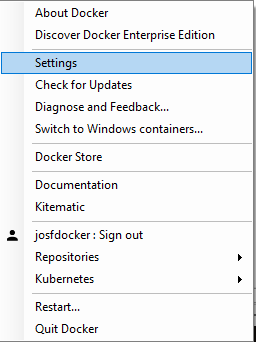
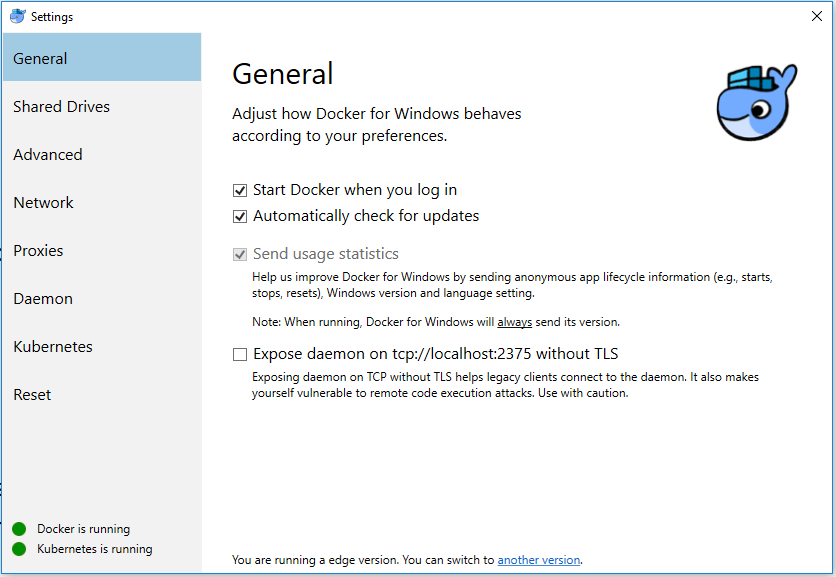
Prerequisites:

1. Docker for Windows Edge Version Latest
2. Minikube
3. Kubernetes
4. [Chocolatey](https://chocolatey.org/)
5. Hyper-V

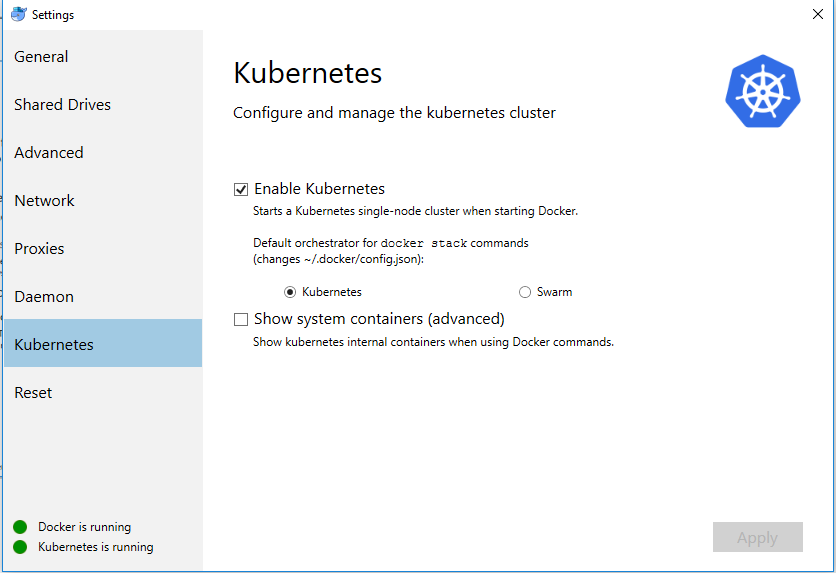
Steps:

1. Install Docker for windows Edge Version Latest
2. Once its Installed and start the Docker for windows
3. Switch to Linux container
4. Right click on Docker icon in taskbar -> click on settings





1. Click on Kubernetes and modify as below and click on apply



Kubernetes Dashboard Installation procedure:

Running the following Commands one after another

1)

kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/master/src/deploy/alternative/kubernetes-dashboard.yaml

2)

kubectl create -f https://raw.githubusercontent.com/kubernetes/heapster/master/deploy/kube-config/influxdb/influxdb.yaml

3)

kubectl create -f https://raw.githubusercontent.com/kubernetes/heapster/master/deploy/kube-config/influxdb/heapster.yaml

4)

kubectl create -f https://raw.githubusercontent.com/kubernetes/heapster/master/deploy/kube-config/influxdb/grafana.yaml

1. Once all the commands are executed without any error. Please run the following command

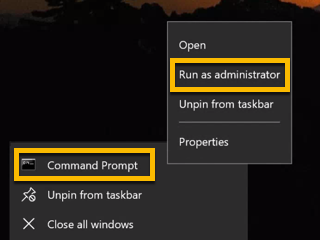
kubectl proxy

1. Then Visit the following link, Kubernetes Dashboard will be visible

<http://127.0.0.1:8001/api/v1/namespaces/kube-system/services/http:kubernetes-dashboard:/proxy/#!/namespace?namespace=_all>

**Installing the Chocolatey Package manager for Windows**

1. Open a command line window as administrator.

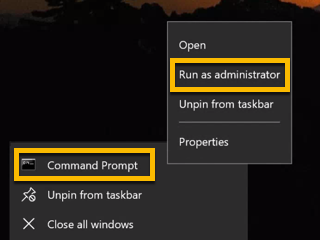


1. Then install Chocolatey using the Windows command line prompt using the following command.

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

**Installing the Minikube for Windows**

Open a command line window with administrator privileges.



Then use the chocolatey package manager to install minikube.

choco install minikube

**Installing the kubernetes for Windows**

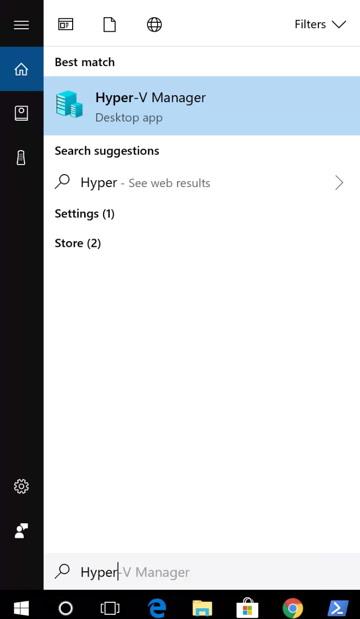
Run the following command to install Kubectl

choco install kubernetes-cli

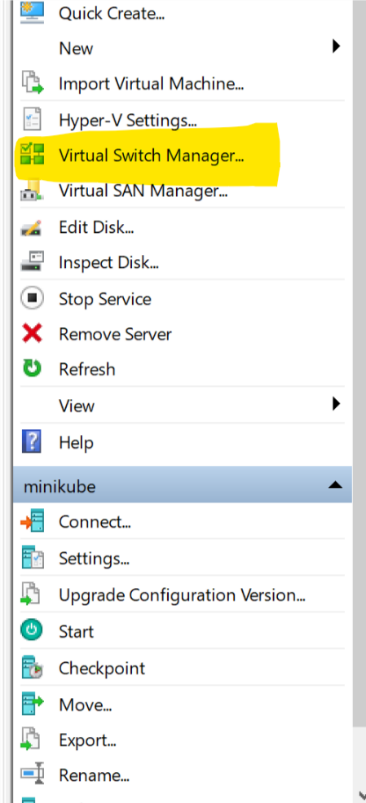
Configuring Hyper V:

Once that is installed you will need to make some changes to the Windows Hyper-V manager.

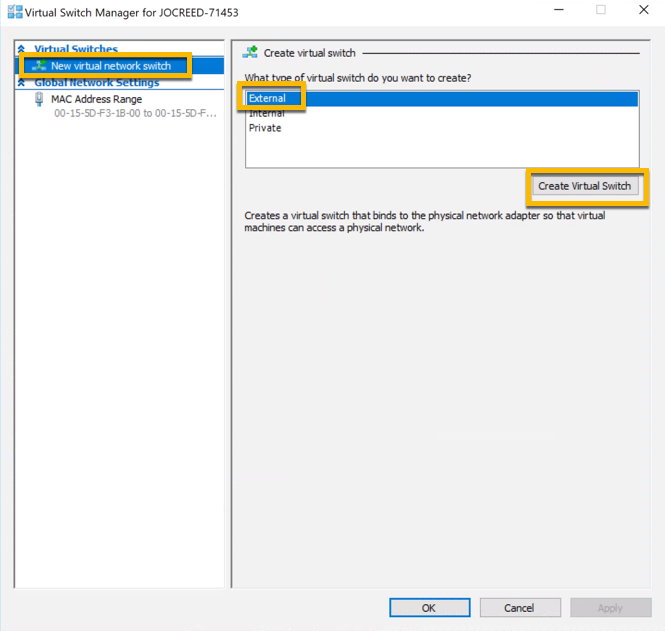
Open the Hyper-V Manager.



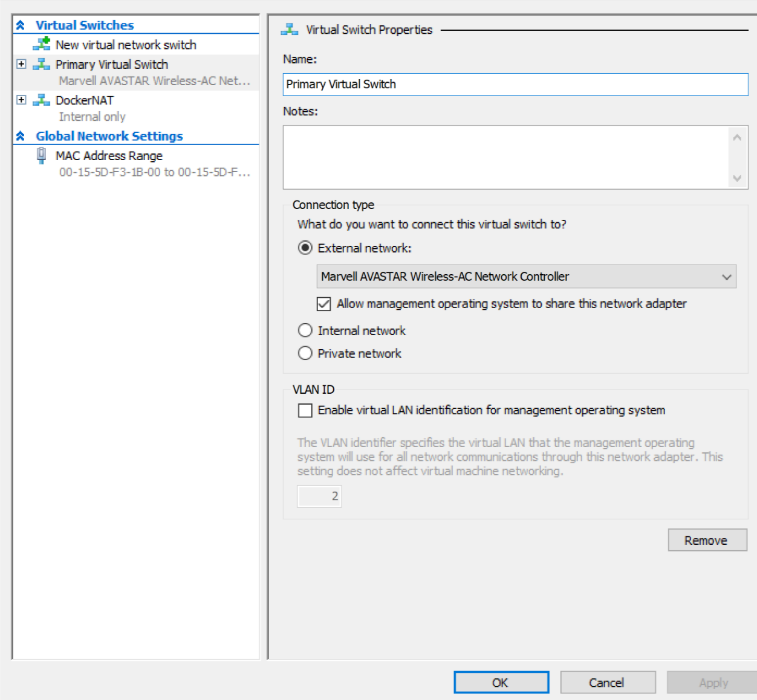
Once in the Hyper-V Manager, on the right panel, select the Virtual Switch Manager.



Next we will create a virtual switch for minikube. Select **New virtual network switch**on the right hand side, select **External**for the network type, and then press the **Create Virtual Switch** button.

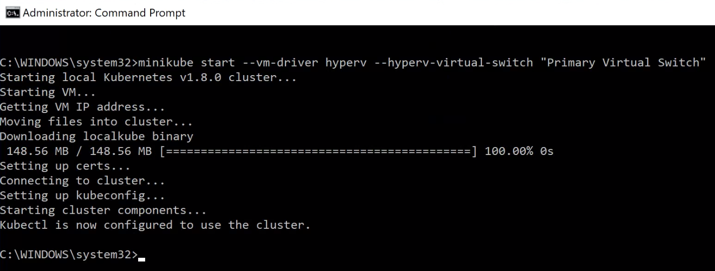


Name the switch **Primary Virtual Switch** and click the apply button.



Once you have the switch created we are now ready to start minikube. Run the following command to start the minikube VM with our applied changes.

minikube start --vm-driver hyperv --hyperv-virtual-switch "Primary Virtual Switch"

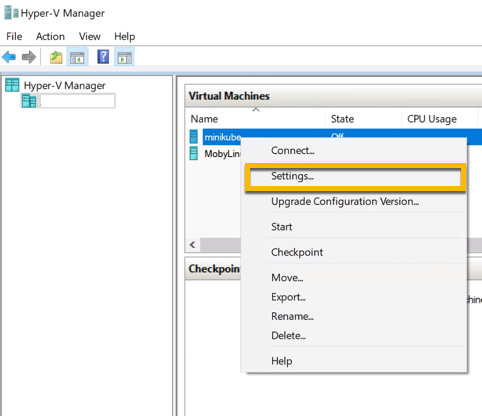


Once the VM is running we will have two more steps to do to address a bug in minikube for Windows. We need to turn off Dynamic Memory for the minikube VM. There is bug reported on this issue which can be [**found here**](https://github.com/kubernetes/minikube/issues/2326)**.** Once this bug has been resolved, I will update the instructions here.

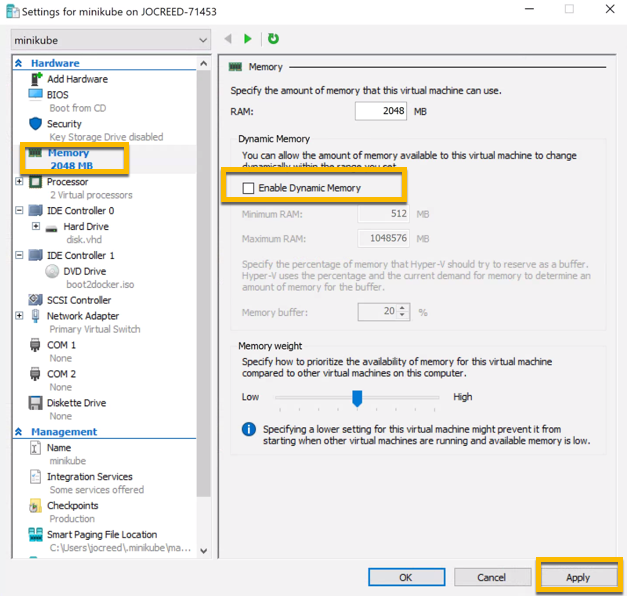
In your administrator command line window stop the minikube VM with the following command.

minikube stop

Once minikube has stopped, open the Hyper-V Manager again and right click on the minikube VM and select settings.



Select the **Memory** option on the left panel, then de-select **Enable DynamicMemory**, and then click **Apply** button.



Close Hyper-V manager and then return to the administrator command line window and restart minikube.

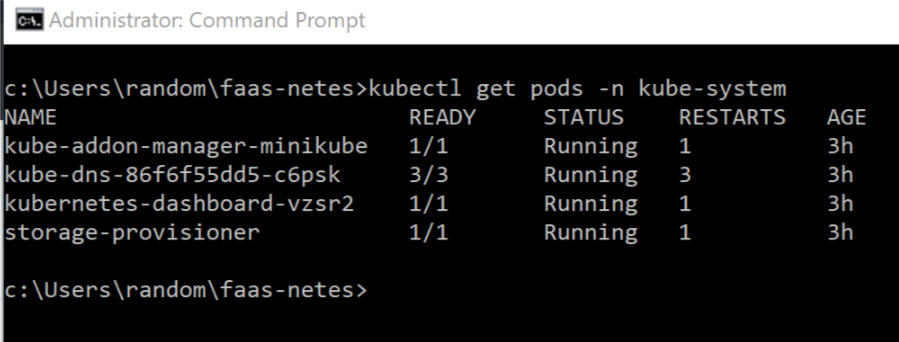
minikube start

Once minikube is restarted you should be ready to go. We will confirm the installation is ready by checking a couple of things.

In the command line window, enter the following command.

kubectl get pods -n kube-system

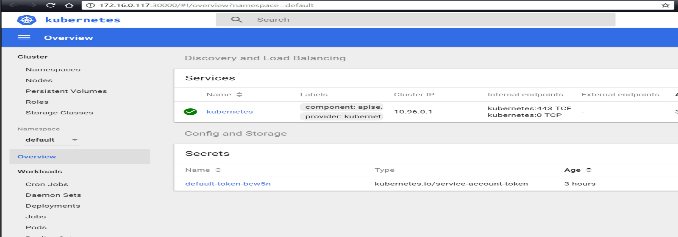
You should see the following result.



If you are a little more visual and want to see the kubernetes dashboard, you can open that up by typing in the following command in the command line window.

minikube dashboard

That will open up the dashboard window in your default browser. Now you are ready to explore and start your journey to learning Kubernetes.



**Build Application’s Docker Image**

1. Create the application by using the following command

dotnet new razor -o kubeaspnetapp

1. Add the file named “Dockerfile” with out filetype with following content inside kubeaspnetapp folder.

FROM microsoft/dotnet:2.1-sdk AS build

WORKDIR /app

# copy csproj and restore as distinct layers

COPY \*.csproj ./

RUN dotnet restore

# copy everything else and build

COPY . ./

RUN dotnet publish -c Release -o out

# build runtime image

FROM microsoft/dotnet:2.1-aspnetcore-runtime AS runtime

WORKDIR /app

COPY --from=build /app/out .

ENTRYPOINT ["dotnet", "kubeaspnetapp.dll"]

1. Create the Docker Image by executing following command.

docker build -t josfdocker/dev-kubeaspnet:latest .

4)Push the docker image using following Command.

docker push josfdocker/dev-kubeaspnet:latest

minikube start --vm-driver="virtualbox" --insecure-registry="0.0.0.0:5000"

1. Create the deployment in Kubernetes using the following command.

kubectl run josephjoychristopherg --image=josfdocker/dev-kubeaspnet:latest --port=80

1. Now expose the deployment via service by using following command.

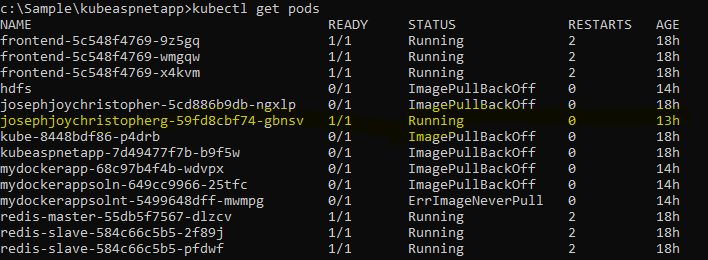
kubectl expose deployment josephjoychristopherg --type=NodePort

1. Now check whether the service is listed by running the following command.

kubectl get services

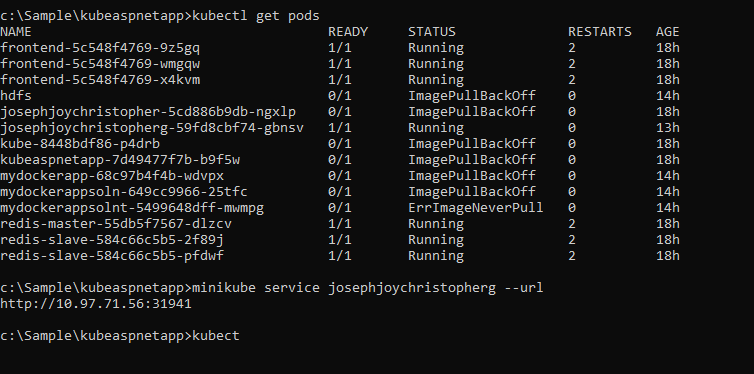
1. Now check whether the pod is listed and running fine without errors by running the following command.

kubectl get pods



1. Now try to get the url of the exposed service via minikube. Kindly run the following command.

minikube service josephjoychristopherg --url



1. To get the deployment yaml file from deployment. Ex. where devtesterla is the deployment name

kubectl get deploy devtesterla -o yaml –export

Utilities Commands

To List the kubectl contexts available in your pc

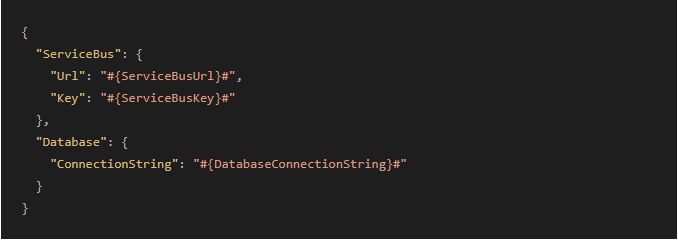
kubectl config get-contexts

to change the kubectl context

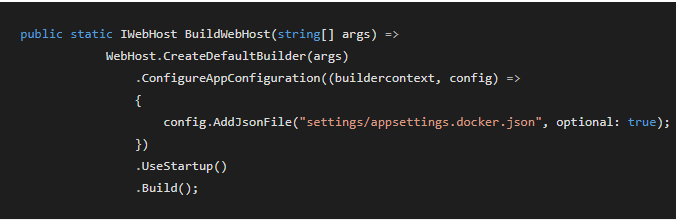
kubectl config use-context docker-for-desktop

**Storing and securing application configuration details via Kubernetes Secrets:**

1. Create a folder named “settings” in the project root.
2. Create a file named “appsettings.docker.json” and place it under settings folder which we created in the step 1.
3. Please mention the following content onto the “appsettings.docker.json” file.



1. In the Program.cs mentioned the following to make sure that the “appsettings.docker.json” is loaded as optional.



1. Create the secret in kubernetes via following command:

kubectl create secret generic settingsdemo-docker-appsettings --from-file=./appsettings.docker.json

1. The following deployment file deploys a Docker Image and mounts a volume to the secret file.

apiVersion: extensions/v1beta1

kind: Deployment

metadata:

name: settingsdemo-deploymentv3

spec:

replicas: 3

template:

metadata:

labels:

app: settingsdemov3

spec:

containers:

- image: josfdocker/mysecretv3:latest

imagePullPolicy: Always

env:

- name: ASPNETCORE\_ENVIRONMENT

value: "Development"

- name: ServiceBus\_\_Url

value: "myServiceBus.servicebus.windows.net"

- name: ServiceBus\_\_Key

value: "abcdefghijklmnopqrstuvwzyz"

- name: Database\_\_ConnectionString

value: "mySqlConnectionstring"

name: settingsdemo

ports:

- containerPort: 80

imagePullPolicy: Always

volumeMounts:

- name: settings

mountPath: /app/settings

readOnly: true

volumes:

- name: settings

secret:

secretName: settingsdemo-docker-appsettings

1. To create the deployment via yaml file. Run the following command.

kubectl create -f c:\sample\yaml\testla.yaml

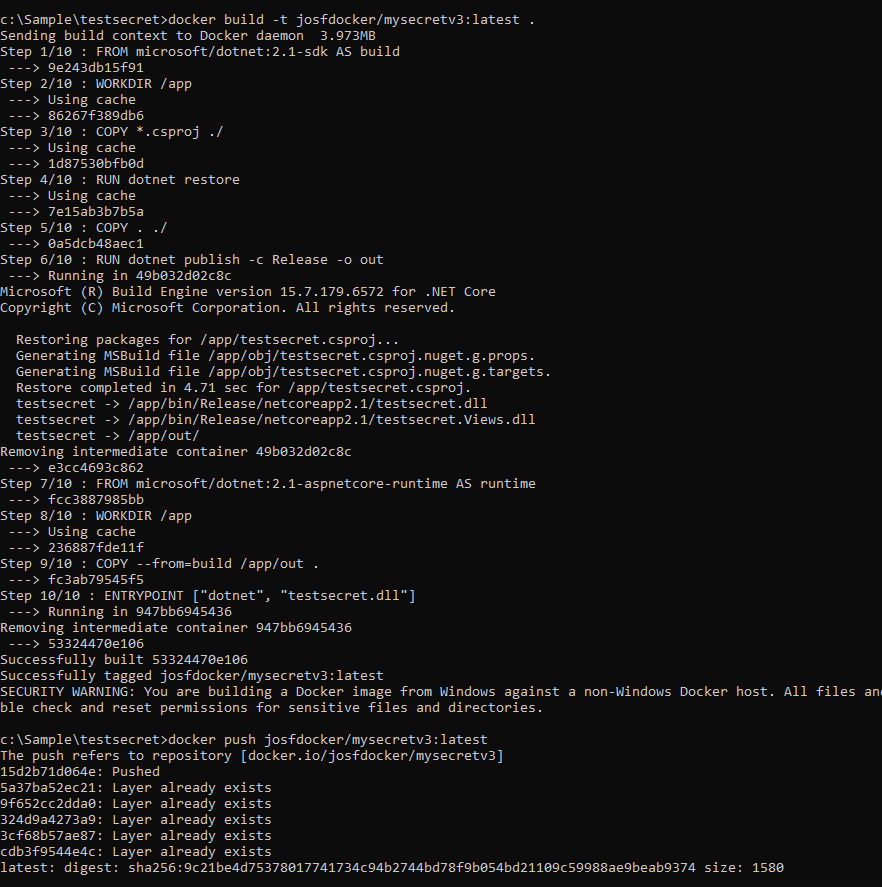
1. Now, expose the deployment as service by executing the following command:

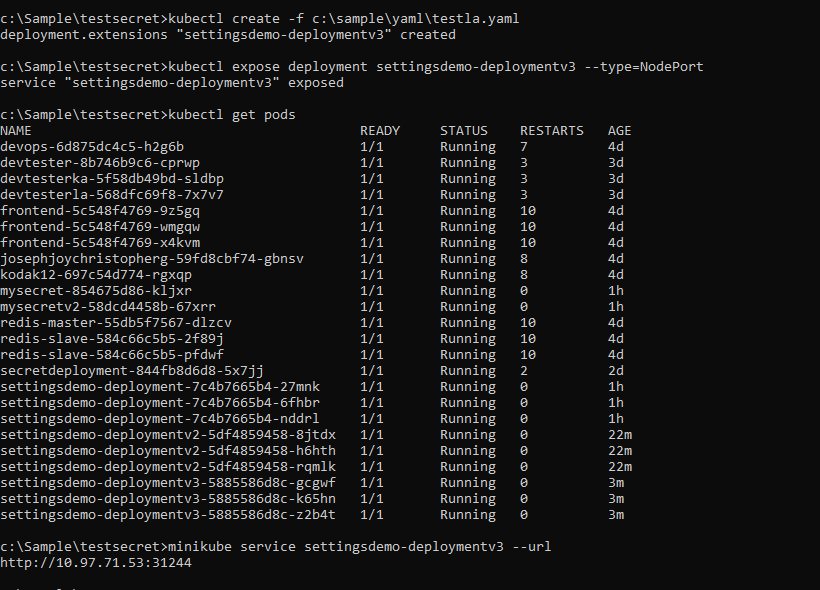
kubectl expose deployment settingsdemo-deploymentv3 --type=NodePort

1. Get the public url to access the deployed the app by executing the following command:

minikube service settingsdemo-deploymentv3 --url

Example execution o/p:





Sample App’s Home Controller:

public class HomeController : Controller

{

public IConfiguration Configuration { get; set; }

public HomeController(IConfiguration config)

{

Configuration = config;

}

public IActionResult Index()

{

ViewBag.status = Configuration is null;

ViewBag.TestConnection = Configuration["Database:ConnectionString"];

return View();

}

}

}

Sample App’s View Controller:

<div class="item active">

<img src="~/images/banner1.svg" alt="ASP.NET" class="img-responsive" />

<div class="carousel-caption" role="option"> <p>

was Configuration null : @ViewBag.status

testconnection:@ViewBag.TestConnection

<a class="btn btn-default" href="https://go.microsoft.com/fwlink/?LinkID=525028&clcid=0x409">

Learn More </a> </p> </div>

