**Kubernetes**

**Prerequsites:**

* 2 CPUs or more
* 2GB of free memory
* 20GB of free disk space
* Internet connection
* Container or virtual machine manager, such as: [Docker](https://minikube.sigs.k8s.io/docs/drivers/docker/), [QEMU](https://minikube.sigs.k8s.io/docs/drivers/qemu/), [Hyperkit](https://minikube.sigs.k8s.io/docs/drivers/hyperkit/), [Hyper-V](https://minikube.sigs.k8s.io/docs/drivers/hyperv/), [KVM](https://minikube.sigs.k8s.io/docs/drivers/kvm2/), [Parallels](https://minikube.sigs.k8s.io/docs/drivers/parallels/), [Podman](https://minikube.sigs.k8s.io/docs/drivers/podman/), [VirtualBox](https://minikube.sigs.k8s.io/docs/drivers/virtualbox/), or [VMware Fusion/Workstation](https://minikube.sigs.k8s.io/docs/drivers/vmware/)

**Note:** I will be using Docker in this document. Make sure you have Docker Desktop installed in your system.

**Steps**

1. Open windows powershell and install minikube. Run the following commands one by one:

New-Item -Path 'c:\' -Name 'minikube' -ItemType Directory -Force

Invoke-WebRequest -OutFile 'c:\minikube\minikube.exe' -Uri 'https://github.com/kubernetes/minikube/releases/latest/download/minikube-windows-amd64.exe' -UseBasicParsing

1. Set environment variables using the following command:

$oldPath = [Environment]::GetEnvironmentVariable('Path', [EnvironmentVariableTarget]::Machine)

if ($oldPath.Split(';') -inotcontains 'C:\minikube'){

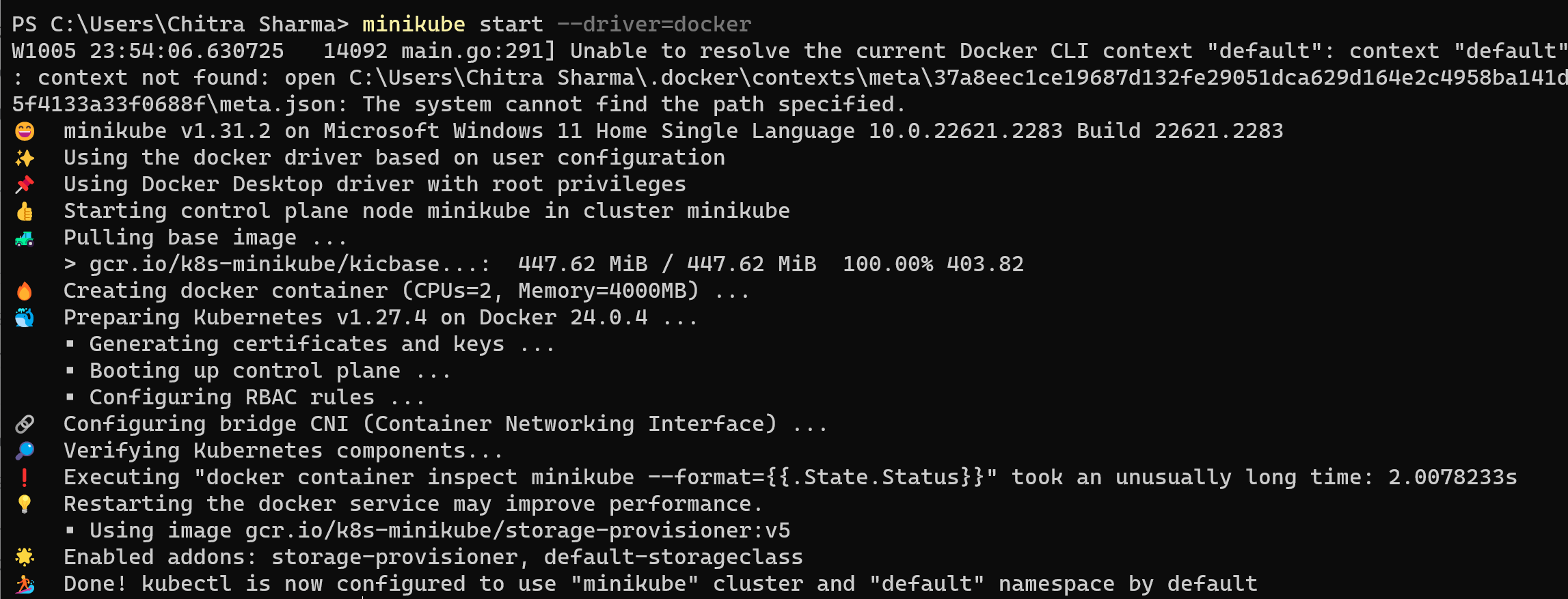
[Environment]::SetEnvironmentVariable('Path', $('{0};C:\minikube' -f $oldPath), [EnvironmentVariableTarget]::Machine)

**Start Your cluster**

1. Now open windows powershell in admin mode and run the command: (will take few minutes)

minikube start --driver=docker

// use this if you are using **docker** or write the hypervisor or container engine that you are using in its place.



**Interact with your cluster**

1. Now, you can interact with the cluster using the kubectl command line tool.

kubectl get po -A

The command "**kubectl get po -A**" is used with Kubernetes, a container orchestration platform. It does the following:

* "**kubectl**" is the command-line tool for interacting with Kubernetes clusters.
* "**get**" is the action, indicating that you want to retrieve information.
* "**po**" is short for "pods," which are the smallest deployable units in Kubernetes.
* "**-A**" or **"--all-namespaces**" specifies that you want to get information about pods in all namespaces (i.e., across the entire Kubernetes cluster), not just in the default namespace.

So, the command retrieves a list of all pods running in all namespaces of the Kubernetes cluster.

1. This command retrieves a list of all pods running in all namespaces within the Minikube Kubernetes cluster.

minikube kubectl -- get po -A

1. The command "alias kubectl="minikube kubectl --"" creates an alias for the "kubectl" command, which is commonly used to interact with Kubernetes clusters.

alias kubectl="minikube kubectl --"

By creating this alias, whenever you type "kubectl" in your terminal, it will actually execute "minikube kubectl --" followed by any additional arguments or commands you provide.

1. This command opens a web-based interface for managing a local Kubernetes cluster created with Minikube.

minikube dashboard

**Deploy Applications**

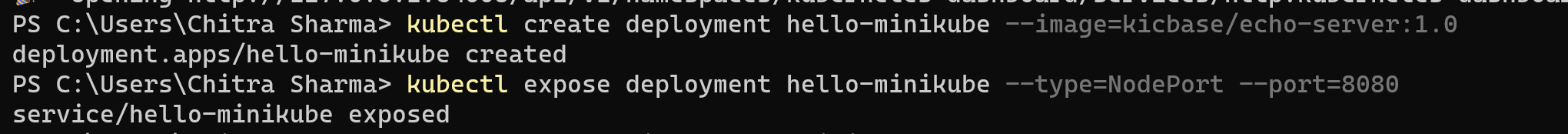
**i)Service**

* 1. Create a sample deployment and expose it on port 8080:

kubectl create deployment hello-minikube --image=kicbase/echo-server:1.0

kubectl expose deployment hello-minikube --type=NodePort --port=8080

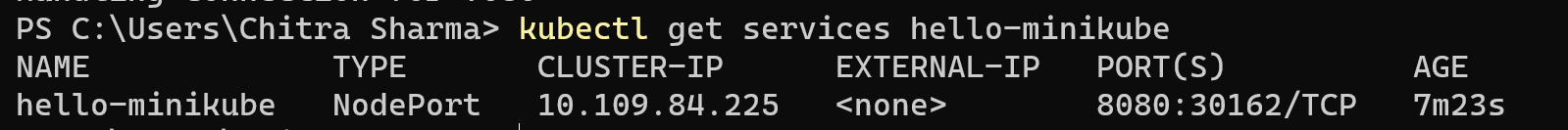
These two "kubectl" commands create a Kubernetes deployment named "hello-minikube" with a specified Docker image and expose it as a NodePort service on port 8080 within the cluster.



* 1. The above command will take a moment but the deployment will soon show up when you run:

kubectl get services hello-minikube

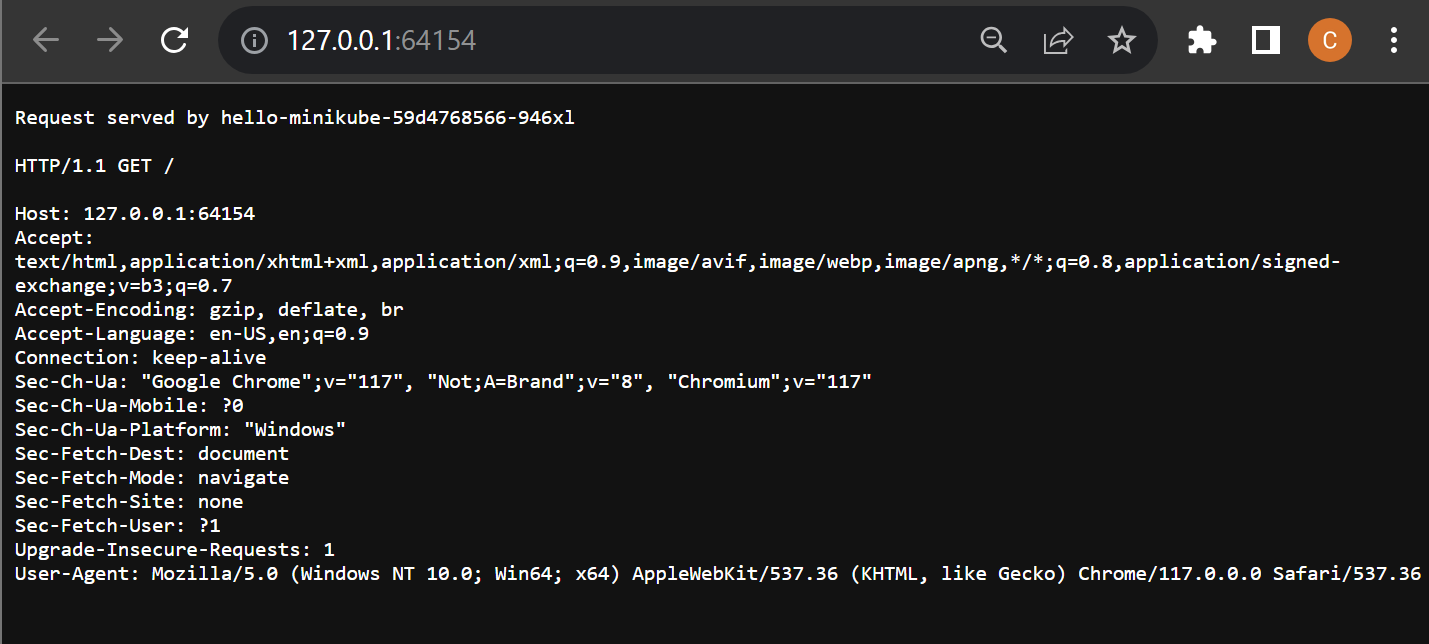
The "kubectl get services hello-minikube" command is used to retrieve information about the service named "hello-minikube" in the current Kubernetes context, which includes details such as the service's name, type, cluster IP, external IP (if applicable), and ports.



* 1. Now, the easiest way to access this service is to let the minikube launch a web browser for you:

minikube service hello-minikube

This command is used to open a web browser and access the service named "hello-minikube" running in a Minikube Kubernetes cluster.



* 1. Alternatively, we can use kubectl to forward the port:

kubectl port-forward service/hello-minikube 7080:8080

Now we can access the application at:

<http://localhost:7080/>

**ii)Load Balancer**

1. To use a LoadBalancer deployment, use the “minikube tunnel” command:

kubectl create deployment balanced --image=kicbase/echo-server:1.0

kubectl expose deployment balanced --type=LoadBalancer --port=8080

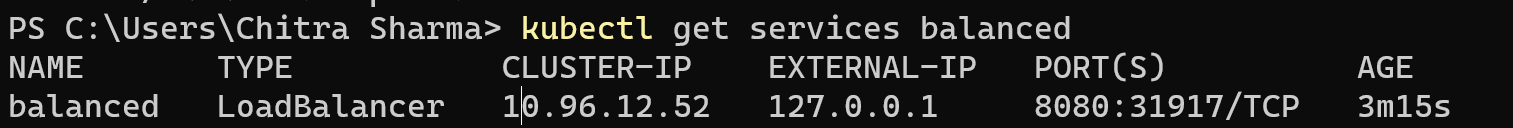
These two "kubectl" commands create a Kubernetes deployment named "balanced" with a specified Docker image and expose it as a LoadBalancer service on port 8080 within the cluster,

1. In another window, start the tunnel to create a routable IP for the ‘balanced’ deployment:

minikube tunnel

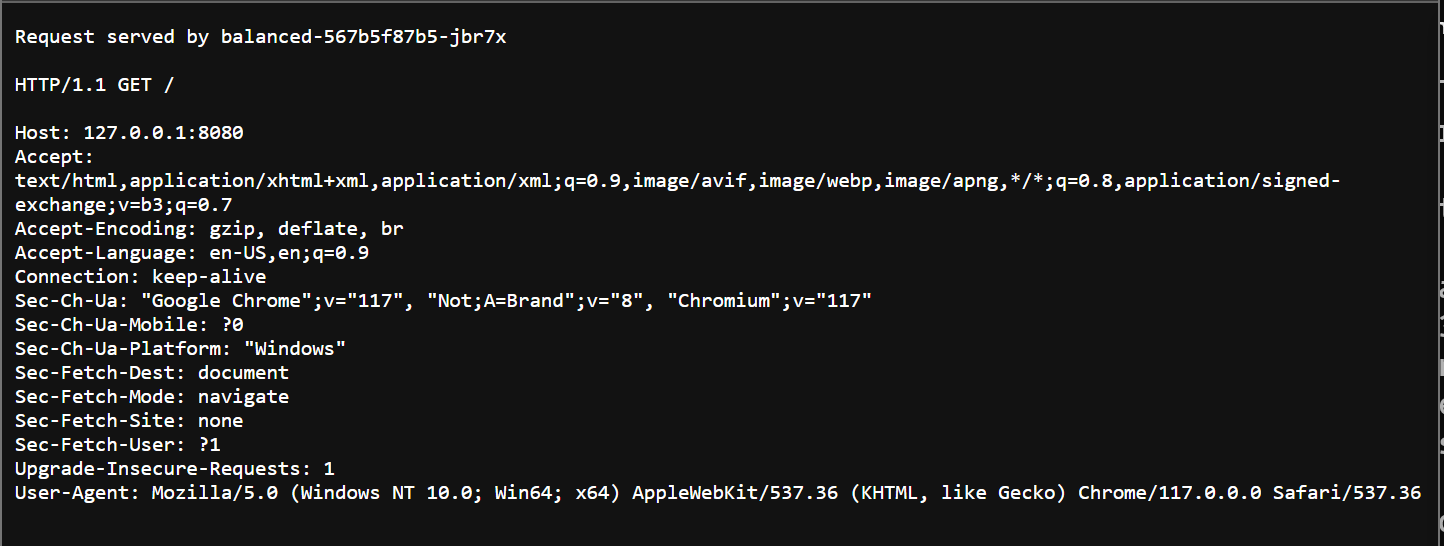
1. To find the routable IP, run this command and examine the EXTERNAL-IP column:

kubectl get services balanced



Your deployment is now available at <EXTERNAL-IP>:8080

Here in my case the External IP will be replaced by : 127.0.0.1



**Manage Your Cluster**

1. To pause Kubernetes without impacting deployed applications, run:

minikube pause

1. To unpause the paused instance, run:

minikube unpause

1. To halt the cluser, run:

minikube stop

1. To change the default memory limit(requires to restart), run:

minikube config set memory 9001

1. To browse the catalog of easily installed Kubernetes services,run:

minikube addons list

1. To create a second cluser running an older Kubernetes release, run:

minikube start -p aged --kubernetes-version=v1.16.1

1. To delete all the minikube clusters, run:

minikube delete –all