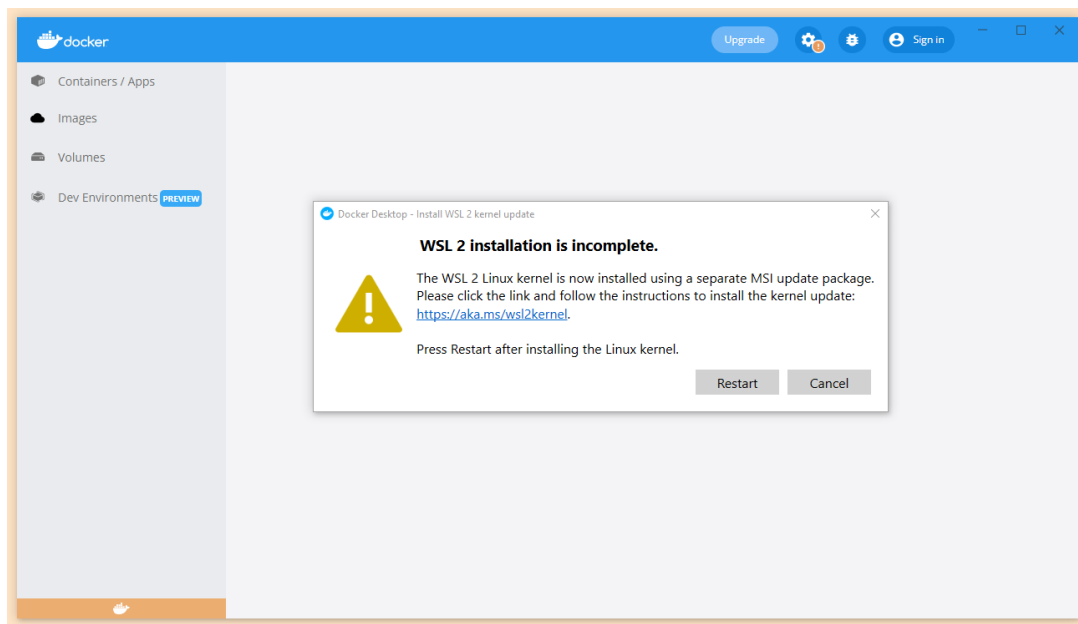
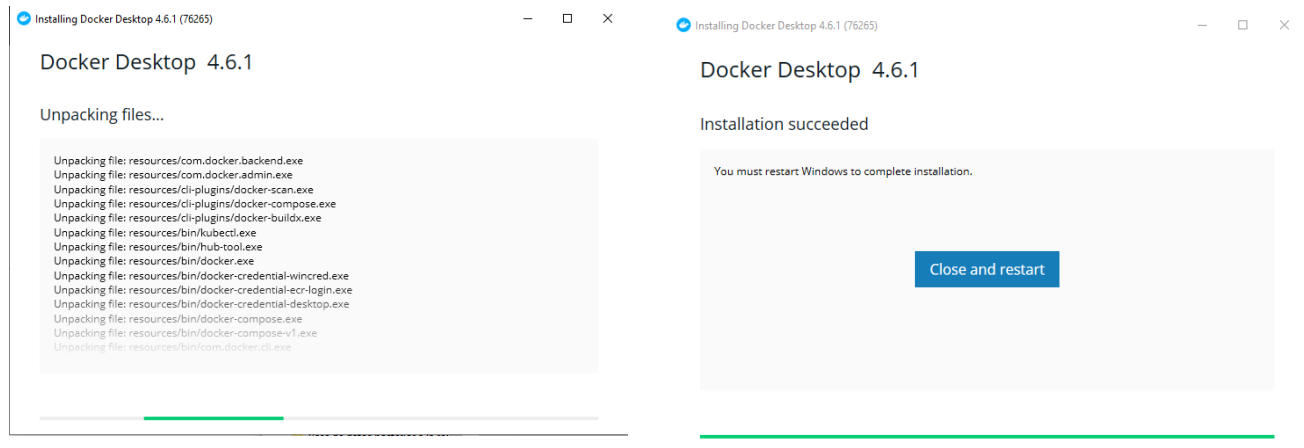
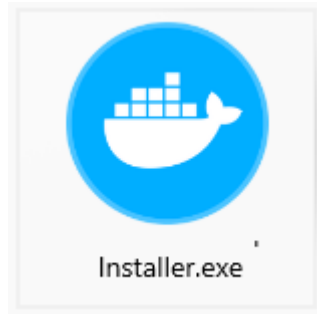


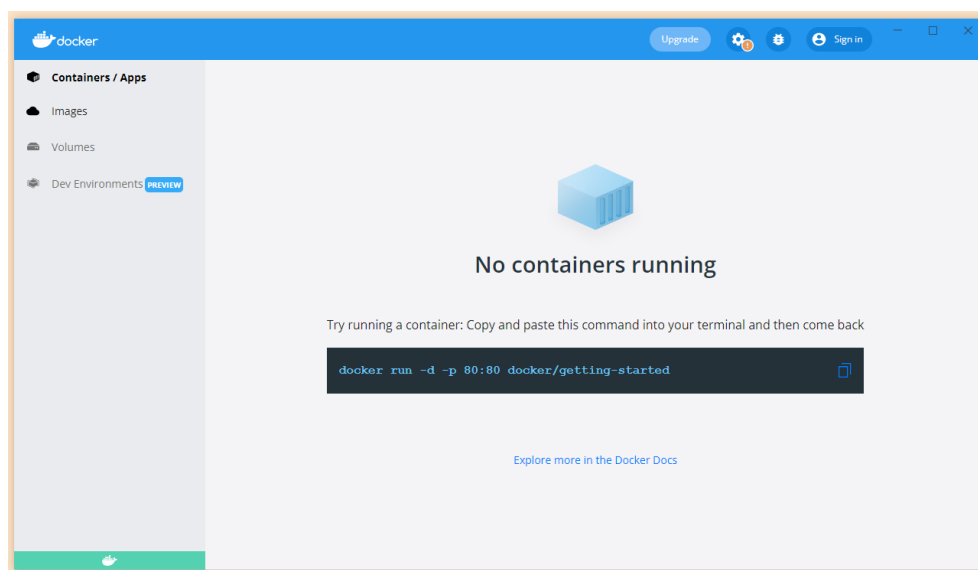
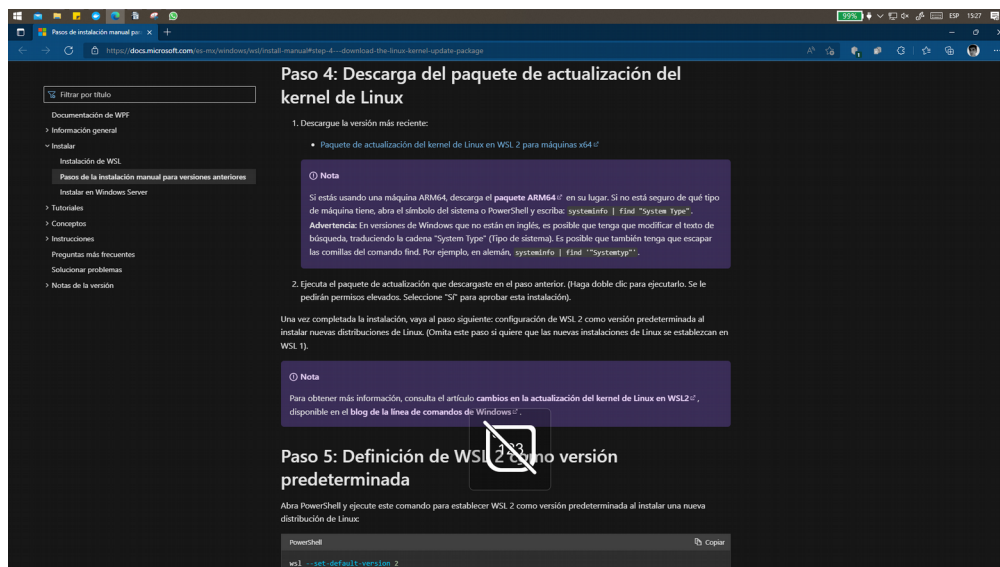
Taller 9: DevOps Kubernetes.

Ing. Luis Felipe Narvaez Gomez. E-mail: luis.narvaez@usantoto.edu.co. Cod: 2312660. Facultad de Ingenieria de sistemas.

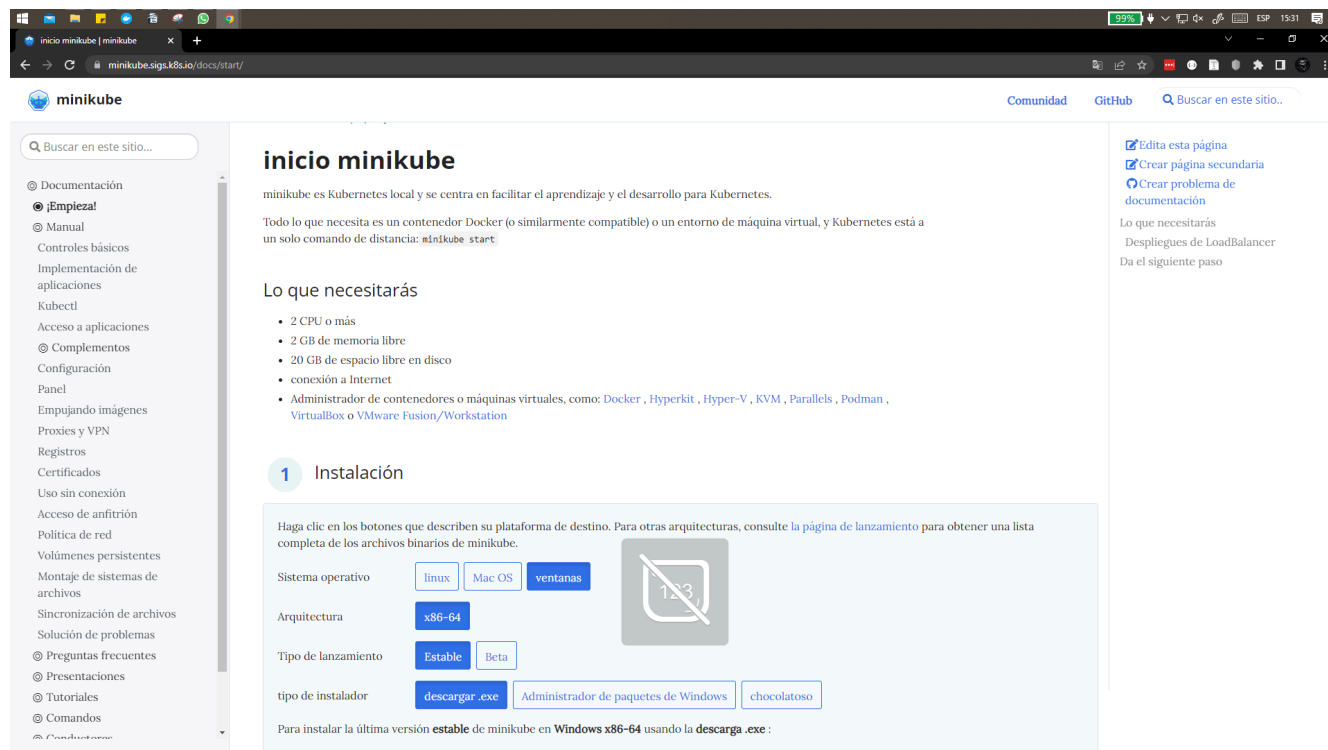
Para este taller es necesario con anterioridad no haber desinstalado Docker, pues Minikube funciona en base a este software. Minikube configura rapidamente los clusters locales de parte de kubernetes que funciona en sisteas operativos basados en Linux, MacOS y Windows.



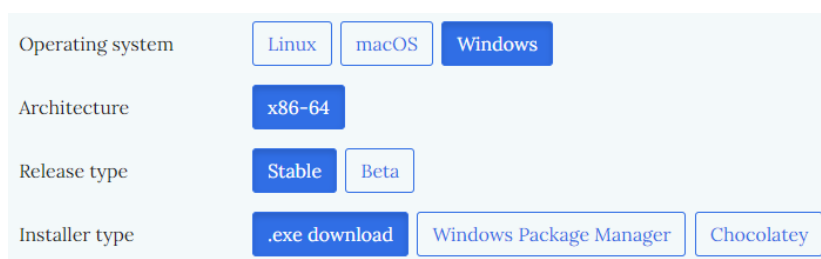
En caso de que salga el anterior error, en vez de dar clic en el boton de RESTART iremos al link de [Pasos de instalación manual para versiones anteriores de WSL | Microsoft Docs](https://docs.microsoft.com/es-mx/windows/wsl/install-manual#step-4---download-the-linux-kernel-update-package) en el que seleccionaremos en el paso 1 la opción “Paquete de actualización del kernel de Linux en WSL 2 para máquinas x64”, este descargará un archivo de extensión MSI que ejecutaremos. Cuando terminamos de instalar la actualización, podemos picar el boton de RESTART del anterior cuadro de dialogo.



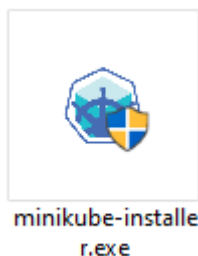
Para instalar Minikube debemos dirigirnos al enlace <https://minikube.sigs.k8s.io/docs/start/> en el encontrara la seccion que le permitira elegir el sistema operativo en que lo quiere instalar y los pasos que debemos seguir para instalar la herramienta. En este caso lo instalaremos en el Sistema Operativo de Microsoft Windows Home Single.

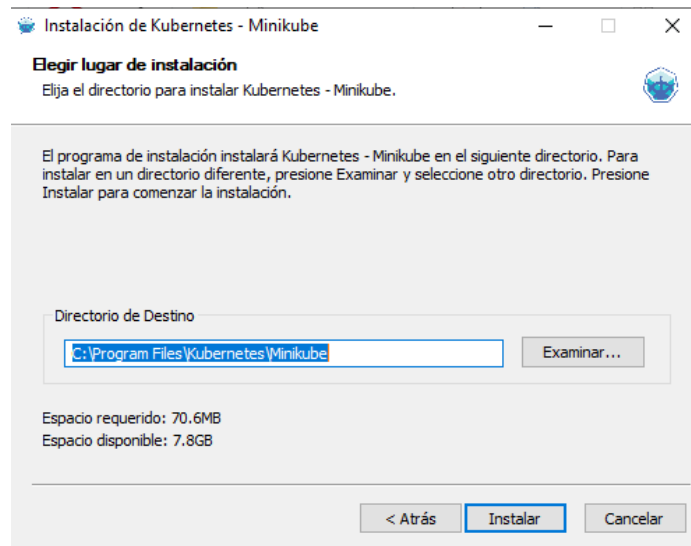


Seleccionamos el sistema operativo en el que queremos instalar.



Descargaremos la ultima version por medio del enlace en texto, aunque tambien se puede utilizar comando de power shell para esta tarea.





Una vez tenemos Minikube en nuestro sistema, ejecutaremos la consola de Power shell de windows pero con permisos de administrador. En el utilizaremos el siguiente comando en morado. Con este comando agregaremos los binarios en el archivo PATH.

```
$oldPath = [Environment]::GetEnvironmentVariable('Path',
[EnvironmentVariableTarget]::Machine)
if ($oldPath.Split(';') -notcontains 'C:\minikube'){
    [Environment]::SetEnvironmentVariable('Path', $('{0};C:\minikube' -f $oldPath),
[EnvironmentVariableTarget]::Machine) `
}
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnología PowerShell multiplataforma https://aka.ms/powershell

PS C:\Windows\system32> $oldPath = [Environment]::GetEnvironmentVariable('Path', [EnvironmentVariableTarget]::Machine)
>> if ($oldPath.Split(';') -notcontains 'C:\minikube'){ `
>> [Environment]::SetEnvironmentVariable('Path', $('{0};C:\minikube' -f $oldPath), [EnvironmentVariableTarget]::Machine) `
>> }
```

El unico cambio que debemos hacer a este comando es el cambio de la ruta de instalacion de Minikube en el sistema, ya que el comando esta hecho para una ruta por defecto de instalacion y en mi caso especifico decidi mover la instalacion a la siguiente ruta D:\SOFTWARE\Minikube\

```
$oldPath = [Environment]::GetEnvironmentVariable('Path',
[EnvironmentVariableTarget]::Machine)
if ($oldPath.Split(';') -notcontains 'D:\SOFTWARE\Kubernetes\Minikube'){
    [Environment]::SetEnvironmentVariable('Path', $('{0};D:\SOFTWARE\Kubernetes\Minikube'
-f $oldPath), [EnvironmentVariableTarget]::Machine) `
}
```

Una vez hecho lo anterior podemos iniciar un cluster, para esto podemos iniciar tanto la consola de CMD de Windows o la Power Shell, pero en cualquiera de los dos casos debemos iniciarla como administradores, en caso de que tengamos un sistema operativo basado en Linux debemos recordar que el modo administrador es el ROOT.

La primera vez que se ejecute el cluster, se descargaran las diferentes dependencias para que este funcione, asi como todas las utilidades que ofrece minikube como en la siguiente imagen.

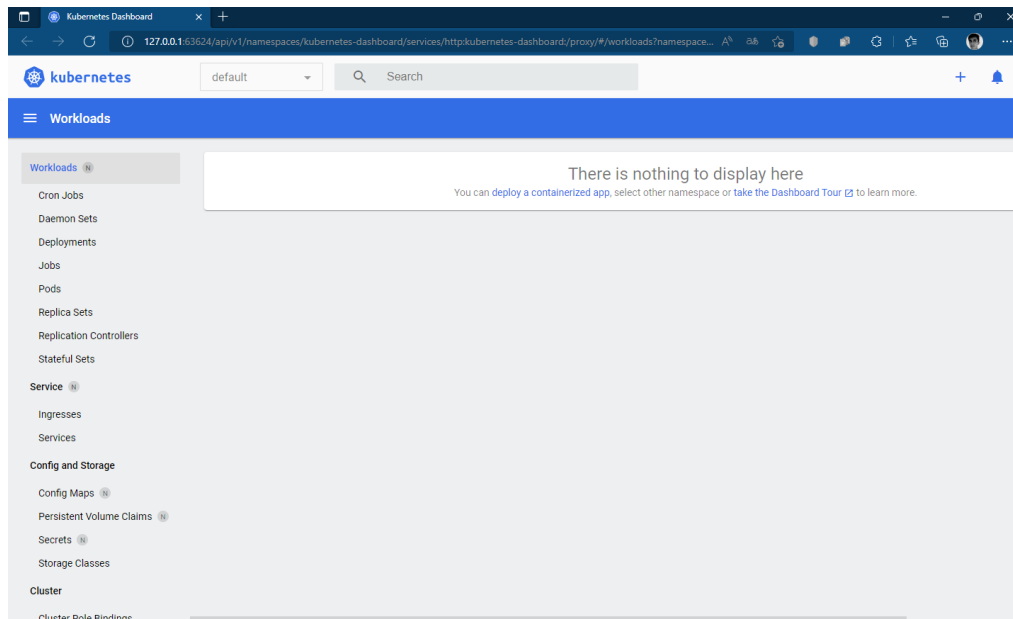
```
Administrador: Windows PowerShell
PS C:\Windows\system32> minikube start
* minikube v1.25.2 en Microsoft Windows 10 Home Single Language 10.0.19044 Build 19044
* Controlador docker seleccionado automáticamente. Otras opciones: hyperv, virtualbox, ssh
* Starting control plane node minikube in cluster minikube
* Pulling base image ...
  > gcr.io/k8s-minikube/kicbase: 379.06 MiB / 379.06 MiB 100.00% 13.28 MiB p
* Creando docker container (CPUs=2, Memory=4000MB) ...E0501 17:39:03.969043 21824 kic.go:267] icacIs failed applying permissions - e
rr - [%s(<nil>)], output - [archivo procesado: C:\Users\Ruiso Local Pc\.minikube\machines\minikube\id_rsa
Se procesaron correctamente 1 archivos; error al procesar 0 archivos]
* Preparando Kubernetes v1.23.3 en Docker 20.10.12...
  - kubelet.housekeeping-interval=5m
  - Generando certificados y llaves
  - Iniciando plano de control
  - Configurando reglas RBAC...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Complementos habilitados: storage-provisioner, default-storageclass
* Done! kubectI is now configured to use "minikube" cluster and "default" namespace by default
PS C:\Windows\system32>
```

En caso de que no se inicie el comando de “minikube Start” pudde revizar que no exista ya un cluster creado a partir de este comando por lo que puede utilizar el comando “minikube delete” el cual eliminara todos los cluster previamente creados en el equipo. En caso de que este no sea el problema puede consultar la pagina de controladores para obtner ayuda en la configuracion del contenedor, la cual sea compatible con su maquina; el enlace es el siguiente [Drivers | minikube \(k8s.io\)](#). Una vez tenemos KUBECTL instalado podemos utilizarlo para abrir el cluster que hemos creado con el comando “kubectI get po -A”.

```
PS C:\Windows\system32> kubectI get po -A
NAMESPACE      NAME                                READY   STATUS    RESTARTS   AGE
kube-system    coredns-64897985d-gkh7h           1/1     Running   0           7m56s
kube-system    etcd-minikube                     1/1     Running   0           8m7s
kube-system    kube-apiserver-minikube           1/1     Running   0           8m7s
kube-system    kube-controller-manager-minikube  1/1     Running   0           8m7s
kube-system    kube-proxy-pz6ds                  1/1     Running   0           7m57s
kube-system    kube-scheduler-minikube           1/1     Running   0           8m7s
kube-system    storage-provisioner               1/1     Running   1 (7m34s ago) 8m4s
PS C:\Windows\system32>
```

De forma alternativa , minikube puede escargar la version mas aecuada de KubectI en caso de que este no se encuentre instalado al utilizar el comando “minikube kubectI -- get po -A”. Otra manera seria crear un alias en la configuracion de Shell con el comando “alias kubectI=“minikube kubectI -””. Inicialmente, es posible que algunos de los servicios como el aprovidionador e almacenamiento de la informacion, aun no esten en ejecucion. Esta es una condicion norma dirante la aparicion del cluster y se resolvera omentaneamente. En caso de querer ver informacion adicional sobre el estado del cluster, minikube agrupa en un panel de Kubernetes este mismo estado con el comando “minikube dashboard”.

```
PS C:\Windows\system32> minikube dashboard
* Habilitando dashboard
  - Using image kubernetesui/dashboard:v2.3.1
  - Using image kubernetesui/metrics-scraper:v1.0.7
* Verifying dashboard health ...
* Launching proxy ...
* Verifying proxy health ...
* Opening http://127.0.0.1:63624/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default bro
wser...
```



Para matar el proceso o salir de la dashboard, en la consola que estavamos (en mi caso Power Shell) presionamos la combinacion de teclas CTRL+C.

```
PS C:\Windows\system32> minikube dashboard
* Habilitando dashboard
  - Using image kubernetesui/dashboard:v2.3.1
  - Using image kubernetesui/metrics-scraper:v1.0.7
* Verifying dashboard health ...
* Launching proxy ...
* Verifying proxy health ...
* Opening http://127.0.0.1:63624/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
E0501 17:53:35.966175    23080 dashboard.go:129] Wait: exit status 0xc000013a
PS C:\Windows\system32>
```

Ahora implementaremos aplicaciones exponiendolas o poniendolas en estado de “Listen” en el puerto 8080. Primero sera crear la aplicación:

```
PS C:\Windows\system32> kubectl create deployment hello-minikube --image=k8s.gcr.io/echoserver:1.4
deployment.apps/hello-minikube created
PS C:\Windows\system32>
```

Y de Segundo esta ponerla a correr o ejecutarla en el puerto 8080 como ejemplo.

```
PS C:\Windows\system32> kubectl expose deployment hello-minikube --type=NodePort --port=8080
service/hello-minikube exposed
PS C:\Windows\system32>
```

Ahora la implementacion aparecera cuando llamemos la implementacion por su nombre, similar a como haciamos con Docker.

```
PS C:\Windows\system32> kubectl get services hello-minikube
NAME          TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
hello-minikube NodePort    10.98.210.218 <none>        8080:32451/TCP   71s
PS C:\Windows\system32>
```

Podemos observar la aplicación corriendo en el navegador de nuestra preferencia al ingresar al puerto anualmente, sin embargo podemos abrirla desde la terminal de la siguiente manera.

```
PS C:\Windows\system32> minikube service hello-minikube
* Starting tunnel for service hello-minikube.
* Opening service default/hello-minikube in default browser...
! Porque estás usando controlador Docker en windows, la terminal debe abrirse para ejecutarlo.
```

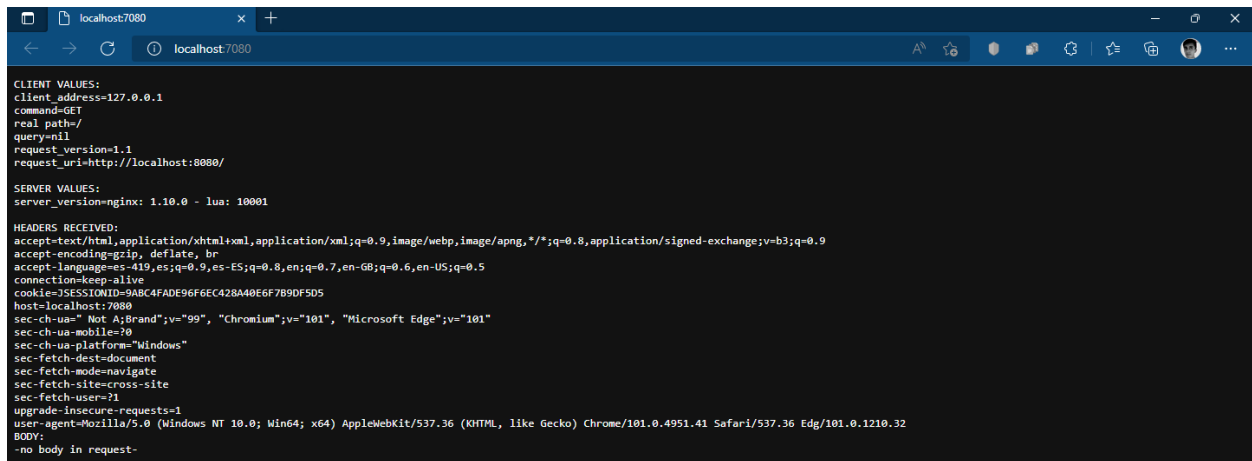
Para terminar con la ejecución del comando también presionamos la combinación de teclas de CTRL+C.

```
C:\Windows\system32>minikube service hello-minikube
* Starting tunnel for service hello-minikube.
* Opening service default/hello-minikube in default browser...
! Porque estás usando controlador Docker en windows, la terminal debe abrirse para ejecutarlo.
* Stopping tunnel for service hello-minikube.

C:\Windows\system32>
```

También podemos reenviar el puerto en la ejecución del servicio, esto moverá la ejecución del servicio al puerto 7080. Cuando queramos terminar con la ejecución del servicio podemos utilizar nuevamente la conjunción de teclas CTRL+C. Al ejecutar este comando podremos visualizar en el navegador los metadatos de NGINX y los resultados de la aplicación.

```
PS C:\Windows\system32> kubectl port-forward service/hello-minikube 7080:8080
Forwarding from 127.0.0.1:7080 -> 8080
Forwarding from [::1]:7080 -> 8080
Handling connection for 7080
Handling connection for 7080
```



```
CLIENT VALUES:
client_address=127.0.0.1
command=GET
real_path=/
query=nil
request_version=1.1
request_uri=http://localhost:8080/

SERVER VALUES:
server_version=nginx: 1.10.0 - lua: 10001

HEADERS RECEIVED:
accept=text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
accept-encoding=gzip, deflate, br
accept-language=es-419;q=0.9,es-ES;q=0.8,en;q=0.7,en-GB;q=0.6,en-US;q=0.5
connection=keep-alive
cookie=JSESSIONID=9ABC4FADE96F6EC428A40E6F7B9DF5D5
host=localhost:7080
sec-ch-ua=Not A;Brand";v="99", "Chromium";v="101", "Microsoft Edge";v="101"
sec-ch-ua-mobile=?0
sec-ch-ua-platform="Windows"
sec-fetch-dest=document
sec-fetch-mode=navigate
sec-fetch-site=cross-site
sec-fetch-user=?1
upgrade-insecure-requests=1
user-agent=Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/101.0.4951.41 Safari/537.36 Edg/101.0.1210.32
BODY:
-no body in request-
```

Ahora, para acceder a una implementación de LoadBalancer, podemos utilizar el comando “minikube tunnel”, esto realizado de la siguiente manera:

```
PS C:\Windows\system32> kubectl create deployment balanced --image=k8s.gcr.io/echoserver:1.4
deployment.apps/balanced created
PS C:\Windows\system32> kubectl expose deployment balanced --type=LoadBalancer --port=8088
service/balanced exposed
PS C:\Windows\system32>
```

En una terminal aparte iniciaremos el túnel para crear una IP enrutable para la implementación equilibrada.

```
Administrador: Windows PowerShell
PS C:\Windows\system32> minikube tunnel
* Tunnel successfully started

* NOTE: Please do not close this terminal as this process must stay alive for the tunnel to be accessible ...

* Starting tunnel for service balanced.
```

Para encontrar la IP enrutable, examinaremos la columna EXTERNAL IP en la terminal Principal bajo el siguiente comando:

```
PS C:\Windows\system32> kubectl get services balanced
NAME         TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
balanced    LoadBalancer 10.99.41.166   127.0.0.1      8088:31768/TCP   3m40s
PS C:\Windows\system32>
```

Para terminar con el tunel, simplemente es ir a la terminal secundaria y dar la suma de teclas CTRL+C, en la principal ya no deberiamos tener el servicio activo.

```
PS C:\Windows\system32> minikube tunnel
* Tunnel successfully started

* NOTE: Please do not close this terminal as this process must stay alive for the tunnel to be accessible ...

* Starting tunnel for service balanced.
* Stopped tunnel for service balanced.
PS C:\Windows\system32> ^C
PS C:\Windows\system32>
```

Ahora podemos empezar a administrar nuestro cluster. Si queremos PAUSAR KUBERNETES sin afectar las aplicaciones implementadas.

```
PS C:\Windows\system32> minikube pause
* Pausing node minikube ...
* Paused 18 containers in: kube-system, kubernetes-dashboard, storage-gluster, istio-operator
PS C:\Windows\system32>
```

Si queremos DESPAUSAR KUBERNETES utilizaremos.

```
PS C:\Windows\system32> minikube unpause
* Unpausing node minikube ...
* Unpaused 18 containers in: kube-system, kubernetes-dashboard, storage-gluster, istio-operator
PS C:\Windows\system32>
```

Si queremos DETENER KUBERNETES entonces utilizaremos.

```
PS C:\Windows\system32> minikube stop
* Stopping node "minikube" ...
* Apagando "minikube" mediante SSH...
* 1 node stopped.
PS C:\Windows\system32>
```

Si queremos AUMENTAR el limite de la MEMORIA de KUBERNETES predeterminada (necesitaremos reinicio) utilizaremos el comando “minikube config set memory 16384” siendo la ultima la cantidad de memoria que asignaremos al cluster. Si queremos ver el CATALOGO de SERVICIOS KUBERNETES podemos listarlos con.


```
1 Node stopped
PS C:\Windows\system32> minikube addons list
```

ADDON NAME	PROFILE	STATUS	MAINTAINER
ambassador	minikube	disabled	third-party (ambassador)
auto-pause	minikube	disabled	google
csi-hostpath-driver	minikube	disabled	kubernetes
dashboard	minikube	enabled ☑	kubernetes
default-storageclass	minikube	enabled ☑	kubernetes
efk	minikube	disabled	third-party (elastic)
freshpod	minikube	disabled	google
gcp-auth	minikube	disabled	google
gvisor	minikube	disabled	google
helm-tiller	minikube	disabled	third-party (helm)
ingress	minikube	disabled	unknown (third-party)
ingress-dns	minikube	disabled	google
istio	minikube	disabled	third-party (istio)
istio-provisioner	minikube	disabled	third-party (istio)
kong	minikube	disabled	third-party (Kong HQ)
kubevirt	minikube	disabled	third-party (kubevirt)
logviewer	minikube	disabled	unknown (third-party)
metallb	minikube	disabled	third-party (metallb)
metrics-server	minikube	disabled	kubernetes
nvidia-driver-installer	minikube	disabled	google
nvidia-gpu-device-plugin	minikube	disabled	third-party (nvidia)
olm	minikube	disabled	third-party (operator framework)
pod-security-policy	minikube	disabled	unknown (third-party)
portainer	minikube	disabled	portainer.io
registry	minikube	disabled	google
registry-aliases	minikube	disabled	unknown (third-party)
registry-creds	minikube	disabled	third-party (upmc enterprises)
storage-provisioner	minikube	enabled ☑	google
storage-provisioner-gluster	minikube	disabled	unknown (third-party)
volumesnapshots	minikube	disabled	kubernetes

```
PS C:\Windows\system32>
```

Ahora bien, si necesitamos crear un segundo cluster en nuestra maquina y asu vez necesitamos que el mismo trabaje cin una version anterior de Kubernetes utilizaremos el siguiente comando. Debemos tener en cuenta que tal como en el proceso del primer cluster, el hecho de tomar otra version anterior de Kunernetes obligara al sistema a descargar las dependencias necesarias para ello, lo cual puede demorar un tiempo.

```

PS C:\Windows\system32> minikube start -p aged --kubernetes-version=v1.16.1
* [aged] minikube v1.25.2 en Microsoft Windows 10 Home Single Language 10.0.19044 Build 19044
* Controlador docker seleccionado automáticamente. Otras opciones: hyperv, virtualbox, ssh
* Starting control plane node aged in cluster aged
* Pulling base image ...
* Creando docker container (CPUs=2, Memory=4000MB) .../ E0501 18:26:04.438250 7464 kic.go:267] icacfs failed applying permissions -
err - [%s(<nil>)], output - [archivo procesado: C:\Users\Ruiso Local Pc\minikube\machines\aged\id_rsa
Se procesaron correctamente 1 archivos; error al procesar 0 archivos]

* Preparando Kubernetes v1.16.1 en Docker 20.10.12...
- kubelet.housekeeping-interval=5m
> kubeadm.shal: 41 B / 41 B [-----] 100.00% ? p/s 0s
> kubect1.shal: 41 B / 41 B [-----] 100.00% ? p/s 0s
> kubelet.shal: 41 B / 41 B [-----] 100.00% ? p/s 0s
> kubeadm: 42.20 MiB / 42.20 MiB [-----] 100.00% 9.43 MiB p/s 4.7s
> kubect1: 44.52 MiB / 44.52 MiB [-----] 100.00% 6.58 MiB p/s 7.0s
> kubelet: 117.43 MiB / 117.43 MiB [-----] 100.00% 11.52 MiB p/s 10s
- Generando certificados y llaves
- Iniciando plano de control
- Configurando reglas RBAC...
* Verifying Kubernetes components...
- Using image gcr.io/k8s-minikube/storage-provisioner:v5
* Complementos habilitados: storage-provisioner, default-storageclass

! C:\Program Files\Docker\Docker\resources\bin\kubect1.exe is version 1.22.5, which may have incompatibilites with Kubernetes 1.16.1.
- Want kubect1 v1.16.1? Try 'minikube kubect1 -- get pods -A'
* Done! kubect1 is now configured to use "aged" cluster and "default" namespace by default
PS C:\Windows\system32>

```

Y por ultimo pero no menos importante, para borrar todos los clusters de minikube utilizaremos el siguiente comando.

```

PS C:\Windows\system32> minikube delete --all
* Eliminando "aged" en docker...
* Eliminando C:\Users\Ruiso Local Pc\minikube\machines\aged...
* Removed all traces of the "aged" cluster.
* Eliminando "minikube" en docker...
* Eliminando C:\Users\Ruiso Local Pc\minikube\machines\minikube...
* Removed all traces of the "minikube" cluster.
* Successfully deleted all profiles
PS C:\Windows\system32>

```

Ahora instalaremos CHOCOLATEY con ayuda de la guia oficial que podemos tener en el enlace [Chocolatey Software | Installing Chocolatey](https://community.chocolatey.org/courses/installation/installing?method=installing-chocolatey#powershell) que enlaza a la pagina oficial de la herramienta. Este es un administrador de paquetes para windows diseñado para permitir a los usuarios el descargar e instalar aplicaciones directamente de Internet con comando de Power Shell o de CMD.

Course Modules:

- Installing Chocolatey
- Upgrading Chocolatey
- Uninstalling Chocolatey

Basic Chocolatey Install

Chocolatey installs in seconds. You are just a few steps from running choco right now!

1. First, ensure that you are using an **administrative shell** - you can also install as a non-admin, check out Non-Administrative Installation.
2. Copy the text specific to your command shell below.

NOTE: Please inspect <https://community.chocolatey.org/install.ps1> prior to running any of these scripts to ensure safety. We already know it's safe, but you should verify the security and contents of **any** script from the Internet you are not familiar with. All of these scripts download a remote PowerShell script and execute it on your machine. We take security very seriously. [Learn more about our security protocols.](#)

Install with cmd.exe **Install with powershell.exe**

Install with powershell.exe

With PowerShell, there is an additional step. You must ensure Get-ExecutionPolicy is not Restricted. We suggest using **bypass** to bypass the policy to get things installed or **AllSigned** for quite a bit more security.

- o Run **Get-ExecutionPolicy**. If it returns **Restricted**, then run **Set-ExecutionPolicy AllSigned** or **Set-ExecutionPolicy Bypass -Scope Process**.

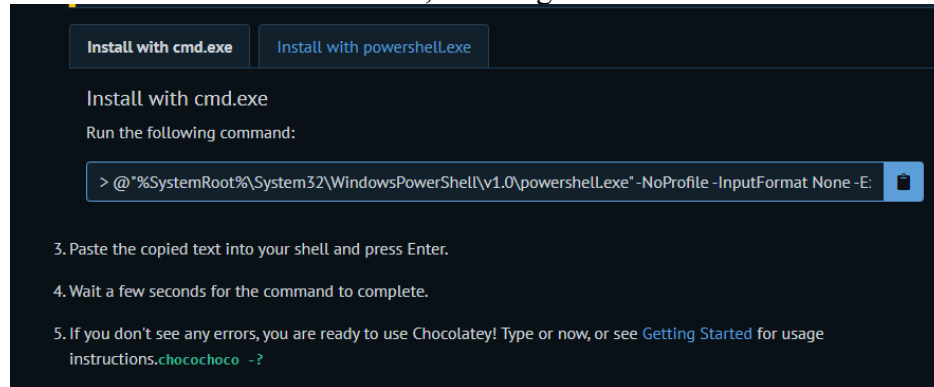
Now run the following command:

```
> Set-ExecutionPolicy Bypass -Scope Process -Force; iex ((New-Object System.Net.WebClient).DownloadStr...
```

3. Paste the copied text into your shell and press Enter.
4. Wait a few seconds for the command to complete.
5. If you don't see any errors, you are ready to use Chocolatey! Type **choco** or **choco -?** now, or see [Getting Started](#) for usage.

Debemos tener en cuenta que al descargar Chocolatey podremos descargar Scripts de aplicaciones directamente a nuestro ordenador, esta accion puede ser aprovechada por terceros para vulnerar a seguridad de nuestra maquina por lo que la pagina oficial de Chocolatey no ofrece una guia en el enlace <https://community.chocolatey.org/install.ps1> que podemos revisar antes de ejecuta cualquiera de los scripts de instalacion.

Chocolatey tiene una forma un tanto distinta de instalarse dependiendo si estamos utilizando la terminal de Windows CMD o la terminal de Power Shell, en esta guia utilizaremos la alternativa de CMD.



Abriremos una consola CMD de Windows como Administrador y ejecutaremos el siguiente comando:

```
@ "%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe" -NoProfile  
-InputFormat None -ExecutionPolicy Bypass -Command "iex ((New-Object  
System.Net.WebClient).DownloadString('https://community.chocolatey.org/  
install.ps1'))" && SET "PATH=%PATH%;%ALLUSERSPROFILE%\chocolatey\bin"
```

```
C:\Windows\system32>@"%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe" -NoProfile -InputFormat None -ExecutionPolicy Bypass -Command "iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))" && SET "PATH=%PATH%;%ALLUSERSPROFILE%\chocolatey\bin"  
ADVERTENCIA: 'choco' was found at 'C:\ProgramData\chocolatey\bin\choco.exe'.  
ADVERTENCIA: An existing Chocolatey installation was detected. Installation will not continue.  
For security reasons, this script will not overwrite existing installations.  
  
Please use choco upgrade chocolatey to handle upgrades of Chocolatey itself.  
C:\Windows\system32>
```

En caso de que vuestro sistema ya posea Chocolatey pero de igual forma quieran actualizarlo pueden utilizar el comando sugerido en la anterior imagen como me sucedió a mi.

```

C:\Windows\system32>choco upgrade chocolatey
Chocolatey v0.12.1
Upgrading the following packages:
chocolatey
By upgrading, you accept licenses for the packages.

You have chocolatey v0.12.1 installed. Version 1.1.0 is available based on your source(s).
Progress: Downloading chocolatey 1.1.0... 100%

chocolatey v1.1.0
chocolatey package files upgrade completed. Performing other installation steps.
The package chocolatey 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

Creating ChocolateyInstall as an environment variable (targeting 'Machine')
Setting ChocolateyInstall to 'C:\ProgramData\chocolatey'
WARNING: It's very 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 folders if they do not already exist.

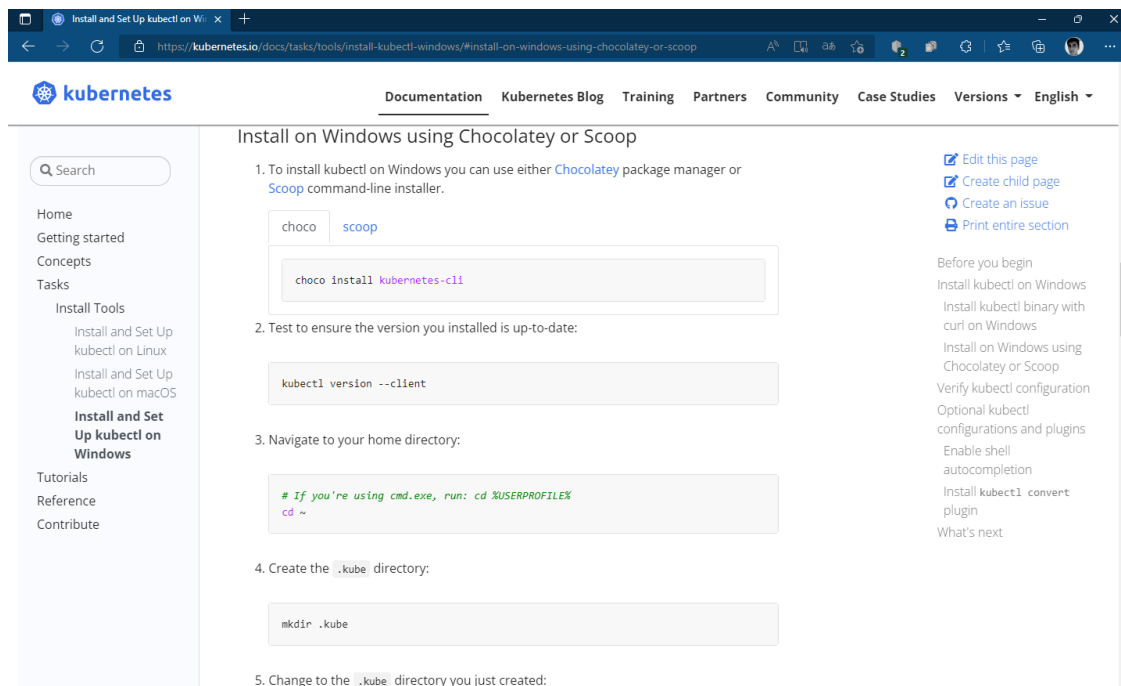
WARNING: You can safely ignore errors related to missing log files when
upgrading from a version of Chocolatey less than 0.9.9.
'Batch file could not be found' is also safe to ignore.
'The system cannot find the file specified' - also safe.
Removing shim C:\ProgramData\chocolatey\redirects\cpack.exe
Removing shim C:\ProgramData\chocolatey\redirects\cver.exe
WARNING: Not setting tab completion: Profile file does not exist at 'C:\Users\Ruiso Local Pc\Documents\WindowsPowerSh
ell\Microsoft.PowerShell_profile.ps1'.
Chocolatey (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.
Removing shim C:\ProgramData\chocolatey\bin\cpack.exe
Removing shim C:\ProgramData\chocolatey\bin\cver.exe
Environment Vars (like PATH) have changed. Close/reopen your shell to
see the changes (or in powershell/cmd.exe just type 'refreshenv').
The upgrade of chocolatey was successful.
Software install location not explicitly set, it could be in package or
default install location of installer.

Chocolatey upgraded 1/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).

C:\Windows\system32>

```

Por medio de Chocolatey tambien podemos instalar Kubernetes, la guia de instalacion de esta herramienta por medio de Chocolatey podemos encontrarla por medio del siguiente enlace [Install and Set Up kubectl on Windows | Kubernetes](#) que dirigira a la pagina principal de Kubernetes.



Para instalar KubeCtl con ayuda de Chocolatey seguiremos la guia oficial de instalacion y sus comandos.

```
C:\Windows\system32>choco install kubernetes-cli
Chocolatey v1.1.0
Installing the following packages:
kubernetes-cli
By installing, you accept licenses for the packages.
Progress: Downloading kubernetes-cli 1.23.6... 100%

kubernetes-cli v1.23.6 [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 kubect1-convert.exe
ShimGen has successfully created a shim for kubect1.exe
The install of kubernetes-cli was successful.
Software installed to 'C:\ProgramData\chocolatey\lib\kubernetes-cli\tools'

Chocolatey installed 1/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).

C:\Windows\system32>
```

Probamos la version que hemos instalado.

```
C:\Windows\system32>kubectl version --client
Client Version: version.Info{Major:"1", Minor:"23", GitVersion:"v1.23.6", GitCommit:"ad3338546da947756e8a88aa6822e9c11e7eac22", GitTreeState:"clean", BuildDate:"2022-04-14T08:49:13Z", GoVersion:"go1.17.9", Compiler:"gc", Platform:"windows/amd64"}

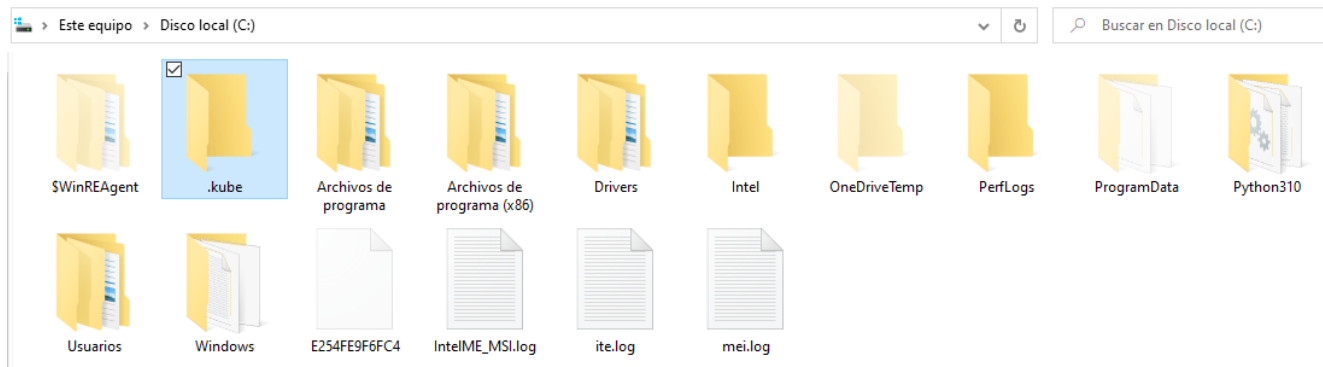
C:\Windows\system32>
```

Nos ubicamos en el directorio principal de Windows.

```
C:\Windows\system32>cd ..  
C:\Windows>cd ..  
C:\>
```

Creamos el directorio desde consola llamado .kube

```
C:\>mkdir .kube  
C:\>
```



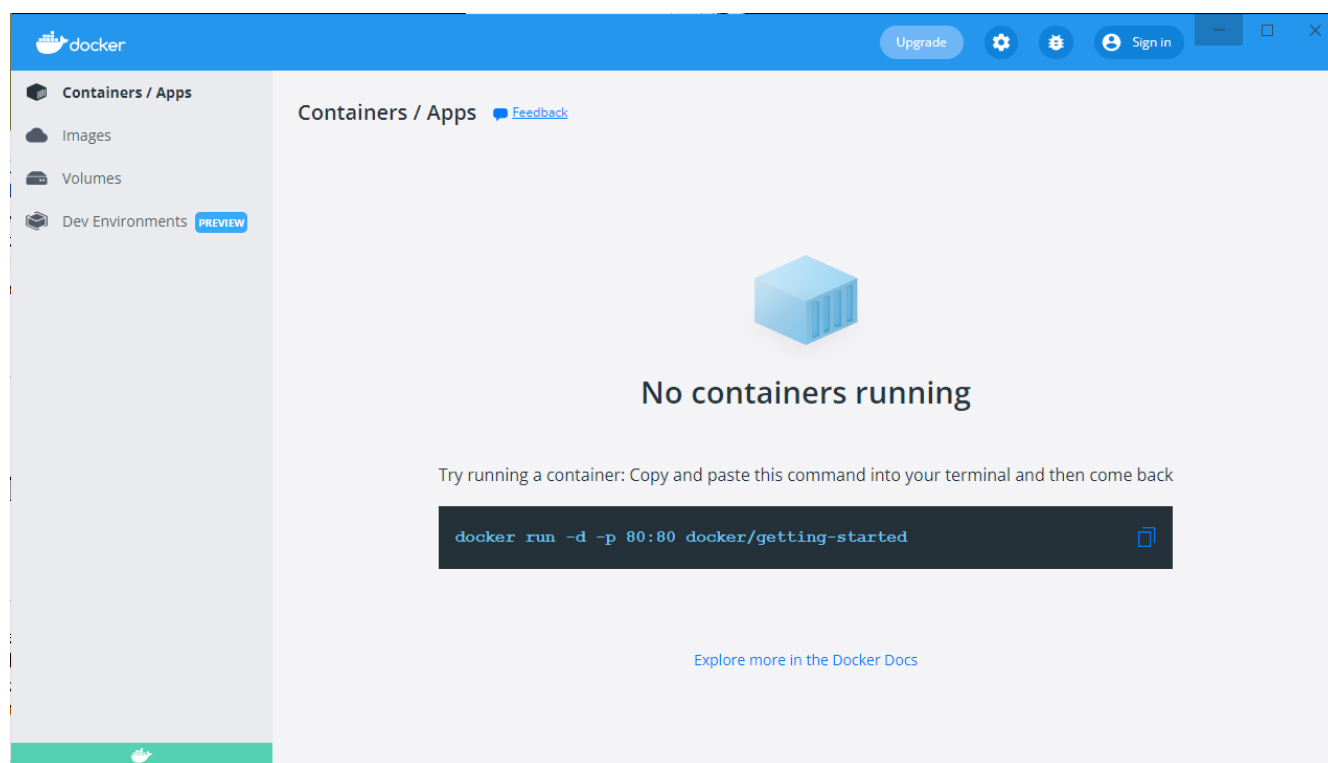
Naveguemos hasta este nuevo directorio y entremos a el.

```
C:\>dir  
El volumen de la unidad C no tiene etiqueta.  
El número de serie del volumen es: 9C27-0323  
  
Directorio de C:\  
  
01/05/2022  19:47    <DIR>          .kube  
22/03/2022  21:05    <DIR>          Drivers  
01/05/2022  17:04    <DIR>          Intel  
12/03/2022  22:16           2.439.112 IntelME_MSI.log  
12/03/2022  22:14           5.976 ite.log  
12/03/2022  22:16           48.170 mei.log  
07/12/2019  04:14    <DIR>          PerfLogs  
01/05/2022  10:18    <DIR>          Program Files  
01/05/2022  10:40    <DIR>          Program Files (x86)  
13/03/2022  11:05    <DIR>          Python310  
12/03/2022  21:45    <DIR>          Users  
26/04/2022  15:03    <DIR>          Windows  
               3 archivos      2.493.258 bytes  
               9 dirs    32.146.055.168 bytes libres  
  
C:\>cd .kube  
C:\.kube>
```

Ahora configuraremos KUBECTL para usar un cluster remoto de Kubernetes con el comando “**New-Item** config -type file”. Para que KubeCTL encuentra y acceda a un cluster de kubernetes, necesita de un archivo “kubeconfig”, que se crea automaticamente cuando se crea un cluster “kube-up.sh” o se implementa correctamente un cluster de Minikube. De forma predeterminada, la configuración de Kubectl se encontrara dentro de la ruta que hemos previamente creado C:\.kube\config. Podemos comprobar que Kubectl esta configurado correctamente obteniendo el estado del cluster.

Kubernetes tambien posee una dashboard basada en la web, en esta se pueden implementar aplicaciones en contenedores con cluster, solucionar problemas con las app, administrar recursos, obtener informacion descriptiva de las aplicaciones se ejecutan en el cluster, crear y modificar recursos individuales de Kubernetes, escalar implementaciones, etc.

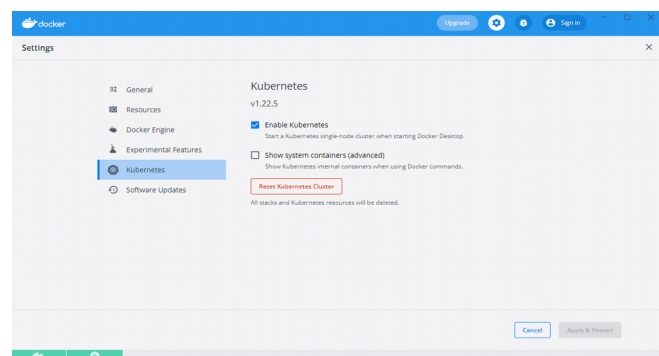
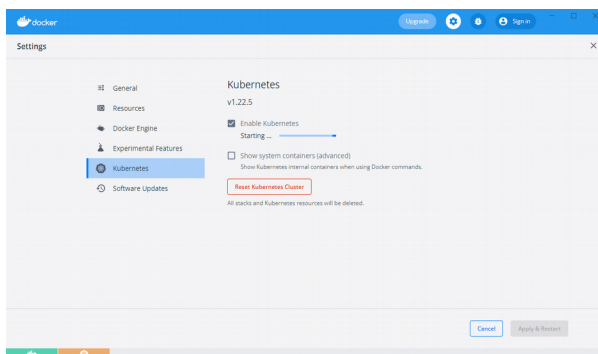
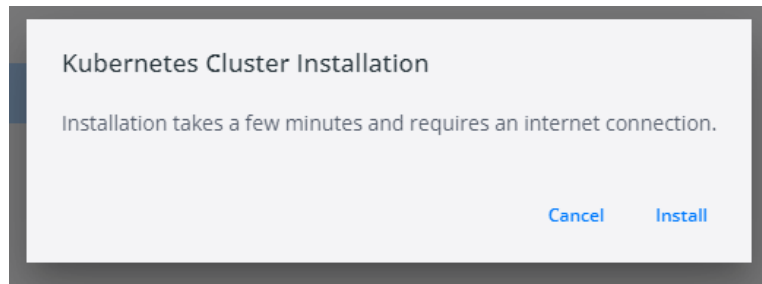
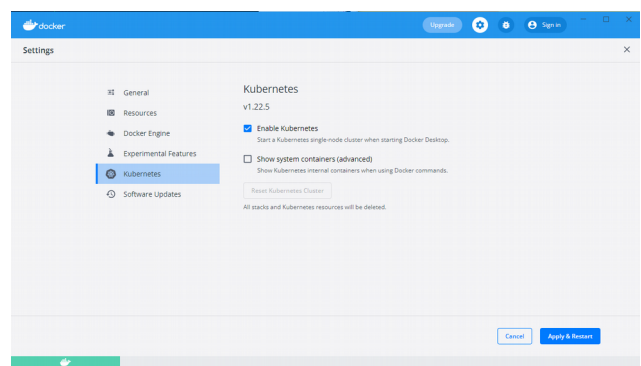
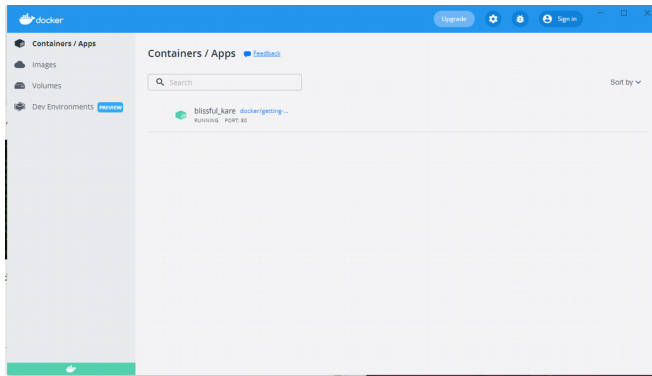
Abrimos Docker para Escritorio.



Ejecutamos el comando en una terminal CMD como Administradores.

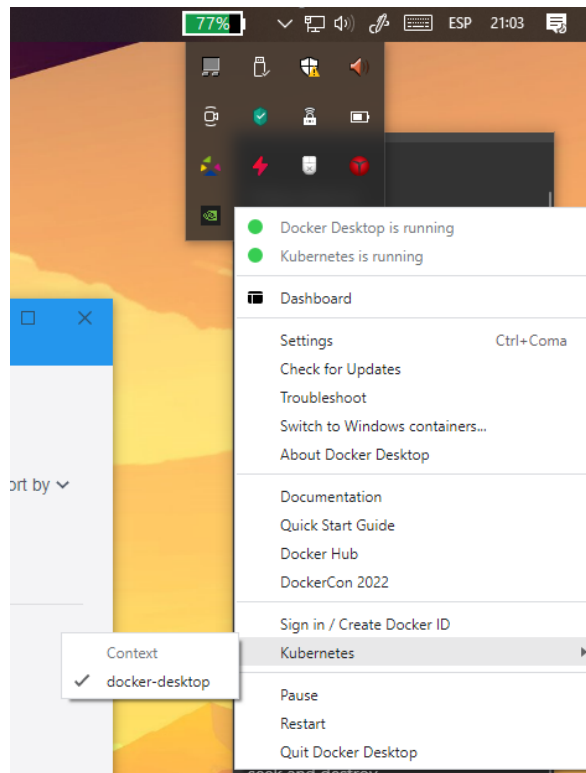
```
C:\Windows\system32>docker run -d -p 80:80 docker/getting-started
Unable to find image 'docker/getting-started:latest' locally
latest: Pulling from docker/getting-started
df9b9388f04a: Pull complete
5867cba5fcbd: Pull complete
4b639e65cb3b: Pull complete
061ed9e2b976: Pull complete
bc19f3e8eeb1: Pull complete
4071be97c256: Pull complete
79b586f1a54b: Pull complete
0c9732f525d6: Pull complete
Digest: sha256:b558be874169471bd4e65bd6eac8c303b271a7ee8553ba47481b73b2bf597aee
Status: Downloaded newer image for docker/getting-started:latest
14e1aac9c2f85dd24fa3a5f51c1fa328619284a5dab93cb98d1062a8fbd6a19c
C:\Windows\system32>
```

Entramos nuevamente a la dashboard de Docker y nos dirigimos a Settings. Alli nos dirigimos al menu lateral vertical izquierdo, opcion Kubernetes y seleccionamos "Enable Kubernetes", consecuentemente damos en "Apply & Restart".



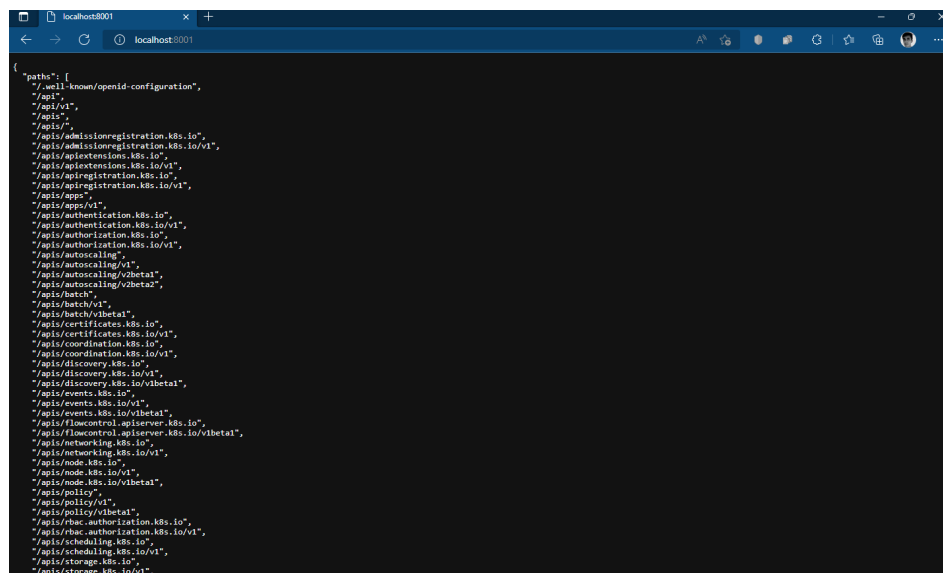
```
C:\Windows\system32>kubectl config get-contexts
CURRENT   NAME             CLUSTER           AUTHINFO           NAMESPACE
*         docker-desktop   docker-desktop    docker-desktop
C:\Windows\system32>
```

En Kubernetes se usan una serie de parametros para agregar una imagen. El nombre es un tipo de contexto facil de recordar que ayuda a nombrar una imagen en particular, añadido a esto, la imagen contiene el nombre del cluster su usuario y el espacio de contexto ara diferenciarlo. Los closters que tengamos en la maquina bien pueden verse con el anterior comando o dirigiendonos a la parte de la barra de tareas de menu de inicio de windows, tareas en segundo plano, el icono de Docker y seleccionando aquellos clusters de Kubernetes.



Ahora iniciemos el servidor y configuremos los archivos relacionados con la Dashboard.

```
C:\Windows\system32>kubectyl proxy
Starting to serve on 127.0.0.1:8001
```



```
C:\Windows\system32>kubectyl get nodes
NAME                STATUS    ROLES                  AGE    VERSION
docker-desktop      Ready     control-plane,master   13m    v1.22.5

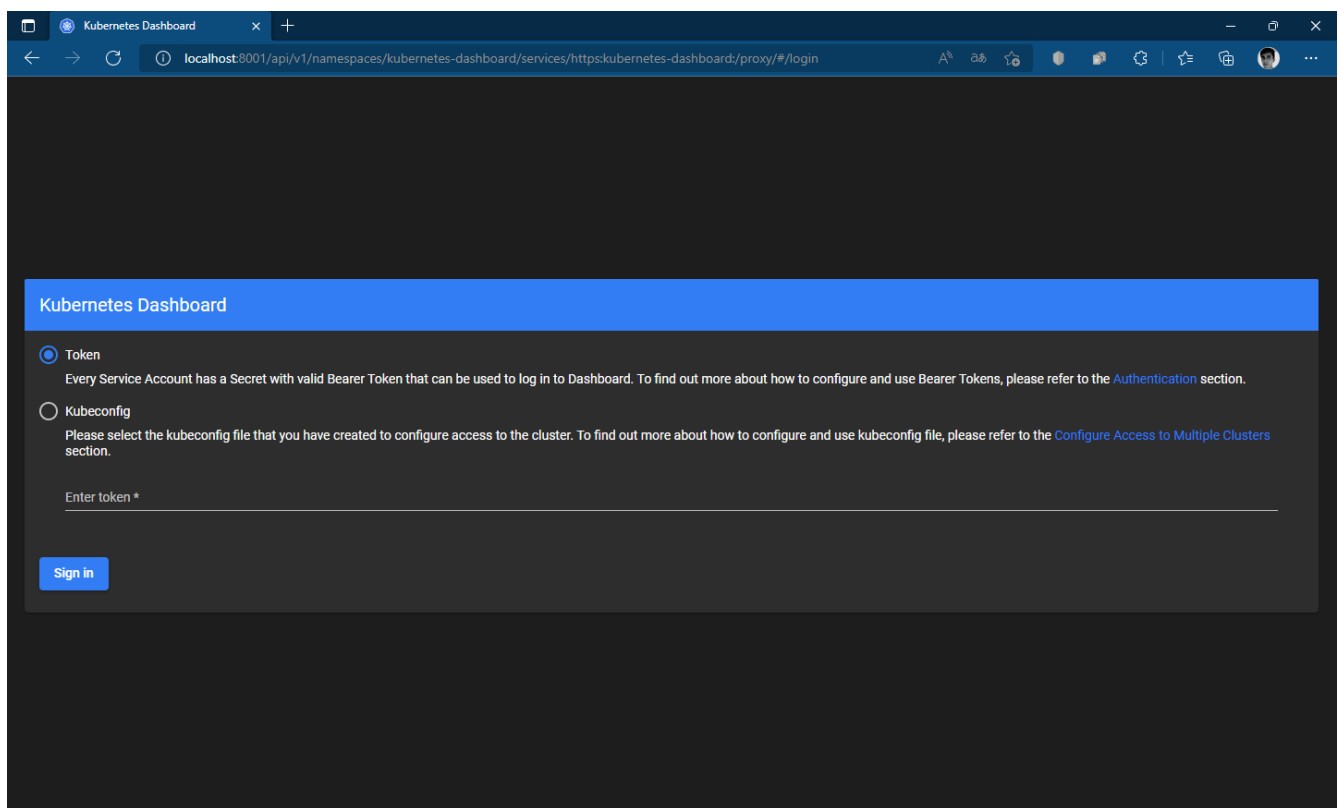
C:\Windows\system32>
```

kubectl apply -f

<https://raw.githubusercontent.com/kubernetes/dashboard/v2.5.0/aio/deploy/recommended.yaml>

```
C:\Windows\system32>kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.5.0/aio/deploy/recommended.yaml
namespace/kubernetes-dashboard created
serviceaccount/kubernetes-dashboard created
service/kubernetes-dashboard created
secret/kubernetes-dashboard-certs created
secret/kubernetes-dashboard-csrf created
secret/kubernetes-dashboard-key-holder created
configmap/kubernetes-dashboard-settings created
role.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrole.rbac.authorization.k8s.io/kubernetes-dashboard created
rolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
clusterrolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created
deployment.apps/kubernetes-dashboard created
service/dashboard-metrics-scraper created
deployment.apps/dashboard-metrics-scraper created

C:\Windows\system32>
```



Ahora obtendremos el token de acceso y configuraremos la cuenta de servicio.

```
C:\Windows\system32>kubectl create serviceaccount geekflare -n default
serviceaccount/geekflare created

C:\Windows\system32>
```

**kubectl create clusterrolebinding geekflare-admin -n default --clusterrole=cluster-admin
--serviceaccount=default:geekflare**

```
C:\Windows\system32>kubectl create clusterrolebinding geekflare-admin -n default --clusterrole=cluster-admin --serviceaccount-default:geekflare
clusterrolebinding.rbac.authorization.k8s.io/geekflare-admin created

C:\Windows\system32>
```

[illegible]

Utilizamos el token generado en la dashboard Web para entrar.

Kubernetes Dashboard

☒ Token

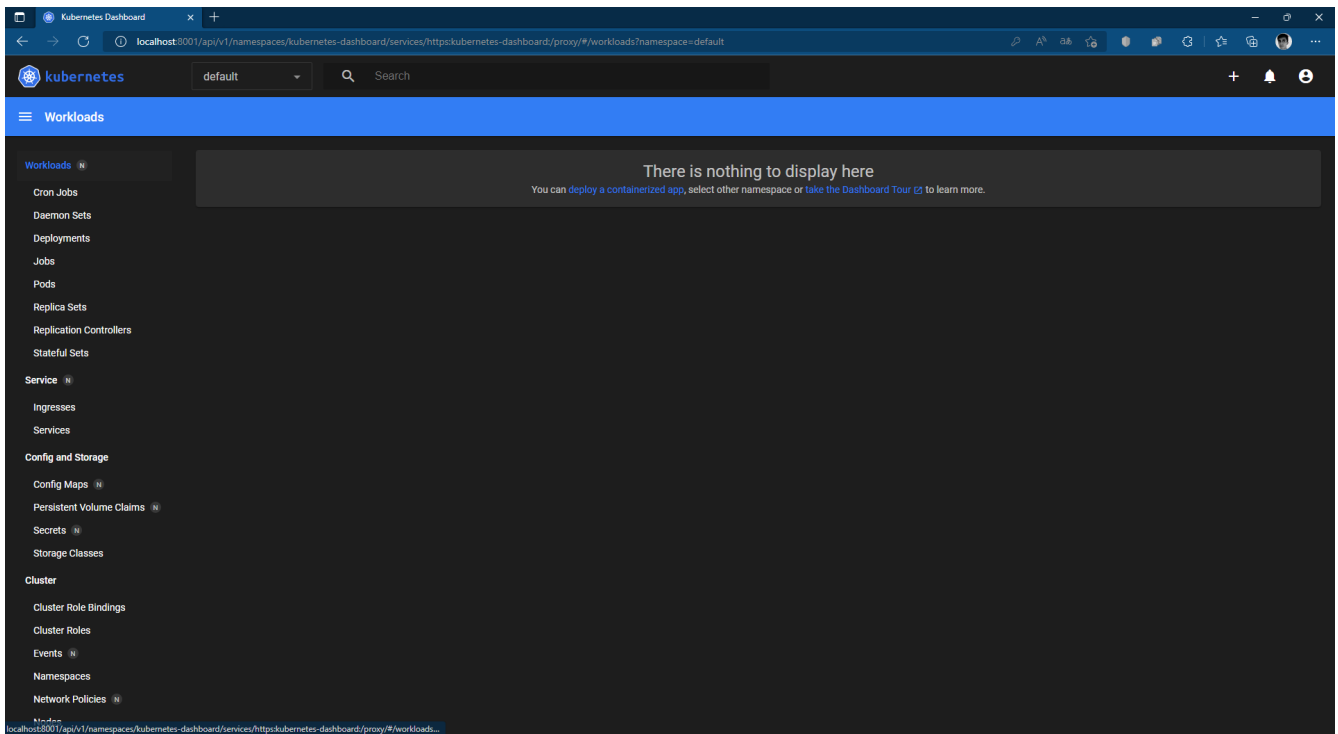
Every Service Account has a Secret with valid Bearer Token that can be used to log in to Dashboard. To find out more about how to configure and use Bearer Tokens, please refer to the [Authentication](#) section.

☐ Kubeconfig

Please select the kubeconfig file that you have created to configure access to the cluster. To find out more about how to configure and use kubeconfig file, please refer to the [Configure Access to Multiple Clusters](#) section.

Enter token *

Sign in



```
C:\Windows\system32>kubectrl proxy
Starting to serve on 127.0.0.1:8001
^C
C:\Windows\system32>
```