago
target	first commit	2 days ago
Dockerfile	first commit	2 days ago
Jenkinsfile	Update Jenkinsfile	12 minutes ago
Jenkinsfile.txt	jenkinfile	28 minutes ago
pom.xml	first commit	2 days ago

README

Add a README

Help people interested in this repository understand your project by adding a README.

Add a README

About

No description, website, or topics provided.

Activity
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

Java 59.1% Dockerfile 40.9%

Suggested workflows

Based on your tech stack

Search

ENG IN

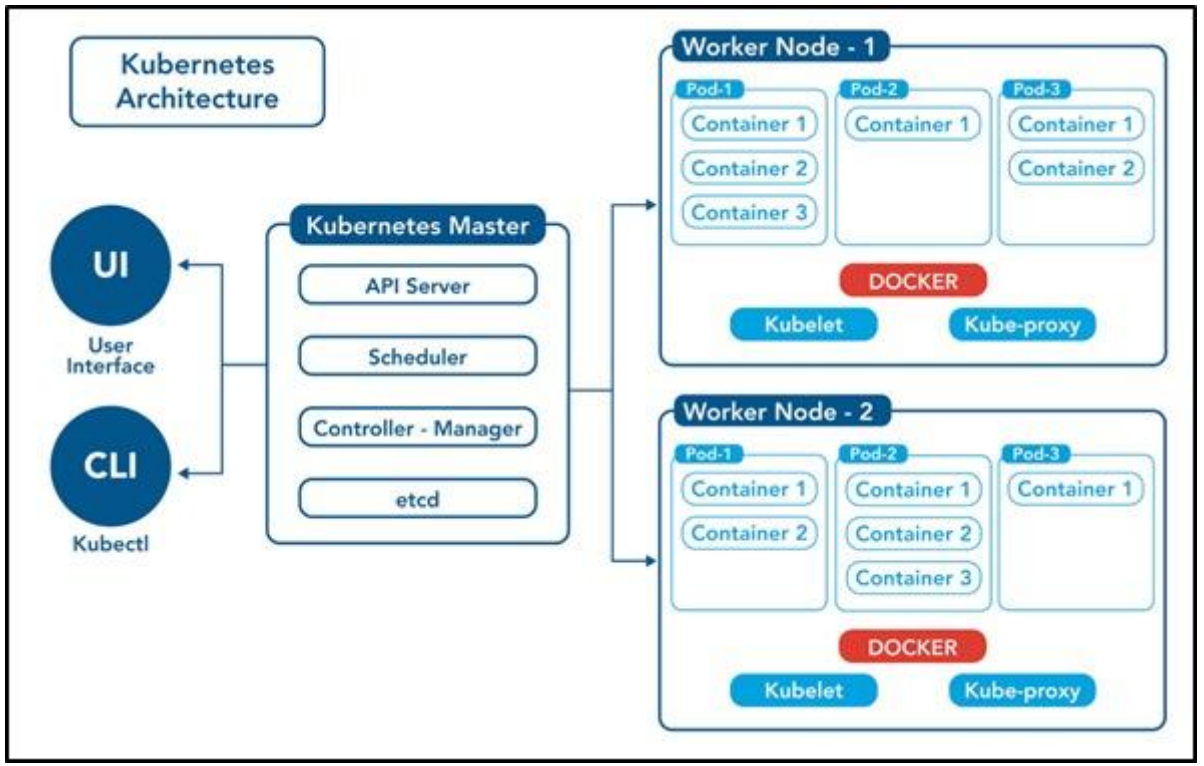
17:01 19-03-2025

Kubernetes (K8s)

Kubernetes is an open source container orchestration engine for automating deployment, scaling, and management of containerized applications. The open source project is hosted by the Cloud Native Computing Foundation (CNCF).
It provides a scalable and resilient framework for automating the deployment, scaling, and management of applications across clusters of servers.

A SMALL HISTORY OF K8S:

- ➔ In the early 2000s, Google started developing a system called Borg to manage their internal containerized applications.
- ➔ Borg enabled Google to run applications at scale, providing features such as automatic scaling, service discovery, and fault tolerance.
- ➔ In 2014, Google open-sourced a version of Borg called Kubernetes.
- ➔ Kubernetes was donated to the Cloud Native Computing Foundation (CNCF), a neutral home for open-source cloud-native projects, in July 2015.
- ➔ Kubernetes 1.8 added significant enhancements for storage, security, and networking. Key features included the stable release of the stateful sets API, expanded support for volume plugins, and improvements in security policies.
- ➔ Check URL: https://kubernetes.io/releases/ for more release details.



Control Plane /Master Node
The control plane's components make global decisions about the cluster (for example, scheduling), as well as detecting and responding to cluster events (for example, starting up a new pod when a deployment's replicas field is unsatisfied).
Control plane components can be run on any machine in the cluster. Do not run user containers on this machine.
Node Components / Worker Nodes
Node components run on every node, maintaining running pods and providing the Kubernetes runtime environment.

1. Master Node: The master node is responsible for managing the cluster and coordinating the overall state of the system. It includes the following components:

a. API Server: The API server is the central control point for all interactions with the cluster. It exposes the Kubernetes API and handles requests from users and other components.

b. Scheduler: The scheduler is responsible for assigning workloads (pods) to individual worker nodes based on resource requirements, constraints, and other policies.

c. Controller Manager: The controller manager runs various controllers that monitor the cluster state and drive it towards the desired state. Examples include the replication controller, node controller, and service controller.

d. etcd: etcd is a distributed key-value store used by Kubernetes to store cluster state and configuration data.

Pod: The basic building block of Kubernetes. A pod represents a single instance of a running process within the cluster. It can encapsulate one or more containers that share the same network and storage resources.

Comments:

1. Create a pod using run command
$ kubectl run <pod-name> --image=<image-name> --port=<container-port>
$ kubectl run my-pod --image=nginx --port=80

2. View all the pods
(In default namespace)
$ kubectl get pods
(In All namespace)

$ kubectl get pods -A
 # For a specific namespace
$ kubectl get pods -n kube-system

 # For a specific type
$ kubectl get pods <pod-name>
$ kubectl get pods <pod-name> -o wide
$ kubectl get pods <pod-name> -o yaml
$ kubectl get pods <pod-name> -o json

3. Describe a pod (View Pod details)
$ kubectl describe pod <pod-name>
$ kubectl describe pod my-pod

4. View Logs of a pod
$ kubectl logs <pod-name>
$ kubectl logs my-pod

5. Execute any command inside Pod (Inside Pod OS)
$ kubectl exec <pod-name> -- <command>

yml file:

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
  labels:
     app: my-web-app
              type: backend
spec:
  containers:
    - name: nginx-container
      image: nginx
      ports:
```

```
      - containerPort: 80

apiVersion: v1
kind: Pod
metadata:
  name: my-app
spec:
  containers:
    - name: my-app-container
      image: <images>      ports:
        - containerPort: 9090
```

ReplicaSet:

```
kind: ReplicaSet
metadata:
  name: my-rs
  labels:
    name: my-rs
spec:
  replicas: 4
  selector:
    matchLabels:
      apptype: web-backend
  template:
    metadata:
      labels:
        apptype: web-backend
    spec:
      containers:
      - name: my-app
        image:
        ports:
          - containerPort: 8080


apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deploy
  labels:
    name: my-deploy
spec:
  replicas: 4
  selector:
    matchLabels:
      apptype: web-backend
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
```

```
    apptype: web-backend
  spec:
    containers:
    - name: my-app
      image:
      ports:
          - containerPort: 7070
```

kubectl create deployment webnginx2 --image=nginx:latest --replicas=1

kubectl scale deploy <deployment-name> --replicas=<desired-replica-count>

Services (short name = svc):
Service is an abstraction that defines a logical set of pods and a policy to access them. Services enable network connectivity and load balancing to the pods that are part of the service, allowing other components within or outside the cluster to interact with the application.

Service Types:
Kubernetes supports different types of services:

1. **NodePort:** Exposes the service on a static port on each selected node's IP. This type makes the service accessible from outside the cluster by the <NodeIP>:<NodePort> combination.

2. **ClusterIP:** Exposes the service on a cluster-internal IP. This type makes the service only reachable within the cluster.

3. **LoadBalancer:** Creates an external load balancer in cloud environments, which routes traffic to the service.

Create Deployment by executing above YAML file
$ kubectl create -f web-deploy.yml
# Do necessary modifications if exist, else create new
$ kubectl create -f web-deploy.yml
# Completely Modify Pod Template
$ kubectl replace –f web-deploy.yml

View Deployments
$ kubectl get deployments
$ kubectl get deploy
$ kubectl get deploy -o wide
$ kubectl get deploy <deployment-name> -o json
$ kubectl get deploy <deployment-name> -o yaml

View Deployment Description
$ kubectl describe deploy <deployment-name>

We can modify generated/updated YAML file
$ kubectl edit deploy <deployment-name>
change replicas: count to any other value then (ESC):wq

# We can modify our YAML file and then execute apply command
$ kubectl apply -f web-deploy.yml

We can Even scale using command also
$ kubectl scale deploy <deployment-name>   --replicas=<desired-replica-count>

Delete Deployment
$ kubectl delete deploy <deployment-name>
$ kubectl delete -f web-deploy.yml

Create ReplicaSet by executing above YAML file
$ kubectl create -f rs-test.yml
# Do necessary modifications if exist, else create new
$ kubectl apply -f rs-test.yml
# Completely Modify Pod Template
$ kubectl replace –f rs-test.yml

View ReplicaSets
$ kubectl get replicasets
$ kubectl get rs
$ kubectl get rs –o wide
$ kubectl get rs <replica-set-name> –o json
$ kubectl get rs <replica-set-name> –o yaml

View ReplicaSet Description
$ kubectl describe rs <replica-set-name>

We can modify generated/updated YAML file
$ kubectl edit rs <replica-set-name>
change replicas: count to any other value then (ESC):wq

# We can modify our YAML file and then execute apply command
$ kubectl apply -f rs-test.yml

We can Even scale using command also
$ kubectl scale replicaset <replicaset-name> --replicas=<desired-replica-count>

Delete ReplicaSet
$ kubectl delete rs <replica-set-name>
$ kubectl delete -f rs-test.yml

Service Creation: yml file

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deploy
  labels:
    name: my-deploy
spec:
  replicas: 1
  selector:
    matchLabels:
      apptype: web-backend
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
```

```
        apptype: web-backend
    spec:
      containers:
      - name: my-app
        image:
        ports:
        - containerPort: 7070
---

apiVersion: v1
kind: Service
metadata:
  name: my-service
  labels:
    app: my-service
    type: backend-app
spec:
  type: NodePort
  ports:
   - targetPort: 7070
     port: 7070
     nodePort: 30002
  selector:
    apptype: web-backend
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deploy
  labels:
    name: my-deploy
spec:
  replicas: 1
  selector:
    matchLabels:
      apptype: web-backend
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        apptype: web-backend
    spec:
      containers:
      - name: my-app
        image:
        ports:
        - containerPort: 9000

---

apiVersion: v1
```

```
kind: Service
metadata:
  name: my-service
  labels:
    app: my-service
spec:
  type: NodePort
  ports:
    - port: 9000
      targetPort: 8080
      nodePort: 30002
  selector:
    apptype: web-backend
```

Namespace (short name = ns):
namespace is a virtual cluster or logical partition within a cluster that provides a way to organize and isolate resources. It allows multiple teams or projects to share the same physical cluster while maintaining resource separation and access control.

```
# To create a namespace:
$ kubectl create namespace <namespace-name>
$ kubectl create ns my-bank
# To switch to a specific namespace: (make this as default type)
$ kubectl config set-context --current --namespace=<namespace-name>
# To list all namespaces:
$ kubectl get namespaces
# To get resources within a specific namespace:
$ kubectl get <resource-type> -n <namespace-name>
$ kubectl get deploy -n my-bank
$ kubectl get deploy --namespace my-bank
$ kubectl get all --namespace my-bank
# To delete a namespace and all associated resources:
$ kubectl delete namespace <namespace-name>
$ kubectl delete ns my-bank
```

```
kubectl create ns mydeploy

kubectl apply -f deploy.yml -n mydeploy

apiVersion: v1
kind: Namespace
metadata:
  name: my-demo-ns


apiVersion: v1
kind: Pod
```

```
metadata:
  name: my-pod
  namespace: my-demo-ns
spec:
  containers:
  - name: my-container
    image: nginx:latest
```

```
32s\": dial tcp 127.0.0.1:32769: connect: connection refused"
The connection to the server 127.0.0.1:32769 was refused - did you specify the right host or port?
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod
E0321 04:54:21.878874    1622 memcache.go:265] "Unhandled Error" err="couldn't get current server API group list: Get \"https://127.0.0.1:32769/api?timeout=
32s\": dial tcp 127.0.0.1:32769: connect: connection refused"
E0321 04:54:21.881465    1622 memcache.go:265] "Unhandled Error" err="couldn't get current server API group list: Get \"https://127.0.0.1:32769/api?timeout=
32s\": dial tcp 127.0.0.1:32769: connect: connection refused"
E0321 04:54:21.883728    1622 memcache.go:265] "Unhandled Error" err="couldn't get current server API group list: Get \"https://127.0.0.1:32769/api?timeout=
32s\": dial tcp 127.0.0.1:32769: connect: connection refused"
E0321 04:54:21.885890    1622 memcache.go:265] "Unhandled Error" err="couldn't get current server API group list: Get \"https://127.0.0.1:32769/api?timeout=
32s\": dial tcp 127.0.0.1:32769: connect: connection refused"
E0321 04:54:21.887491    1622 memcache.go:265] "Unhandled Error" err="couldn't get current server API group list: Get \"https://127.0.0.1:32769/api?timeout=
32s\": dial tcp 127.0.0.1:32769: connect: connection refused"
The connection to the server 127.0.0.1:32769 was refused - did you specify the right host or port?
prasanndh_raaju@DESKTOP-SO3TDPN:~$ minikube start
😄  minikube v1.35.0 on Ubuntu 24.04 (amd64)
✨  Using the docker driver based on existing profile
👍  Starting "minikube" primary control-plane node in "minikube" cluster
🚜  Pulling base image v0.0.46 ...
🔄  Restarting existing docker container for "minikube" ...
🐳  Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
🔎  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
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod
No resources found in default namespace.
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano deploy.yml
[sudo] password for prasanndh_raaju:
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl apply -f deploy.yml
deployment.apps/my-deploy created
service/my-service created
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod
NAME                         READY   STATUS    RESTARTS   AGE
my-deploy-5fd77cc989-z2qkv   1/1     Running   0          8s
prasanndh_raaju@DESKTOP-SO3TDPN:~$ minikube service my-service
|-----------|------------|-------------|---------------------------|
| NAMESPACE |    NAME    | TARGET PORT |            URL            |
|-----------|------------|-------------|---------------------------|
| default   | my-service |        9000 | http://192.168.49.2:30002 |
|-----------|------------|-------------|---------------------------|
🏃  Starting tunnel for service my-service.
|-----------|------------|-------------|------------------------|
| NAMESPACE |    NAME    | TARGET PORT |          URL           |
|-----------|------------|-------------|------------------------|
| default   | my-service |             | http://127.0.0.1:39637 |
|-----------|------------|-------------|------------------------|
```

```
E0321 04:54:21.887491    1622 memcache.go:265] "Unhandled Error" err="couldn't get current server API group list: Get \"https://127.0.0.1:32769/api?timeout=
32s\": dial tcp 127.0.0.1:32769: connect: connection refused"
The connection to the server 127.0.0.1:32769 was refused - did you specify the right host or port?
prasanndh_raaju@DESKTOP-SO3TDPN:~$ minikube start
😄  minikube v1.35.0 on Ubuntu 24.04 (amd64)
✨  Using the docker driver based on existing profile
👍  Starting "minikube" primary control-plane node in "minikube" cluster
🚜  Pulling base image v0.0.46 ...
🔄  Restarting existing docker container for "minikube" ...
🐳  Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
🔎  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
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod
No resources found in default namespace.
prasanndh_raaju@DESKTOP-SO3TDPN:~$ sudo nano deploy.yml
[sudo] password for prasanndh_raaju:
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl apply -f deploy.yml
deployment.apps/my-deploy created
service/my-service created
prasanndh_raaju@DESKTOP-SO3TDPN:~$ kubectl get pod
NAME                         READY   STATUS    RESTARTS   AGE
my-deploy-5fd77cc989-z2qkv   1/1     Running   0          8s
prasanndh_raaju@DESKTOP-SO3TDPN:~$ minikube service my-service
|-----------|------------|-------------|---------------------------|
| NAMESPACE |    NAME    | TARGET PORT |            URL            |
|-----------|------------|-------------|---------------------------|
| default   | my-service |        9000 | http://192.168.49.2:30002 |
|-----------|------------|-------------|---------------------------|
🏃  Starting tunnel for service my-service.
|-----------|------------|-------------|------------------------|
| NAMESPACE |    NAME    | TARGET PORT |          URL           |
|-----------|------------|-------------|------------------------|
| default   | my-service |             | http://127.0.0.1:39637 |
|-----------|------------|-------------|------------------------|
🎉  Opening service default/my-service in default browser...
👉  http://127.0.0.1:39637
❗  Because you are using a Docker driver on linux, the terminal needs to be open to run it.
^C✋  Stopping tunnel for service my-service.
prasanndh_raaju@DESKTOP-SO3TDPN:~$ curl  http://192.168.49.2:30002/maven-web-app/
<html>
<body>
<h2>Hello World!</h2>
</body>
</html>
prasanndh_raaju@DESKTOP-SO3TDPN:~$
```