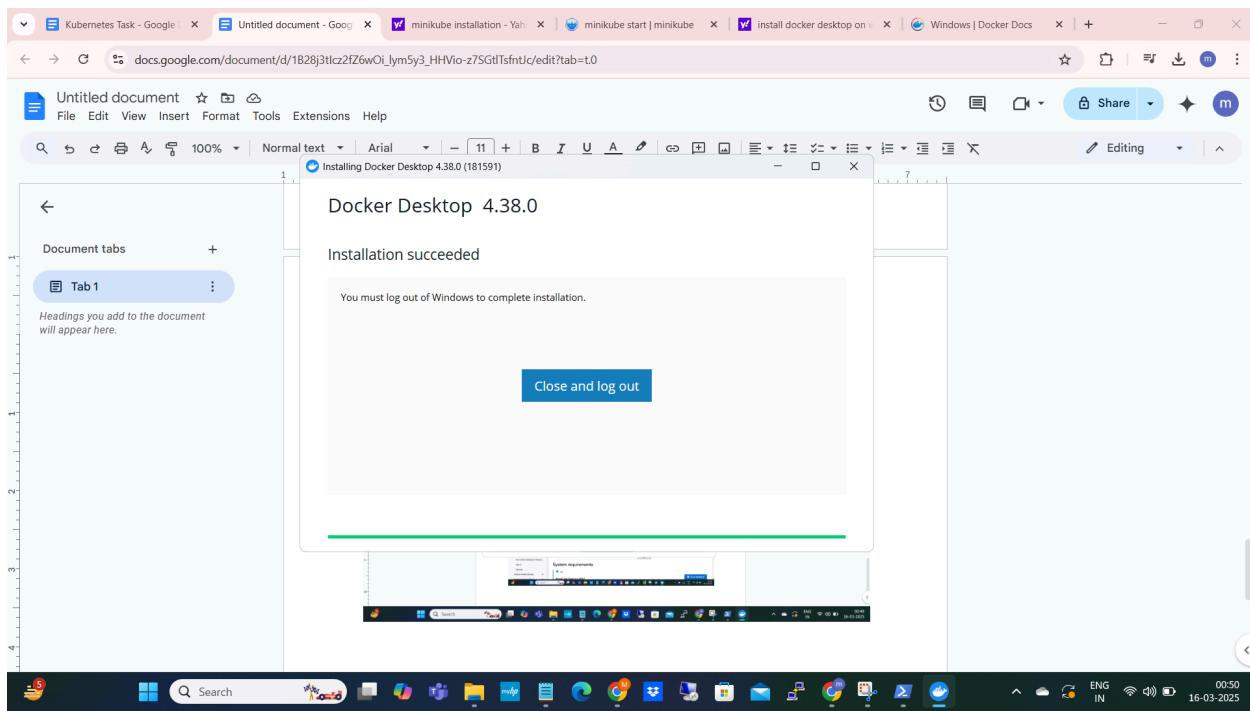
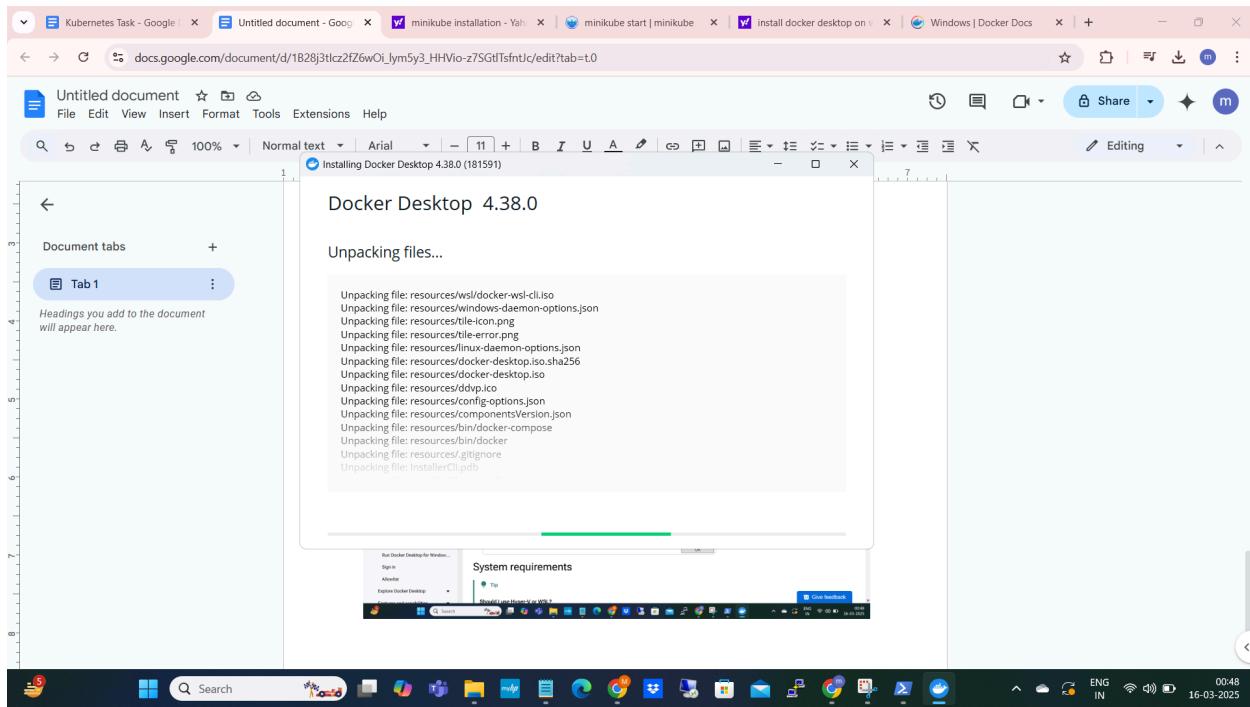


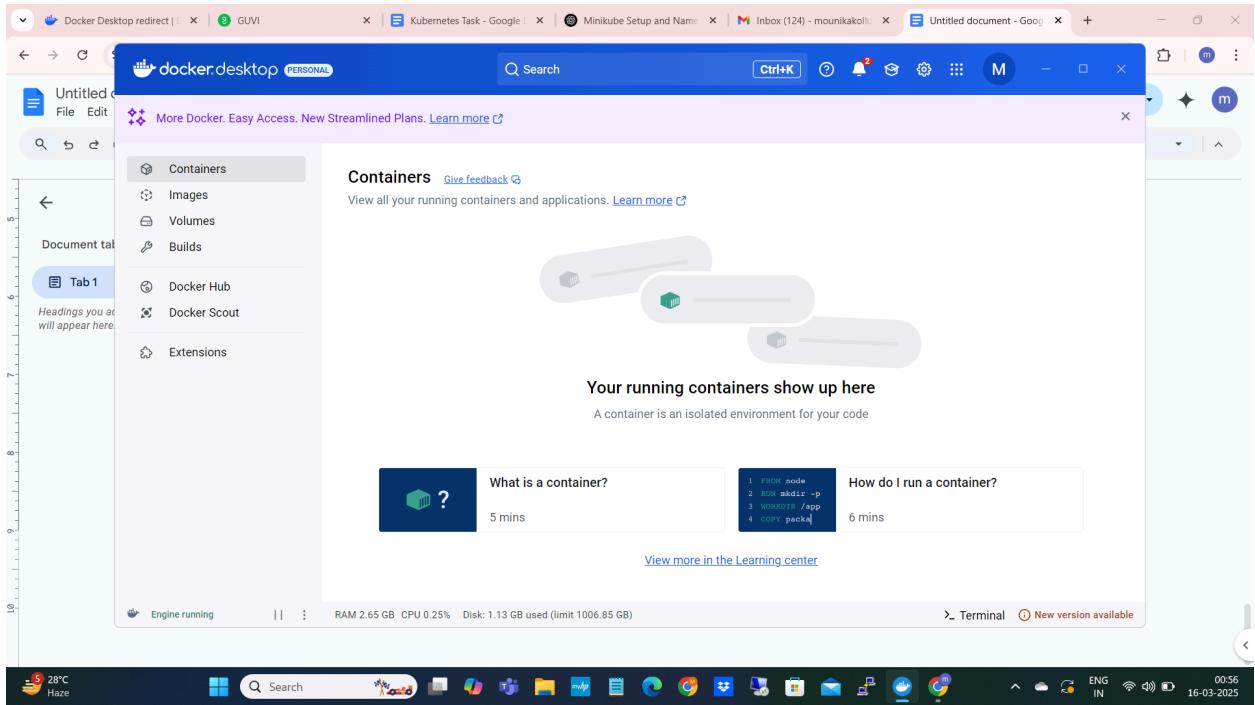
## Kubernetes Task

Install Docker Desktop:  
Download .exe file

The screenshot shows a Microsoft Edge browser window with the URL [docs.docker.com/desktop/setup/install/windows-install/](https://docs.docker.com/desktop/setup/install/windows-install/). The page title is "Install Docker Desktop on Windows". The left sidebar is expanded, showing the "Windows" section under "Docker Desktop". The main content area contains instructions and download links for Docker Desktop for Windows (x86\_64 and Arm Beta). A "System requirements" section is also present. A "Tip" sidebar suggests using Hyper-V or WSL2. A "Recent download history" sidebar shows a download for "Docker Desktop Installer.exe". The taskbar at the bottom shows various pinned icons.

The screenshot shows a Microsoft Edge browser window with the URL [docs.docker.com/desktop/setup/install/windows-install/](https://docs.docker.com/desktop/setup/install/windows-install/). The page title is "Installing Docker Desktop 4.38.0 (181591)". A configuration dialog box is open in the foreground, prompting the user to "Add shortcut to desktop". The "OK" button is visible at the bottom right of the dialog. The rest of the page content, including the "System requirements" section and sidebar, is visible in the background. The taskbar at the bottom shows various pinned icons.





## Install Chocolatey:

```
$ Set-ExecutionPolicy Bypass -Scope Process -Force;
[System.Net.ServicePointManager]::SecurityProtocol =
[System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object
System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))
```

```
PS C:\WINDOWS\system32>
+ FullyQualifiedErrorId : CommandNotFoundException
PS C:\WINDOWS\system32> Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))
Forcing web requests to allow TLS v1.2 (Required for requests to Chocolatey.org)
Getting latest version of the Chocolatey package for download.
Not using proxy.
Getting Chocolatey from https://community.chocolatey.org/api/v2/package/chocolatey/2.4.3.
Downloading https://community.chocolatey.org/api/v2/package/chocolatey/2.4.3 to C:\Users\mouni\AppData\Local\Temp\chocoInstall\chocolatey.zip
Not using proxy.
Extracting C:\Users\mouni\AppData\Local\Temp\chocolatey\chocoInstall\chocolatey.zip to C:\Users\mouni\AppData\Local\Temp\chocolatey\chocoInstall
Installing Chocolatey on the local machine
Creating ChocolateyInstall as an environment variable (targeting 'Machine')
Setting ChocolateyInstall to C:\ProgramData\chocolatey
WARNING: It is highly likely you will need to close and reopen your shell
before you can use choco.
Restricting write permissions to Administrators
We are setting up the Chocolatey package repository.
The packages themselves go to 'C:\ProgramData\chocolatey\lib'
(i.e. C:\ProgramData\chocolatey\lib\yourPackageName).
A shim file for the command line goes to 'C:\ProgramData\chocolatey\bin'
and points to an executable in 'C:\ProgramData\chocolatey\lib\yourPackageName'.
Creating Chocolatey CLI folders if they do not already exist.

chocolatey.nupkg file not installed in lib.
Attempting to locate it from bootstrapper.
PATH environment variable does not have C:\ProgramData\chocolatey\bin in it. Adding...
WARNING: Not setting PATH environment variable file does not exist at 'C:\Users\mouni\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1'.
Chocolatey CLI (choco.exe) is now ready.
You can call choco from anywhere, command line or powershell by typing choco.
Run choco /? for a list of functions.
You may need to shut down and restart powershell and/or consoles
first prior to using choco.
Ensuring Chocolatey commands are on the path
Ensuring chocolatey.nupkg is in the lib folder
PS C:\WINDOWS\system32>
```

## Install kubectl using chocolatey

```
$ choco install kubernetes-cli
```

```
Select Administrator: Windows PowerShell
PATH environment variable does not have C:\ProgramData\chocolatey\bin in it. Adding...
WARNING: Not setting tab completion: Profile file does not exist at 'C:\Users\mounti\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1'.
Chocolatey v2.4.3 (choco.exe) is now ready.
You can call choco from the command line or powershell by typing choco.
Run choco /? for a list of functions.
You may need to shut down and restart powershell and/or consoles
first prior to using choco.
Ensuring Chocolatey commands are on the path
Ensuring chocolatey.nupkg is in the lib folder
PS C:\WINDOWS\system32> choco --version
2.4.3
PS C:\WINDOWS\system32> choco install kubernetes-cli
Chocolatey v2.4.3
3 validations performed. 2 success(es), 1 warning(s), and 0 error(s).

Validation Warnings:
- A pending system reboot request has been detected, however, this is
being ignored due to the current Chocolatey configuration. If you
want to halt when this occurs, then either set the global feature
using:
  choco feature enable --name="exitOnRebootDetected"
or pass the option --exit-when-reboot-detected.

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 1.32.3... 100%
kubernetes-cli v1.32.3 [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): yes

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>
```

## Install Minikube:

```
$ choco install minikube
```

```
Select Administrator: Windows PowerShell
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading kubernetes-cli 1.32.3... 100%
kubernetes-cli v1.32.3 [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): yes

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> choco install minikube
Chocolatey v2.4.3
3 validations performed. 2 success(es), 1 warning(s), and 0 error(s).

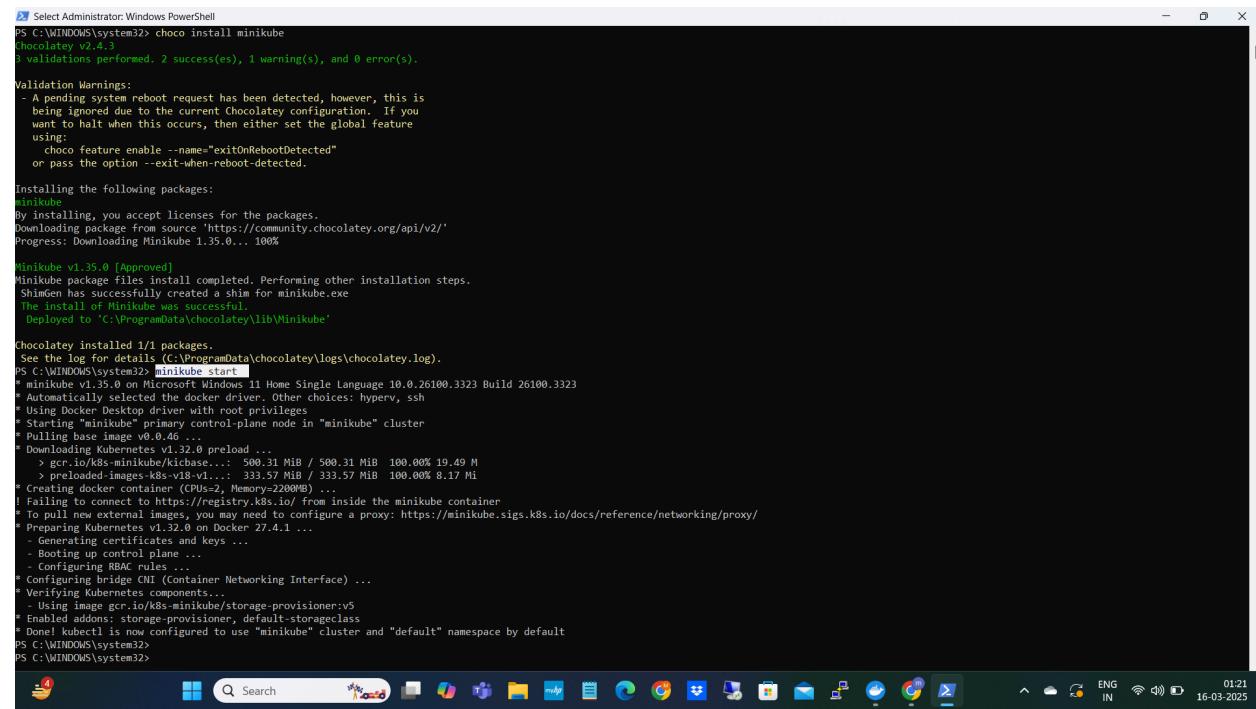
Validation Warnings:
- A pending system reboot request has been detected, however, this is
being ignored due to the current Chocolatey configuration. If you
want to halt when this occurs, then either set the global feature
using:
  choco feature enable --name="exitOnRebootDetected"
or pass the option --exit-when-reboot-detected.

Installing the following packages:
minikube
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading Minikube 1.35.0... 100%
Minikube v1.35.0 [Approved]
Minikube 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'

Chocolatey installed 1/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\WINDOWS\system32>
```

## Start Minikube:

```
$ minikube start
```

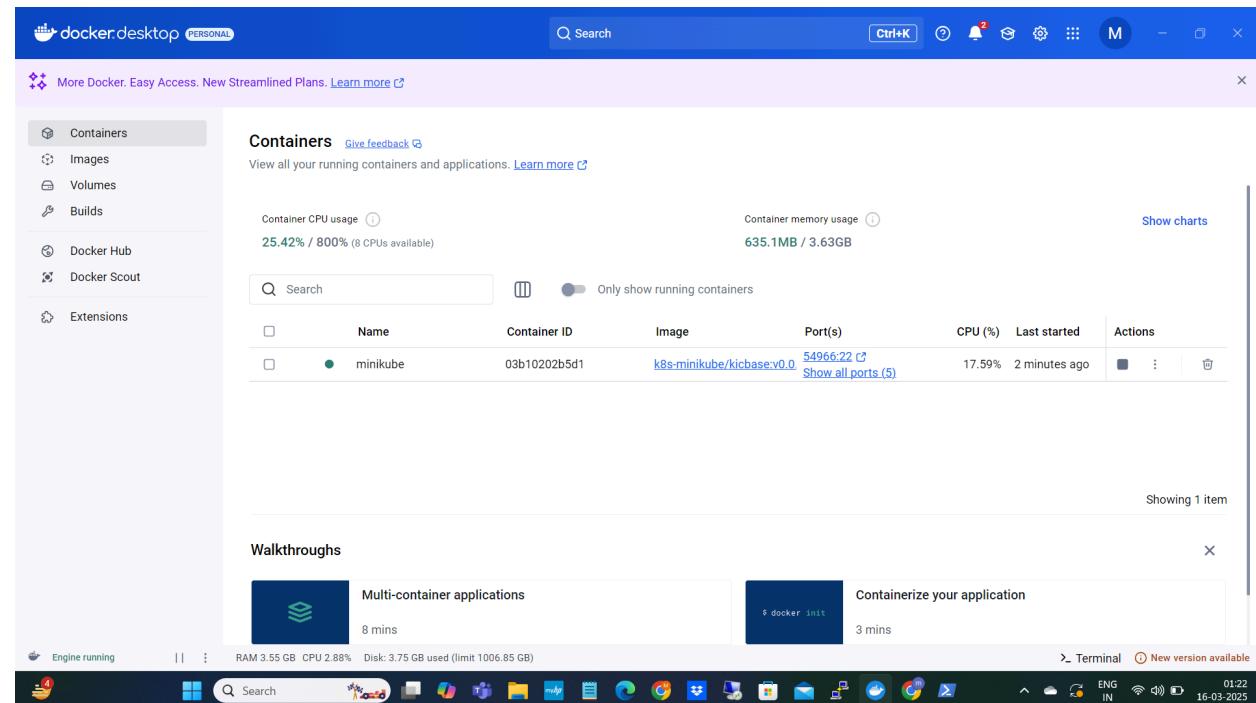


```
Select Administrator: Windows PowerShell
PS C:\WINDOWS\system32> choco install minikube
Chocolatey v2.4.3
3 validations performed. 2 success(es), 1 warning(s), and 0 error(s).

Validation Warnings:
- A pending system reboot request has been detected, however, this is
  being ignored due to the current Chocolatey configuration. If you
  want to halt when this occurs, then either set the global feature
  using:
    choco feature enable --name="exitOnRebootDetected"
  or pass the option --exit-when-reboot-detected.

Installing the following packages:
minikube
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading Minikube 1.35.0... 100%
Minikube v1.35.0 [Approved]
Minikube 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'

Chocolatey installed 1/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\WINDOWS\system32> minikube start
* minikube v1.35.0 on Microsoft Windows 11 Home Single Language 10.0.26100.3323 Build 26100.3323
* Automatically selected the docker driver. Other choices: hyperv, ssh
* Using Docker daemon with root privileges
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.46
* Downloading Kubernetes v1.32.0 prelease ...
  > gcr.io/k8s-minikube/kicbase ...: 500.31 MiB / 500.31 MiB 100.00% 19.49 M
  > preloaded-images-k8s-v18-v1...: 333.57 MiB / 333.57 MiB 100.00% 8.17 Mi
* Creating docker container (CPUs=2, Memory=2200MiB) ...
! 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 v1.32.0 on Docker 27.4.1 ...
- Generating certificates and keys ...
- Booting up control plane ...
  Configuration: RUNTIME
* Configuring bridge CNL (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>
PS C:\WINDOWS\system32>
```



Containers [Give feedback](#)

View all your running containers and applications. [Learn more](#)

Container CPU usage: 25.42% / 800% (8 CPUs available)

Container memory usage: 635.1MB / 3.63GB

Show charts

	Name	Container ID	Image	Port(s)	CPU (%)	Last started	Actions
	minikube	03b10202b5d1	k8s-minikube/kicbase/v0.0.46	54966:22	17.59%	2 minutes ago	

Showing 1 item

Walkthroughs

Multi-container applications [\\$ docker init](#) Containerize your application

Engine running | RAM 3.55 GB CPU 2.88% Disk: 3.75 GB used (limit 1006.85 GB)

Terminal [New version available](#)

## Check info about mini kube cluster if any

\$ kubectl cluster-info

```

Select Administrator: Windows PowerShell
- A pending system reboot request has been detected, however, this is
  being ignored due to the current Chocolatey configuration. If you
  want to halt when this occurs, then either set the global feature
  using:
    choco feature enable --name="exitOnRebootDetected"
  or pass the option --exit-when-reboot-detected.

Installing the following packages:
minikube
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading Minikube 1.35.0... 100%
Minikube v1.35.0 [Approved]
Minikube 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'

Chocolatey installed 1/1 packages.
See the log for details: (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\WINDOWS\system32> minikube start
* minikube v1.35.0 on Microsoft Windows 11 Home Single Language 10.0.26100.3323 Build 26100.3323
* Automatically selected the docker driver. Other choices: hyperv, ssh
* Using Docker Desktop driver with root privileges
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.46 ...
* Downloading Kubernetes v1.32.0 preloaded ...
  > gcr.io/k8s-minikube/kicbase... 500.31 MiB / 500.31 MiB 100.00% 19.49 M
  > preloaded-images-k8s-v18-v1...: 333.57 MiB / 333.57 MiB 100.00% 8.17 Mi
* Creating docker container (CPU=2, Memory=2200MiB) ...
! 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 v1.32.0 on Docker 27.4.1 ...
  - 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> kubectl cluster-info
Kubernetes control plane is running at https://127.0.0.1:54970
CoreDNS is running at https://127.0.0.1:54970/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
PS C:\WINDOWS\system32>
```

## Check the Running nodes:

\$ kubectl get nodes

```

Select Administrator: Windows PowerShell
using:
  choco feature enable --name="exitOnRebootDetected"
  or pass the option --exit-when-reboot-detected.

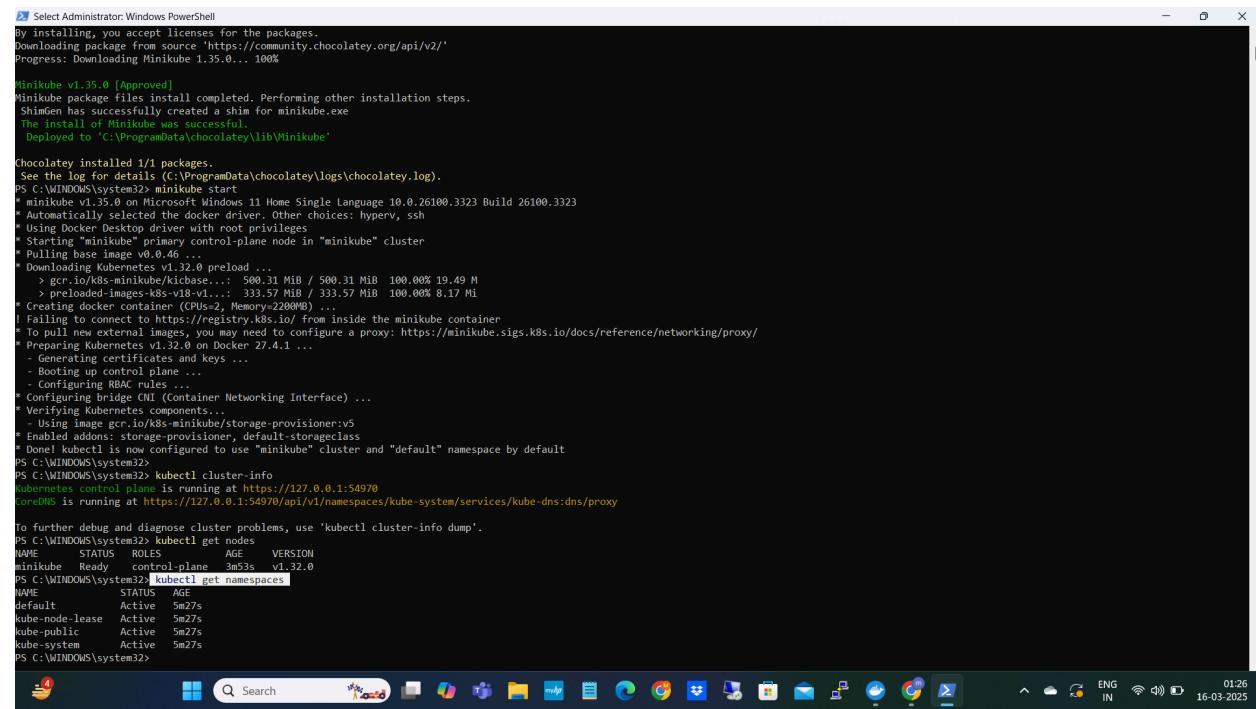
Installing the following packages:
minikube
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading Minikube 1.35.0... 100%
Minikube v1.35.0 [Approved]
Minikube package files install completed. Performing other installation steps.
ShimGen has successfully created a shim for "minikube.exe"
The install of Minikube was successful.
Deployed to 'C:\ProgramData\chocolatey\lib\minikube'

Chocolatey installed 1/1 packages.
See the log for details: (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\WINDOWS\system32> minikube start
* minikube v1.35.0 on Microsoft Windows 11 Home Single Language 10.0.26100.3323 Build 26100.3323
* Automatically selected the docker driver. Other choices: hyperv, ssh
* Using Docker Desktop driver with root privileges
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.46 ...
* Downloading Kubernetes v1.32.0 preloaded ...
  > gcr.io/k8s-minikube/kicbase... 500.31 MiB / 500.31 MiB 100.00% 19.49 M
  > preloaded-images-k8s-v18-v1...: 333.57 MiB / 333.57 MiB 100.00% 8.17 Mi
* Creating docker container (CPU=2, Memory=2200MiB) ...
! 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 v1.32.0 on Docker 27.4.1 ...
  - 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> kubectl cluster-info
Kubernetes control plane is running at https://127.0.0.1:54970
CoreDNS is running at https://127.0.0.1:54970/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
PS C:\WINDOWS\system32> kubectl get nodes
NAME      STATUS   ROLES    AGE     VERSION
minikube  Ready    control-plane   3m53s   v1.32.0
PS C:\WINDOWS\system32>
```

## Check Namespaces:

```
$ kubectl get namespaces
```



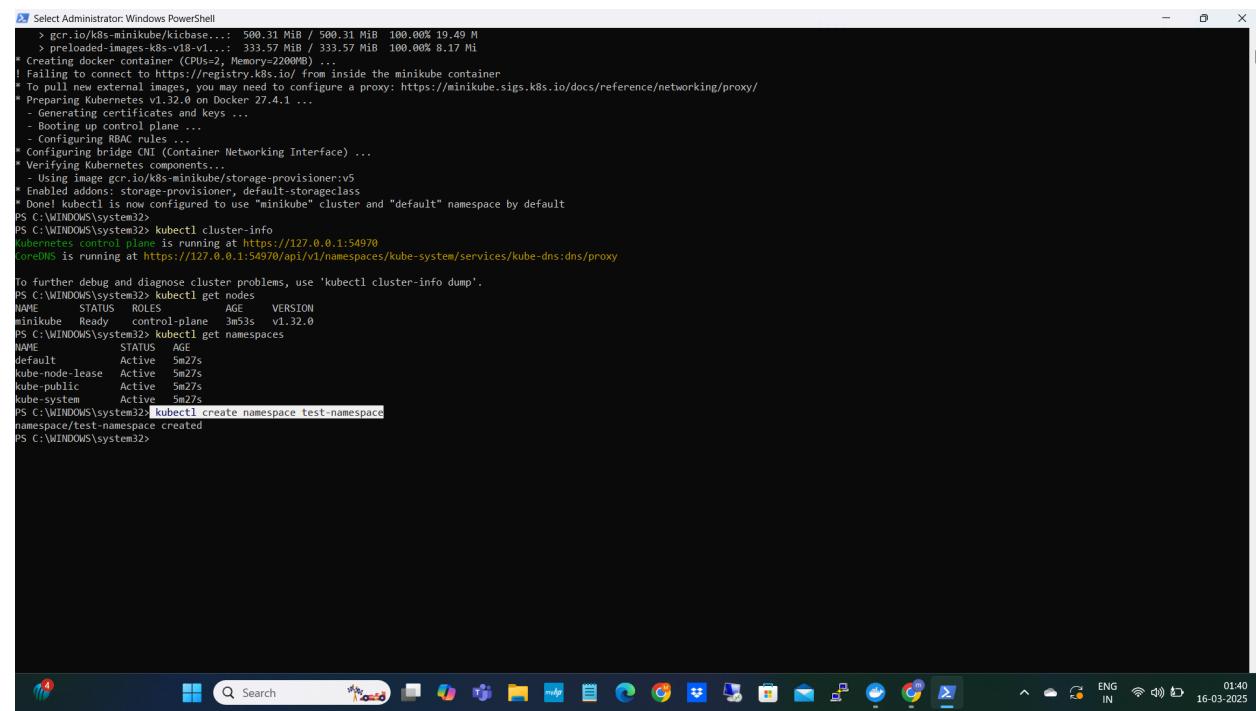
```
Select Administrator: Windows PowerShell
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading Minikube 1.35.0... 100%
Minikube v1.35.0 [Approved]
Minikube package files install completed. Performing other installation steps.
ShimGen has successfully created a shim for minikube.exe
The install of Minikube was successful.
Deployed to 'C:\ProgramData\chocolatey\lib\minikube'

Chocolatey installed 1/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\WINDOWS\system32> minikube start
* minikube v1.35.0 Microsoft Windows 11 Home Single Language 10.0.26100.3323 Build 26100.3323
* Automatically selected the docker driver. Other choices: hyperv, ssh
* Using the User Docker driver with root privileges
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.46
* Downloading Kubernetes v1.32.0 preload ...
  > gcr.io/k8s-minikube/kicbase...: 500.31 MiB / 500.31 MiB 100.00% 19.49 M
  > preloaded-images-k8s-v18-v1...: 333.57 MiB / 333.57 MiB 100.00% 8.17 Mi
* Creating docker container (CPUs=2, Memory=2200MiB) ...
! 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 v1.32.0 on Docker 27.4.1 ...
  - 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> kubectl cluster-info
Kubernetes control plane is running at https://127.0.0.1:54970
CoreDNS is running at https://127.0.0.1:54970/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
PS C:\WINDOWS\system32> kubectl get nodes
NAME      STATUS   ROLES    AGE     VERSION
minikube  Ready    control-plane   3m53s   v1.32.0
PS C:\WINDOWS\system32> kubectl get namespaces
NAME        STATUS   AGE
default     Active   5m27s
kube-node-lease  Active   5m27s
kube-public   Active   5m27s
kube-system   Active   5m27s
PS C:\WINDOWS\system32>
```

## Create a Namespace

```
$ kubectl create namespace test-namespace
```

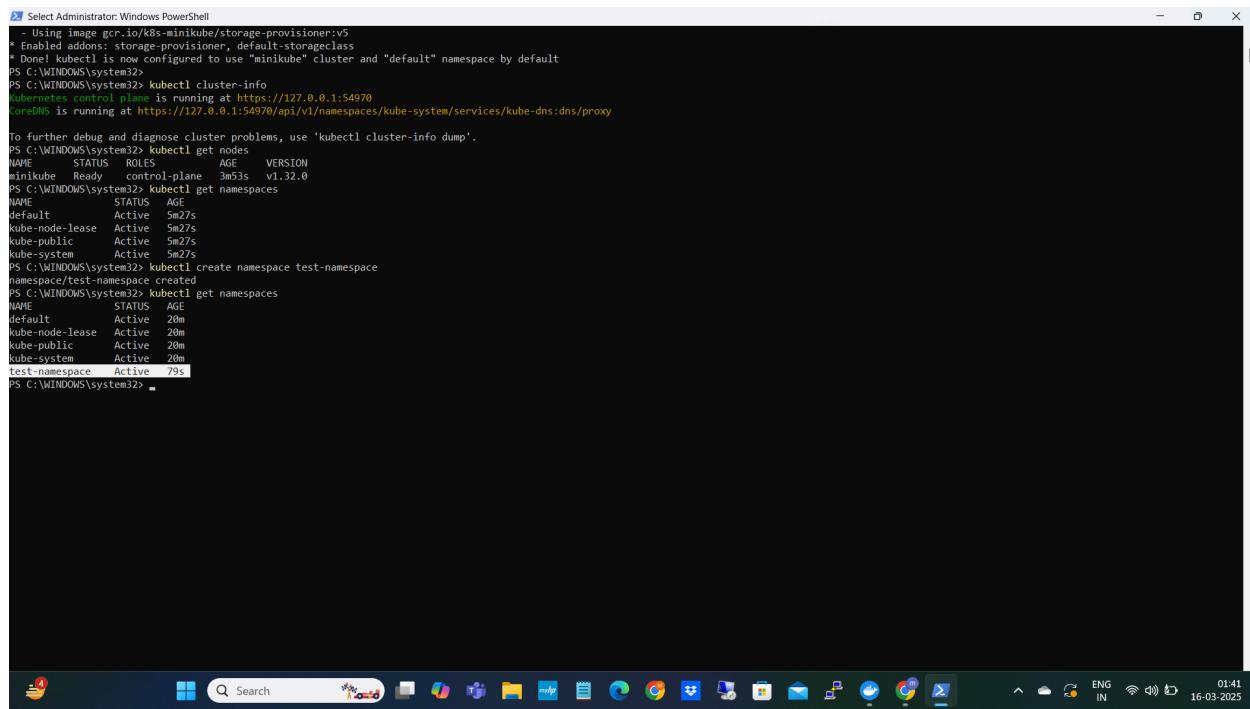


```
Select Administrator: Windows PowerShell
> gcr.io/k8s-minikube/kicbase...: 500.31 MiB / 500.31 MiB 100.00% 19.49 M
> preloaded-images-k8s-v18-v1...: 333.57 MiB / 333.57 MiB 100.00% 8.17 Mi
* Creating docker container (CPUs=2, Memory=2200MiB) ...
! Failing to connect to https://registry.k8s.io/ from 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 v1.32.0 on Docker 27.4.1 ...
  - 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> kubectl cluster-info
Kubernetes control plane is running at https://127.0.0.1:54970
CoreDNS is running at https://127.0.0.1:54970/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
PS C:\WINDOWS\system32> kubectl get nodes
NAME      STATUS   ROLES    AGE     VERSION
minikube  Ready    control-plane   3m53s   v1.32.0
PS C:\WINDOWS\system32> kubectl get namespaces
NAME        STATUS   AGE
default     Active   5m27s
kube-node-lease  Active   5m27s
kube-public   Active   5m27s
kube-system   Active   5m27s
PS C:\WINDOWS\system32> kubectl create namespace test-namespace
namespace/test-namespace created
PS C:\WINDOWS\system32>
```

Check the test-namespace is created

\$ kubectl get namespaces

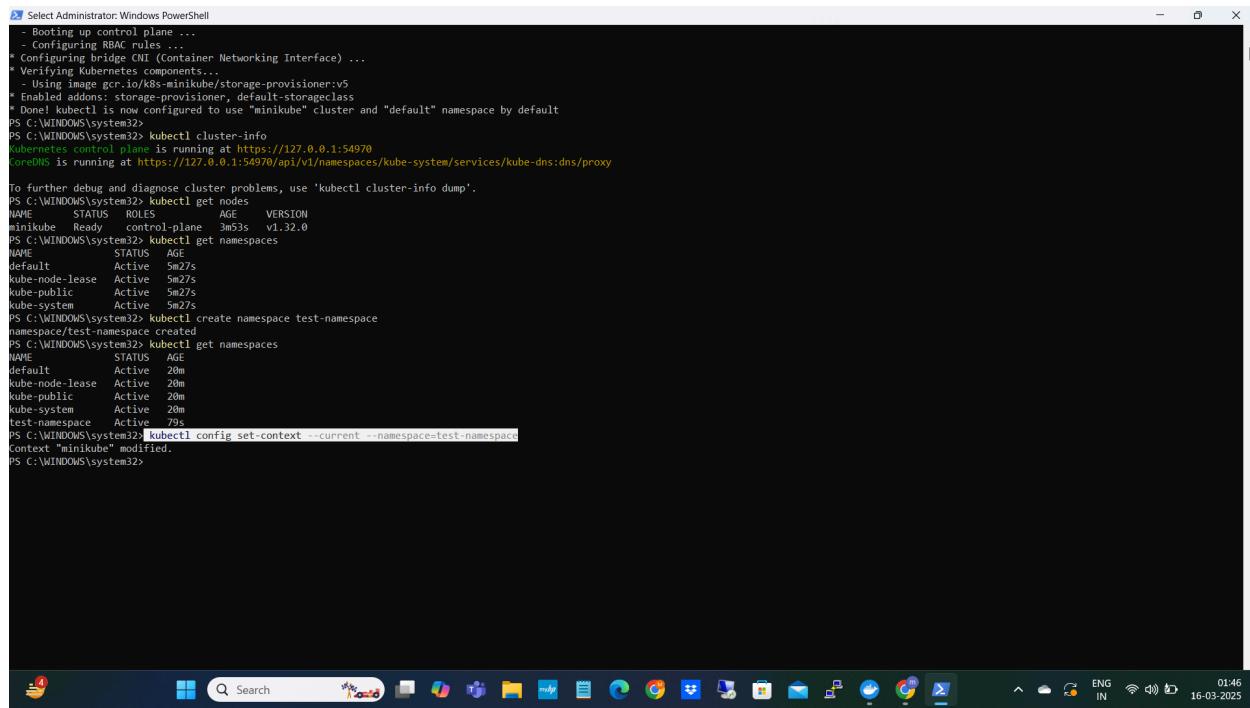


```
PS C:\WINDOWS\system32> kubectl cluster-info
Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Enabled addons: storage-provisioner, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
PS C:\WINDOWS\system32>
PS C:\WINDOWS\system32> kubectl cluster-info
Kubernetes control plane is running at https://127.0.0.1:54970
CoreDNS is running at https://127.0.0.1:54970/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
PS C:\WINDOWS\system32> kubectl get nodes
NAME STATUS ROLES AGE VERSION
minikube Ready control-plane 3m53s v1.32.0
PS C:\WINDOWS\system32> kubectl get namespaces
NAME STATUS AGE
default Active 5m27s
kube-node-lease Active 5m27s
kube-public Active 5m27s
kube-system Active 5m27s
PS C:\WINDOWS\system32> kubectl create namespace test-namespace
namespace/test-namespace created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME STATUS AGE
default Active 20m
kube-node-lease Active 20m
kube-public Active 20m
kube-system Active 20m
test-namespace Active 79s
PS C:\WINDOWS\system32>
```

To use or operate within the test-namespace:

\$ kubectl config set-context --current --namespace=test-namespace



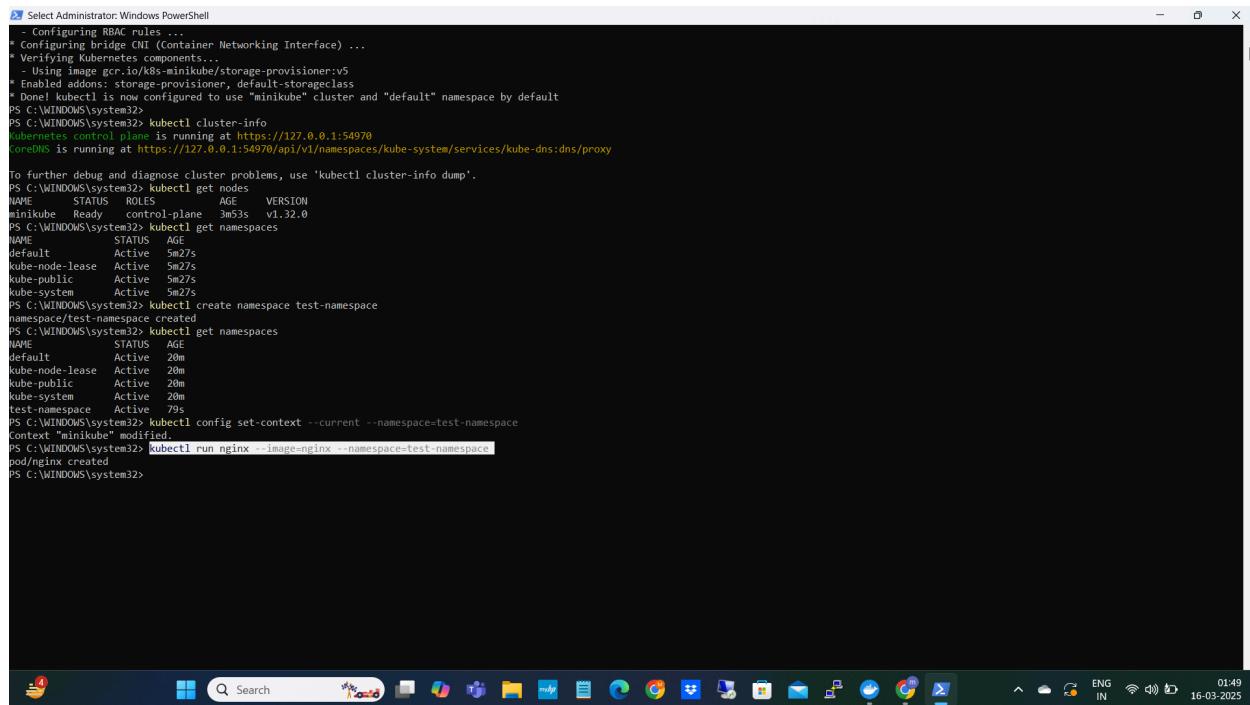
```
PS C:\WINDOWS\system32> kubectl cluster-info
- Booting up control plane ...
- Configuring RBAC roles ...
* Configuring bridge NET (Container Networking Interface) ...
* Verifying Kubernetes components...
- Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Enabled addons: storage-provisioner, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
PS C:\WINDOWS\system32>
PS C:\WINDOWS\system32> kubectl cluster-info
Kubernetes control plane is running at https://127.0.0.1:54970
CoreDNS is running at https://127.0.0.1:54970/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
PS C:\WINDOWS\system32> kubectl get nodes
NAME STATUS ROLES AGE VERSION
minikube Ready control-plane 3m53s v1.32.0
PS C:\WINDOWS\system32> kubectl get namespaces
NAME STATUS AGE
default Active 5m27s
kube-node-lease Active 5m27s
kube-public Active 5m27s
kube-system Active 5m27s
PS C:\WINDOWS\system32> kubectl create namespace test-namespace
namespace/test-namespace created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME STATUS AGE
default Active 20m
kube-node-lease Active 20m
kube-public Active 20m
kube-system Active 20m
test-namespace Active 79s
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=test-namespace
Context "minikube" modified.
PS C:\WINDOWS\system32>
```

## Create Resources inside the test-namespace

Create a pod using nginx image:

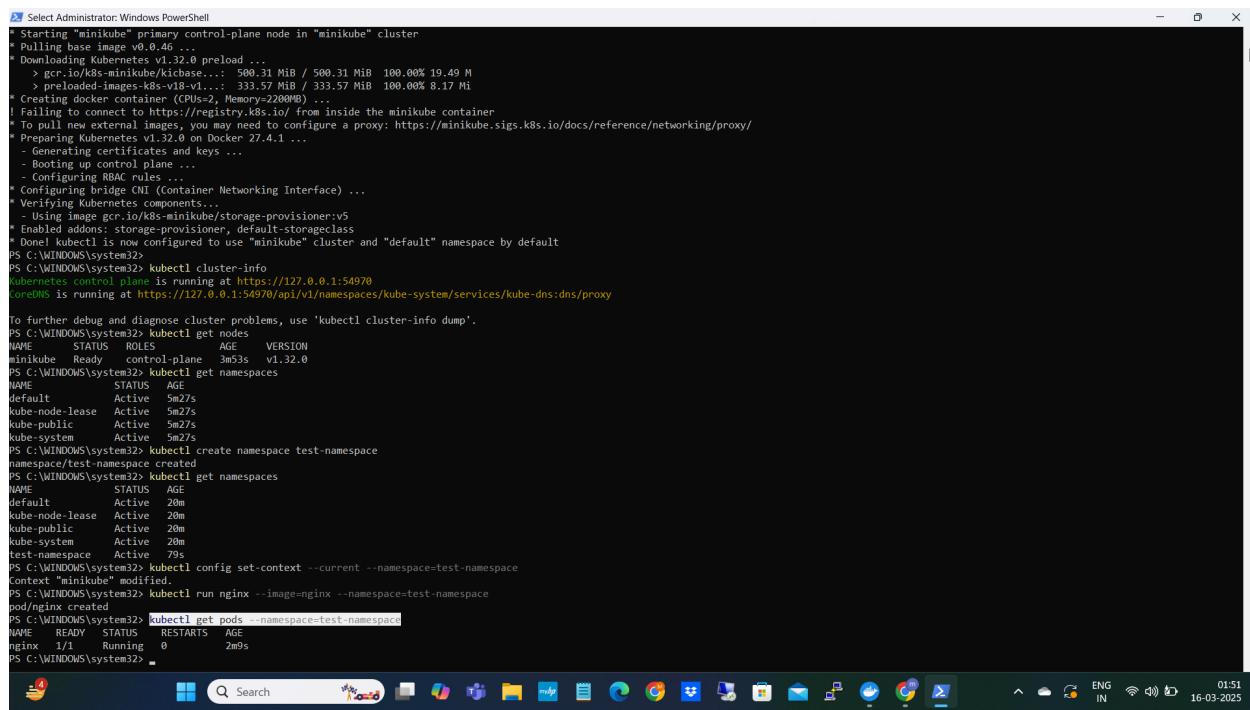
```
$ kubectl run nginx --image=nginx --namespace=test-namespace
```



```
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=test-namespace
Context "minikube" modified.
PS C:\WINDOWS\system32> kubectl run nginx --image=nginx --namespace=test-namespace
pod/nginx created
PS C:\WINDOWS\system32>
```

## Verify the pod creation in the test-namespace

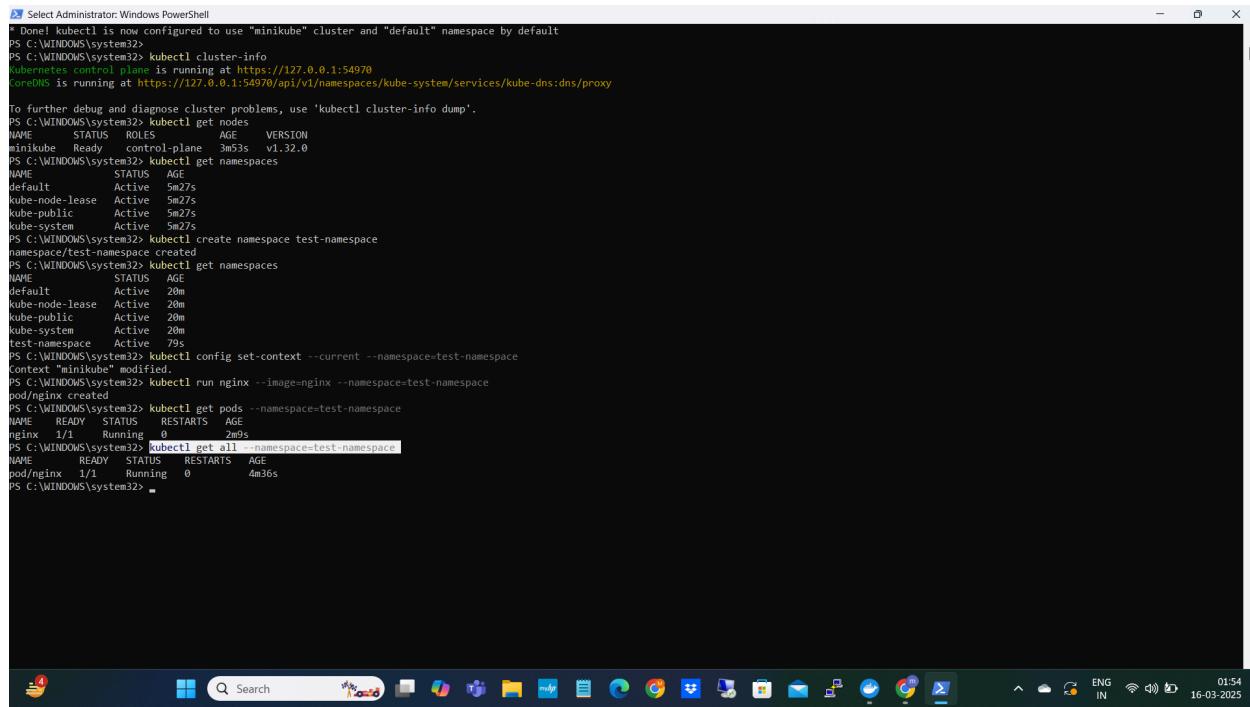
```
$ kubectl get pods --namespace=test-namespace
```



```
PS C:\WINDOWS\system32> kubectl get pods --namespace=test-namespace
NAME      READY   STATUS    RESTARTS   AGE
nginx   1/1     Running   0          2m9s
PS C:\WINDOWS\system32>
```

Nginx pod created inside the test-namespace

List all resources inside a namespace  
\$ kubectl get all --namespace=test-namespace



```

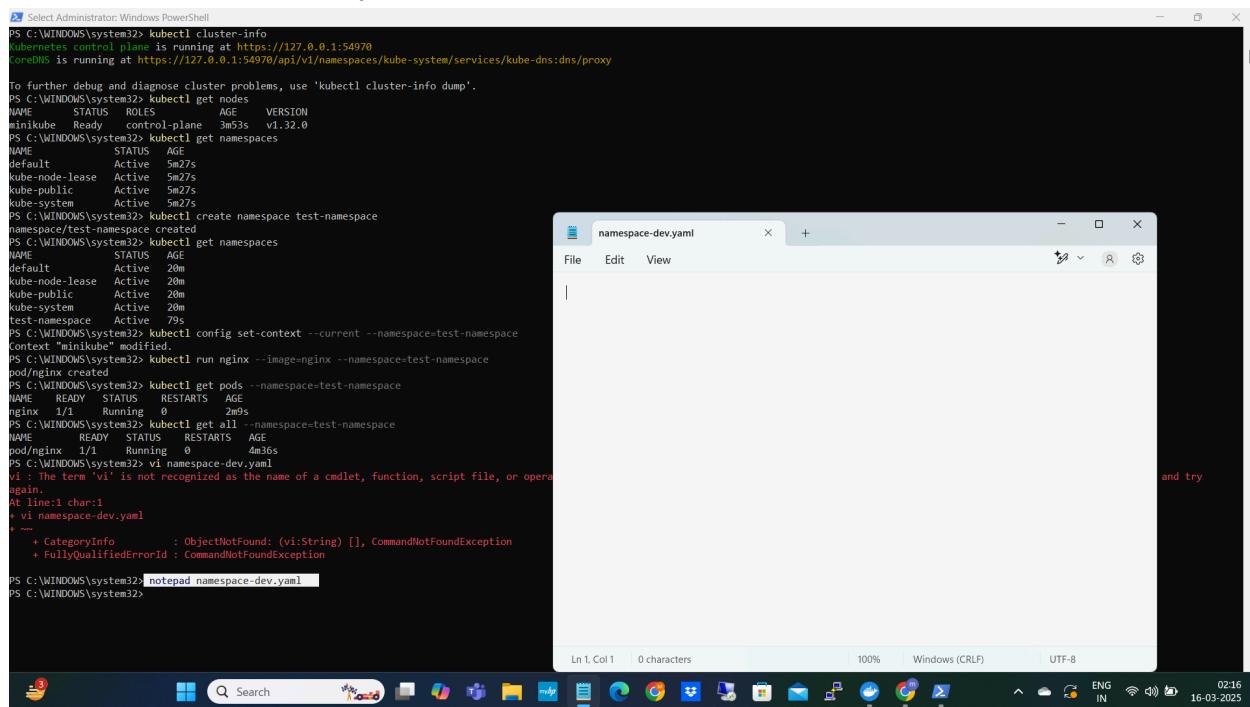
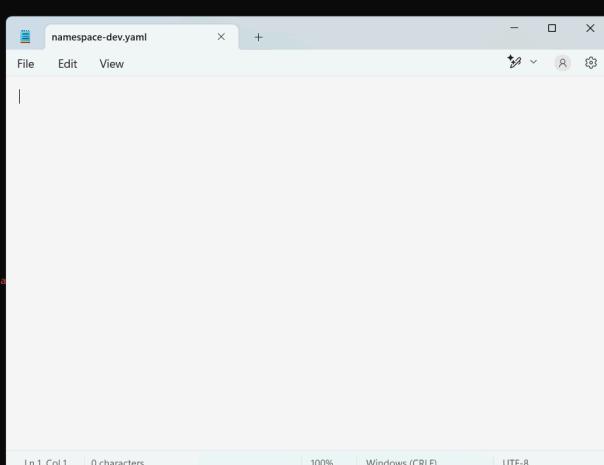
PS C:\WINDOWS\system32> Select Administrator: Windows PowerShell
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
PS C:\WINDOWS\system32> kubectl cluster-info
Kubernetes control plane is running at https://127.0.0.1:54970
CoreDNS is running at https://127.0.0.1:54970/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
PS C:\WINDOWS\system32> kubectl get nodes
NAME STATUS ROLES AGE VERSION
minikube Ready control-plane 3m53s v1.32.0
PS C:\WINDOWS\system32> kubectl get namespaces
NAME STATUS AGE
default Active 5m27s
kube-node-lease Active 5m27s
kube-public Active 5m27s
kube-system Active 5m27s
PS C:\WINDOWS\system32> kubectl create namespace test-namespace
namespace/test-namespace created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME STATUS AGE
default Active 20m
kube-node-lease Active 20m
kube-public Active 20m
kube-system Active 20m
test-namespace Active 79s
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=test-namespace
Context "minikube" modified.
PS C:\WINDOWS\system32> kubectl run nginx --image=nginx --namespace=test-namespace
pod/nginx created
PS C:\WINDOWS\system32> kubectl get pods --namespace=test-namespace
NAME READY STATUS RESTARTS AGE
nginx 1/1 Running 0 2m9s
PS C:\WINDOWS\system32> kubectl get all --namespace=test-namespace
NAME READY STATUS RESTARTS AGE
pod/nginx 1/1 Running 0 4m36s
PS C:\WINDOWS\system32>

```

Create Namespace using yaml file:

Lets create dev and prod name spaces using yaml files  
\$ notepad namespace-dev.yaml

```

PS C:\WINDOWS\system32> Select Administrator: Windows PowerShell
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
PS C:\WINDOWS\system32> kubectl cluster-info
Kubernetes control plane is running at https://127.0.0.1:54970
CoreDNS is running at https://127.0.0.1:54970/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
PS C:\WINDOWS\system32> kubectl get nodes
NAME STATUS ROLES AGE VERSION
minikube Ready control-plane 3m53s v1.32.0
PS C:\WINDOWS\system32> kubectl get namespaces
NAME STATUS AGE
default Active 5m27s
kube-node-lease Active 5m27s
kube-public Active 5m27s
kube-system Active 5m27s
PS C:\WINDOWS\system32> kubectl create namespace test-namespace
namespace/test-namespace created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME STATUS AGE
default Active 20m
kube-node-lease Active 20m
kube-public Active 20m
kube-system Active 20m
test-namespace Active 79s
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=test-namespace
Context "minikube" modified.
PS C:\WINDOWS\system32> kubectl run nginx --image=nginx --namespace=test-namespace
pod/nginx created
PS C:\WINDOWS\system32> kubectl get pods --namespace=test-namespace
NAME READY STATUS RESTARTS AGE
nginx 1/1 Running 0 2m9s
PS C:\WINDOWS\system32> kubectl get all --namespace=test-namespace
NAME READY STATUS RESTARTS AGE
pod/nginx 1/1 Running 0 4m36s
PS C:\WINDOWS\system32> vi namespace-dev.yaml
vi : The term 'vi' is not recognized as the name of a cmdlet, function, script file, or oper
again.
At line:1 char:1
+ vi namespace-dev.yaml
+ ~~~~~
+ CategoryInfo          : ObjectNotFound: (vi:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException

PS C:\WINDOWS\system32> notepad namespace-dev.yaml
PS C:\WINDOWS\system32>

```

```
$ cat namespace-dev.yaml
```

```
Select Administrator: Windows PowerShell
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=test-namespace
Context "minikube" modified.
PS C:\WINDOWS\system32> kubectl run nginx --image=nginx --namespace=test-namespace
pod/nginx created
PS C:\WINDOWS\system32> kubectl get pods --namespace=test-namespace
NAME      READY   STATUS    RESTARTS   AGE
nginx   1/1     Running   0          2m9s
PS C:\WINDOWS\system32> kubectl get all --namespace=test-namespace
NAME      READY   STATUS    RESTARTS   AGE
pod/nginx  1/1     Running   0          4m36s
PS C:\WINDOWS\system32> vi namespace-dev.yaml
vi : The term 'vi' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the spelling of the name, or if a path was included, verify that the path is correct and try again.
At line:1 char:1
+ vi namespace-dev.yaml
+ ~~~
+ CategoryInfo          : ObjectNotFound: (vi:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException

PS C:\WINDOWS\system32> notepad namespace-dev.yaml
PS C:\WINDOWS\system32> cat namespace-dev.yaml
apiVersion: v1
kind: Namespace
metadata:
  name: dev
PS C:\WINDOWS\system32>
```

Executeor apply yaml file to create dev namespace

```
$ kubectl apply -f namespace-dev.yaml
```

```
Select Administrator: Windows PowerShell
apiVersion: v1
kind: Namespace
metadata:
  name: dev
PS C:\WINDOWS\system32> kubectl get namespaces
NAME      STATUS   AGE
default   Active   57m
kube-node-lease   Active   57m
kube-public   Active   57m
kube-system   Active   57m
test-namespace   Active   38m
PS C:\WINDOWS\system32> kubectl apply -f namespace-dev.yaml
error: unknown command "apply" for "kubectl"

Did you mean this?
  apply
PS C:\WINDOWS\system32> kubectl apply -f namespace-dev.yaml
namespace/dev created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME      STATUS   AGE
default   Active   58m
dev      Active   24s
kube-node-lease   Active   58m
kube-public   Active   58m
kube-system   Active   58m
test-namespace   Active   39m
PS C:\WINDOWS\system32>
```

Create namespace prod  
\$ notepad namespace-prod.yaml

Administrator: Windows PowerShell

```
NAME      STATUS  AGE
default   Active  57m
kube-node-lease  Active  57m
kube-public  Active  57m
kube-system  Active  57m
test-namespace Active  38m
PS C:\WINDOWS\system32> kubectl apply -f namespace-dev.yaml
error: unknown command "apply" for "kubectl"

Did you mean this?
    apply
PS C:\WINDOWS\system32> kubectl apply -f namespace-dev.yaml
namespace/dev created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME      STATUS  AGE
default   Active  58m
dev      Active  24s
kube-node-lease  Active  58m
kube-public  Active  58m
kube-system  Active  58m
test-namespace Active  39m
PS C:\WINDOWS\system32> notepad namespace-prod.yaml
PS C:\WINDOWS\system32>
```

File Edit View

namespace-dev.yaml namespace-prod.yaml

Ln 1, Col 1 0 characters 100% Windows (CRLF) UTF-8

02:23 16-03-2025

Administrator: Windows PowerShell

```
NAME      STATUS  AGE
default   Active  57m
kube-node-lease  Active  57m
kube-public  Active  57m
kube-system  Active  57m
test-namespace Active  38m
PS C:\WINDOWS\system32> kubectl apply -f namespace-dev.yaml
error: unknown command "apply" for "kubectl"

Did you mean this?
    apply
PS C:\WINDOWS\system32> kubectl apply -f namespace-dev.yaml
namespace/dev created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME      STATUS  AGE
default   Active  58m
dev      Active  24s
kube-node-lease  Active  58m
kube-public  Active  58m
kube-system  Active  58m
test-namespace Active  39m
PS C:\WINDOWS\system32> notepad namespace-prod.yaml
PS C:\WINDOWS\system32>
```

File Edit View

namespace-dev.yaml namespace-prod.yaml

```
apiVersion: v1
kind: Namespace
metadata:
  name: prod
```

Ln 5, Col 1 54 characters 100% Windows (CRLF) UTF-8

02:24 16-03-2025

```
$ cat namespace-prod.yaml
```

```
PS C:\WINDOWS\system32> Select-Object -Property Name, Status, Age -InputObject (Get-Content .\namespace-dev.yaml | ConvertFrom-Yaml).items[0].spec | Format-Table
PS C:\WINDOWS\system32> kubectl get namespaces
NAME      STATUS   AGE
default   Active   57m
kube-node-lease  Active  57m
kube-public  Active  57m
kube-system  Active  57m
test-namespace  Active  38m
PS C:\WINDOWS\system32> kubectl apply -f namespace-dev.yaml
error: unknown command "apply" for "kubectl"

Did you mean this?
    apply
PS C:\WINDOWS\system32> kubectl apply -f namespace-dev.yaml
namespace/dev created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME      STATUS   AGE
default   Active   58m
dev       Active   24s
kube-node-lease  Active  58m
kube-public  Active  58m
kube-system  Active  58m
test-namespace  Active  39m
PS C:\WINDOWS\system32> notepad namespace-prod.yaml
PS C:\WINDOWS\system32> cat namespace-prod.yaml
apiVersion: v1
kind: Namespace
metadata:
  name: prod
PS C:\WINDOWS\system32>
```

Execute or apply yaml file to create prod namespace

```
$ kubectl apply -f namespace-prod.yaml
```

```
PS C:\WINDOWS\system32> Select-Object -Property Name, Status, Age -InputObject (Get-Content .\namespace-dev.yaml | ConvertFrom-Yaml).items[0].spec | Format-Table
PS C:\WINDOWS\system32> notepad namespace-dev.yaml
PS C:\WINDOWS\system32> cat namespace-dev.yaml
apiVersion: v1
kind: Namespace
metadata:
  name: dev
PS C:\WINDOWS\system32> kubectl get namespaces
NAME      STATUS   AGE
default   Active   57m
kube-node-lease  Active  57m
kube-public  Active  57m
kube-system  Active  57m
test-namespace  Active  38m
PS C:\WINDOWS\system32> kubectl apply -f namespace-dev.yaml
error: unknown command "apply" for "kubectl"

Did you mean this?
    apply
PS C:\WINDOWS\system32> kubectl apply -f namespace-dev.yaml
namespace/dev created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME      STATUS   AGE
default   Active   58m
dev       Active   24s
kube-node-lease  Active  58m
kube-public  Active  58m
kube-system  Active  58m
test-namespace  Active  39m
PS C:\WINDOWS\system32> notepad namespace-prod.yaml
PS C:\WINDOWS\system32> cat namespace-prod.yaml
apiVersion: v1
kind: Namespace
metadata:
  name: prod
PS C:\WINDOWS\system32> kubectl apply -f namespace-prod.yaml
namespace/prod created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME      STATUS   AGE
default   Active   6dm
dev       Active  5m31s
kube-node-lease  Active  6dm
kube-public  Active  6dm
kube-system  Active  6dm
prod      Active   6s
test-namespace  Active  45m
PS C:\WINDOWS\system32>
```

Check both dev and prod namespaces are created  
\$ kubectl get namespaces

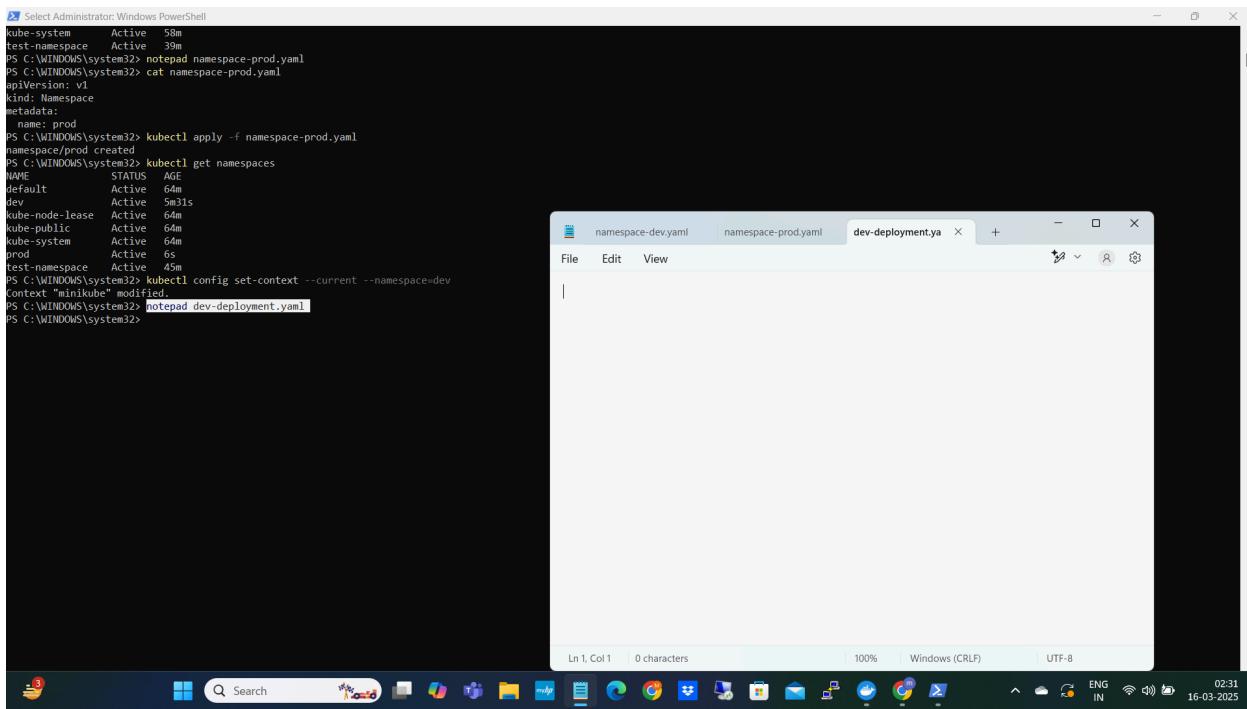
```
PS C:\WINDOWS\system32> notepad namespace-dev.yaml
PS C:\WINDOWS\system32> cat namespace-dev.yaml
apiVersion: v1
kind: Namespace
metadata:
  name: dev
PS C:\WINDOWS\system32> kubectl get namespaces
NAME      STATUS   AGE
default   Active   57m
kube-node-lease  Active  57m
kube-public   Active  57m
kube-system   Active  57m
test-namespace Active  38m
PS C:\WINDOWS\system32> kubectl apply -f namespace-dev.yaml
error: unknown command "apply" for "kubectl"
Did you mean this?
  apply
PS C:\WINDOWS\system32> kubectl apply -f namespace-dev.yaml
namespace/dev created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME      STATUS   AGE
default   Active   58m
dev       Active   24s
kube-node-lease  Active  58m
kube-public   Active  58m
kube-system   Active  58m
test-namespace Active  39m
PS C:\WINDOWS\system32> notepad namespace-prod.yaml
PS C:\WINDOWS\system32> cat namespace-prod.yaml
apiVersion: v1
kind: Namespace
metadata:
  name: prod
PS C:\WINDOWS\system32> kubectl apply -f namespace-prod.yaml
namespace/prod created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME      STATUS   AGE
default   Active   64m
dev       Active   5m31s
kube-node-lease  Active  64m
kube-public   Active  64m
kube-system   Active  64m
prod      Active   6s
test-namespace Active  45m
PS C:\WINDOWS\system32>
```

Set namespace as default to operate with in the namespace  
\$ kubectl config set-context --current --namespace=dev

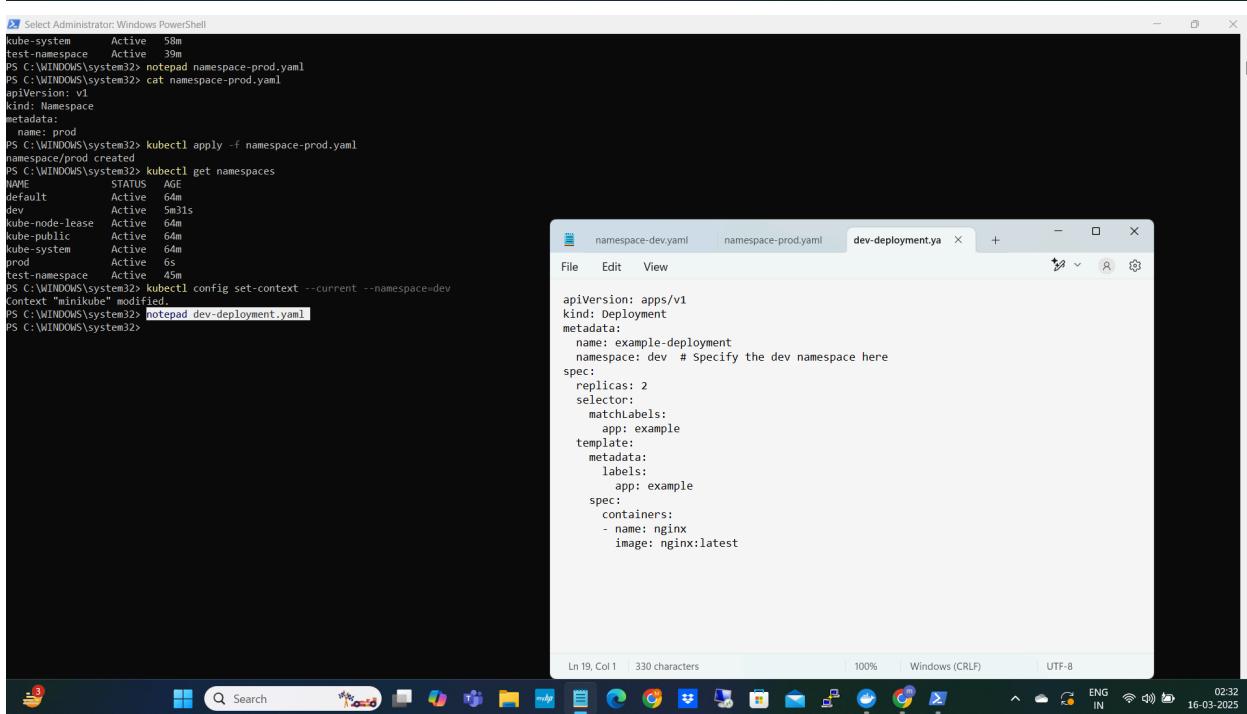
```
PS C:\WINDOWS\system32> notepad namespace-prod.yaml
PS C:\WINDOWS\system32> cat namespace-prod.yaml
apiVersion: v1
kind: Namespace
metadata:
  name: prod
PS C:\WINDOWS\system32> kubectl apply -f namespace-prod.yaml
namespace/prod created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME      STATUS   AGE
default   Active   64m
dev       Active   5m31s
kube-node-lease  Active  64m
kube-public   Active  64m
kube-system   Active  64m
prod      Active   6s
test-namespace Active  45m
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=dev
Context "minikube" modified.
PS C:\WINDOWS\system32>
```

## Create deployment in the dev namespace

\$ notepad dev-deployment.yaml



```
PS C:\WINDOWS\system32> kubectl get namespaces
NAME     STATUS   AGE
default  Active   58m
test-namespace  Active   39m
PS C:\WINDOWS\system32> notepad namespace-prod.yaml
PS C:\WINDOWS\system32> cat namespace-prod.yaml
apiVersion: v1
kind: Namespace
metadata:
  name: prod
PS C:\WINDOWS\system32> kubectl apply -f namespace-prod.yaml
namespace/prod created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME     STATUS   AGE
default  Active   64m
dev      Active   5m31s
kube-node-lease  Active   64m
kube-public  Active   64m
kube-system  Active   64m
prod      Active   6s
test-namespace  Active   45m
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=dev
Context "minikube" modified.
PS C:\WINDOWS\system32> notepad dev-deployment.yaml
PS C:\WINDOWS\system32>
```



```
PS C:\WINDOWS\system32> kubectl get namespaces
NAME     STATUS   AGE
default  Active   58m
test-namespace  Active   39m
PS C:\WINDOWS\system32> notepad namespace-prod.yaml
PS C:\WINDOWS\system32> cat namespace-prod.yaml
apiVersion: v1
kind: Namespace
metadata:
  name: prod
PS C:\WINDOWS\system32> kubectl apply -f namespace-prod.yaml
namespace/prod created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME     STATUS   AGE
default  Active   64m
dev      Active   5m31s
kube-node-lease  Active   64m
kube-public  Active   64m
kube-system  Active   64m
prod      Active   6s
test-namespace  Active   45m
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=dev
Context "minikube" modified.
PS C:\WINDOWS\system32> notepad dev-deployment.yaml
PS C:\WINDOWS\system32>
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: example-deployment
  namespace: dev # Specify the dev namespace here
spec:
  replicas: 2
  selector:
    matchLabels:
      app: example
  template:
    metadata:
      labels:
        app: example
    spec:
      containers:
        - name: nginx
          image: nginx:latest
```

```
$ cat dev-deployment.yaml
```

```
PS C:\WINDOWS\system32> kubectl get namespaces
NAME        STATUS   AGE
default     Active   58m
dev         Active   24s
kube-node-lease  Active  50m
kube-public  Active   58m
kube-system  Active   58m
test-namespace Active  39m
PS C:\WINDOWS\system32> notepad namespace-prod.yaml
PS C:\WINDOWS\system32> cat namespace-prod.yaml
apiVersion: v1
kind: Namespace
metadata:
  name: prod
PS C:\WINDOWS\system32> kubectl apply -f namespace-prod.yaml
namespace/prod created
PS C:\WINDOWS\system32> kubectl get namespaces
NAME        STATUS   AGE
default     Active   64m
dev         Active   5m31s
kube-node-lease  Active  64m
kube-public  Active   64m
kube-system  Active   64m
prod        Active   6s
test-namespace Active  45m
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=dev
Context "minikube" modified.
PS C:\WINDOWS\system32> notepad dev-deployment.yaml
PS C:\WINDOWS\system32> cat dev-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: example-deployment
  namespace: dev # Specify the dev namespace here
spec:
  replicas: 2
  selector:
    matchLabels:
      app: example
  template:
    metadata:
      labels:
        app: example
    spec:
      containers:
        - name: nginx
          image: nginx:latest
PS C:\WINDOWS\system32>
```

Create deployment by applying or executing dev-deployment.yaml file

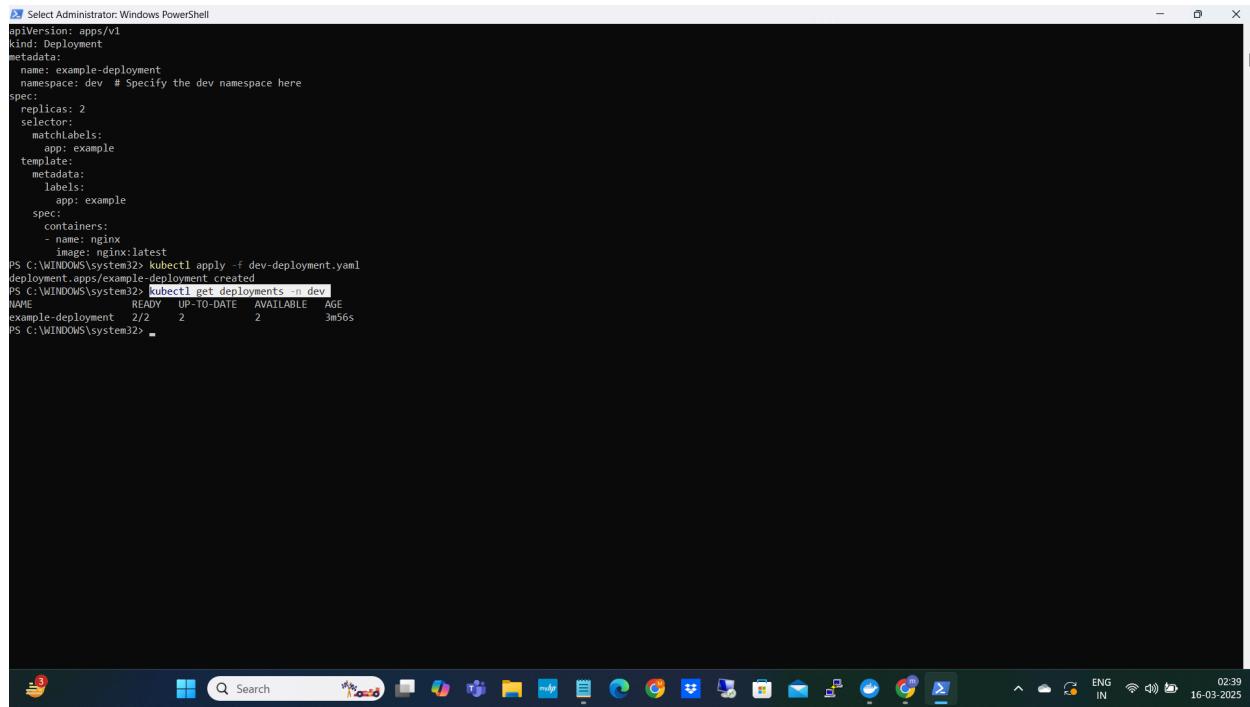
```
$ kubectl apply -f dev-deployment.yaml
```

```
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=dev
Context "minikube" modified.
PS C:\WINDOWS\system32> notepad dev-deployment.yaml
PS C:\WINDOWS\system32> cat dev-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: example-deployment
  namespace: dev # Specify the dev namespace here
spec:
  replicas: 2
  selector:
    matchLabels:
      app: example
  template:
    metadata:
      labels:
        app: example
    spec:
      containers:
        - name: nginx
          image: nginx:latest
PS C:\WINDOWS\system32> kubectl apply -f dev-deployment.yaml
deployment.apps/example-deployment created
PS C:\WINDOWS\system32>
```

This will deploy the nginx pod in the dev namespace

## Verify the deployment in the dev namespace

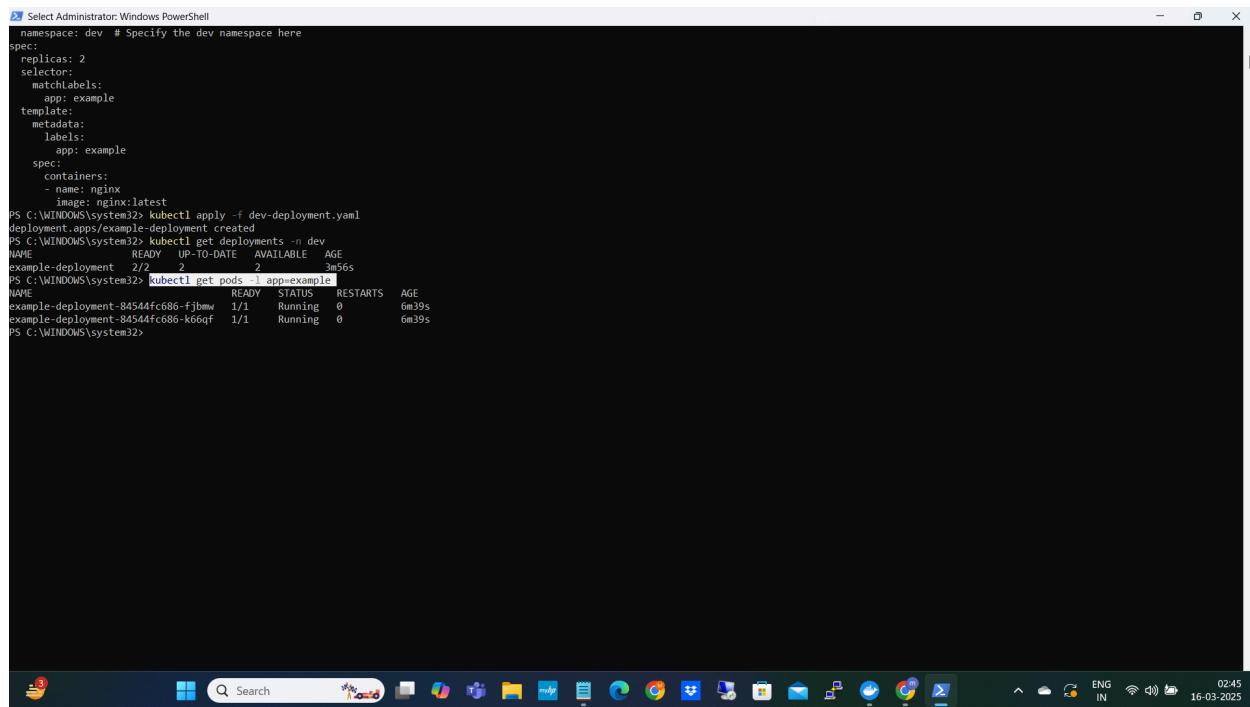
```
$ kubectl get deployments -n dev
```



```
Select Administrator: Windows PowerShell
apiVersion: apps/v1
kind: Deployment
metadata:
  name: example-deployment
  namespace: dev # Specify the dev namespace here
spec:
  replicas: 2
  selector:
    matchLabels:
      app: example
  template:
    metadata:
      labels:
        app: example
    spec:
      containers:
        - name: nginx
          image: nginx:latest
PS C:\WINDOWS\system32> kubectl apply -f dev-deployment.yaml
deployment.apps/example-deployment created
PS C:\WINDOWS\system32> kubectl get deployments -n dev
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
example-deployment 2/2     2           2           3m56s
PS C:\WINDOWS\system32>
```

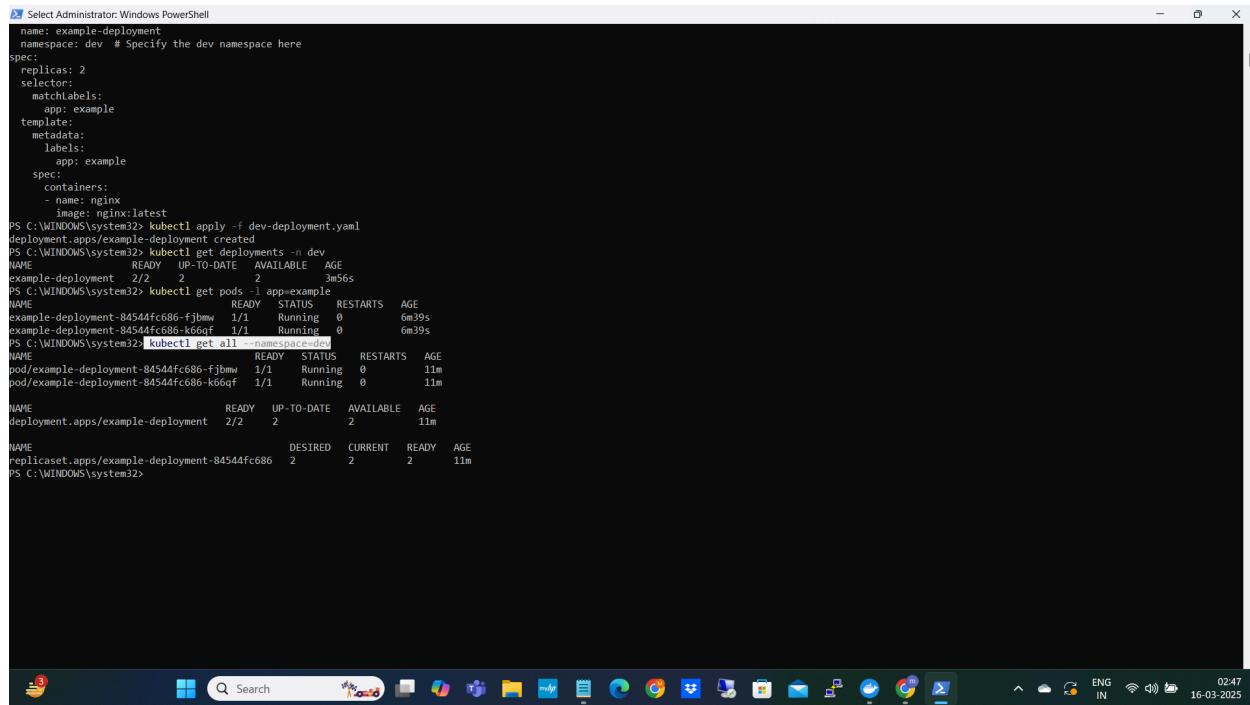
## List all the pods with in the label example

```
$ kubectl get pods -l app=example
```



```
Select Administrator: Windows PowerShell
namespace: dev # Specify the dev namespace here
spec:
  replicas: 2
  selector:
    matchLabels:
      app: example
  template:
    metadata:
      labels:
        app: example
    spec:
      containers:
        - name: nginx
          image: nginx:latest
PS C:\WINDOWS\system32> kubectl apply -f dev-deployment.yaml
deployment.apps/example-deployment created
PS C:\WINDOWS\system32> kubectl get deployments -n dev
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
example-deployment 2/2     2           2           3m56s
PS C:\WINDOWS\system32> kubectl get pods -l app=example
NAME                           READY   STATUS    RESTARTS   AGE
example-deployment-84544fc686-fjbmw 1/1     Running   0          6m39s
example-deployment-84544fc686-k66qf 1/1     Running   0          6m39s
PS C:\WINDOWS\system32>
```

Get all resources inside the dev namespace  
\$ kubectl get all --namespace=dev



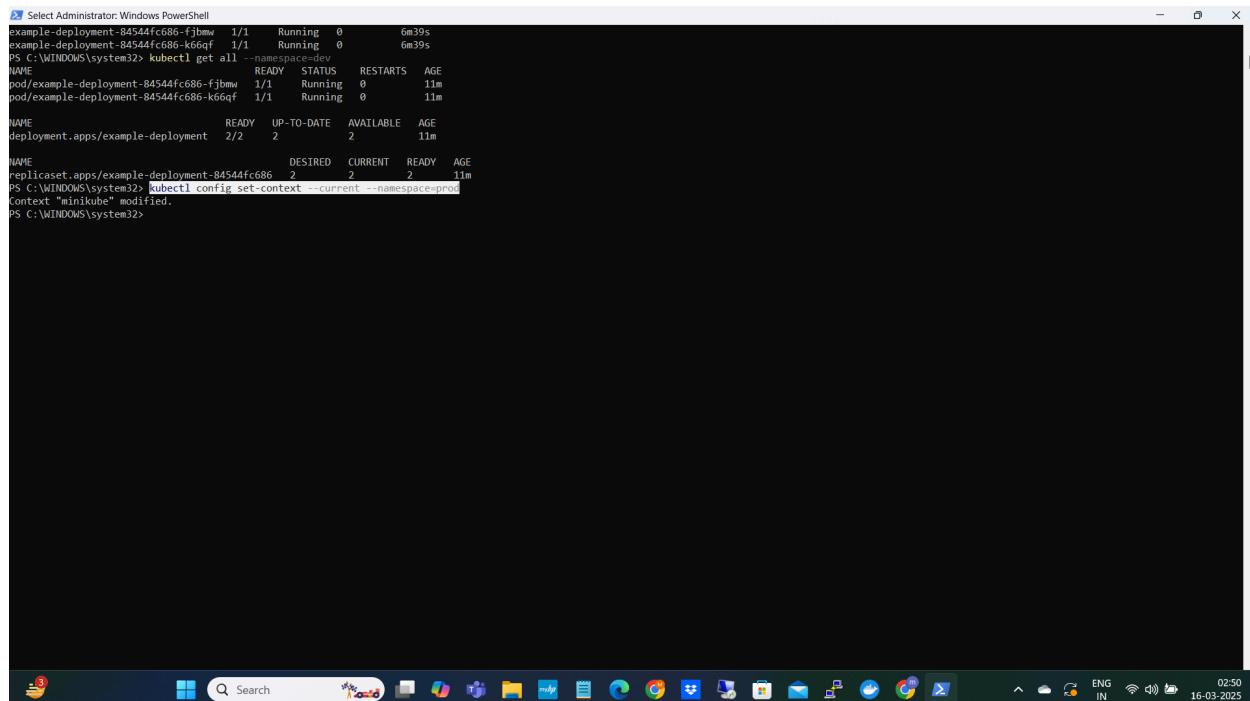
A screenshot of a Windows PowerShell window titled "Select Administrator: Windows PowerShell". The command \$ kubectl get all --namespace=dev is run, displaying the following output:

```
PS C:\WINDOWS\system32> kubectl apply -f dev-deployment.yaml
deployment.apps/example-deployment created
PS C:\WINDOWS\system32> kubectl get deployments -n dev
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
example-deployment   2/2     2           2           3m56s
PS C:\WINDOWS\system32> kubectl get pods -l app=example
NAME           READY   STATUS    RESTARTS   AGE
example-deployment-84544fc686-fjbmw 1/1     Running   0          6m39s
example-deployment-84544fc686-k66qf 1/1     Running   0          6m39s
PS C:\WINDOWS\system32> kubectl get all --namespace=dev
NAME           READY   STATUS    RESTARTS   AGE
pod/example-deployment-84544fc686-fjbmw 1/1     Running   0          11m
pod/example-deployment-84544fc686-k66qf 1/1     Running   0          11m
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
replicaset.apps/example-deployment-84544fc686 2        2           2           11m
PS C:\WINDOWS\system32>
```

The taskbar at the bottom shows various icons for system tools like File Explorer, Task View, and Control Panel.

Here we can see  
Pods, deployments and also replicaset

Let's switch to the prod namespace  
\$ kubectl config set-context --current --namespace=prod



A screenshot of a Windows PowerShell window titled "Select Administrator: Windows PowerShell". The command \$ kubectl config set-context --current --namespace=prod is run, followed by \$ kubectl get all --namespace=prod, displaying the following output:

```
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=prod
Context "minikube" modified.
PS C:\WINDOWS\system32> kubectl get all --namespace=prod
NAME           READY   STATUS    RESTARTS   AGE
pod/example-deployment-84544fc686-fjbmw 1/1     Running   0          11m
pod/example-deployment-84544fc686-k66qf 1/1     Running   0          11m
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
replicaset.apps/example-deployment-84544fc686 2        2           2           11m
PS C:\WINDOWS\system32>
```

The taskbar at the bottom shows various icons for system tools like File Explorer, Task View, and Control Panel.

## Create prod-deployment.yaml file

\$ notepad prod-deployment.yaml

```

app: example
spec:
  containers:
    - name: nginx
      image: nginx:latest
PS C:\WINDOWS\system32> kubectl apply -f dev-deployment.yaml
deployment.apps/example-deployment created
PS C:\WINDOWS\system32> kubectl get deployments -n dev
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
example-deployment   2/2     2          2           3m56s
PS C:\WINDOWS\system32> kubectl get pods -l app=example
NAME           READY   STATUS    RESTARTS   AGE
example-deployment-84544fc686-fjbmw 1/1   Running   0          6m39s
example-deployment-84544fc686-k66qf 1/1   Running   0          6m39s
PS C:\WINDOWS\system32> kubectl get all -n namespace=dev
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
pod/example-deployment-84544fc686-fjbmw 1/1   Running   0          11m
pod/example-deployment-84544fc686-k66qf 1/1   Running   0          11m
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/example-deployment 2/2     2          2           11m
NAME           DESIRED  CURRENT  READY   AGE
replicaset.apps/example-deployment-84544fc686 2        2        2       11m
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=prod
Context "minikube" modified.
PS C:\WINDOWS\system32> notepad prod-deployment.yaml
PS C:\WINDOWS\system32>

```

```

app: example
spec:
  containers:
    - name: nginx
      image: nginx:latest
PS C:\WINDOWS\system32> kubectl apply -f dev-deployment.yaml
deployment.apps/example-deployment created
PS C:\WINDOWS\system32> kubectl get deployments -n dev
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
example-deployment   2/2     2          2           3m56s
PS C:\WINDOWS\system32> kubectl get pods -l app=example
NAME           READY   STATUS    RESTARTS   AGE
example-deployment-84544fc686-fjbmw 1/1   Running   0          6m39s
example-deployment-84544fc686-k66qf 1/1   Running   0          6m39s
PS C:\WINDOWS\system32> kubectl get all -n namespace=dev
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
pod/example-deployment-84544fc686-fjbmw 1/1   Running   0          11m
pod/example-deployment-84544fc686-k66qf 1/1   Running   0          11m
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/example-deployment 2/2     2          2           11m
NAME           DESIRED  CURRENT  READY   AGE
replicaset.apps/example-deployment-84544fc686 2        2        2       11m
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=prod
Context "minikube" modified.
PS C:\WINDOWS\system32> notepad prod-deployment.yaml
PS C:\WINDOWS\system32>

```

```

apiVersion: apps/v1
kind: Deployment
metadata:
  name: production-deployment
  namespace: prod # Specify the prod namespace here
spec:
  replicas: 3 # You can change this to the desired number of replicas for your production environment
  selector:
    matchLabels:
      app: prod-nginx
  template:
    metadata:
      labels:
        app: prod-nginx
    spec:
      containers:
        - name: nginx
          image: nginx:latest # You can replace this with your app's image
          ports:
            - containerPort: 80

```

```
$ cat prod-deployment.yaml
```

```
Select Administrator: Windows PowerShell
  app: example
spec:
  containers:
    - name: nginx
      image: nginx:latest
PS C:\WINDOWS\system32> kubectl apply -f dev-deployment.yaml
deployment.apps/example created
PS C:\WINDOWS\system32> kubectl get deployments -n dev
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
example-deployment  2/2     2           2           3m56s
PS C:\WINDOWS\system32> kubectl get pods -l app=example
NAME          READY   STATUS    RESTARTS   AGE
example-deployment-84544fc686-fjbmw  1/1     Running   0          6m39s
example-deployment-84544fc686-k66qf  1/1     Running   0          6m39s
PS C:\WINDOWS\system32> kubectl get all -n namespace=dev
NAME          READY   STATUS    RESTARTS   AGE
pod/example-deployment-84544fc686-fjbmw  1/1     Running   0          11m
pod/example-deployment-84544fc686-k66qf  1/1     Running   0          11m
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/example-deployment  2/2     2           2           11m
NAME          DESIRED  CURRENT  READY   AGE
replicaset.apps/example-deployment-84544fc686  2        2        2       11m
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=prod
Context "minikube" modified.
PS C:\WINDOWS\system32> notepad prod-deployment.yaml
PS C:\WINDOWS\system32> cat prod-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: production-deployment
  namespace: prod # Specify the prod namespace here
spec:
  replicas: 3 # You can change this to the desired number of replicas for your production environment
  selector:
    matchLabels:
      app: prod-nginx
  template:
    metadata:
      labels:
        app: prod-nginx
    spec:
      containers:
        - name: nginx
          image: nginx:latest # You can replace this with your app's image
          ports:
            - containerPort: 80
PS C:\WINDOWS\system32> _
```



ENG IN 02:55 16-03-2025

To create or apply deployment

```
$ kubectl apply -f prod-deployment.yaml
```

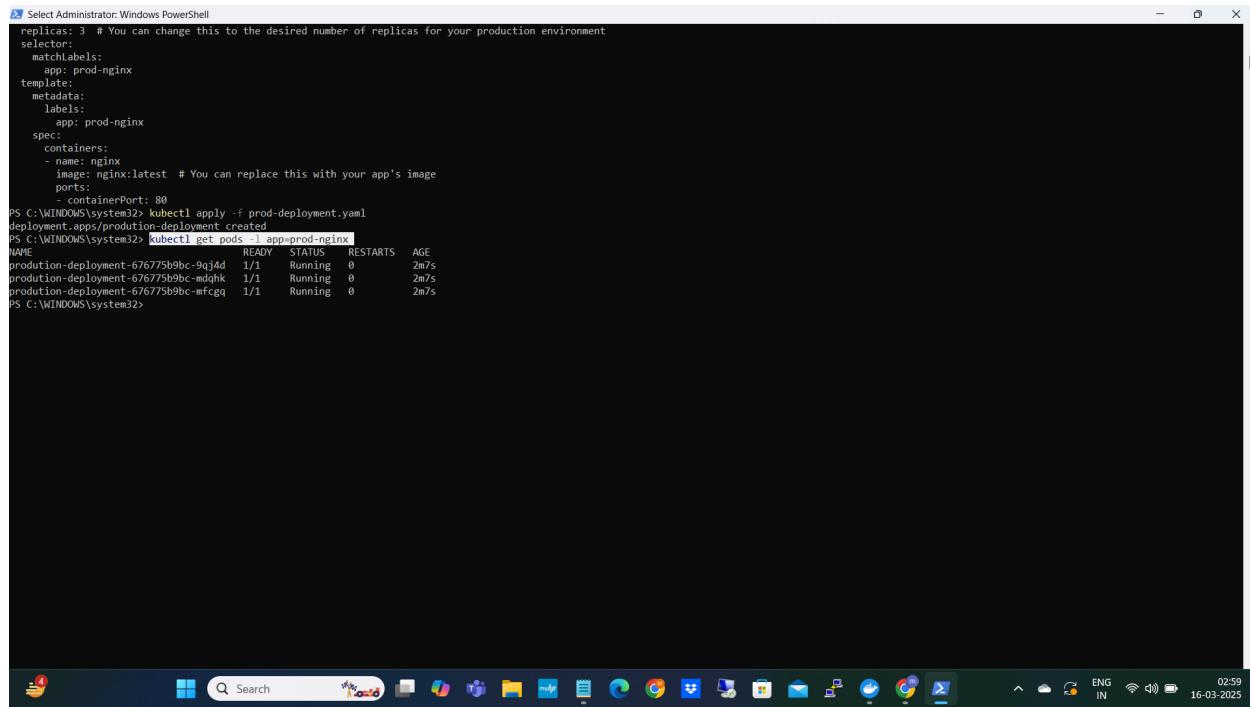
```
Select Administrator: Windows PowerShell
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/example-deployment  2/2     2           2           11m
NAME          DESIRED  CURRENT  READY   AGE
replicaset.apps/example-deployment-84544fc686  2        2        2       11m
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=prod
Context "minikube" modified.
PS C:\WINDOWS\system32> notepad prod-deployment.yaml
PS C:\WINDOWS\system32> cat prod-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: production-deployment
  namespace: prod # Specify the prod namespace here
spec:
  replicas: 3 # You can change this to the desired number of replicas for your production environment
  selector:
    matchLabels:
      app: prod-nginx
  template:
    metadata:
      labels:
        app: prod-nginx
    spec:
      containers:
        - name: nginx
          image: nginx:latest # You can replace this with your app's image
          ports:
            - containerPort: 80
PS C:\WINDOWS\system32> kubectl apply -f prod-deployment.yaml
deployment.apps/production-deployment created
PS C:\WINDOWS\system32> _
```



ENG IN 02:57 16-03-2025

List all the pods with in the label prod-nginx

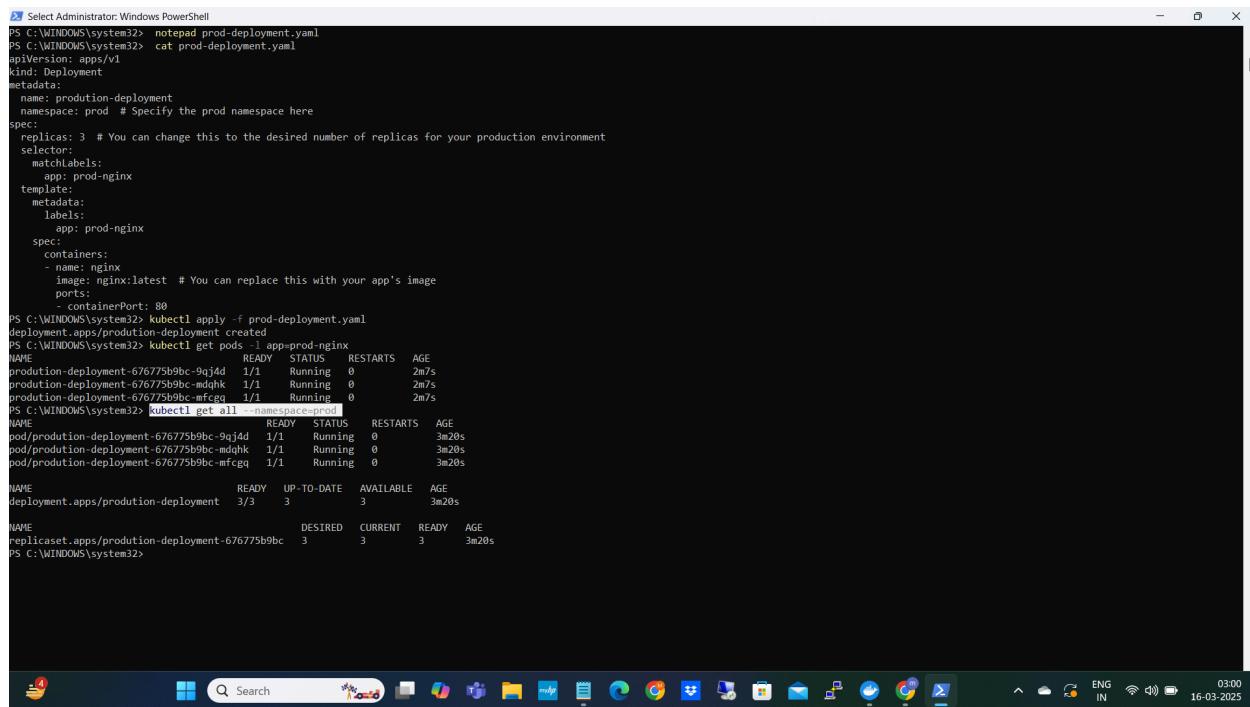
```
$ kubectl get pods -l app=prod-nginx
```



```
PS C:\WINDOWS\system32> kubectl apply -f prod-deployment.yaml
deployment.apps/production-deployment created
PS C:\WINDOWS\system32> kubectl get pods -l app=prod-nginx
NAME           READY   STATUS    RESTARTS   AGE
production-deployment-676775b9bc-9qj4d  1/1     Running   0          2m7s
production-deployment-676775b9bc-mdqhk  1/1     Running   0          2m7s
production-deployment-676775b9bc-mfcgq  1/1     Running   0          2m7s
PS C:\WINDOWS\system32>
```

Get all resources inside the prod namespace

```
kubectl get all --namespace=prod
```



```
PS C:\WINDOWS\system32> notePad prod-deployment.yaml
PS C:\WINDOWS\system32> cat prod-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: production-deployment
  namespace: prod # Specify the prod namespace here
spec:
  replicas: 3 # You can change this to the desired number of replicas for your production environment
  selector:
    matchLabels:
      app: prod-nginx
  template:
    metadata:
      labels:
        app: prod-nginx
    spec:
      containers:
        - name: nginx
          image: nginx:latest # You can replace this with your app's image
          ports:
            - containerPort: 80
PS C:\WINDOWS\system32> kubectl apply -f prod-deployment.yaml
deployment.apps/production-deployment created
PS C:\WINDOWS\system32> kubectl get pods -l app=prod-nginx
NAME           READY   STATUS    RESTARTS   AGE
production-deployment-676775b9bc-9qj4d  1/1     Running   0          2m7s
production-deployment-676775b9bc-mdqhk  1/1     Running   0          2m7s
production-deployment-676775b9bc-mfcgq  1/1     Running   0          2m7s
PS C:\WINDOWS\system32> kubectl get all --namespace=prod
NAME           READY   STATUS    RESTARTS   AGE
pod/production-deployment-676775b9bc-9qj4d  1/1     Running   0          3m20s
pod/production-deployment-676775b9bc-mdqhk  1/1     Running   0          3m20s
pod/production-deployment-676775b9bc-mfcgq  1/1     Running   0          3m20s
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/production-deployment   3/3      3           3           3m20s
NAME           DESIRED  CURRENT    READY     AGE
replicaset.apps/production-deployment-676775b9bc  3        3           3           3m20s
PS C:\WINDOWS\system32>
```

We can see pods, deployment and replica set

List all the namespaces:

\$ kubectl get namespaces

```
Select Administrator: Windows PowerShell
* minikube v1.35.0 on Microsoft Windows 11 Home Single Language 10.0.26100.3323 Build 26100.3323
* Using the docker driver based on existing profile
* Starting minikube binary control-plane node in "minikube" cluster
* Pulling base image: v0.1.0 ...
* Restarting existing docker container for "minikube" ...
! 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 v1.32.0 on Docker 27.4.1 ...
* Verifying Kubernetes components...
- Using image gcr.io/k8s-minikube/storage-provisioner:v5
- 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

* Enabled addons: storage-provisioner, dashboard, default-storageclass
Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
PS C:\WINDOWS\system32> kubectl get all --namespace=prod
NAME           READY   STATUS    RESTARTS   AGE
pod/production-deployment-676775b9bc-9g4d  1/1     Running   1 (11m ago)  35m
pod/production-deployment-676775b9bc-mdghk  0/1     Completed  0          35m
pod/production-deployment-676775b9bc-mfcga  0/1     Completed  0          35m

NAME        TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
service/prod-service  LoadBalancer  10.101.26.127  <pending>      80:32577/TCP  23m

NAME           READY   UP-TO-DATE   CURRENT   READY   AGE
deployment.apps/production-deployment  1/3     3           1         1       35m

NAME           DESIRED   CURRENT   READY   AGE
replicaset.apps/production-deployment-676775b9bc  3         3           1         35m
PS C:\WINDOWS\system32> kubectl get namespaces
NAME          STATUS   AGE
default       Active   132m
dev           Active   74m
kube-node-lease  Active  132m
kube-public    Active   132m
kube-system    Active   132m
kubernetes-dashboard  Active  22m
prod           Active   68m
test-namespace  Active   113m
PS C:\WINDOWS\system32>
```

To delete name spaces:

\$ kubectl delete namespace prod

```
Select Administrator: Windows PowerShell
- 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

* Enabled addons: storage-provisioner, dashboard, default-storageclass
Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
PS C:\WINDOWS\system32> kubectl get all --namespace=prod
NAME           READY   STATUS    RESTARTS   AGE
pod/production-deployment-676775b9bc-9g4d  1/1     Running   1 (11m ago)  35m
pod/production-deployment-676775b9bc-mdghk  0/1     Completed  0          35m
pod/production-deployment-676775b9bc-mfcga  0/1     Completed  0          35m

NAME        TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
service/prod-service  LoadBalancer  10.101.26.127  <pending>      80:32577/TCP  23m

NAME           READY   UP-TO-DATE   CURRENT   READY   AGE
deployment.apps/production-deployment  1/3     3           1         1       35m

NAME           DESIRED   CURRENT   READY   AGE
replicaset.apps/production-deployment-676775b9bc  3         3           1         35m
PS C:\WINDOWS\system32> kubectl get namespaces
NAME          STATUS   AGE
default       Active   132m
dev           Active   74m
kube-node-lease  Active  132m
kube-public    Active   132m
kube-system    Active   132m
kubernetes-dashboard  Active  22m
prod           Active   68m
test-namespace  Active   113m
PS C:\WINDOWS\system32> kubectl delete namespace prod
namespace "prod" deleted
PS C:\WINDOWS\system32>
```

## List all the namespaces ‘prod-namespace’ is deleted

```
PS C:\WINDOWS\system32> Select-Object -Property Name, Status, Age -InputObject $(Get-Service -Name kubelet).Statuses | Where-Object { $_.Name -eq "Kubelet" } | Format-Table
PS C:\WINDOWS\system32> kubectl get all --field-selector metadata.namespace=prod
NAME                                     READY   STATUS    RESTARTS   AGE
pod/production-deployment-676775b9bc-9qj4d   1/1    Running   1 (11m ago)   35m
pod/production-deployment-676775b9bc-mdgk   0/1    Completed   0   35m
pod/production-deployment-676775b9bc-mfcgq  0/1    Completed   0   35m

NAME          TYPE      CLUSTER-IP     EXTERNAL-IP   PORT(S)   AGE
service/prod-service LoadBalancer  10.101.26.127 <pending>   80:32577/TCP 23m

NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/production-deployment   1/3     3           1           35m

NAME          DESIRED  CURRENT   READY   AGE
replicaset.apps/production-deployment-676775b9bc 3         3           1           35m
PS C:\WINDOWS\system32> kubectl get namespaces
NAME          STATUS   AGE
default       Active   132m
dev          Active   74m
kube-node-lease  Active  132m
kube-public    Active  132m
kube-system    Active  132m
kubernetes-dashboard  Active  22m
prod          Active  68m
test-namespace Active  113m
PS C:\WINDOWS\system32> kubectl delete namespace prod
namespace "prod" deleted
PS C:\WINDOWS\system32> kubectl get namespaces
NAME          STATUS   AGE
default       Active   136m
dev          Active   77m
kube-node-lease  Active  136m
kube-public    Active  136m
kube-system    Active  136m
kubernetes-dashboard  Active  26m
test-namespace Active  117m
PS C:\WINDOWS\system32>
```

## Switch to dev namespace and check the dashboard

```
PS C:\WINDOWS\system32> Select-Object -Property Name, Status, Age -InputObject $(Get-Service -Name kubelet).Statuses | Where-Object { $_.Name -eq "Kubelet" } | Format-Table
PS C:\WINDOWS\system32> kubectl get all --field-selector metadata.namespace=prod
NAME                                     READY   STATUS    RESTARTS   AGE
pod/production-deployment-676775b9bc-mdgk   0/1    Completed   0   35m
pod/production-deployment-676775b9bc-mfcgq  0/1    Completed   0   35m

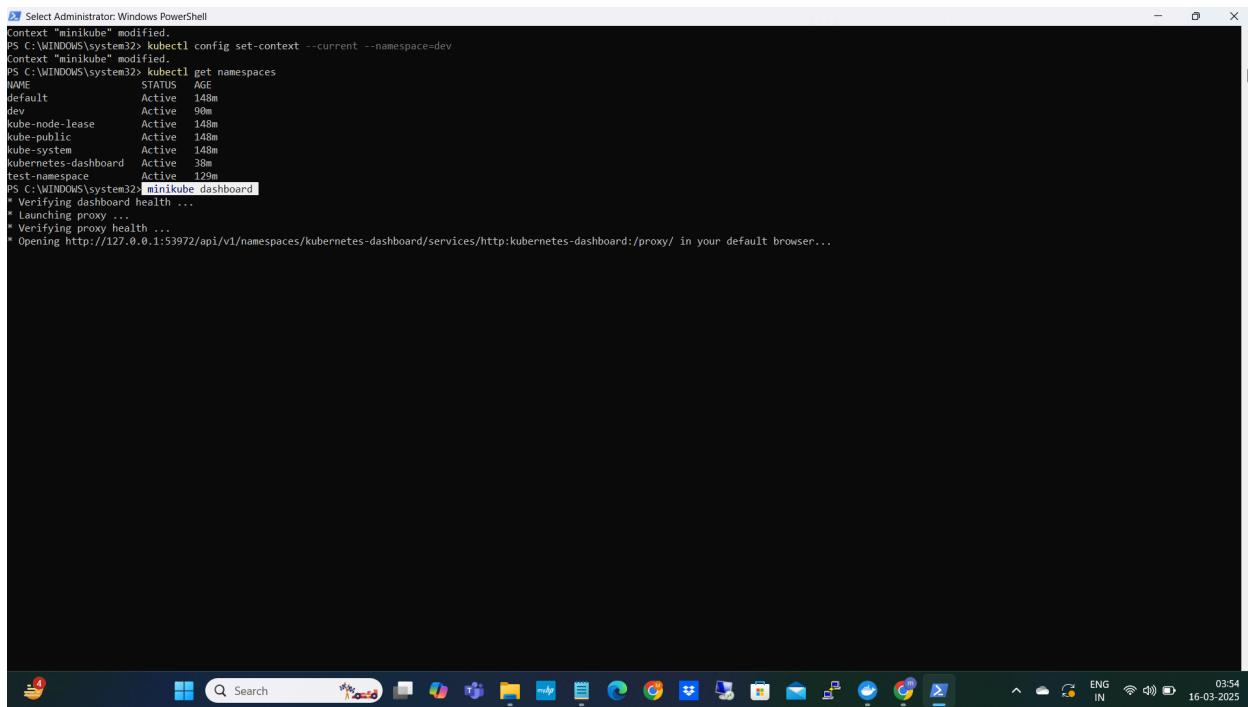
NAME          TYPE      CLUSTER-IP     EXTERNAL-IP   PORT(S)   AGE
service/prod-service LoadBalancer  10.101.26.127 <pending>   80:32577/TCP 23m

NAME          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/production-deployment   1/3     3           1           35m

NAME          DESIRED  CURRENT   READY   AGE
replicaset.apps/production-deployment-676775b9bc 3         3           1           35m
PS C:\WINDOWS\system32> kubectl get namespaces
NAME          STATUS   AGE
default       Active   132m
dev          Active   74m
kube-node-lease  Active  132m
kube-public    Active  132m
kube-system    Active  132m
kubernetes-dashboard  Active  22m
prod          Active  68m
test-namespace Active  113m
PS C:\WINDOWS\system32> kubectl delete namespace prod
namespace "prod" deleted
PS C:\WINDOWS\system32> kubectl get namespaces
NAME          STATUS   AGE
default       Active   136m
dev          Active   77m
kube-node-lease  Active  136m
kube-public    Active  136m
kube-system    Active  136m
kubernetes-dashboard  Active  26m
test-namespace Active  117m
PS C:\WINDOWS\system32> minikube dashboard
* Verifying dashboard health ...
* Launching proxy ...
* Verifying proxy health ...
* Opening http://127.0.0.1:64371/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=my-namespace
Context "minikube" modified.
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=dev
Context "minikube" modified.
PS C:\WINDOWS\system32> kubectl get namespaces
NAME          STATUS   AGE
default       Active   148m
dev          Active   90m
kube-node-lease  Active  148m
kube-public    Active  148m
kube-system    Active  148m
kubernetes-dashboard  Active  38m
PS C:\WINDOWS\system32>
```

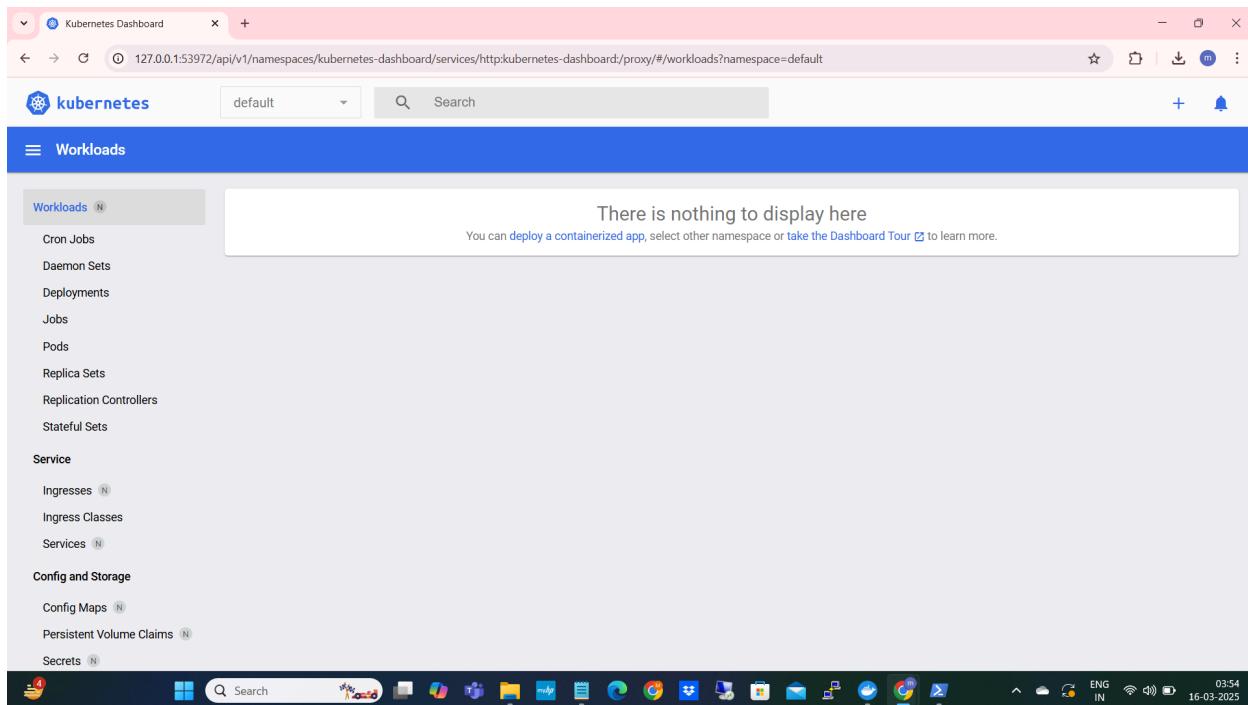
## Minikube dashboard launch

```
$ minikube dashboard
```



```
Select Administrator: Windows PowerShell
Context "minikube" modified.
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=dev
Context "minikube" modified.
PS C:\WINDOWS\system32> kubectl get namespaces
NAME          STATUS   AGE
default       Active   148m
dev           Active   99m
kube-node-lease   Active   148m
kube-public    Active   148m
kube-system    Active   148m
kubernetes-dashboard   Active   38m
test-namespace  Active   129m
PS C:\WINDOWS\system32> minikube dashboard
* Verifying dashboard health ...
* Launching proxy ...
* Verifying proxy health ...
Opening http://127.0.0.1:53972/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
```

It can launch kubernetes dashboard



Kubernetes Dashboard

127.0.0.1:53972/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/#/workloads?namespace=default

kubernetes

default

Workloads

Workloads

Cron Jobs

Daemon Sets

Deployments

Jobs

Pods

Replica Sets

Replication Controllers

Stateful Sets

Service

Ingresses

Ingress Classes

Services

Config and Storage

Config Maps

Persistent Volume Claims

Secrets

Select the **dev** namespace:

The screenshot shows the Kubernetes Dashboard interface. On the left, a sidebar menu is open under the 'Workloads' section. A dropdown menu titled 'NAMESPACES' is displayed, listing several namespaces: default, dev, kube-node-lease, kube-public, and kube-system. The 'dev' namespace is currently selected. The main content area displays a message: 'There is nothing to display here. You can [deploy a containerized app](#), select other namespace or [take the Dashboard Tour](#) to learn more.'

The screenshot shows the Kubernetes Dashboard interface with the 'dev' namespace selected. The main content area is titled 'Workload Status' and displays three large green circles representing the status of Deployments, Pods, and Replica Sets, all showing a value of 1. Below this, the 'Deployments' section is expanded, showing a table with one row:

Name	Images	Labels	Pods	Created
example-deployment	nginx:latest	-	2 / 2	an hour ago

Here we can see the resources inside the **dev** namespace:

The screenshot shows the Kubernetes Dashboard interface with the namespace set to **dev**. The main area displays three sections: **Deployments**, **Pods**, and **Replica Sets**. In the **Deployments** section, there is one entry: **example-deployment** with the image **nginx:latest**. In the **Pods** section, there are two pods: **example-deployment-84544fc686-fjbmw** and **example-deployment-84544fc686-k66qf**, both running on the node **minikube**. In the **Replica Sets** section, there is one entry: **example-deployment-84544fc686** with the image **nginx:latest**. The sidebar on the left lists various resource types: Workloads, Cron Jobs, Daemon Sets, Deployments, Jobs, Pods, Replica Sets, Replication Controllers, Stateful Sets, Service, Ingresses, Ingress Classes, Services, Config and Storage, Config Maps, Persistent Volume Claims, and Secrets.

Select **test-namespace**:

The screenshot shows the same Kubernetes Dashboard interface, but the namespace dropdown in the top right has been opened, revealing a list of namespaces: **kube-node-lease**, **kube-public**, **kube-system**, **kubernetes-dashboard**, and **test-namespace**. The **test-namespace** option is highlighted. The main content area shows the same deployment and pod information as the previous screenshot, but the sidebar and other details remain consistent with the first screenshot.

Here we can see the resources with in it

The screenshot shows the Kubernetes Dashboard interface. In the top left, there's a sidebar with categories like Workloads, Service, Config and Storage, and more. The main area is titled "Workload Status" and shows a large green circle indicating "Running: 1". Below this is a table titled "Pods" with one entry:

Name	Images	Labels	Node	Status	Restarts	CPU Usage (cores)	Memory Usage (bytes)	Created
nginx	nginx	run: nginx	minikube	Running	1	-	-	2.hours.ago

Stop minikube

\$ minikube stop

```
PS C:\WINDOWS\system32> kubectl delete namespace prod
namespace "prod" deleted
PS C:\WINDOWS\system32> kubectl get namespaces
NAME          STATUS   AGE
default       Active   132m
dev           Active   74m
kube-node-lease  Active  132m
kube-public    Active  132m
kube-system    Active  132m
kubernetes-dashboard  Active  22m
prod          Active   68m
test-namespace  Active  113m
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=my-namespace
Context "minikube" modified.
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=dev
Context "minikube" modified.
PS C:\WINDOWS\system32> kubectl get namespaces
NAME          STATUS   AGE
default       Active   148m
dev           Active   98m
kube-node-lease  Active  148m
kube-public    Active  148m
kube-system    Active  148m
kubernetes-dashboard  Active  38m
test-namespace  Active  129m
PS C:\WINDOWS\system32> minikube dashboard
* Verifying dashboard health ...
* Launching proxy ...
* Verifying proxy health ...
* Opening http://127.0.0.1:53972/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
PS C:\WINDOWS\system32> kubectl config set-context --current --namespace=minikube
Context "minikube" modified.
PS C:\WINDOWS\system32> minikube stop
* Stopping node "minikube" ...
* Powering off "minikube" via SSH ...
* 1 node stopped.
PS C:\WINDOWS\system32>
```



### This site can't be reached

127.0.0.1 refused to connect.

Try:

- Checking the connection
- [Checking the proxy and the firewall](#)

ERR\_CONNECTION\_REFUSED

[Reload](#)

[Details](#)



As minikube stopped dashboard is unreachable

Docker container also stopped

Name	Container ID	Image	Port(s)	CPU (%)	Last started	Actions
minikube	03b10202b5d1	k8s-minikube/kicbase:v0.0	0.22 <a href="#">Show all ports (5)</a>	N/A	29 minutes ago	<a href="#">D</a> <a href="#">⋮</a> <a href="#">X</a>

Showing 1 item

**Walkthroughs**

Multi-container applications      Containerize your application

8 mins      3 mins

Engine running      RAM 2.09 GB CPU 0.00% Disk: 4.18 GB used (limit 1006.85 GB)      Terminal      New version available

