```
deepa@ubuntu:~/devops/kind$ ls
go1.25.3.linux-amd64.tar.gz
```

```
deepa@ubuntu:~/devops/kind$ sudo tar -C /usr/local -xzf go1.25.3.linux-amd64.tar.gz
deepa@ubuntu:~/devops/kind$ ls
go1.25.3.linux-amd64.tar.gz
deepa@ubuntu:~/devops/kind$ sudo tar -C /usr/local -xzf go1.25.3.linux-amd64.tar.gz
deepa@ubuntu:~/devops/kind$ ls
go1.25.3.linux-amd64.tar.gz
deepa@ubuntu:~/devops/kind$ ls /usr/local/go
api  bin  codereview.cfg  CONTRIBUTING.md  doc  go.env  lib  LICENSE  misc  PATENTS  pkg  README.md  SECURITY.md  src  test  VERSION
```

```
deepa@ubuntu:~/devops/kind$ vi ~/.bashrc
```

```
deepa@ubuntu:~/devops/kind$ ls /usr/local/go/bin
go  gofmt
deepa@ubuntu:~/devops/kind$ source ~/.bashrc
deepa@ubuntu:~/devops/kind$ go version
go version go1.25.3 linux/amd64
deepa@ubuntu:~/devops/kind$ go install sigs.k8s.io/kind@latest
go: downloading sigs.k8s.io/kind v0.30.0
go: downloading github.com/spf13/pflag v1.0.5
go: downloading al.essio.dev/pkg/shellescape v1.5.1
go: downloading github.com/spf13/cobra v1.8.0
go: downloading github.com/pkg/errors v0.9.1
go: downloading github.com/mattn/go-isatty v0.0.20
go: downloading github.com/pelletier/go-toml v1.9.5
go: downloading github.com/BurntSushi/toml v1.4.0
go: downloading github.com/evanphx/json-patch/v5 v5.6.0
go: downloading go.yaml.in/yaml/v3 v3.0.4
go: downloading sigs.k8s.io/yaml v1.4.0
go: downloading golang.org/x/sys v0.6.0
deepa@ubuntu:~/devops/kind$ kind version
kind v0.30.0 go1.25.3 linux/amd64
```

```
deepa@ubuntu:~/devops/kind$ cat kind-config.yaml
kind: Cluster
apiVersion: kind.x-k8s.io/v1alpha4
nodes:
  - role: control-plane
  - role: worker
  - role: worker

deepa@ubuntu:~/devops/kind$
```

```
deepa@ubuntu:~/devops/kind$ kind create cluster --name dev-cluster --config kind-config.yaml
Creating cluster "dev-cluster" ...
 ✓ Ensuring node image (kindest/node:v1.34.0)
 ✓ Preparing nodes
 ✓ Writing configuration
 ✓ Starting control-plane
 ✓ Installing CNI
 ✓ Installing StorageClass
 ✓ Joining worker nodes
Set kubectl context to "kind-dev-cluster"
You can now use your cluster with:

kubectl cluster-info --context kind-dev-cluster

Have a question, bug, or feature request? Let us know! https://kind.sigs.k8s.io/#community
deepa@ubuntu:~/devops/kind$ kubectl config get-contexts
CURRENT   NAME              CLUSTER           AUTHINFO          NAMESPACE
*         kind-dev-cluster  kind-dev-cluster  kind-dev-cluster
          minikube          minikube          minikube          default
```

```
deepa@ubuntu:~/devops/kind$ kubectl create deployment hello-world --image=nginx --replicas=2
deployment.apps/hello-world created
deepa@ubuntu:~/devops/kind$ kubectl create deployment hello-world --image=nginx --replicas=2 --dry-run=client
deployment.apps/hello-world created (dry run)
```

```
deepa@ubuntu:~/devops/kind$ kubectl create deployment hello-world --image=nginx --replicas=2 --dry-run=client -o yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: hello-world
  name: hello-world
spec:
  replicas: 2
  selector:
    matchLabels:
      app: hello-world
  strategy: {}
  template:
    metadata:
      labels:
        app: hello-world
    spec:
      containers:
      - image: nginx
        name: nginx
        resources: {}
status: {}
```

```
deepa@ubuntu:~/devops/kind$ kubectl create deployment hello-world --image=nginx --replicas=2 --dry-run=client -o yaml > dep.yaml
deepa@ubuntu:~/devops/kind$ ls
dep.yaml  go1.25.3.linux-amd64.tar.gz  kind-config.yaml
```

Now deployment with replicas=2 is created.

Then expose it:

```
deepa@ubuntu:~/devops/kind$ kubectl expose deployment hello-world port=80 --type=NodePort
couldn't find port via --port flag or introspection
Error from server (NotFound): deployments.apps "port=80" not found
deepa@ubuntu:~/devops/kind$ kubectl expose deployment hello-world --port=80 --type=NodePort
service/hello-world exposed
deepa@ubuntu:~/devops/kind$
```

```
deepa@ubuntu:~/devops/kind$ kubectl get svc
NAME          TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
hello-world   NodePort    10.96.240.220   <none>        80:32179/TCP   45s
kubernetes    ClusterIP   10.96.0.1       <none>        443/TCP        16m
deepa@ubuntu:~/devops/kind$ kubectl get svc -o wide
NAME          TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE   SELECTOR
hello-world   NodePort    10.96.240.220   <none>        80:32179/TCP   73s   app=hello-world
kubernetes    ClusterIP   10.96.0.1       <none>        443/TCP        17m   <none>
```

```
deepa@ubuntu:~/devops/kind$ kubectl get nodes -o wide
NAME                          STATUS   ROLES           AGE   VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE                         KERNEL-VERSION       CONTAINER-RUNTIME
dev-cluster-control-plane     Ready    control-plane   18m   v1.34.0   172.21.0.2    <none>        Debian GNU/Linux 12 (bookworm)   5.15.0-139-generic   containerd://2.1.3
dev-cluster-worker            Ready    <none>          17m   v1.34.0   172.21.0.3    <none>        Debian GNU/Linux 12 (bookworm)   5.15.0-139-generic   containerd://2.1.3
dev-cluster-worker2           Ready    <none>          17m   v1.34.0   172.21.0.4    <none>        Debian GNU/Linux 12 (bookworm)   5.15.0-139-generic   containerd://2.1.3
deepa@ubuntu:~/devops/kind$ kubectl get nodes
NAME                          STATUS   ROLES           AGE   VERSION
dev-cluster-control-plane     Ready    control-plane   19m   v1.34.0
dev-cluster-worker            Ready    <none>          19m   v1.34.0
dev-cluster-worker2           Ready    <none>          19m   v1.34.0
deepa@ubuntu:~/devops/kind$
```

```
deepa@ubuntu:~/devops/kind$ curl http://172.21.0.2:32179
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

```
deepa@ubuntu:~/devops/kind$ wget  http://172.21.0.2:32179
--2025-10-31 00:06:42--  http://172.21.0.2:32179/
Connecting to 172.21.0.2:32179... connected.
HTTP request sent, awaiting response... 200 OK
Length: 615 [text/html]
Saving to: 'index.html'

index.html                          100%[=======================================================================================>]     615  --.-KB/s

2025-10-31 00:06:42 (48.9 MB/s) - 'index.html' saved [615/615]

deepa@ubuntu:~/devops/kind$
```

Best Ways to Access Services in Kind

🧿 Option 1: Use Docker IP (what you did)

bash

curl http://172.21.0.2:32179

Option 2: Port Forwarding (to use localhost)

```
deepa@ubuntu:~/devops/kind$ kubectl port-forward svc/hello-world 8080:80
Unable to listen on port 8080: Listeners failed to create with the following errors: [unable to create listener: Error listen tcp4 127.0.0.1:8080: bind: add
ress already in use unable to create listener: Error listen tcp6 [::1]:8080: bind: address already in use]
error: unable to listen on any of the requested ports: [{8080 80}]
```

8080 is occupied

Find the PID:

Kill the process

```
deepa@ubuntu:~/devops/kind$ sudo lsof -i :8080
[sudo] password for deepa:
COMMAND  PID    USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
java    3102 jenkins    8u  IPv6  62174      0t0  TCP *:http-alt (LISTEN)
deepa@ubuntu:~/devops/kind$ sudo kill -9 3102
```

```
deepa@ubuntu:~/devops/kind$ kubectl port-forward svc/hello-world 8080:80
Forwarding from 127.0.0.1:8080 → 80
Forwarding from [::1]:8080 → 80
Handling connection for 8080
```

Open a new terminal

```
deepa@ubuntu:~$ cd devops/kind
deepa@ubuntu:~/devops/kind$ curl http://localhost:8080
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
deepa@ubuntu:~/devops/kind$
```

Updated dep.yaml file:

Section	**Purpose**
selector.matchLabels	Tells the Deployment which Pods to manage
template.metadata.labels	Applies labels to the Pods it creates

Present:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: hello-world
  name: hello-world
spec:
  replicas: 2
  selector:
    matchLabels:
      app: hello-world
  strategy: {}
  template:
    metadata:
      labels:
        app: hello-world
    spec:
      containers:
      - image: nginx
        name: nginx
        resources: {}
status: {}
~
```

When port exposed:

```
deepa@ubuntu:~/devops/kind$ cat dep.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: hello-world
  name: hello-world
spec:
  replicas: 2
  selector:
    matchLabels:
      app: hello-world
  template:
    metadata:
      labels:
        app: hello-world
    spec:
      containers:
      - image: nginx
        name: nginx
        ports:
        -  containerPort: 80
        resources: {}
status: {}
deepa@ubuntu:~/devops/kind$
```

```
deepa@ubuntu:~/devops/kind$ kubectl apply -f dep.yaml
Warning: resource deployments/hello-world is missing the kubectl.kubernetes.io/last-applied-configuration annotation which is required by kubectl apply. kubectl apply should only be used on resources created declaratively by either kubectl create --save-config or kubectl apply. The missing annotation will be patched automatically.
deployment.apps/hello-world configured
```

```
deepa@ubuntu:~/devops/kind$ kubectl get deployments
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
hello-world   2/2     2            2           60m
deepa@ubuntu:~/devops/kind$ kubectl describe deployment hello-world
Name:                   hello-world
Namespace:              default
CreationTimestamp:      Thu, 30 Oct 2025 23:45:59 -0700
Labels:                 app=hello-world
Annotations:            deployment.kubernetes.io/revision: 2
Selector:               app=hello-world
Replicas:               2 desired | 2 updated | 2 total | 2 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=hello-world
  Containers:
   nginx:
    Image:         nginx
    Port:          80/TCP
    Host Port:     0/TCP
    Environment:   <none>
    Mounts:        <none>
  Volumes:         <none>
  Node-Selectors:  <none>
  Tolerations:     <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets:  hello-world-66b477f4d9 (0/0 replicas created)
NewReplicaSet:   hello-world-7f6f4c8b6 (2/2 replicas created)
Events:
  Type    Reason             Age    From                   Message
  ----    ------             ----   ----                   -------
  Normal  ScalingReplicaSet  3m51s  deployment-controller  Scaled up replica set hello-world-7f6f4c8b6 from 0 to 1
  Normal  ScalingReplicaSet  3m46s  deployment-controller  Scaled down replica set hello-world-66b477f4d9 from 2 to 1
  Normal  ScalingReplicaSet  3m46s  deployment-controller  Scaled up replica set hello-world-7f6f4c8b6 from 1 to 2
  Normal  ScalingReplicaSet  3m40s  deployment-controller  Scaled down replica set hello-world-66b477f4d9 from 1 to 0
deepa@ubuntu:~/devops/kind$
```

What RollingUpdate Does

When you update a Deployment (e.g., change the image version), Kubernetes:

- **Creates new Pods** with the updated spec
- **Gradually replaces old Pods** with new ones
- Ensures **some Pods stay available** during the update

⚙️ **Key Parameters You Can Customize**

You can fine-tune how the rolling update behaves:

yaml

strategy:

type: RollingUpdate

rollingUpdate:

maxUnavailable: 1

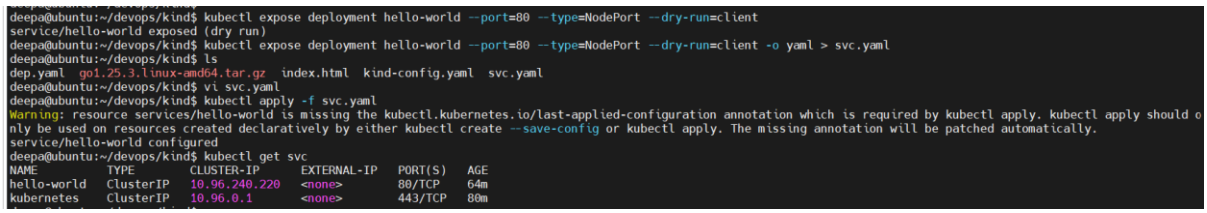
maxSurge: 1

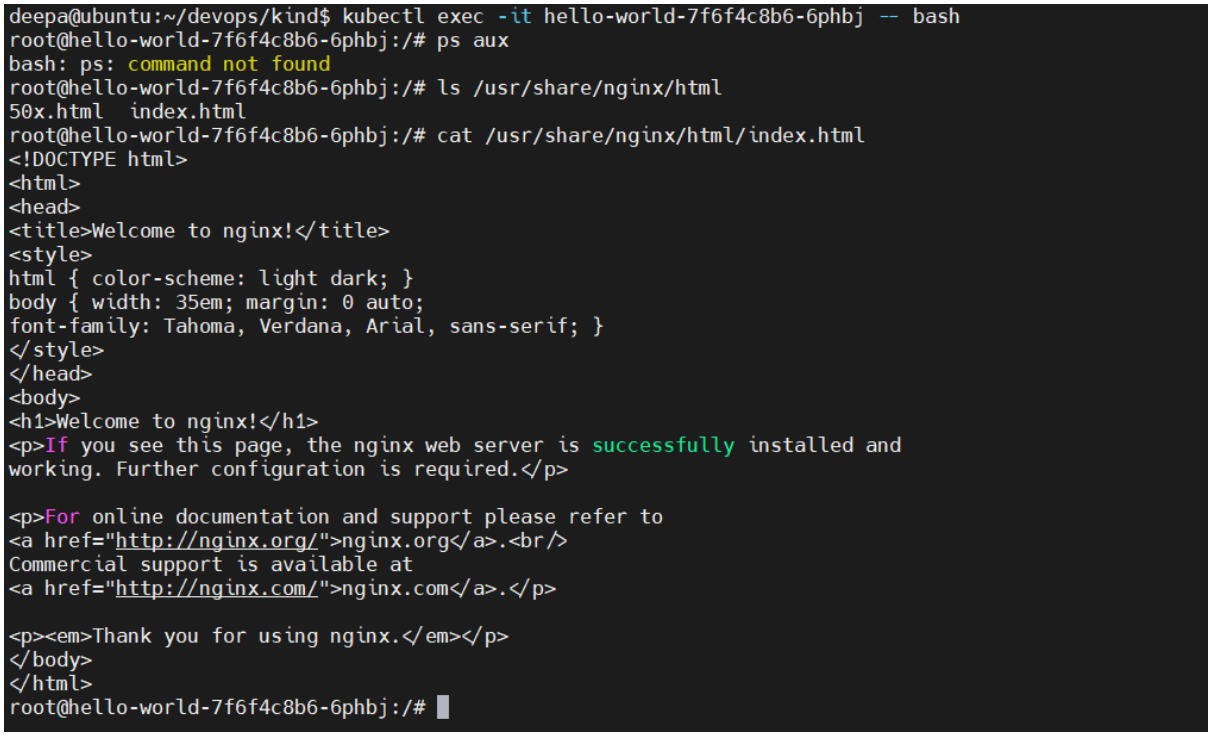
- maxUnavailable: How many Pods can be **unavailable** during the update (e.g., 1 means at least 1 Pod stays up)

- maxSurge: How many **extra Pods** can be created temporarily during the update

🧠 **Why It's Useful**

- Ensures **zero downtime** during updates
- Ideal for **stateless apps** like web servers
- Works well with **readiness probes** to avoid routing traffic to unhealthy Pods





To check internal cluster connectivity:

```
root@hello-world-7f6f4c8b6-6phbj:/# apt update && apt install -y curl
Get:1 http://deb.debian.org/debian trixie InRelease [140 kB]
Get:2 http://deb.debian.org/debian trixie-updates InRelease [47.3 kB]
Get:3 http://deb.debian.org/debian-security trixie-security InRelease [43.4 kB]
Get:4 http://deb.debian.org/debian trixie/main amd64 Packages [9669 kB]
Get:5 http://deb.debian.org/debian trixie-updates/main amd64 Packages [5412 B]
Get:6 http://deb.debian.org/debian-security trixie-security/main amd64 Packages [69.0 kB]
Fetched 9974 kB in 27s (373 kB/s)
All packages are up to date.
curl is already the newest version (8.14.1-2).
Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 0
root@hello-world-7f6f4c8b6-6phbj:/#
```

```
root@hello-world-7f6f4c8b6-6phbj:/# curl http://hello-clusterip
curl: (6) Could not resolve host: hello-clusterip
root@hello-world-7f6f4c8b6-6phbj:/# curl http://hello-world
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@hello-world-7f6f4c8b6-6phbj:/#
```

```
apiVersion: v1
kind: Service
metadata:
  labels:
    app: hello-world
  name: hello-world
spec:
  ports:
  - port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: hello-world
  type: NodePort
status:
```

Yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  labels:
    app: hello-world
  name: hello-world
spec:
  ports:
  - port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: hello-world
  type: NodePort
status:
  loadBalancer: {}
```

```
apiVersion: v1
kind: Service
metadata:
  labels:
    app: hello-world
  name: hello-world
spec:
  ports:
  - port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: hello-world
  type: ClusterIP
```

```
deepa@ubuntu:~/devops/kind$ cat svc.yaml
apiVersion: v1
kind: Service
metadata:
  labels:
    app: hello-world
  name: hello-world
spec:
  ports:
  - port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: hello-world
  type: LoadBalancer
deepa@ubuntu:~/devops/kind$ kubectl apply -f svc.yaml
service/hello-world configured
deepa@ubuntu:~/devops/kind$
```

On **cloud platforms** (like AWS, GCP, Azure), EXTERNAL-IP will show a real IP.

On **Kind**, it stays <pending> — because Kind doesn't support external LoadBalancers by default.

```
deepa@ubuntu:~/devops/kind$ kubectl get svc
NAME          TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
hello-world   LoadBalancer   10.96.240.220   <pending>     80:30959/TCP   98m
kubernetes    ClusterIP      10.96.0.1       <none>        443/TCP        114m
deepa@ubuntu:~/devops/kind$ kubectl get svc hello-world
NAME          TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
hello-world   LoadBalancer   10.96.240.220   <pending>     80:30959/TCP   98m
deepa@ubuntu:~/devops/kind$
```

```
deepa@ubuntu:~/devops/kind$ kubectl port-forward svc/hello-world 8081:80
Forwarding from 127.0.0.1:8081 → 80
Forwarding from [::1]:8081 → 80
Handling connection for 8081
```

```
deepa@ubuntu:~/devops/kind$ curl http://localhost:8081
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
deepa@ubuntu:~/devops/kind$
```

Load balancer:

```
deepa@ubuntu:~/devops/kind$ az login
A web browser has been opened at https://login.microsoftonline.com/organizations/oauth2/v2.0/authorize. Please continue the login in the web browser. If no web browser is a
vailable or if the web browser fails to open, use device code flow with `az login --use-device-code`.
^Cdeepa@ubuntu:~/devops/kind$ az login --use-device-code
To sign in, use a web browser to open the page https://microsoft.com/devicelogin and enter the code SQ9G3XLPA to authenticate.

Retrieving tenants and subscriptions for the selection...

[Tenant and subscription selection]

No     Subscription name     Subscription ID                       Tenant
-----  --------------------  ------------------------------------  ----------------
[1] *  Azure subscription 1  d9a16617-0a23-4a2d-a693-f181d21ca740  Default Directory

The default is marked with an *; the default tenant is 'Default Directory' and subscription is 'Azure subscription 1' (d9a16617-0a23-4a2d-a693-f181d21ca740).

Select a subscription and tenant (Type a number or Enter for no changes):

Tenant: Default Directory
Subscription: Azure subscription 1 (d9a16617-0a23-4a2d-a693-f181d21ca740)

[Announcements]
With the new Azure CLI login experience, you can select the subscription you want to use more easily. Learn more about it and its configuration at https://go.microsoft.com/
fwlink/?linkid=2271236

If you encounter any problem, please open an issue at https://aka.ms/azclibug

[Warning] The login output has been updated. Please be aware that it no longer displays the full list of available subscriptions by default.
```

```
deepa@ubuntu:~/devops/kind$ az aks create \
>   --resource-group myResourceGroup \
>   --name myAKSCluster \
>   --node-count 1 \
>   --node-vm-size Standard_B2s \
>   --enable-managed-identity \
>   --generate-ssh-keys
The default value of '--node-vm-size' will be changed to 'Dynamically Selected By Azure' from 'Standard_DS2_V2 (Linux), Standard_DS2_V3 (Windows)' in next breaking change r
elease(2.73.0) scheduled for May 2025.
The cluster 'myAKSCluster' under resource group 'myResourceGroup' already exists. Please use command 'az aks update' to update the existing cluster, or select a different c
luster name to create a new cluster.
deepa@ubuntu:~/devops/kind$
```

that message means AKS cluster **already exists**, so we don't need to create it again. We ready to go.

✅ **Next Steps**

1. Connect to AKS Cluster

Make sure kubectl is configured to talk to the AKS cluster:

```
deepa@ubuntu:~/devops/kind$
deepa@ubuntu:~/devops/kind$ az aks get-credentials --resource-group myResourceGroup --name myAKSCluster

Merged "myAKSCluster" as current context in /home/deepa/.kube/config
deepa@ubuntu:~/devops/kind$
```

all set — local kubectl is now connected to AKS cluster! That message confirms:

AKS cluster myAKSCluster is now **active Kubernetes context**.

confirm the connection with:

```
deepa@ubuntu:~/devops/kind$ kubectl get nodes
NAME                                STATUS   ROLES    AGE   VERSION
aks-nodepool1-74711264-vmss000000   Ready    <none>   31h   v1.32.7
aks-nodepool1-74711264-vmss000001   Ready    <none>   31h   v1.32.7
deepa@ubuntu:~/devops/kind$
```

Perfect — AKS cluster is up and running with two healthy nodes! now fully connected and ready to deploy services that use Azure's infrastructure, including LoadBalancer.

```
deepa@ubuntu:~/devops/kind$ cat dep.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: hello-world
  name: hello-world
spec:
  replicas: 2
  selector:
    matchLabels:
      app: hello-world
  template:
    metadata:
      labels:
        app: hello-world
    spec:
      containers:
      - image: nginx
        name: nginx
        ports:
        -  containerPort: 80
        resources: {}
status: {}
```

```
deepa@ubuntu:~/devops/kind$ cat svc.yaml
apiVersion: v1
kind: Service
metadata:
  labels:
    app: hello-world
  name: hello-world
spec:
  ports:
  - port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: hello-world
  type: LoadBalancer
```

```
deepa@ubuntu:~/devops/kind$ kubectl apply -f dep.yaml
deployment.apps/hello-world created
deepa@ubuntu:~/devops/kind$
```

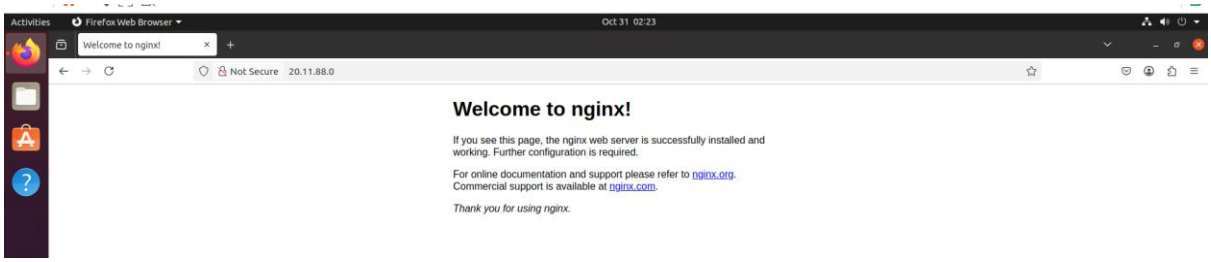
```
deepa@ubuntu:~/devops/kind$ kubectl get svc
NAME               TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)        AGE
apache-deploy      ClusterIP      10.0.144.142   <none>          8080/TCP       100m
hello-world        LoadBalancer   10.0.77.200    <pending>       80:30352/TCP   6s
kubernetes         ClusterIP      10.0.0.1       <none>          443/TCP        31h
nginx-deploy       LoadBalancer   10.0.158.211   4.254.116.111   80:31780/TCP   121m
nginx-deploy1      ClusterIP      10.0.29.237    <none>          80/TCP         102m
nginx-deployment   NodePort       10.0.216.120   <none>          80:31076/TCP   31h
deepa@ubuntu:~/devops/kind$
deepa@ubuntu:~/devops/kind$
deepa@ubuntu:~/devops/kind$ kubectl delete svc nginx-deploy
service "nginx-deploy" deleted from default namespace
deepa@ubuntu:~/devops/kind$
```

```
deepa@ubuntu:~/devops/kind$ kubectl get svc hello-world
NAME          TYPE           CLUSTER-IP    EXTERNAL-IP   PORT(S)        AGE
hello-world   LoadBalancer   10.0.77.200   20.11.88.0    80:30352/TCP   3m
deepa@ubuntu:~/devops/kind$
```

```
deepa@ubuntu:~/devops/kind$
deepa@ubuntu:~/devops/kind$ curl http://20.11.88.0
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
deepa@ubuntu:~/devops/kind$
```



Delete the AKS Cluster

To delete the AKS cluster and all associated resources:

bash

az aks delete --name myAKSCluster --resource-group myResourceGroup --yes --no-wait

This will:

- Delete the AKS cluster
- Detach and clean up the node pool
- Free up your Azure credits

```
deepa@ubuntu:~/devops/kind$ kubectl delete svc hello-world

service "hello-world" deleted from default namespace
deepa@ubuntu:~/devops/kind$
deepa@ubuntu:~/devops/kind$ az aks delete --name myAKSCluster --resource-group myResourceGroup --yes --no-wait

deepa@ubuntu:~/devops/kind$
deepa@ubuntu:~/devops/kind$
```