QUES 1 : **Create a Kubernetes cluster using minikube**

SOLN:

What is Minikube?

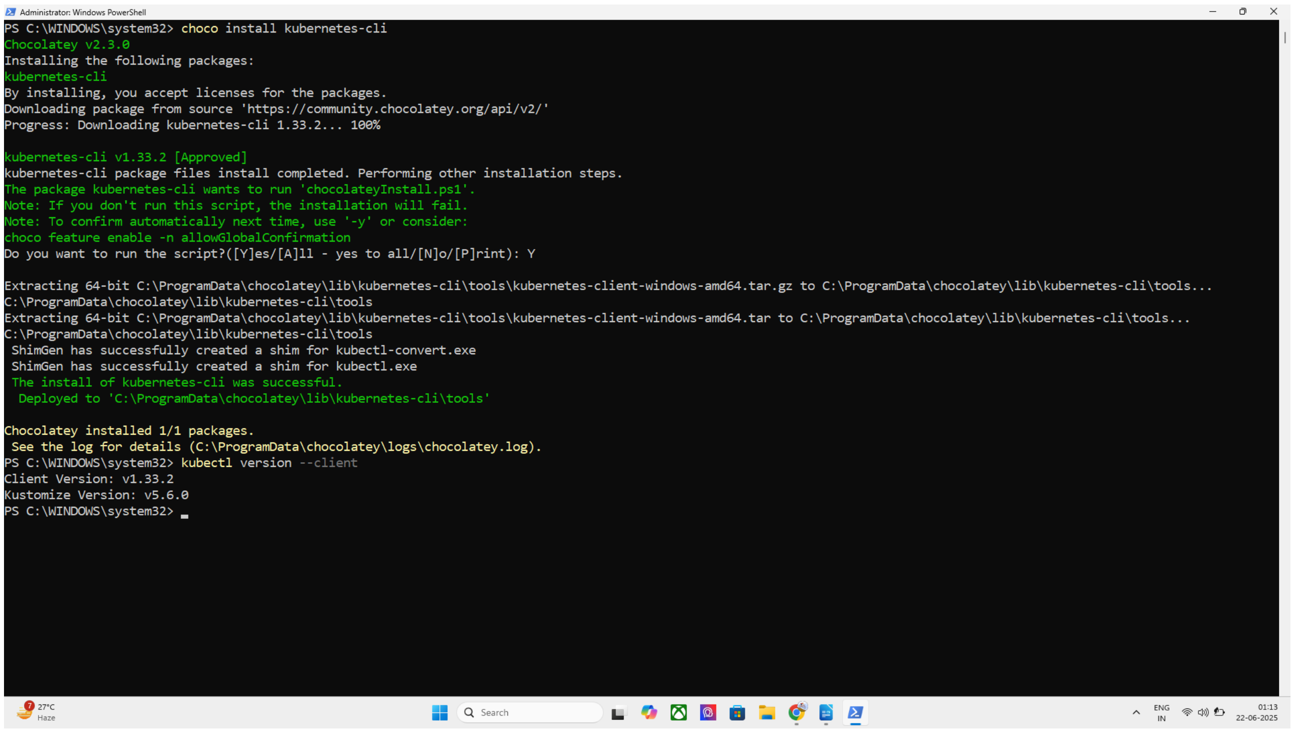
Minikube is a lightweight tool that creates a local Kubernetes cluster on your machine for development and testing purposes. It runs a single-node Kubernetes cluster inside a VM or container.

Install Required Tools

- **Minikube**
- **kubectl (Kubernetes CLI)**
- **Virtualization enabled** (like Hyper-V, VirtualBox, Docker, etc.)

For Windows (Using PowerShell or CMD)

✓ 1. Install kubectl



```
PS C:\WINDOWS\system32> choco install kubernetes-cli
Chocolatey v2.3.0
Installing the following packages:
kubernetes-cli
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading kubernetes-cli 1.33.2... 100%

kubernetes-cli v1.33.2 [Approved]
kubernetes-cli package files install completed. Performing other installation steps.
The package kubernetes-cli wants to run 'chocolateyInstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[A]ll - yes to all/[N]o/[P]rint): Y

Extracting 64-bit C:\ProgramData\chocolatey\lib\kubernetes-cli\tools\kubernetes-client-windows-amd64.tar.gz to C:\ProgramData\chocolatey\lib\kubernetes-cli\tools...
C:\ProgramData\chocolatey\lib\kubernetes-cli\tools
Extracting 64-bit C:\ProgramData\chocolatey\lib\kubernetes-cli\tools\kubernetes-client-windows-amd64.tar to C:\ProgramData\chocolatey\lib\kubernetes-cli\tools...
C:\ProgramData\chocolatey\lib\kubernetes-cli\tools
 ShimGen has successfully created a shim for kubectl-convert.exe
 ShimGen has successfully created a shim for kubectl.exe
 The install of kubernetes-cli was successful.
  Deployed to 'C:\ProgramData\chocolatey\lib\kubernetes-cli\tools'

Chocolatey installed 1/1 packages.
 See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\WINDOWS\system32> kubectl version --client
Client Version: v1.33.2
Kustomize Version: v5.6.0
PS C:\WINDOWS\system32>
```

`COMMAND : choco install kubernetes-cli`

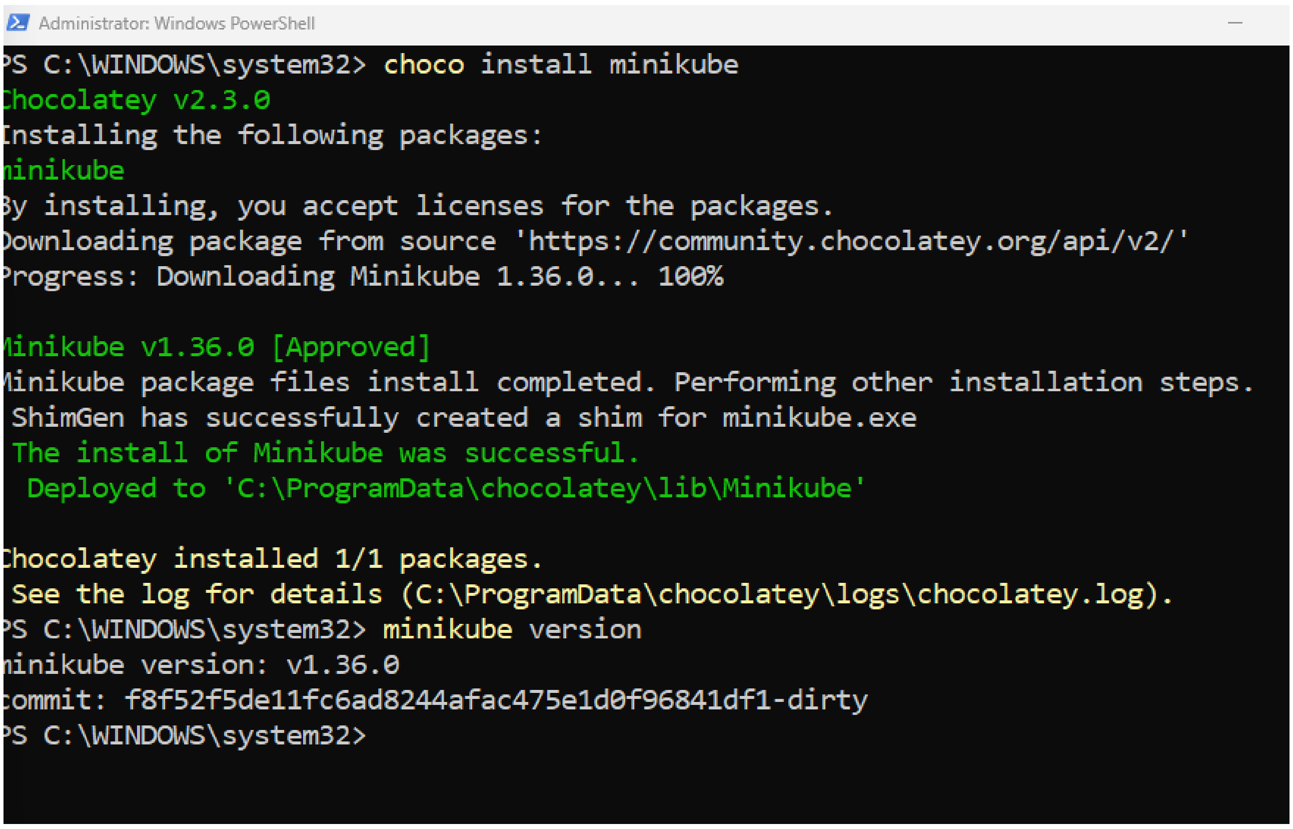
`You need Chocolatey installed.`

To verify:

`COMMAND : kubectl version --client`

2. **Install Minikube**

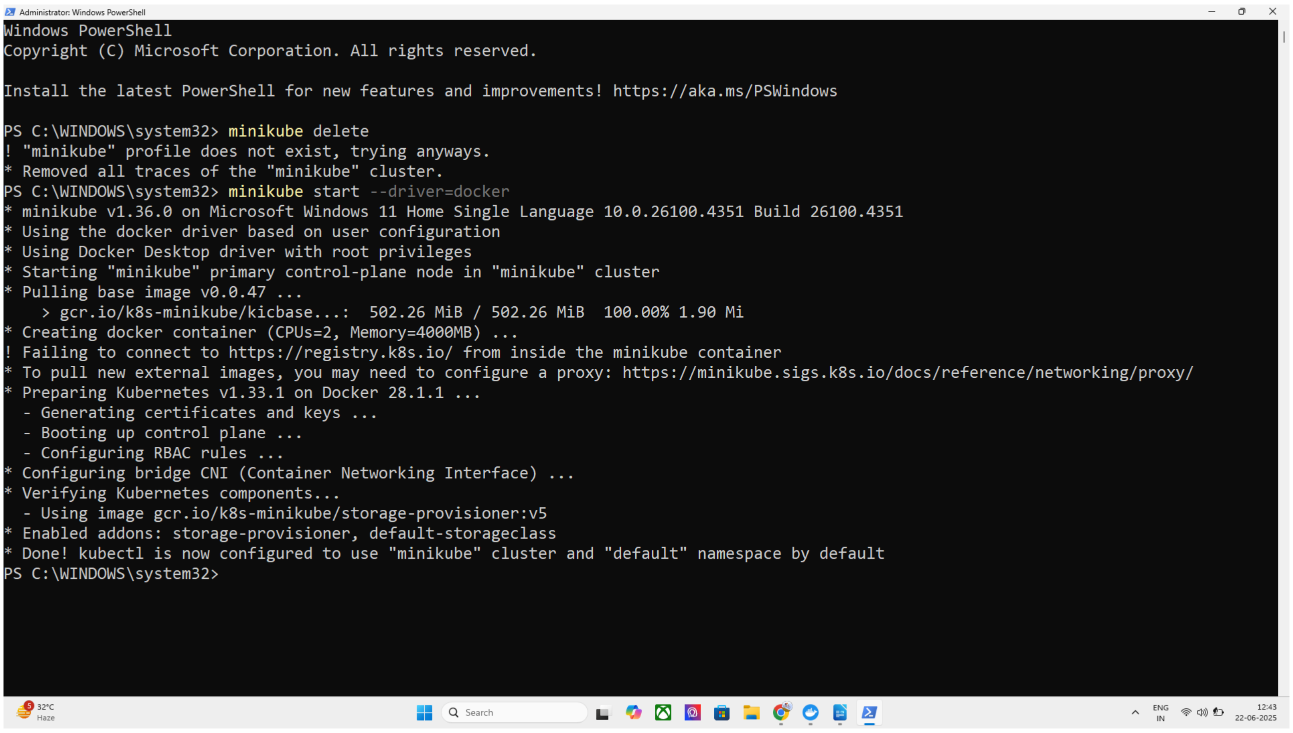
COMMAND : choco install minikube
TO VERIFY : minikube version



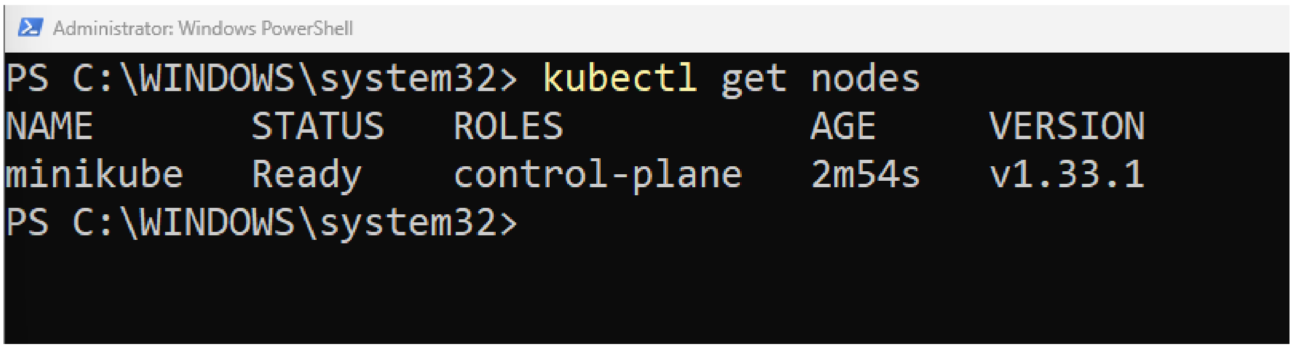
```
PS C:\WINDOWS\system32> choco install minikube
hocolatey v2.3.0
nstalling the following packages:
inikube
y installing, you accept licenses for the packages.
ownloading package from source 'https://community.chocolatey.org/api/v2/'
rogress: Downloading Minikube 1.36.0... 100%

inikube v1.36.0 [Approved]
inikube package files install completed. Performing other installation steps.
 ShimGen has successfully created a shim for minikube.exe
 The install of Minikube was successful.
  Deployed to 'C:\ProgramData\chocolatey\lib\Minikube'

hocolatey installed 1/1 packages.
 See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\WINDOWS\system32> minikube version
inikube version: v1.36.0
ommit: f8f52f5de11fc6ad8244afac475e1d0f96841df1-dirty
PS C:\WINDOWS\system32>
```

Start Minikube (After Installation)

COMMAND : minikube start / minikube start --driver=docker



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\WINDOWS\system32> minikube delete
! "minikube" profile does not exist, trying anyways.
* Removed all traces of the "minikube" cluster.
PS C:\WINDOWS\system32> minikube start --driver=docker
* minikube v1.36.0 on Microsoft Windows 11 Home Single Language 10.0.26100.4351 Build 26100.4351
* Using the docker driver based on user configuration
* Using Docker Desktop driver with root privileges
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.47 ...
    > gcr.io/k8s-minikube/kicbase...:  502.26 MiB / 502.26 MiB  100.00% 1.90 Mi
* Creating docker container (CPUs=2, Memory=4000MB) ...
! Failing to connect to https://registry.k8s.io/ from inside the minikube container
* To pull new external images, you may need to configure a proxy: https://minikube.sigs.k8s.io/docs/reference/networking/proxy/
* Preparing Kubernetes v1.33.1 on Docker 28.1.1 ...
  - Generating certificates and keys ...
  - Booting up control plane ...
  - Configuring RBAC rules ...
* Configuring bridge CNI (Container Networking Interface) ...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Enabled addons: storage-provisioner, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
PS C:\WINDOWS\system32>
```



Administrator: Windows PowerShell

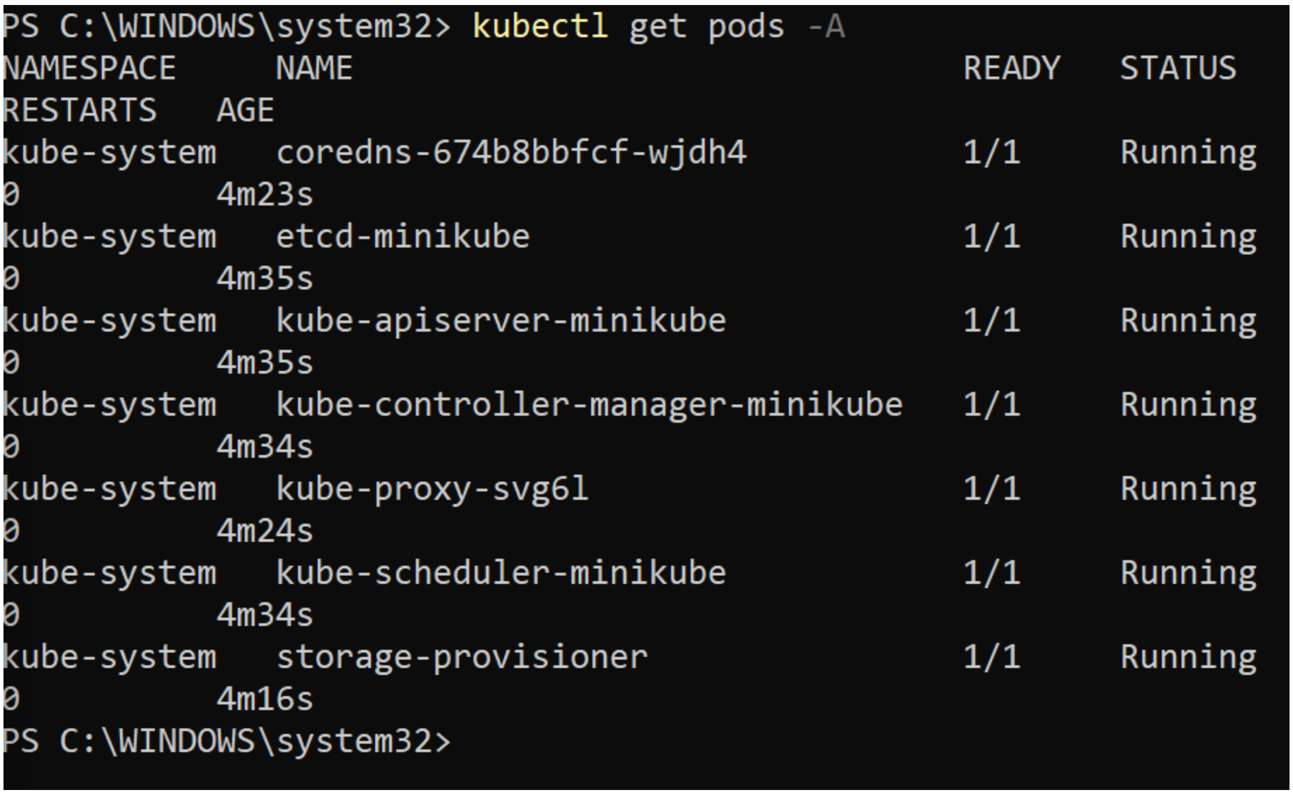
```
PS C:\WINDOWS\system32> kubectl get nodes
NAME       STATUS   ROLES           AGE     VERSION
minikube   Ready    control-plane   2m54s   v1.33.1
PS C:\WINDOWS\system32>
```

COMMAND TO GET NODES : kubectl get nodes

AND ALSO MINIKUBE IS A SINGLE NODE ARCHITECTURE

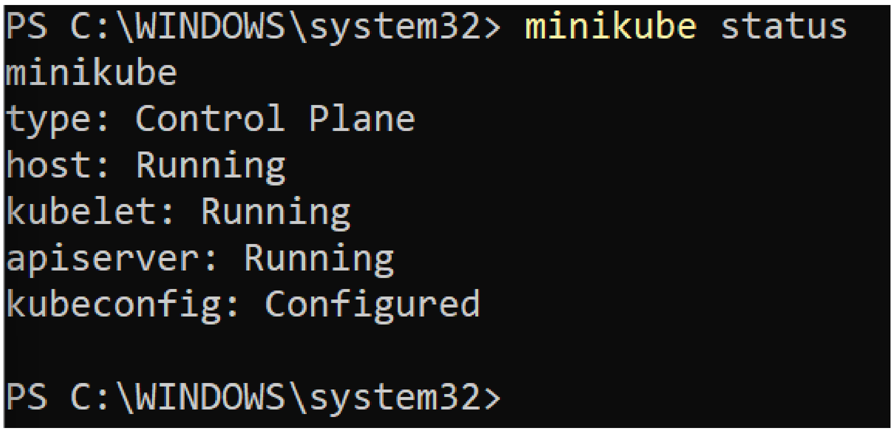
TO GET ALL THE PODS THAT ARE RUNNING IN NODE:

COMMAND USED : kubectl get pods -A



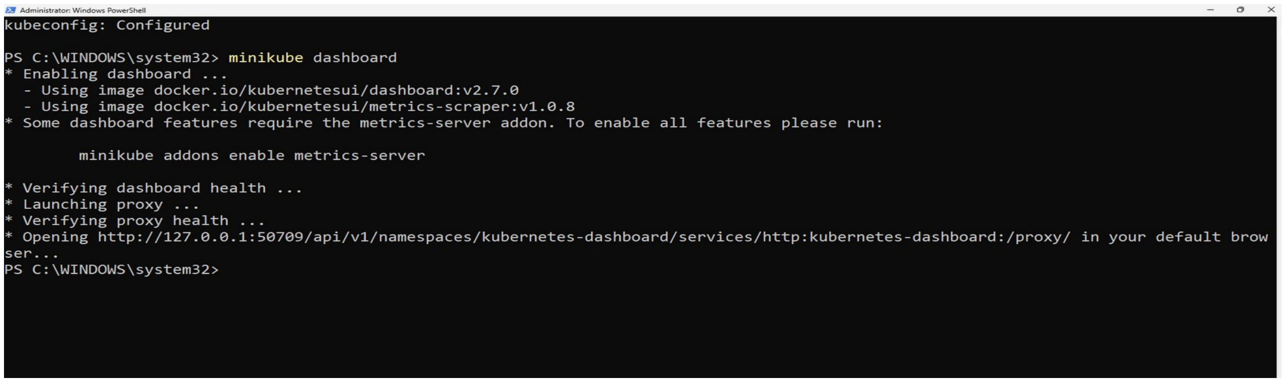
```
PS C:\WINDOWS\system32> kubectl get pods -A
NAMESPACE     NAME                               READY   STATUS
RESTARTS   AGE
kube-system   coredns-674b8bbfcf-wjdh4           1/1     Running
0          4m23s
kube-system   etcd-minikube                      1/1     Running
0          4m35s
kube-system   kube-apiserver-minikube            1/1     Running
0          4m35s
kube-system   kube-controller-manager-minikube   1/1     Running
0          4m34s
kube-system   kube-proxy-svg6l                   1/1     Running
0          4m24s
kube-system   kube-scheduler-minikube            1/1     Running
0          4m34s
kube-system   storage-provisioner                1/1     Running
0          4m16s
PS C:\WINDOWS\system32>
```

TO CHECK STATUS :
command used :
minikube status

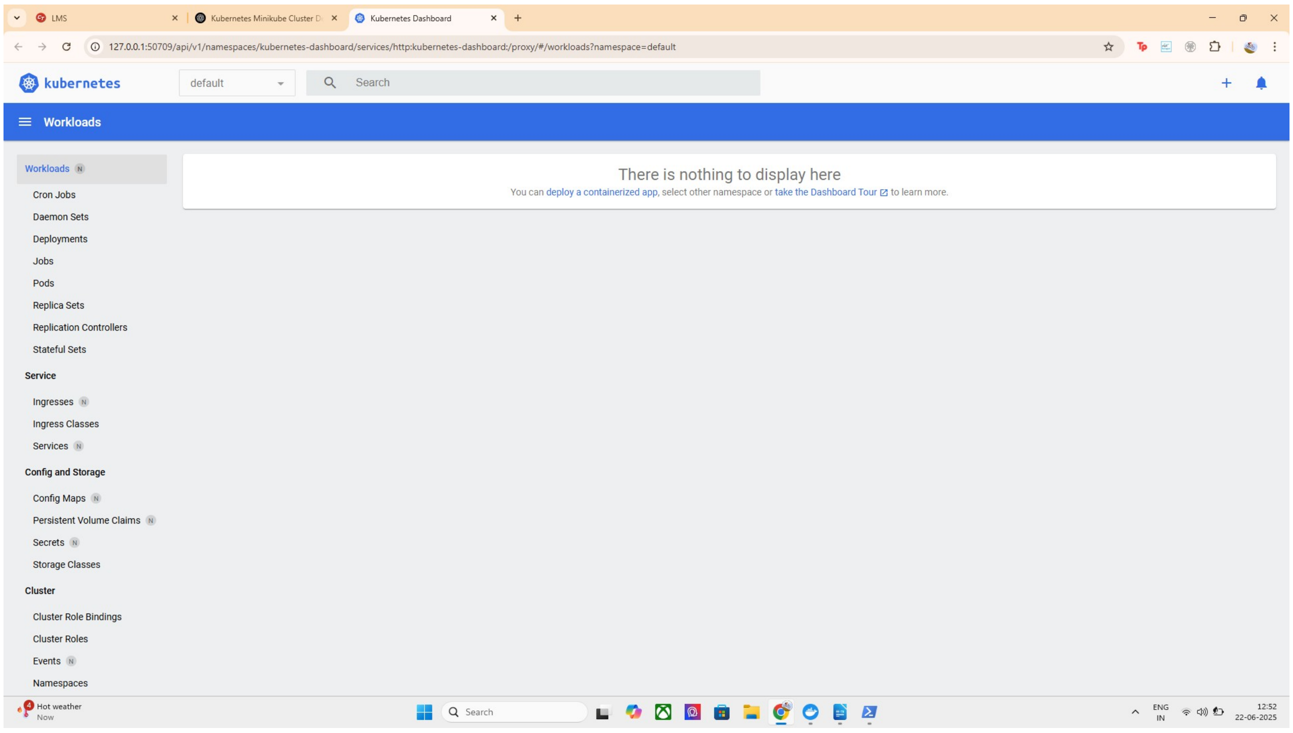


```
PS C:\WINDOWS\system32> minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured

PS C:\WINDOWS\system32>
```

TO OPEN THE DASHBOARD WE USE : minikube dashboard



```
kubeconfig: Configured

PS C:\WINDOWS\system32> minikube dashboard
* Enabling dashboard ...
  - 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 features please run:

        minikube addons enable metrics-server

* Verifying dashboard health ...
* Launching proxy ...
* Verifying proxy health ...
* Opening http://127.0.0.1:50709/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default brow
ser...
PS C:\WINDOWS\system32>
```



There is nothing to display here
You can deploy a containerized app, select other namespace or take the Dashboard Tour to learn more.

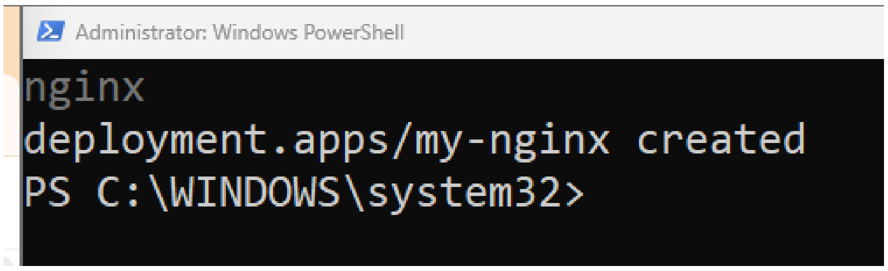
FOR NOW THERE IS NO DEPLOYMENT HENCE DASHBOARD IS EMPTY SO LETS US CREATE A DEPLOYMENT

Lets Deploy a Sample Nginx Application First :

command used : kubectl create deployment my-nginx –image=nginx

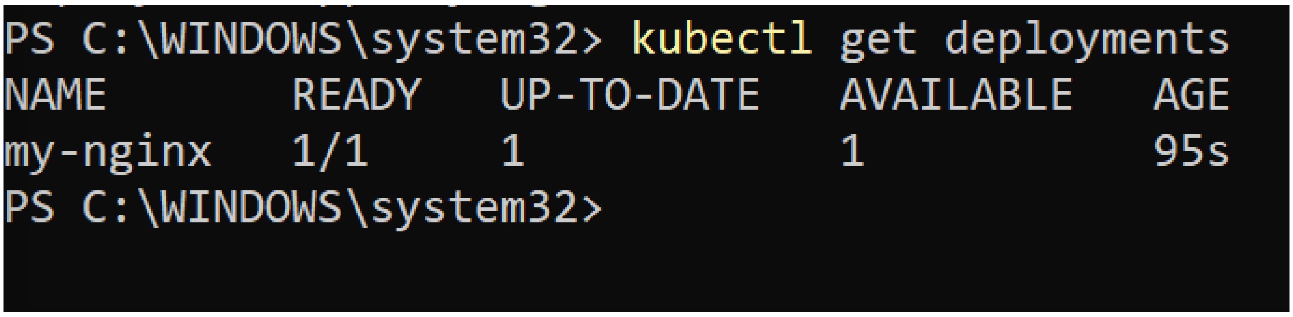
This creates a deployment named `my-nginx` using the official Nginx container image.

- Kubernetes will automatically create a pod running Nginx.



```
nginx
deployment.apps/my-nginx created
PS C:\WINDOWS\system32>
```

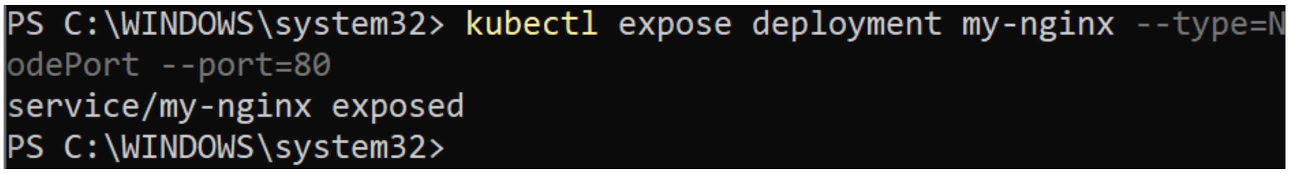
TO GET YOUR DEPLOYMENTS : kubectl get deployments



```
PS C:\WINDOWS\system32> kubectl get deployments
NAME       READY   UP-TO-DATE   AVAILABLE   AGE
my-nginx   1/1     1            1           95s
PS C:\WINDOWS\system32>
```

Now Lets Expose the Nginx Deployment

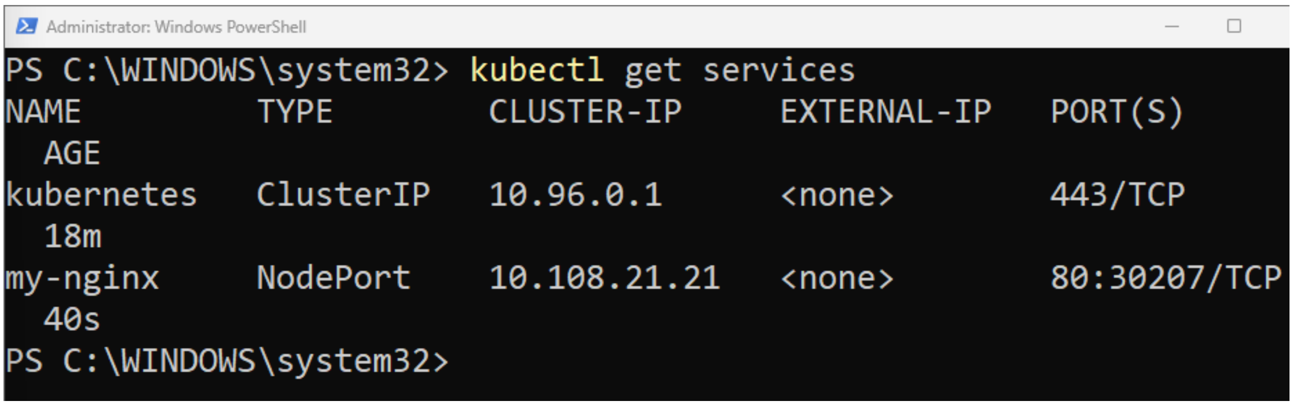
command used : kubectl expose deployment my-nginx --type=NodePort –port=80



```
PS C:\WINDOWS\system32> kubectl expose deployment my-nginx --type=N
odePort --port=80
service/my-nginx exposed
PS C:\WINDOWS\system32>
```

Exposes the Nginx deployment so it can be accessed outside the cluster via a NodePort service.

Command used : kubectl get services



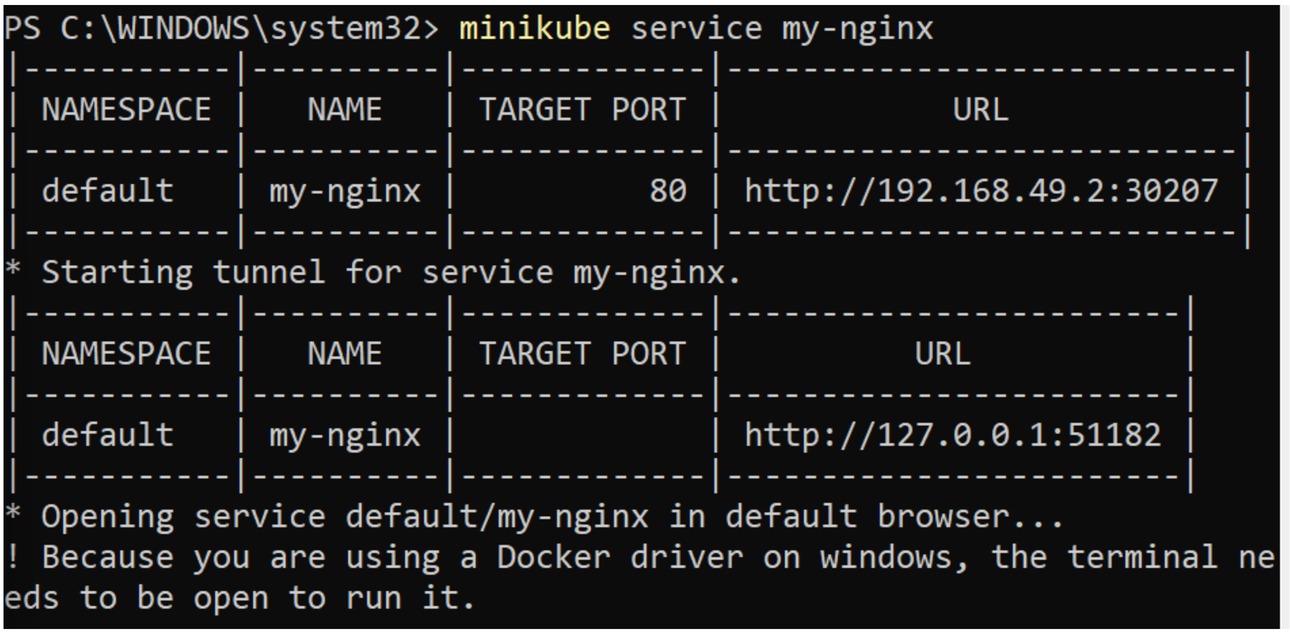
Administrator: Windows PowerShell

```
PS C:\WINDOWS\system32> kubectl get services
NAME         TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)
  AGE
kubernetes   ClusterIP   10.96.0.1      <none>        443/TCP
  18m
my-nginx     NodePort    10.108.21.21   <none>        80:30207/TCP
  40s
PS C:\WINDOWS\system32>
```

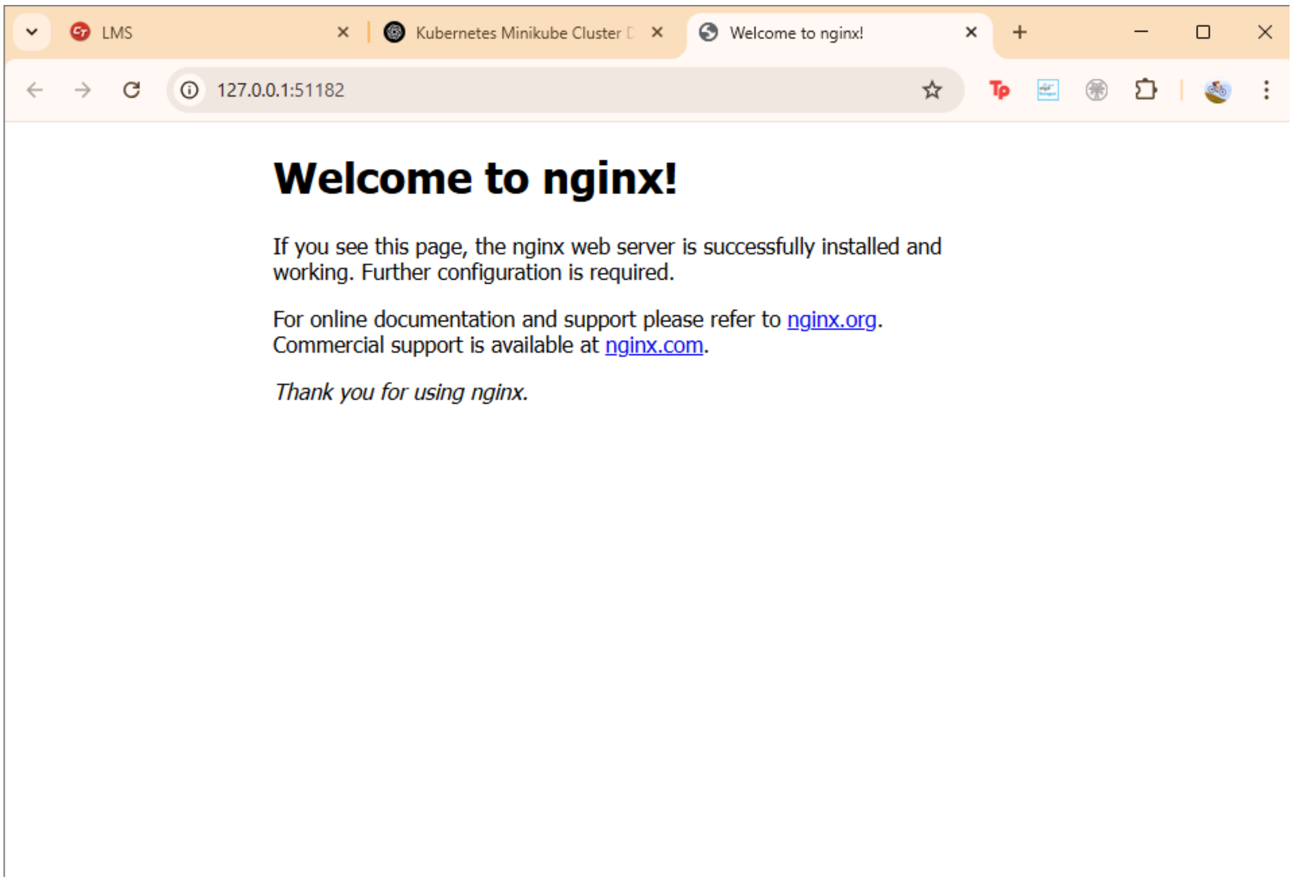
Access the App via Minikube Service

command used : minikube service my-nginx

Service running successfully:



```
PS C:\WINDOWS\system32> minikube service my-nginx
|-----------|----------|-------------|---------------------------|
| NAMESPACE |   NAME   | TARGET PORT |            URL            |
|-----------|----------|-------------|---------------------------|
| default   | my-nginx |          80 | http://192.168.49.2:30207 |
|-----------|----------|-------------|---------------------------|
* Starting tunnel for service my-nginx.
|-----------|----------|-------------|------------------------|
| NAMESPACE |   NAME   | TARGET PORT |          URL           |
|-----------|----------|-------------|------------------------|
| default   | my-nginx |             | http://127.0.0.1:51182 |
|-----------|----------|-------------|------------------------|
* Opening service default/my-nginx in default browser...
! Because you are using a Docker driver on windows, the terminal ne
eds to be open to run it.
```



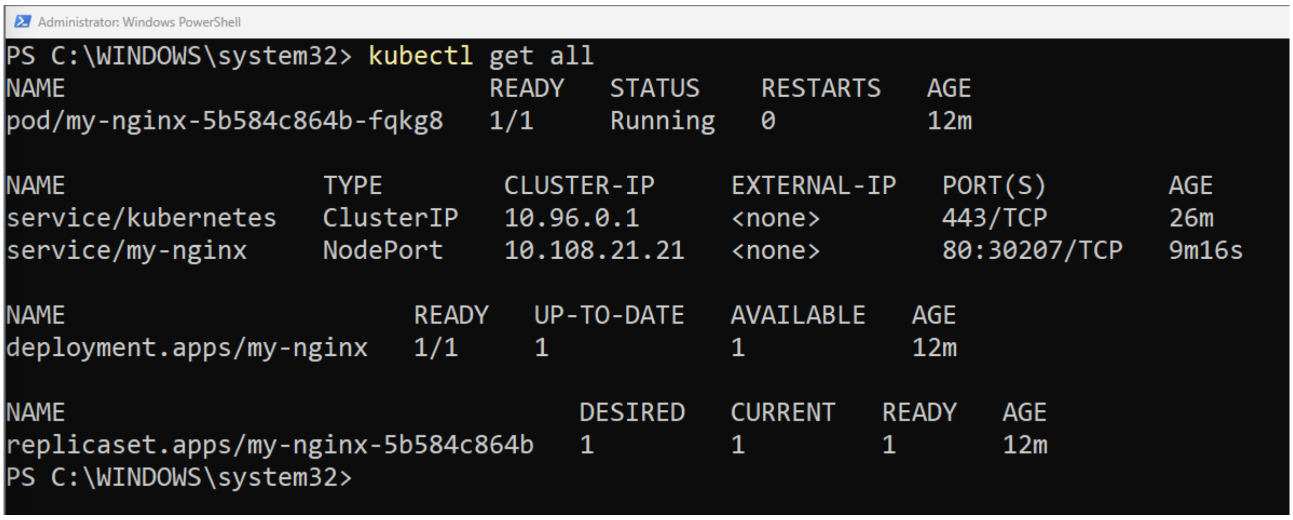
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

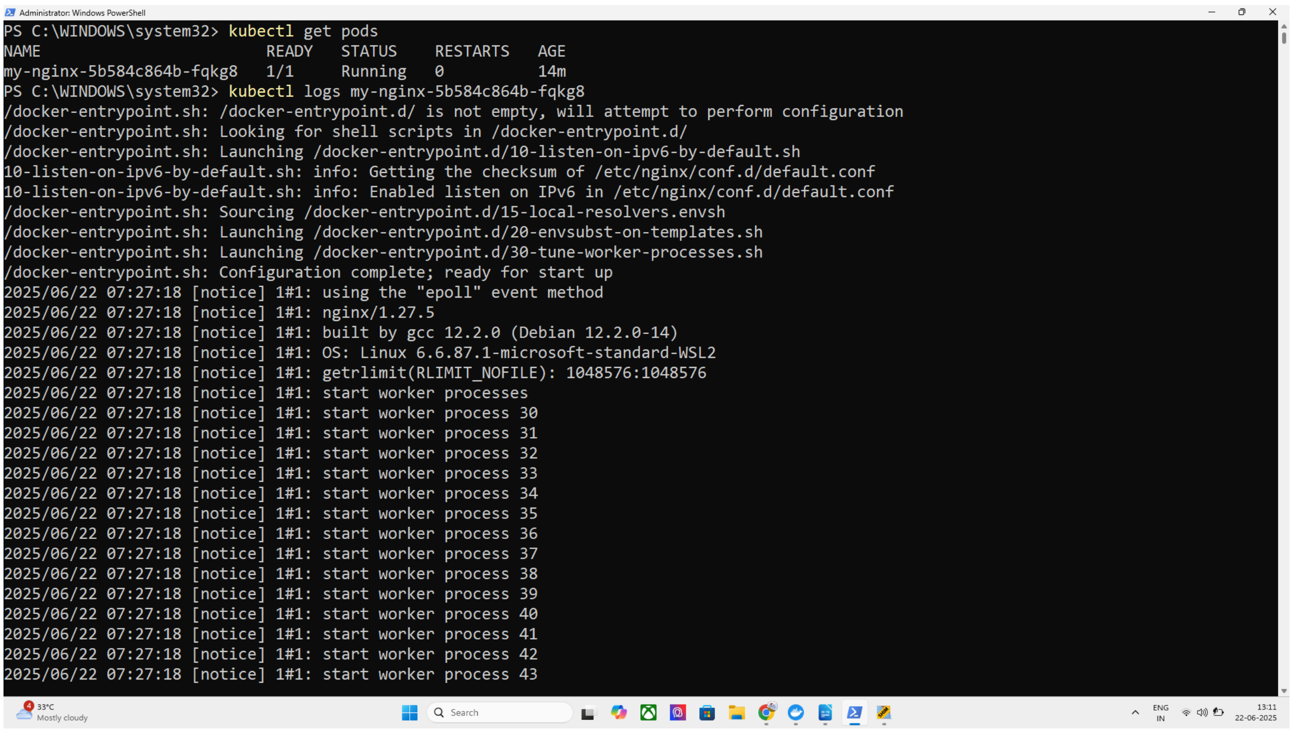
7. View Cluster Resources : kubectl get all



```
PS C:\WINDOWS\system32> kubectl get all
NAME                       READY   STATUS    RESTARTS   AGE
pod/my-nginx-5b584c864b-fqkg8   1/1     Running   0          12m

NAME                 TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)        AGE
service/kubernetes   ClusterIP   10.96.0.1       <none>        443/TCP        26m
service/my-nginx     NodePort    10.108.21.21    <none>        80:30207/TCP   9m16s

NAME                       READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/my-nginx   1/1     1            1           12m

NAME                                  DESIRED   CURRENT   READY   AGE
replicaset.apps/my-nginx-5b584c864b   1         1         1       12m
PS C:\WINDOWS\system32>
```

Check Logs of Running Pod :

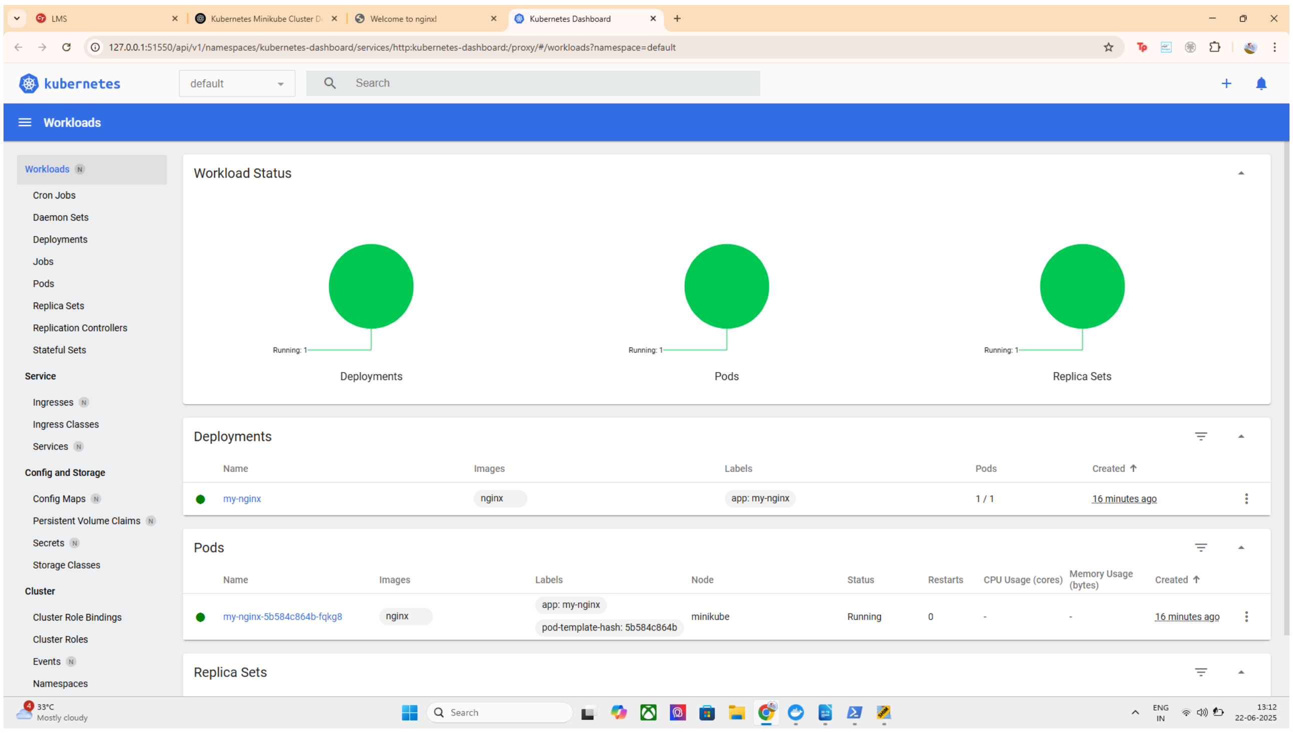
kubectl get pods
kubectl logs <pod-name>



```
PS C:\WINDOWS\system32> kubectl get pods
NAME                        READY   STATUS    RESTARTS   AGE
my-nginx-5b584c864b-fqkg8   1/1     Running   0          14m
PS C:\WINDOWS\system32> kubectl logs my-nginx-5b584c864b-fqkg8
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: Getting the checksum of /etc/nginx/conf.d/default.conf
10-listen-on-ipv6-by-default.sh: info: Enabled listen on IPv6 in /etc/nginx/conf.d/default.conf
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
2025/06/22 07:27:18 [notice] 1#1: using the "epoll" event method
2025/06/22 07:27:18 [notice] 1#1: nginx/1.27.5
2025/06/22 07:27:18 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2025/06/22 07:27:18 [notice] 1#1: OS: Linux 6.6.87.1-microsoft-standard-WSL2
2025/06/22 07:27:18 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2025/06/22 07:27:18 [notice] 1#1: start worker processes
2025/06/22 07:27:18 [notice] 1#1: start worker process 30
2025/06/22 07:27:18 [notice] 1#1: start worker process 31
2025/06/22 07:27:18 [notice] 1#1: start worker process 32
2025/06/22 07:27:18 [notice] 1#1: start worker process 33
2025/06/22 07:27:18 [notice] 1#1: start worker process 34
2025/06/22 07:27:18 [notice] 1#1: start worker process 35
2025/06/22 07:27:18 [notice] 1#1: start worker process 36
2025/06/22 07:27:18 [notice] 1#1: start worker process 37
2025/06/22 07:27:18 [notice] 1#1: start worker process 38
2025/06/22 07:27:18 [notice] 1#1: start worker process 39
2025/06/22 07:27:18 [notice] 1#1: start worker process 40
2025/06/22 07:27:18 [notice] 1#1: start worker process 41
2025/06/22 07:27:18 [notice] 1#1: start worker process 42
2025/06/22 07:27:18 [notice] 1#1: start worker process 43
```

Lets open the dashboard again : Open the Kubernetes Dashboard

Now you can see deployments are ready



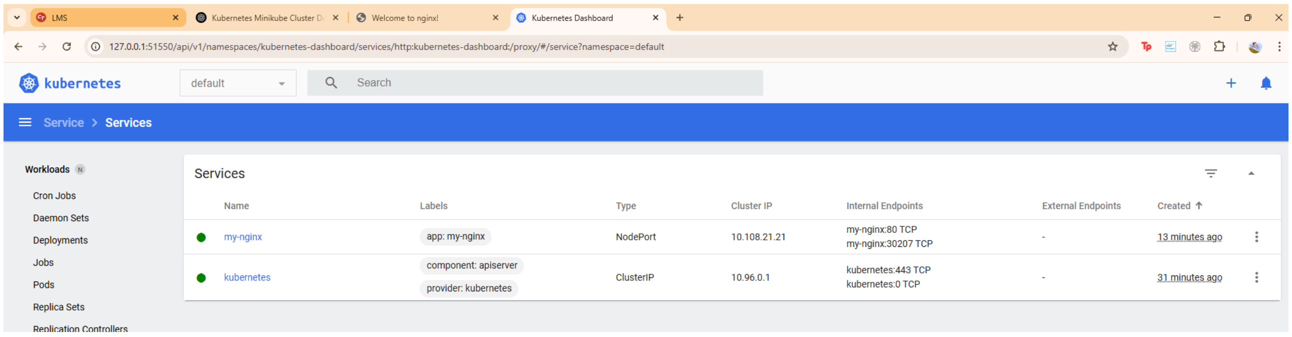
Deployments

	Name	Images	Labels	Pods	Created ↑
●	my-nginx	nginx	app: my-nginx	1 / 1	16 minutes ago

Pods

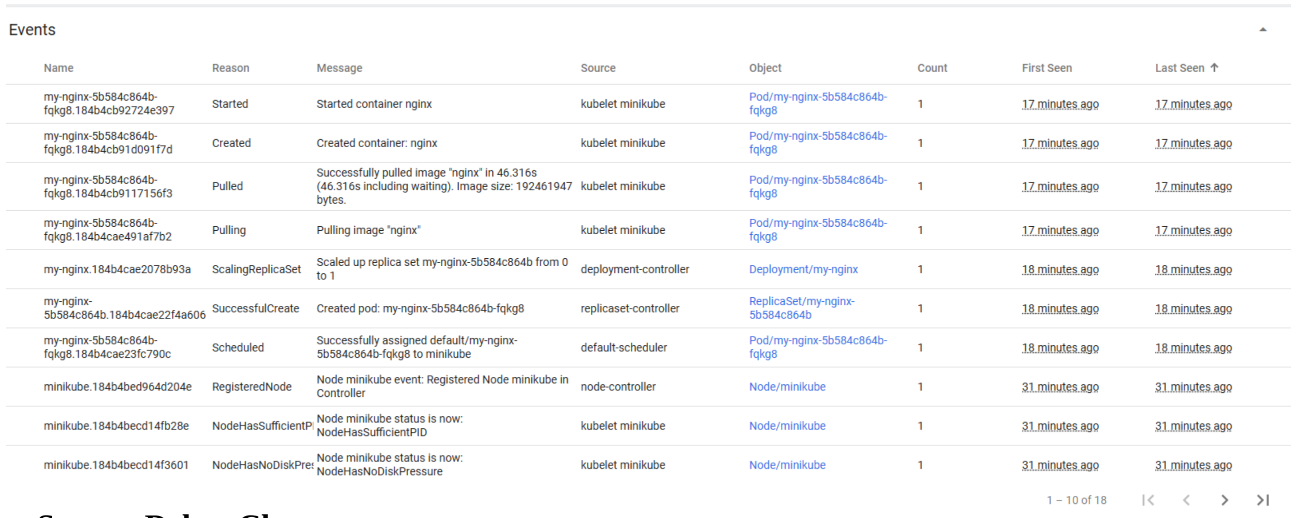
	Name	Images	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created ↑
●	my-nginx-5b584c864b-fqkg8	nginx	app: my-nginx pod-template-hash: 5b584c864b	minikube	Running	0	-	-	16 minutes ago

Replica Sets



Services

	Name	Labels	Type	Cluster IP	Internal Endpoints	External Endpoints	Created ↑
●	my-nginx	app: my-nginx	NodePort	10.108.21.21	my-nginx:80 TCP my-nginx:30207 TCP	-	13 minutes ago
●	kubernetes	component: apiserver provider: kubernetes	ClusterIP	10.96.0.1	kubernetes:443 TCP kubernetes:0 TCP	-	31 minutes ago



Events

Name	Reason	Message	Source	Object	Count	First Seen	Last Seen ↑
my-nginx-5b584c864b-fqkg8.184b4cb92724e397	Started	Started container nginx	kubelet minikube	Pod/my-nginx-5b584c864b-fqkg8	1	17 minutes ago	17 minutes ago
my-nginx-5b584c864b-fqkg8.184b4cb91d091f7d	Created	Created container: nginx	kubelet minikube	Pod/my-nginx-5b584c864b-fqkg8	1	17 minutes ago	17 minutes ago
my-nginx-5b584c864b-fqkg8.184b4cb9117156f3	Pulled	Successfully pulled image "nginx" in 46.316s (46.316s including waiting). Image size: 192461947 bytes.	kubelet minikube	Pod/my-nginx-5b584c864b-fqkg8	1	17 minutes ago	17 minutes ago
my-nginx-5b584c864b-fqkg8.184b4cae491af7b2	Pulling	Pulling image "nginx"	kubelet minikube	Pod/my-nginx-5b584c864b-fqkg8	1	17 minutes ago	17 minutes ago
my-nginx.184b4cae2078b93a	ScalingReplicaSet	Scaled up replica set my-nginx-5b584c864b from 0 to 1	deployment-controller	Deployment/my-nginx	1	18 minutes ago	18 minutes ago
my-nginx-5b584c864b.184b4cae22f4a606	SuccessfulCreate	Created pod: my-nginx-5b584c864b-fqkg8	replicaset-controller	ReplicaSet/my-nginx-5b584c864b	1	18 minutes ago	18 minutes ago
my-nginx-5b584c864b-fqkg8.184b4cae23fc790c	Scheduled	Successfully assigned default/my-nginx-5b584c864b-fqkg8 to minikube	default-scheduler	Pod/my-nginx-5b584c864b-fqkg8	1	18 minutes ago	18 minutes ago
minikube.184b4bed964d204e	RegisteredNode	Node minikube event: Registered Node minikube in Controller	node-controller	Node/minikube	1	31 minutes ago	31 minutes ago
minikube.184b4becd14fb28e	NodeHasSufficientP	Node minikube status is now: NodeHasSufficientPID	kubelet minikube	Node/minikube	1	31 minutes ago	31 minutes ago
minikube.184b4becd14f3601	NodeHasNoDiskPres	Node minikube status is now: NodeHasNoDiskPressure	kubelet minikube	Node/minikube	1	31 minutes ago	31 minutes ago

1 – 10 of 18

Stop or Delete Cluster :

```
minikube stop       # Stops the cluster
minikube delete     # Deletes the Minikube VM and cleans up resources
```

LETS GET SOME MORE INSIGHTS

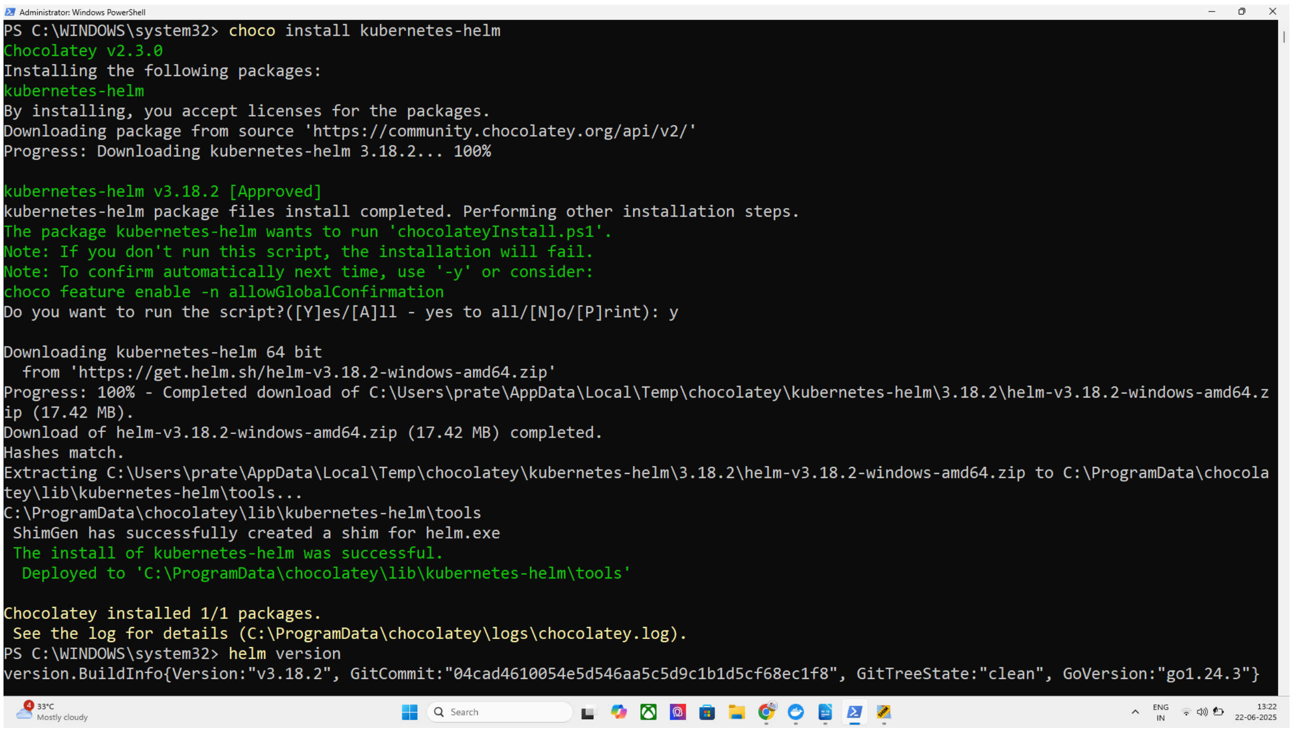
BY DEPLOYING SOME

1. Helm Chart

Step 1: Install Helm

For Windows (Using Chocolatey):

```
choco install kubernetes-helm

helm version or helm --version
```

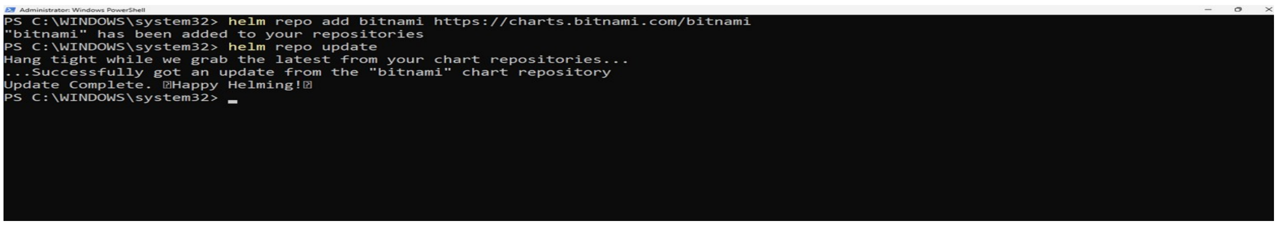


```
PS C:\WINDOWS\system32> choco install kubernetes-helm
Chocolatey v2.3.0
Installing the following packages:
kubernetes-helm
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading kubernetes-helm 3.18.2... 100%

kubernetes-helm v3.18.2 [Approved]
kubernetes-helm package files install completed. Performing other installation steps.
The package kubernetes-helm wants to run 'chocolateyInstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[A]ll - yes to all/[N]o/[P]rint): y

Downloading kubernetes-helm 64 bit
  from 'https://get.helm.sh/helm-v3.18.2-windows-amd64.zip'
Progress: 100% - Completed download of C:\Users\prate\AppData\Local\Temp\chocolatey\kubernetes-helm\3.18.2\helm-v3.18.2-windows-amd64.z
ip (17.42 MB).
Download of helm-v3.18.2-windows-amd64.zip (17.42 MB) completed.
Hashes match.
Extracting C:\Users\prate\AppData\Local\Temp\chocolatey\kubernetes-helm\3.18.2\helm-v3.18.2-windows-amd64.zip to C:\ProgramData\chocola
tey\lib\kubernetes-helm\tools...
C:\ProgramData\chocolatey\lib\kubernetes-helm\tools
 ShimGen has successfully created a shim for helm.exe
 The install of kubernetes-helm was successful.
  Deployed to 'C:\ProgramData\chocolatey\lib\kubernetes-helm\tools'

Chocolatey installed 1/1 packages.
 See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\WINDOWS\system32> helm version
version.BuildInfo{Version:"v3.18.2", GitCommit:"04cad4610054e5d546aa5c5d9c1b1d5cf68ec1f8", GitTreeState:"clean", GoVersion:"go1.24.3"}
```

Step 2: Add Bitnami Helm Repository

helm repo add bitnami https://charts.bitnami.com/bitnami

helm repo update



```
PS C:\WINDOWS\system32> helm repo add bitnami https://charts.bitnami.com/bitnami
"bitnami" has been added to your repositories
PS C:\WINDOWS\system32> helm repo update
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "bitnami" chart repository
Update Complete. ⎈Happy Helming!⎈
PS C:\WINDOWS\system32> _
```

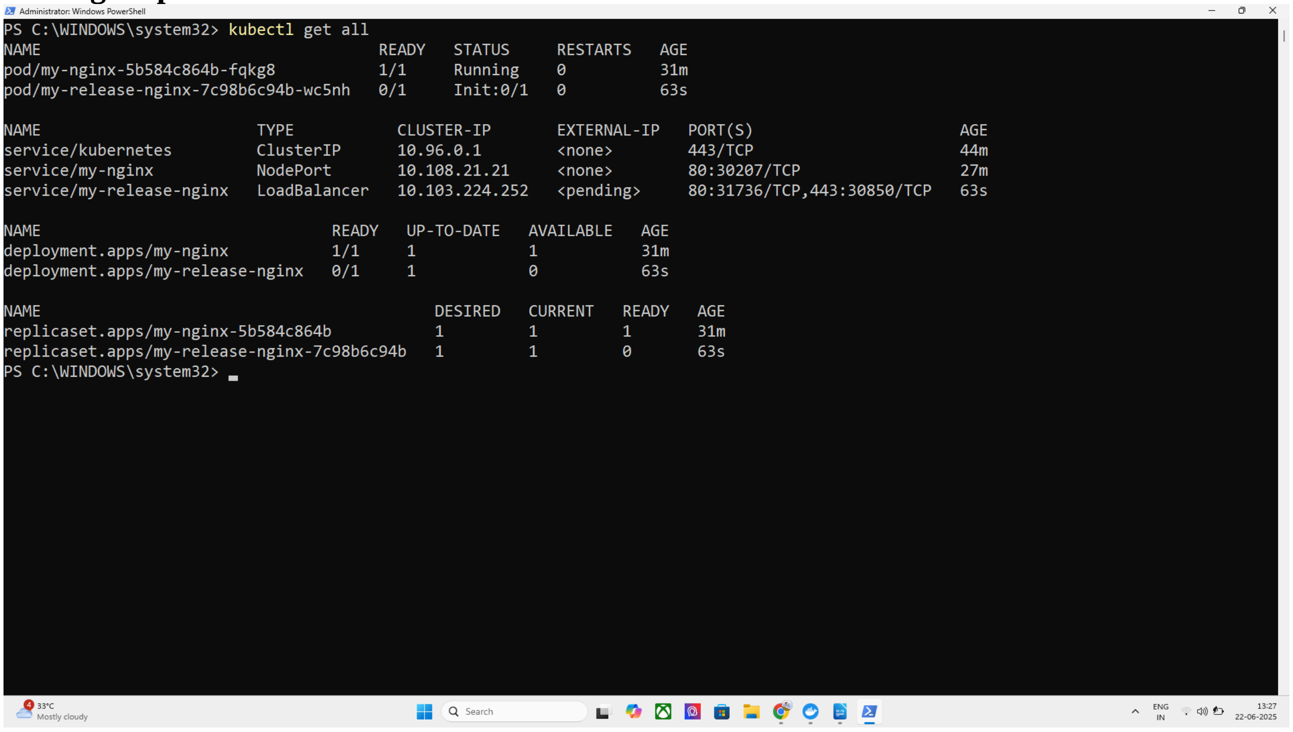
Step 3: Install NGINX via Helm

helm install my-release bitnami/nginx

Step 4: Verify Deployment

kubectl get all

New nginx pod and service formed



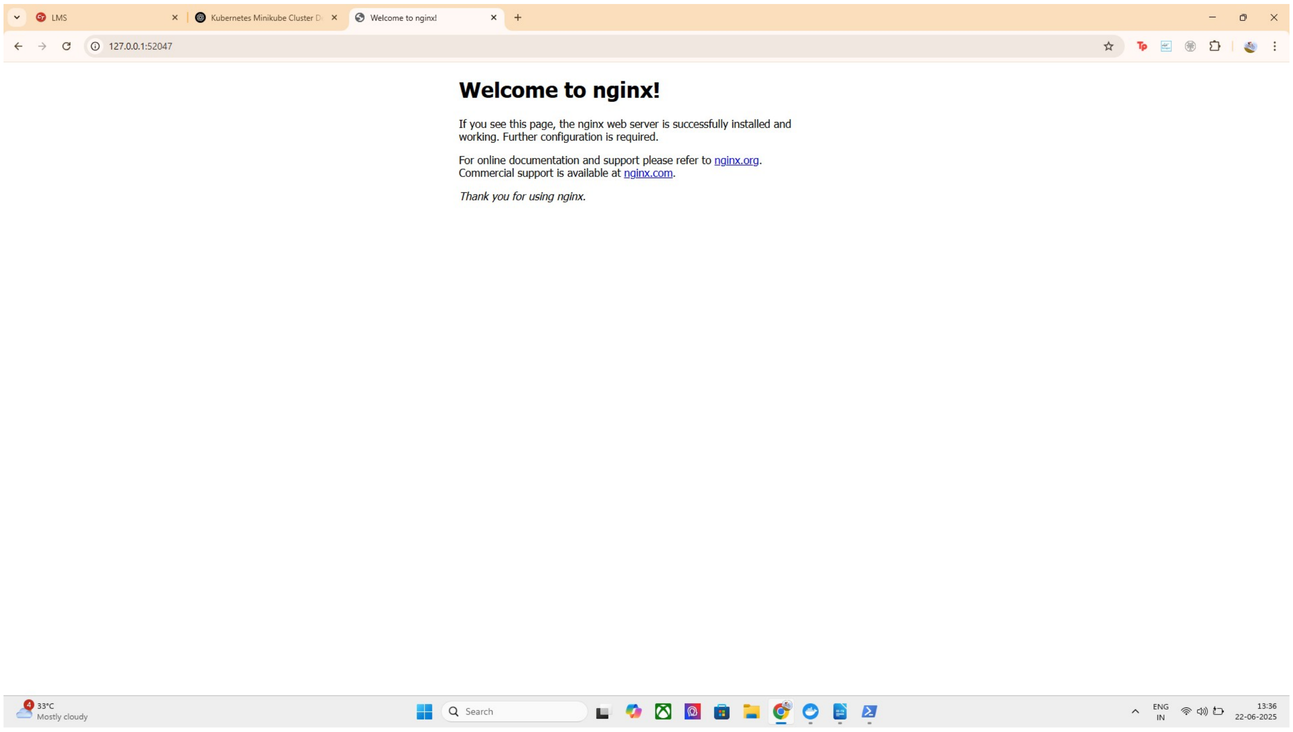
```
PS C:\WINDOWS\system32> kubectl get all
NAME                                 READY   STATUS     RESTARTS   AGE
pod/my-nginx-5b584c864b-fqkg8        1/1     Running    0          31m
pod/my-release-nginx-7c98b6c94b-wc5nh   0/1     Init:0/1   0          63s

NAME                       TYPE           CLUSTER-IP       EXTERNAL-IP   PORT(S)                      AGE
service/kubernetes         ClusterIP      10.96.0.1        <none>        443/TCP                      44m
service/my-nginx           NodePort       10.108.21.21     <none>        80:30207/TCP                 27m
service/my-release-nginx   LoadBalancer   10.103.224.252   <pending>     80:31736/TCP,443:30850/TCP   63s

NAME                               READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/my-nginx           1/1     1            1           31m
deployment.apps/my-release-nginx   0/1     1            0           63s

NAME                                          DESIRED   CURRENT   READY   AGE
replicaset.apps/my-nginx-5b584c864b           1         1         1       31m
replicaset.apps/my-release-nginx-7c98b6c94b   1         1         0       63s
PS C:\WINDOWS\system32>
```

Step 5: Access the App

kubectl get svc

minikube service my-nginx-svc

successfully accessed on local host : http://127.0.0.1:52047

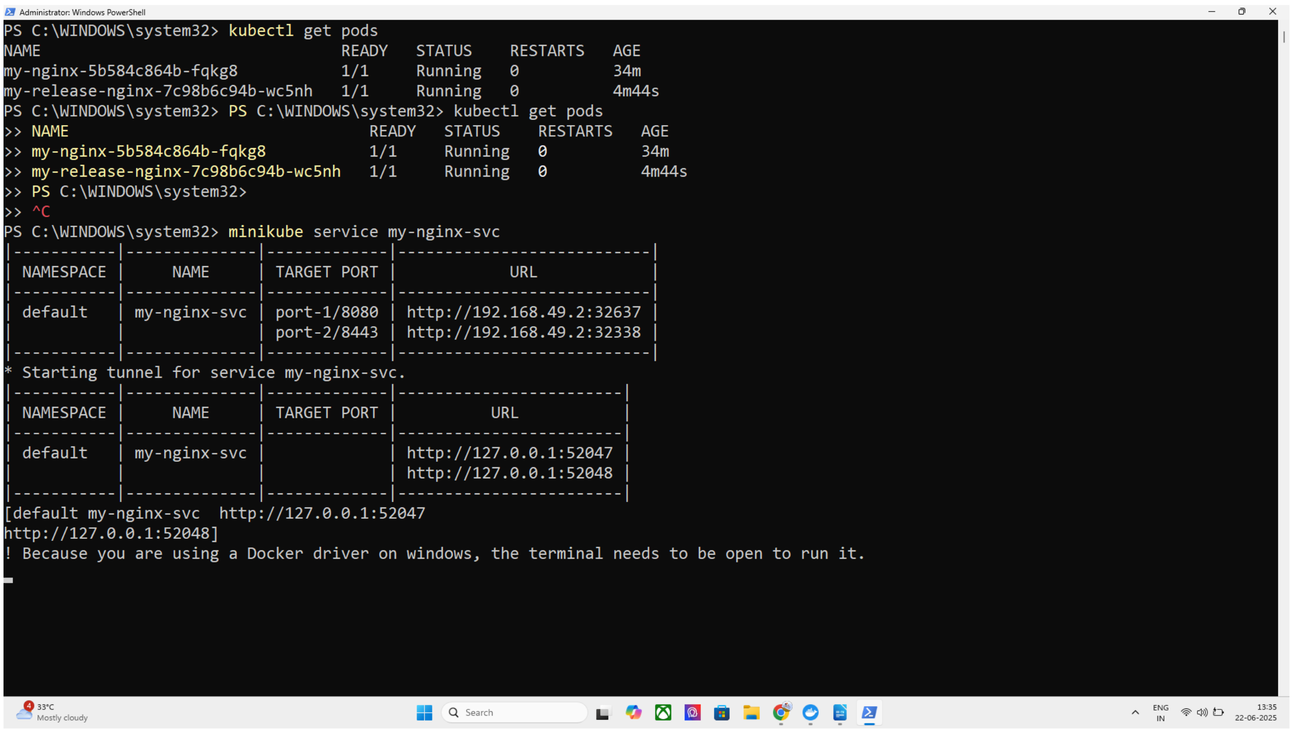


Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.



```
PS C:\WINDOWS\system32> kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
my-nginx-5b584c864b-fqkg8          1/1     Running   0          34m
my-release-nginx-7c98b6c94b-wc5nh  1/1     Running   0          4m44s
PS C:\WINDOWS\system32> PS C:\WINDOWS\system32> kubectl get pods
>> NAME                               READY   STATUS    RESTARTS   AGE
>> my-nginx-5b584c864b-fqkg8          1/1     Running   0          34m
>> my-release-nginx-7c98b6c94b-wc5nh  1/1     Running   0          4m44s
>> PS C:\WINDOWS\system32>
>> ^C
PS C:\WINDOWS\system32> minikube service my-nginx-svc
|-----------|--------------|-------------|---------------------------|
| NAMESPACE |     NAME     | TARGET PORT |            URL            |
|-----------|--------------|-------------|---------------------------|
| default   | my-nginx-svc | port-1/8080 | http://192.168.49.2:32637 |
|           |              | port-2/8443 | http://192.168.49.2:32338 |
|-----------|--------------|-------------|---------------------------|
* Starting tunnel for service my-nginx-svc.
|-----------|--------------|-------------|------------------------|
| NAMESPACE |     NAME     | TARGET PORT |          URL           |
|-----------|--------------|-------------|------------------------|
| default   | my-nginx-svc |             | http://127.0.0.1:52047 |
|           |              |             | http://127.0.0.1:52048 |
|-----------|--------------|-------------|------------------------|
[default my-nginx-svc  http://127.0.0.1:52047
http://127.0.0.1:52048]
! Because you are using a Docker driver on windows, the terminal needs to be open to run it.
```

NOW THERE ARE 2 DEPLOYMENTS ONE WITH HELM CHART

What is a Helm Chart?

A **Helm Chart** is a collection of:

- Kubernetes YAML files
- Templates

- Predefined configurations

That together **automate the deployment** of a complete application

Why It's Powerful

Instead of writing multiple YAML files (for Deployment, Service, ConfigMap, PVC, etc.), you:

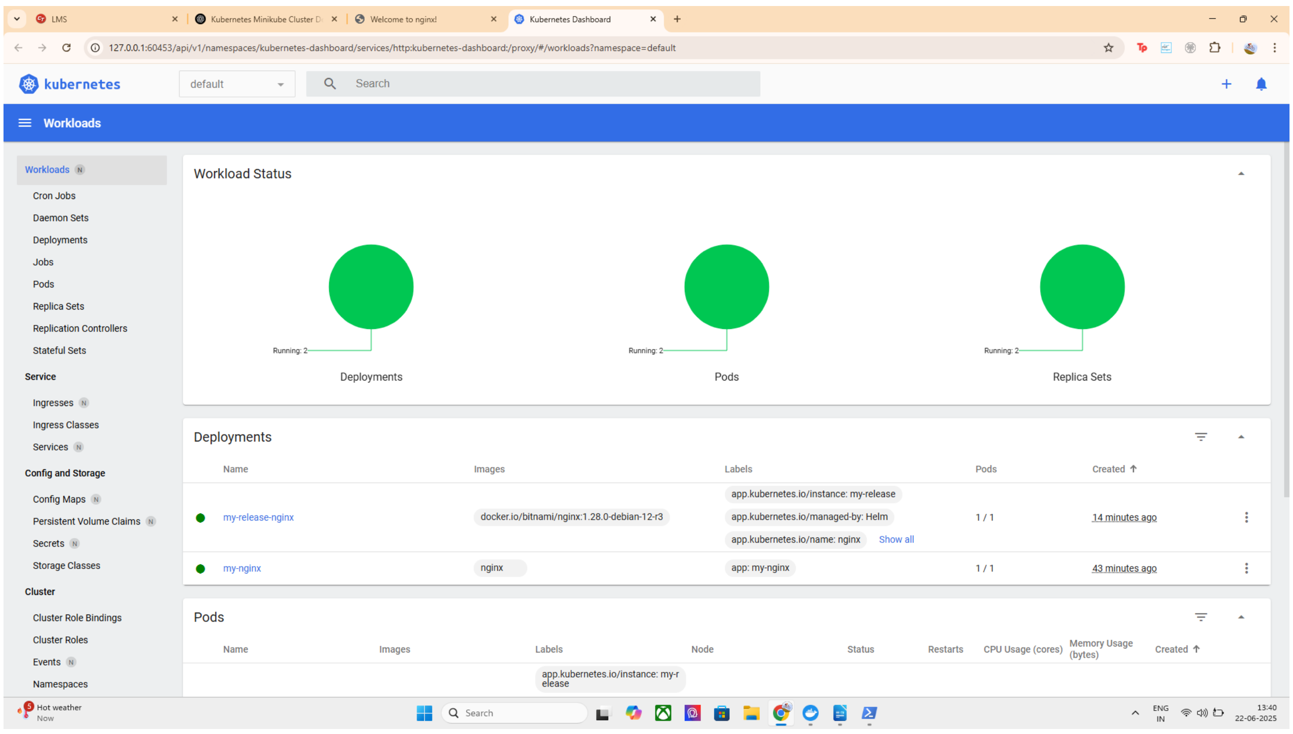
- Run **one command**
- Helm installs the app with **best practices**
- You can customize it later with `--set` or values.yaml

Example: What Happens When You Run This?

```
helm install my-release bitnami/nginx
```

You are telling Helm:

- Fetch the **NGINX chart** from Bitnami's repo
- Install it in your cluster
- Use default settings unless overridden
- Name the release `my-release`



127.0.0.1:60453/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/#/workloads?namespace=default

Workload Status

Deployments

	Name	Images	Labels	Pods	Created ↑
●	my-release-nginx	docker.io/bitnami/nginx:1.28.0-debian-12-r3	app.kubernetes.io/instance: my-release app.kubernetes.io/managed-by: Helm app.kubernetes.io/name: nginx Show all	1 / 1	14 minutes ago
●	my-nginx	nginx	app: my-nginx	1 / 1	43 minutes ago

Pods

Name	Images	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created ↑
		app.kubernetes.io/instance: my-r elease						