Contents

[1 Azure container Registry and creation 1](#_Toc76752489)

[2 Docker setup on local machine 3](#_Toc76752490)

[3 Create sample .net core MVC app with docker file 4](#_Toc76752491)

[3 Create and Push image to azure container registry 5](#_Toc76752492)

[4 AKS – create and deploy ACR image to cluster/pod 7](#_Toc76752493)

[5 POD : SCALING 12](#_Toc76752494)

[Additional Information 13](#_Toc76752495)

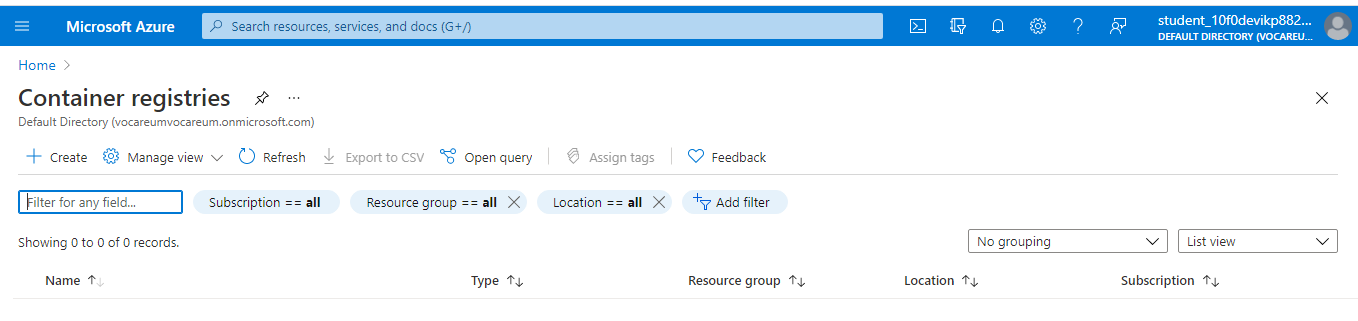
[Hosting image in container instance 13](#_Toc76752496)

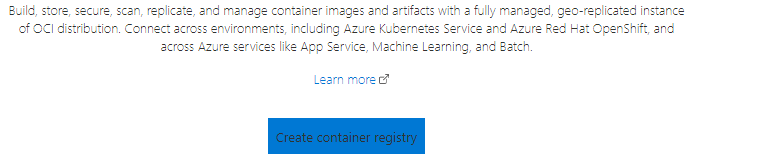
[Using Visual Studio: 16](#_Toc76752497)

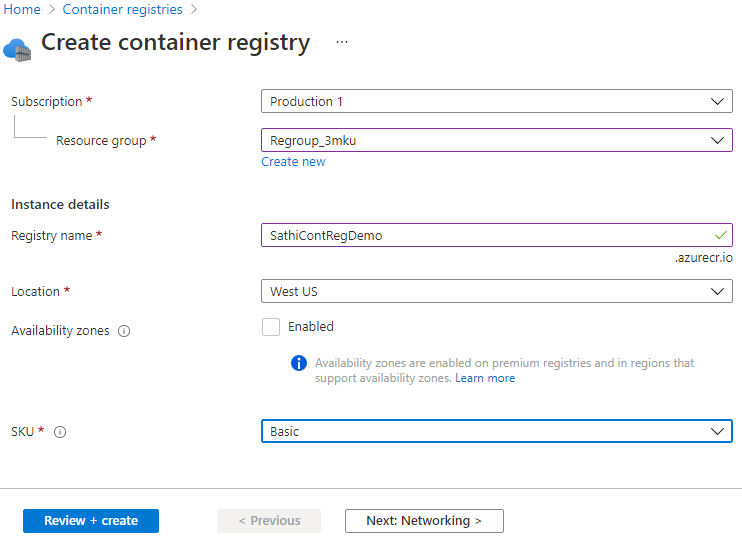
# 1 Azure container Registry and creation

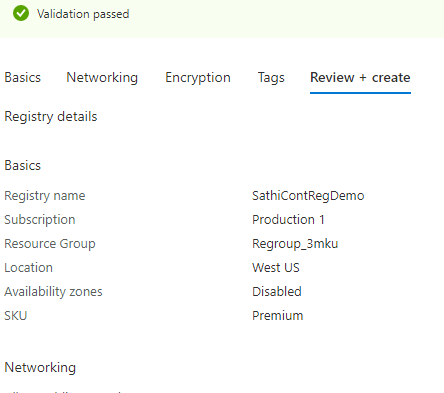
Azure Container Registry allows you to build, store, and manage container images and artifacts in a private registry for all types of container deployments. Use Azure container registries with your existing container development and deployment pipelines. Use Azure Container Registry Tasks to build container images in Azure on-demand, or automate builds triggered by source code updates, updates to a container's base image, or timers. *Documentation* : <https://docs.microsoft.com/en-us/azure/container-registry/>

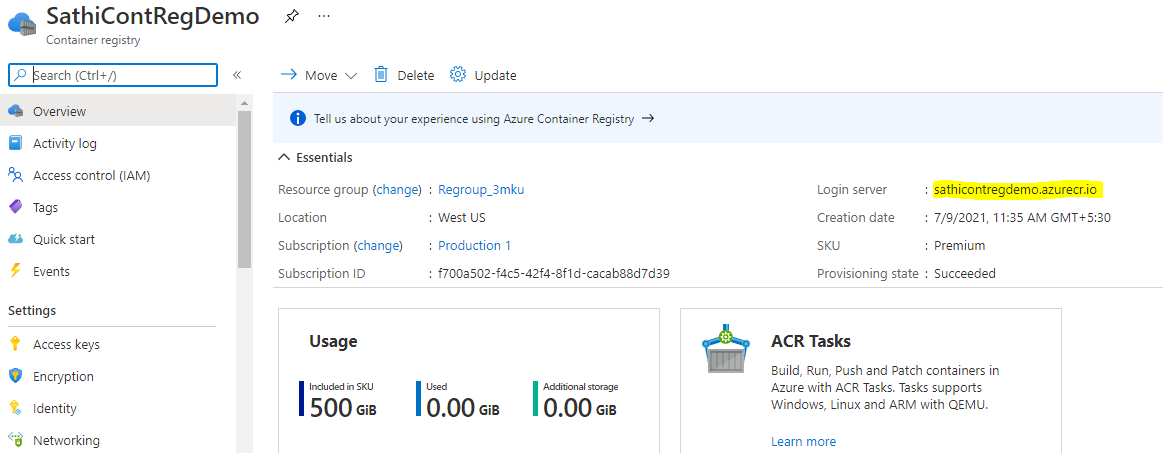
Create container Register

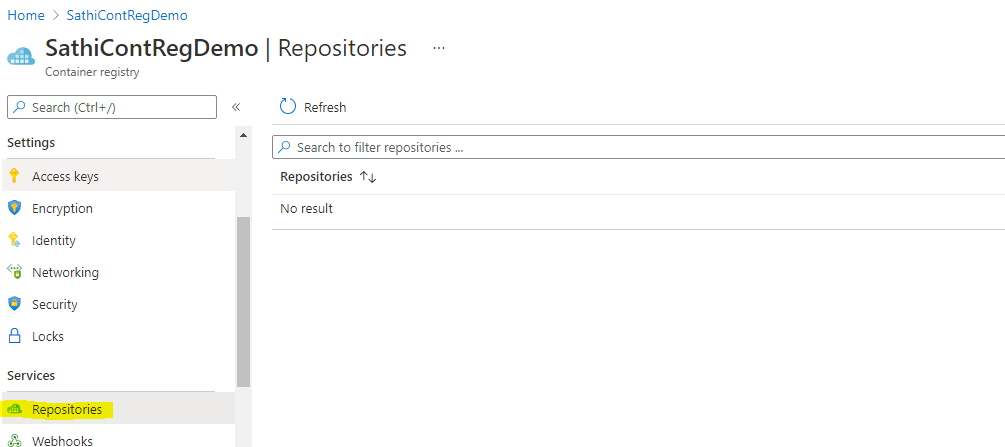








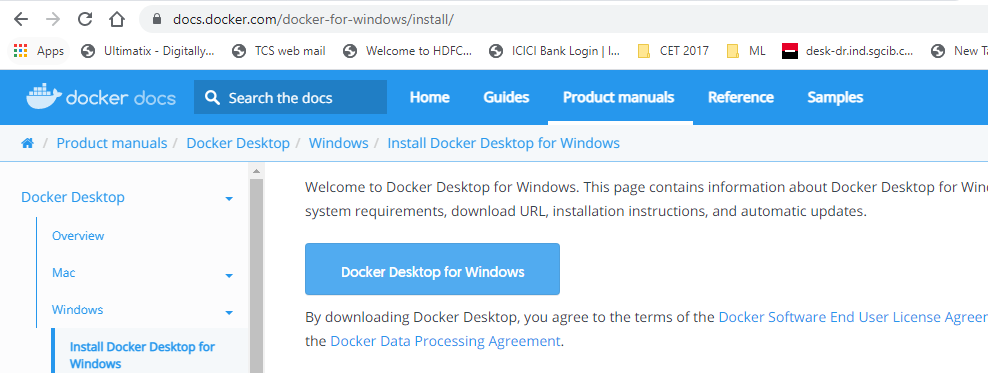




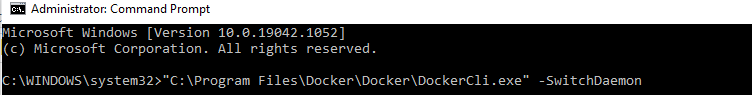
# 2 Docker setup on local machine

<https://docs.microsoft.com/en-us/windows/wsl/install-win10>  
<https://docs.docker.com/docker-for-windows/install/>  
<https://docs.docker.com/get-started/>

Download and Install wsl and docker



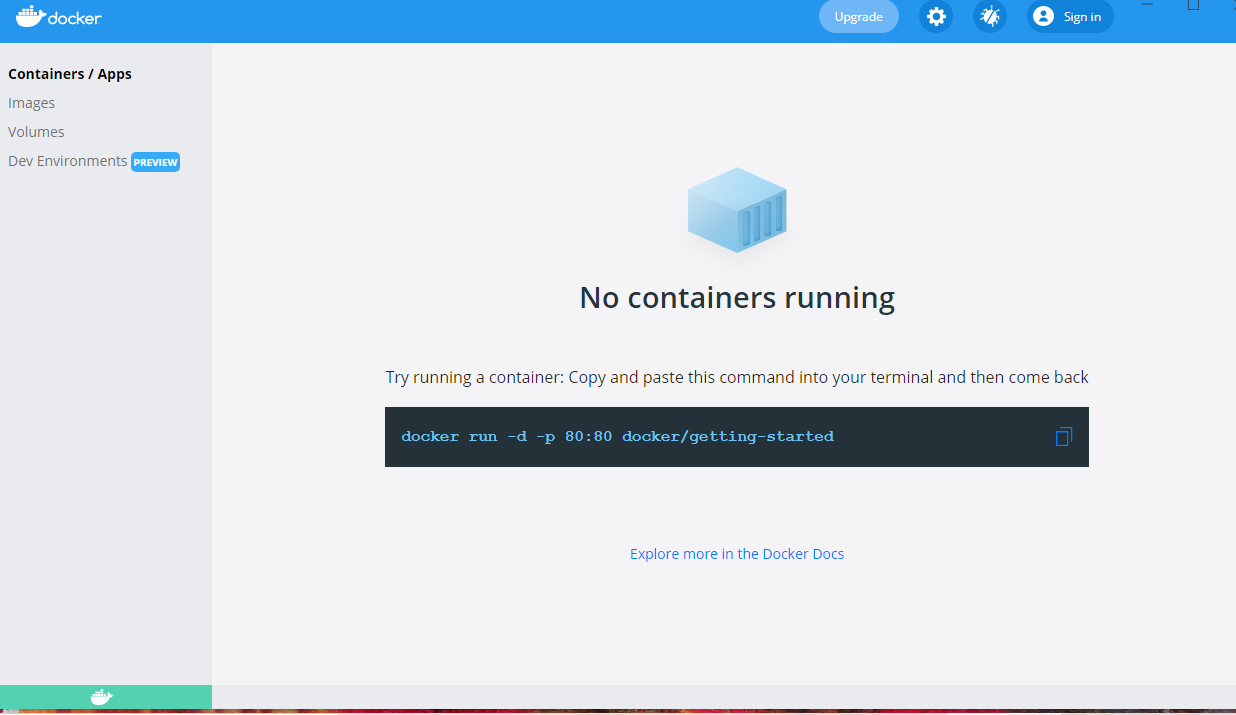
Run Daemon



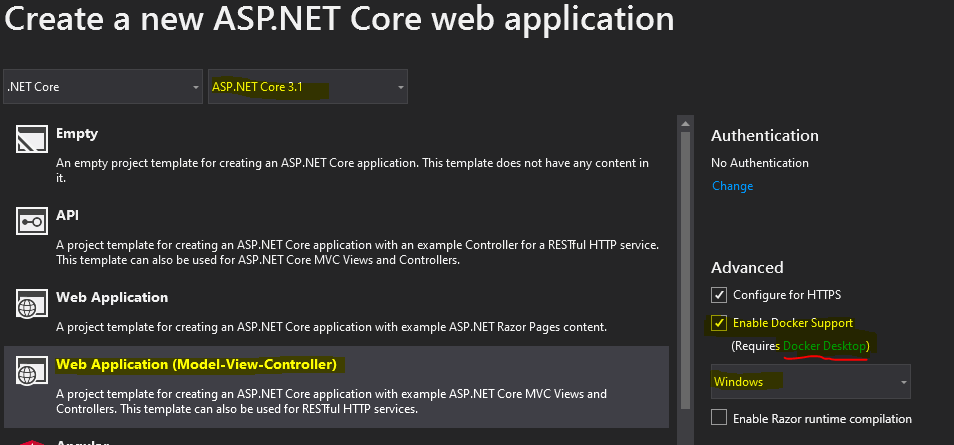
Check Version

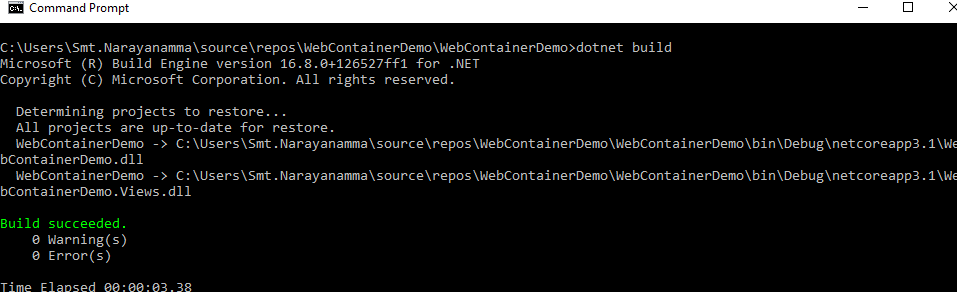


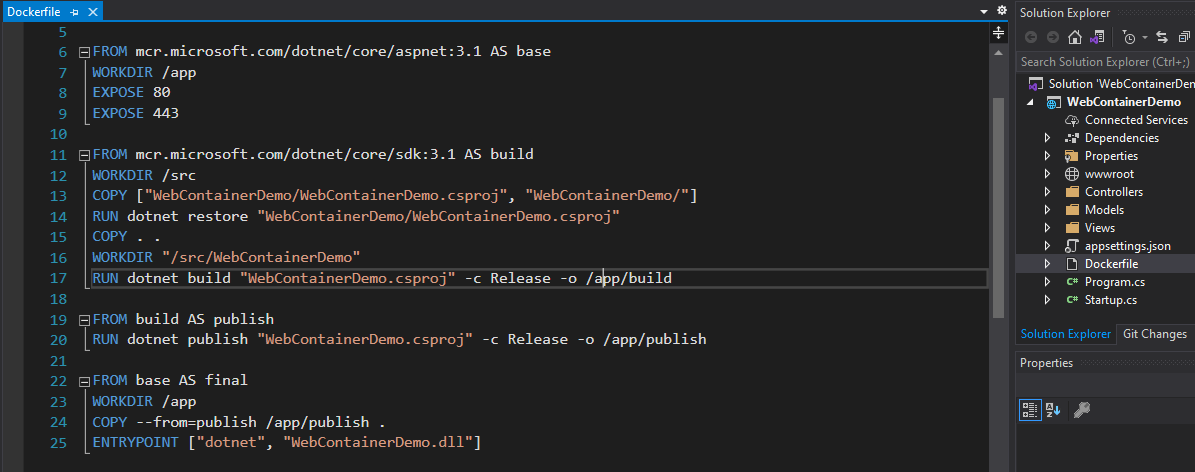
Run –Green indicator shows its successsfully runing



# 3 Create sample .net core MVC app with docker file





DOCKER FILE

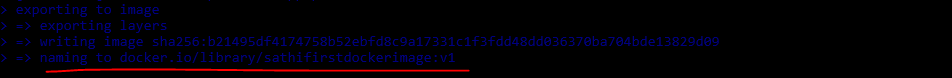
# 3 Create and Push image to azure container registry

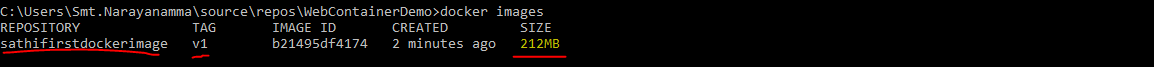


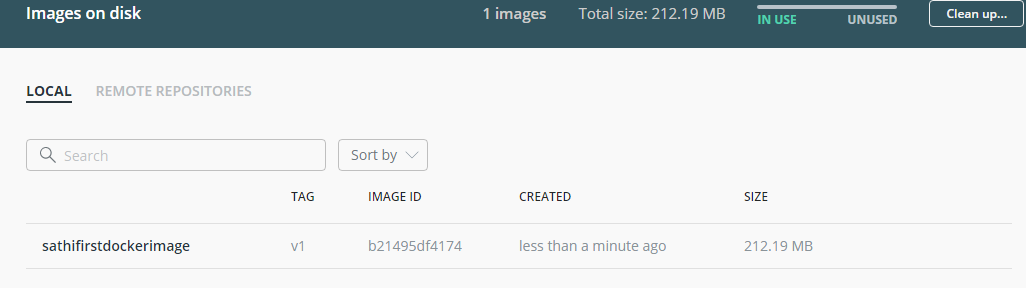


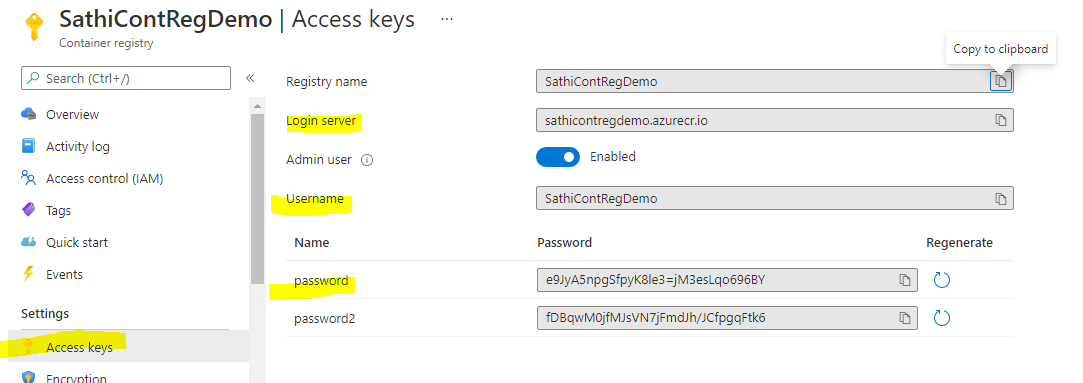
When docker file and solution are not in same folder , naviagte to solution level folder and specfiy docker file path eg: *-f WebContainerDemo/Dockerfile* , if not you wiill get cache key error . Also make sure that your docker is running and internet is up to download metadata from microsoft . *build.* says that source is from same folder *–t sathisfirstdockerimage* will be image name , v1 will be version tag. First time build should take good time as it will download .net run time(212MB) along with our project which is prerequest /dependency



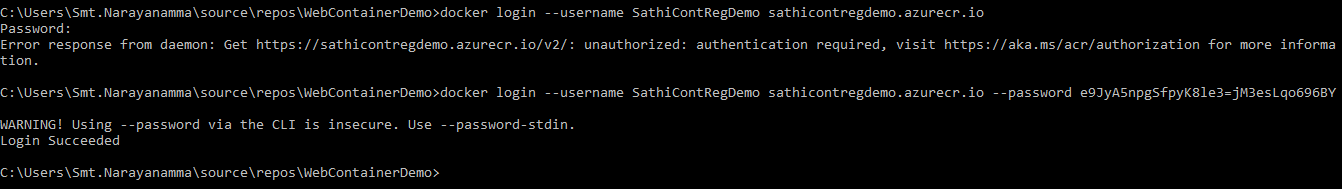








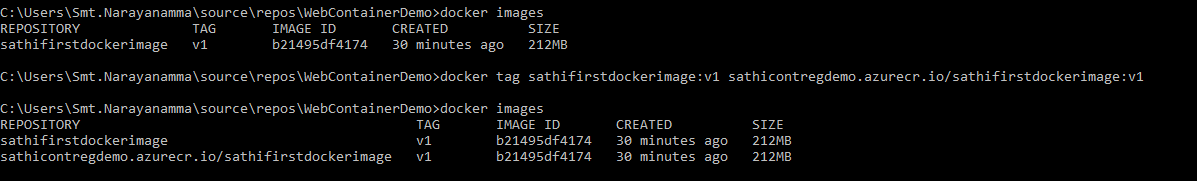
docker login –-username SathiContRegDemo sathicontregdemo.azurecr.io



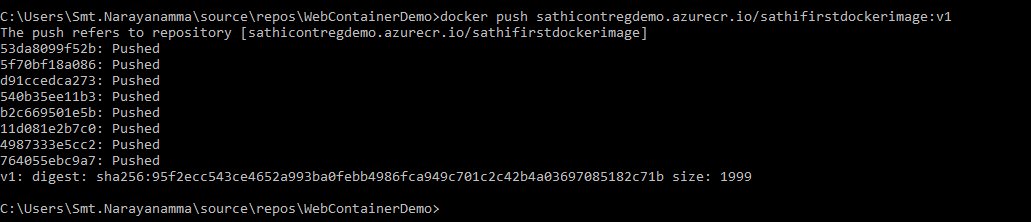
Tag the image b/n local and remote location

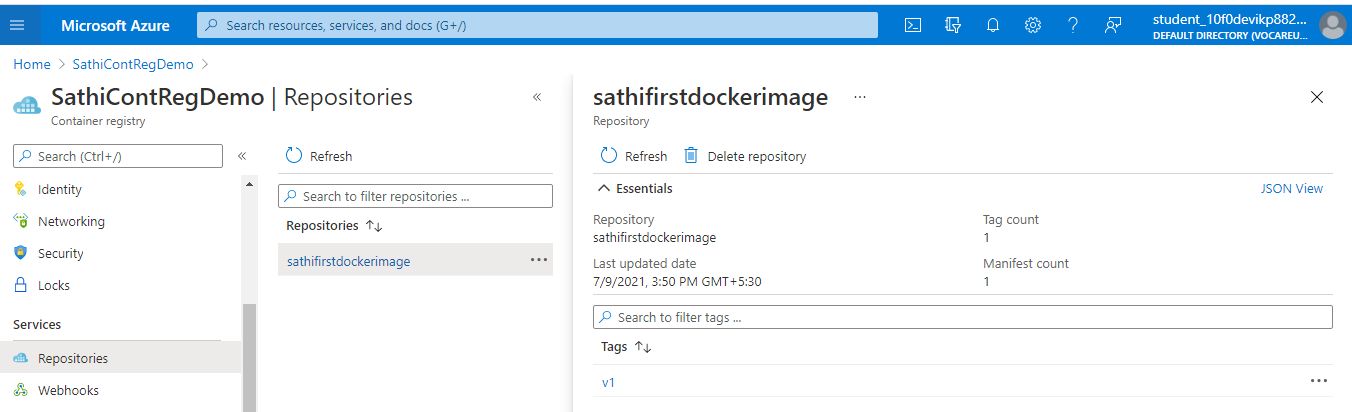


docker tag sathifirstdockerimage:v1 sathicontregdemo.azurecr.io/sathifirstdockerimage:v1



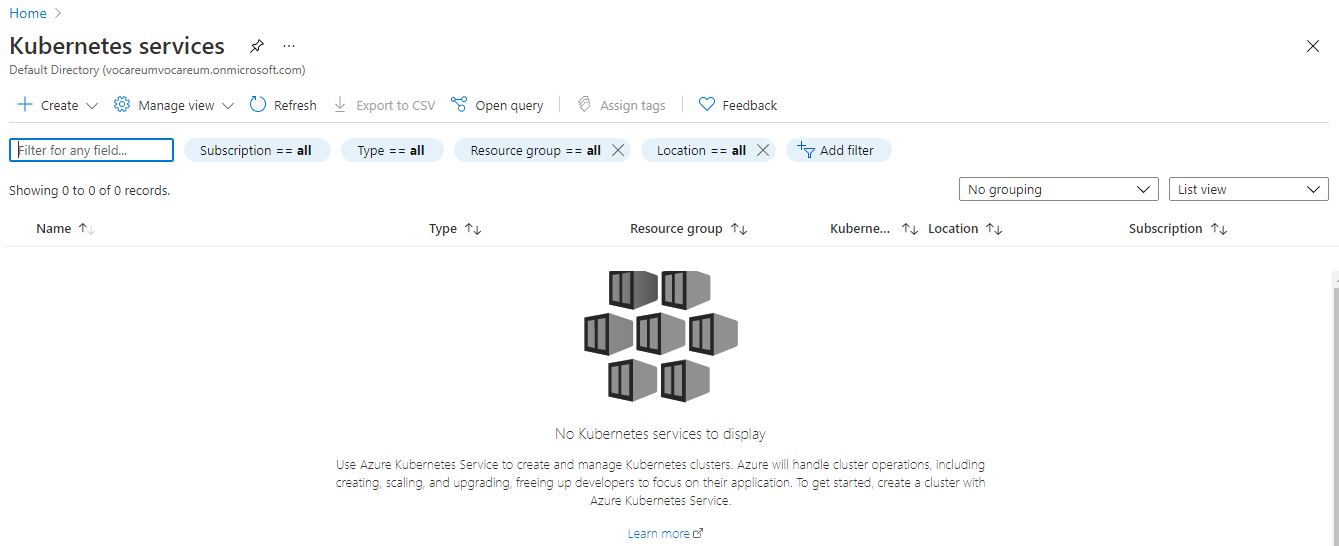
docker push sathicontregdemo.azurecr.io/sathifirstdockerimage:v1

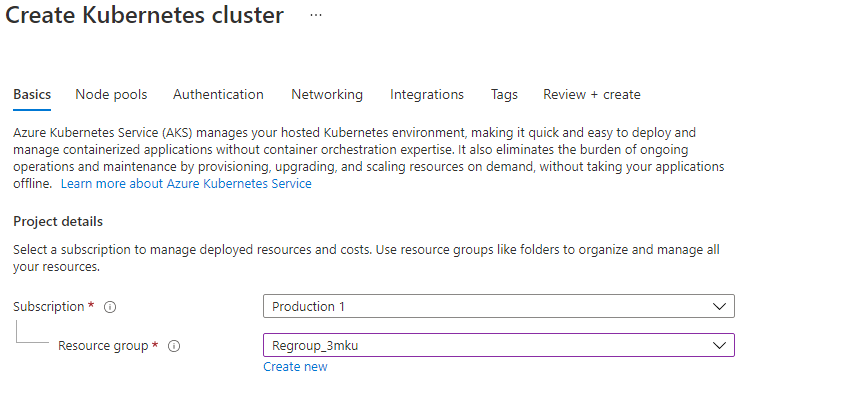


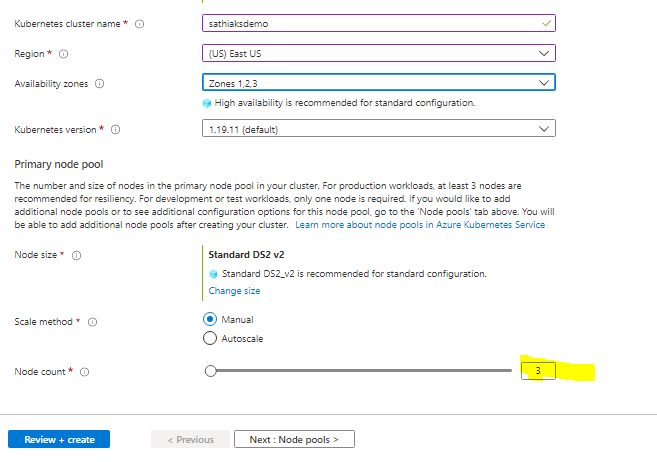


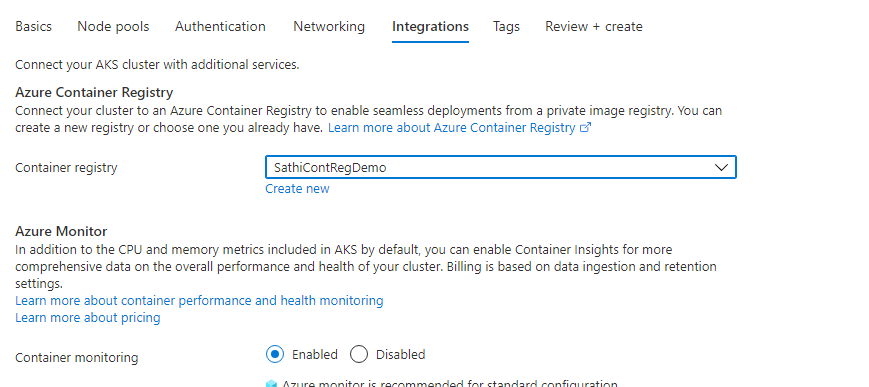
# 4 AKS – create and deploy ACR image to cluster/pod

Azure Kubernetes Service (AKS) is a managed Kubernetes service that lets you quickly deploy and manage clusters:   
**Kubernetes Service**.

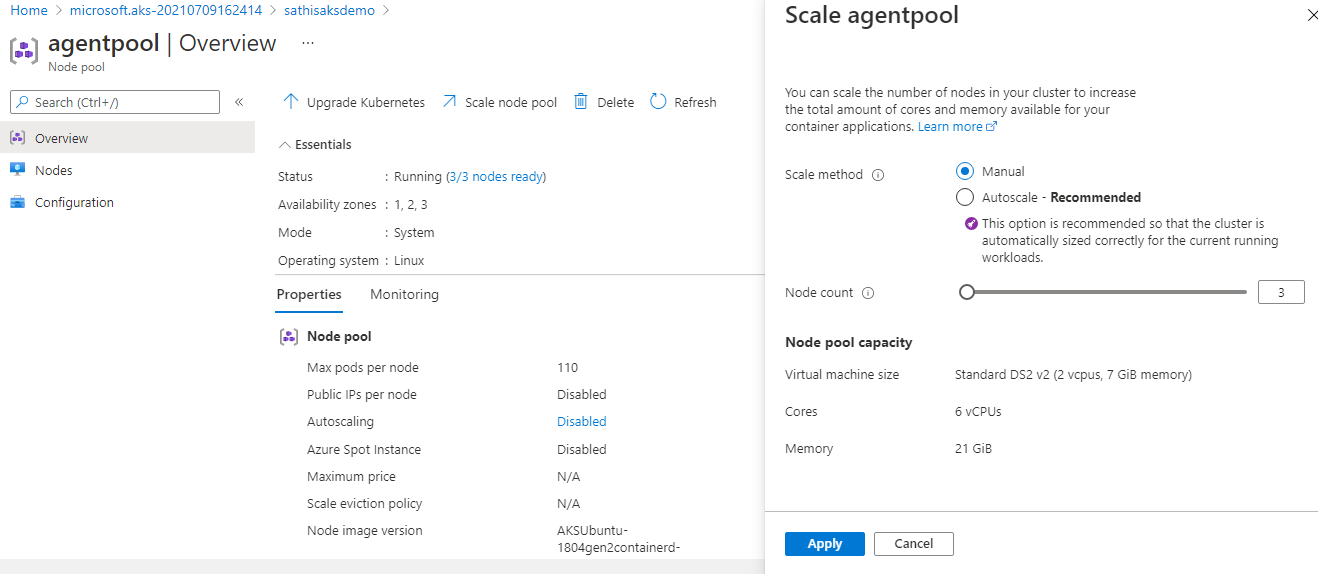








Disable above container moitoring as it not working



Create Yaml File : <https://docs.microsoft.com/en-us/azure/aks/kubernetes-walkthrough-portal#run-the-application>

*apiVersion: apps/v1*

*kind: Deployment*

*metadata:*

*name: sathifirstdockerimage*

*spec:*

*replicas: 1*

*selector:*

*matchLabels:*

*app: sathifirstdockerimage*

*template:*

*metadata:*

*labels:*

*app: sathifirstdockerimage*

*spec:*

*nodeSelector:*

*"beta.kubernetes.io/os": linux*

*containers:*

*- name: sathifirstdockerimage*

*image: sathicontregdemo.azurecr.io/sathifirstdockerimage:v1*

*resources:*

*requests:*

*cpu: 100m*

*memory: 128Mi*

*limits:*

*cpu: 250m*

*memory: 256Mi*

*ports:*

*- containerPort: 80*

*env:*

*- name: REDIS*

*value: "sathifirstdockerimage"*

*---*

*apiVersion: v1*

*kind: Service*

*metadata:*

*name: sathifirstdockerimage*

*spec:*

*type: LoadBalancer*

*ports:*

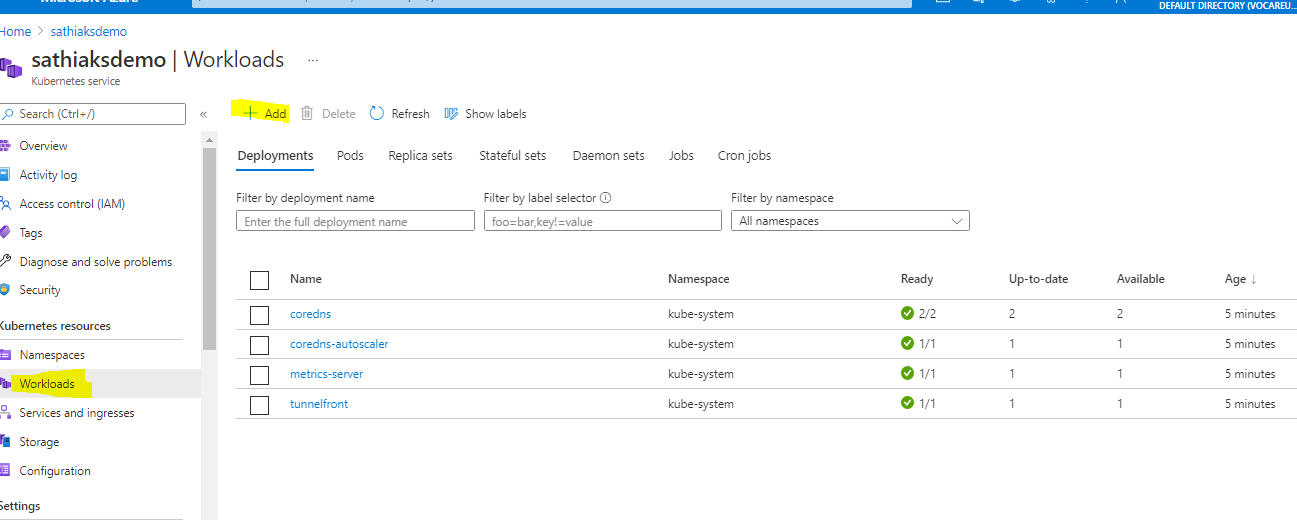
*- port: 80*

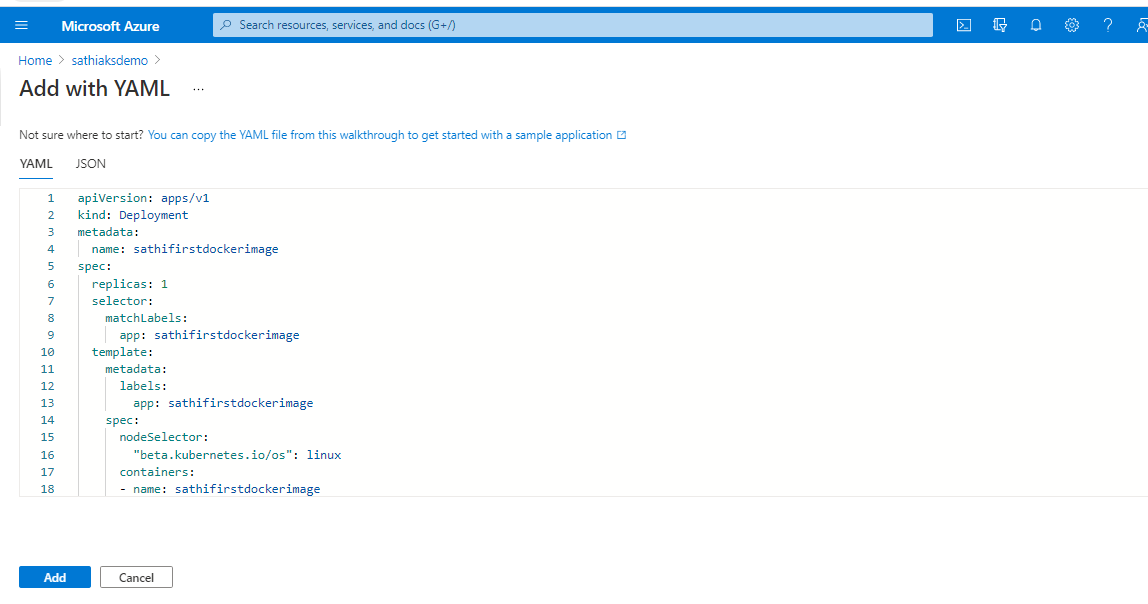
*selector:*

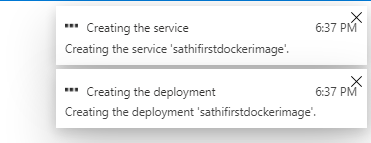
*app: sathifirstdockerimage*

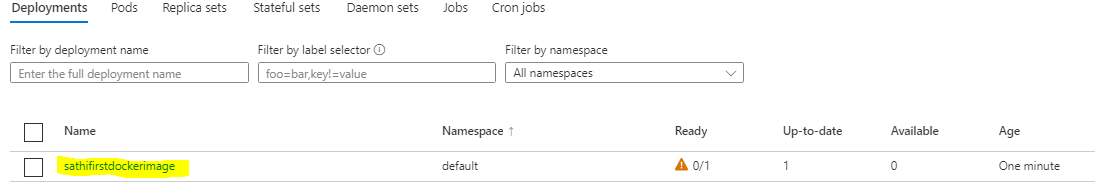
<https://docs.microsoft.com/en-us/azure/aks/tutorial-kubernetes-prepare-app>

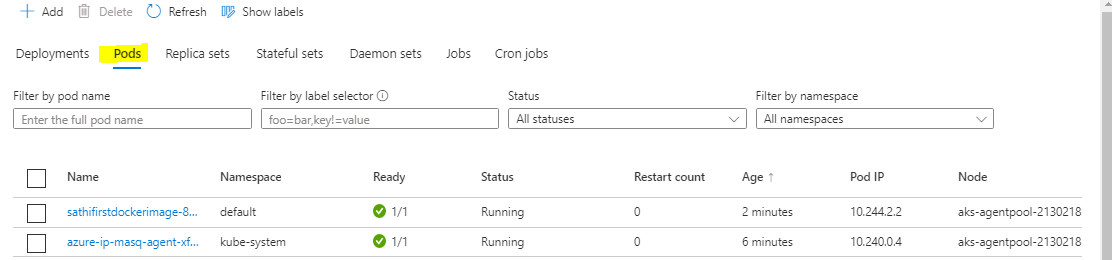
Deploy above yaml using portal as below or command kubectl apply -f *sathifirstdockerimage*.yaml

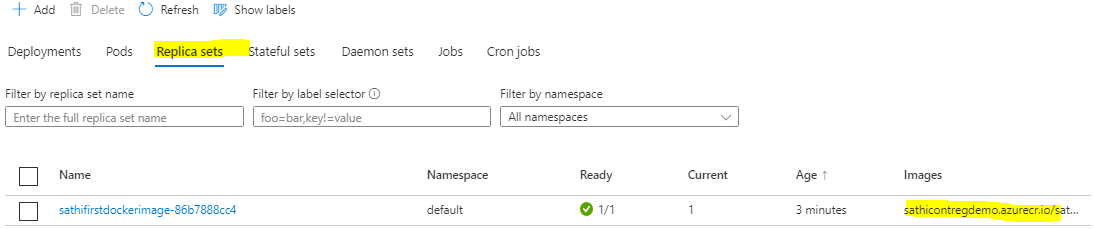


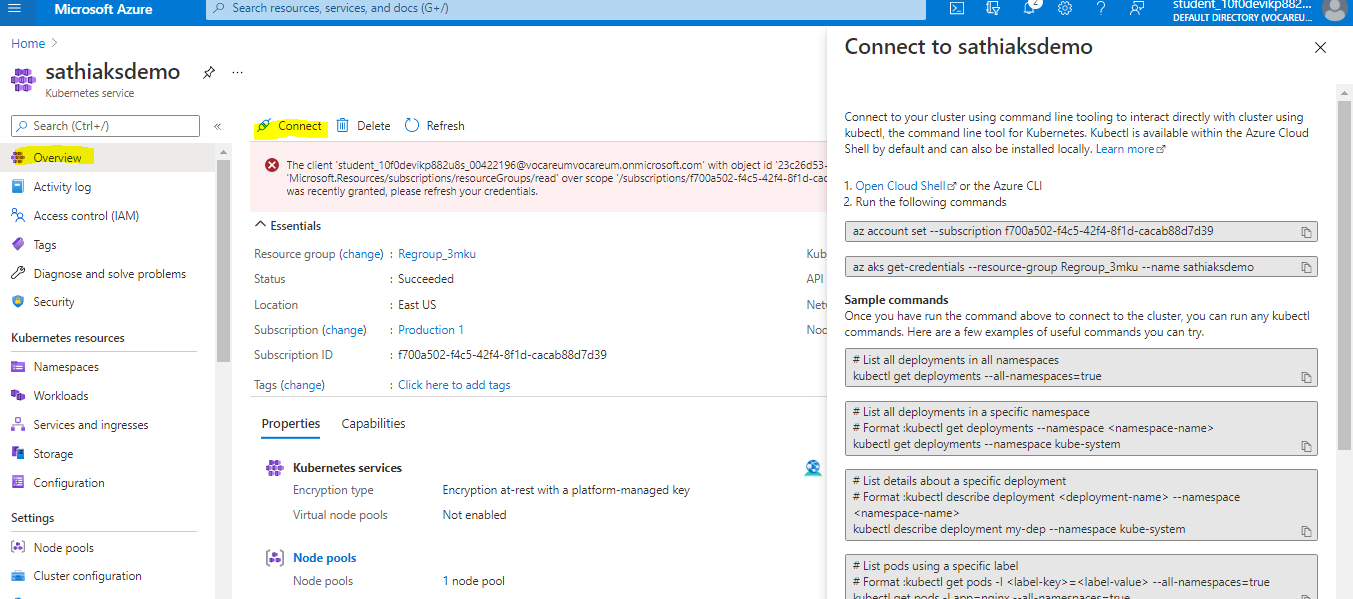




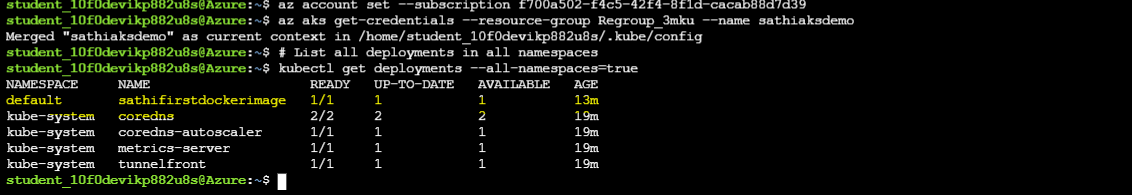






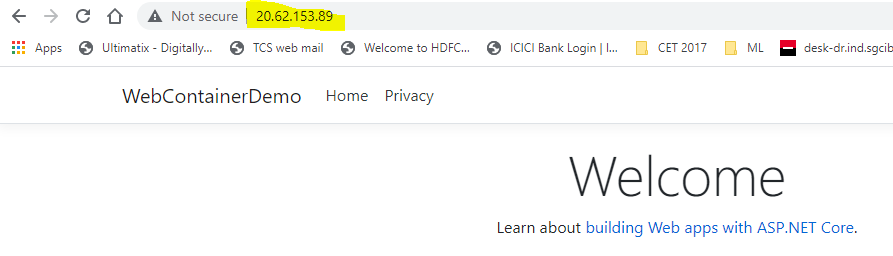


Connect To Azure CLI – (Enter required storage details)



kubectl get service *sathifirstdockerimage* --watch

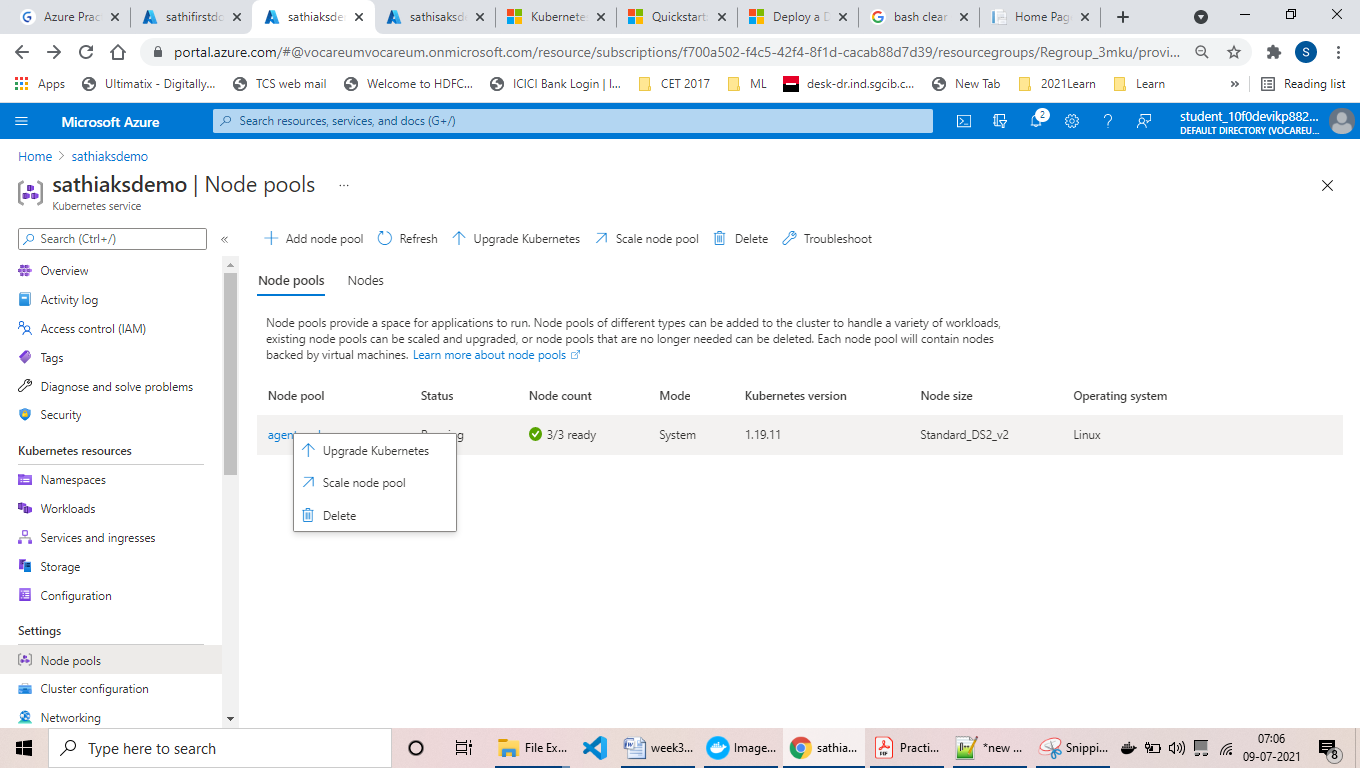


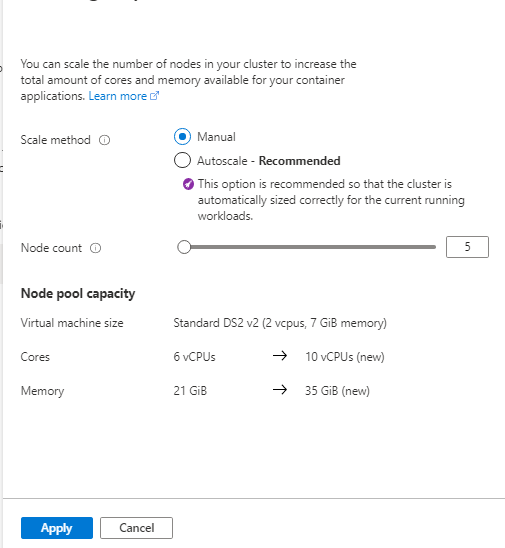


# 5 POD : SCALING

A pod is the smallest unit in Kubernetes that you create or deploy. A pod represents the application as a running process on your cluster.

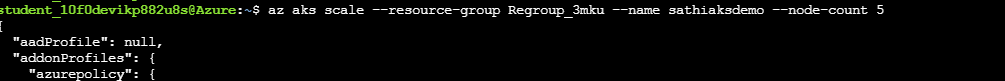
Manually scale the number of Kubernetes pods from 3 to 5

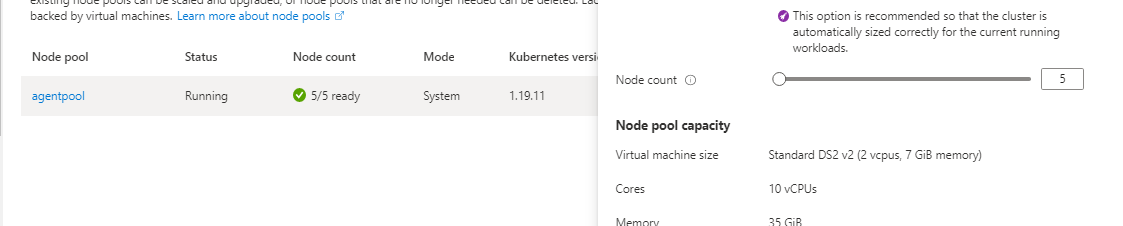




Do not apply above changes, we will apply via command

az aks scale --resource-group Regroup\_3mku --name sathiaksdemo --node-count 5



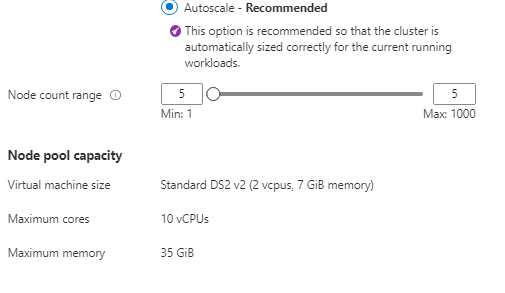


Create an autoscaling policy where the number of nodes is increased when the CPU

utilization of the cluster exceeds 80% of their capacity up to a maximum of 15 pods

Console command :

kubectl autoscale deployment sathifirstdockerimage --cpu-percent=80 --min=3 --max=15



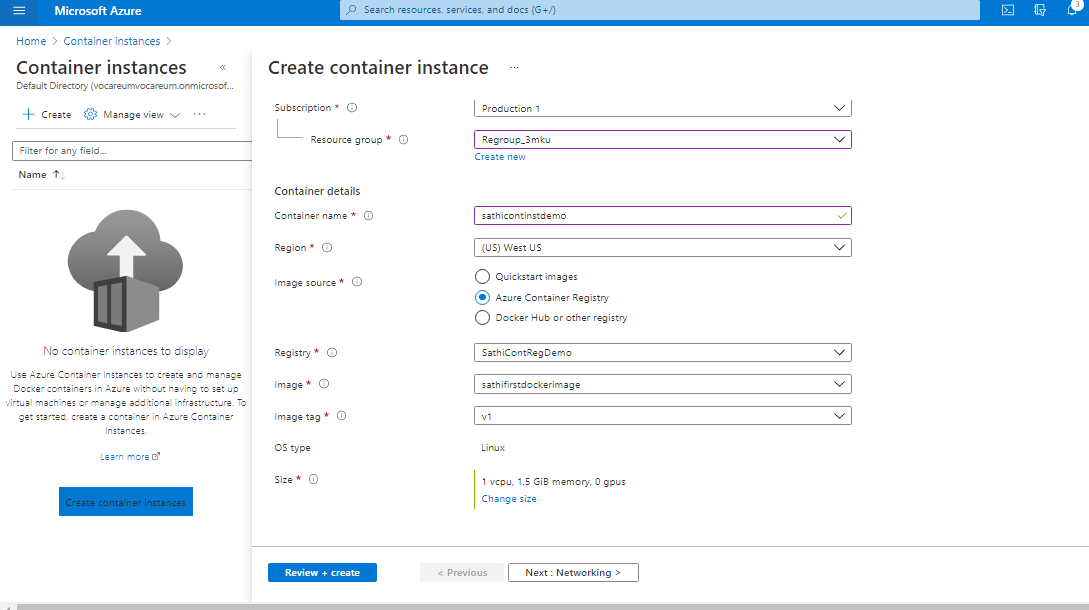


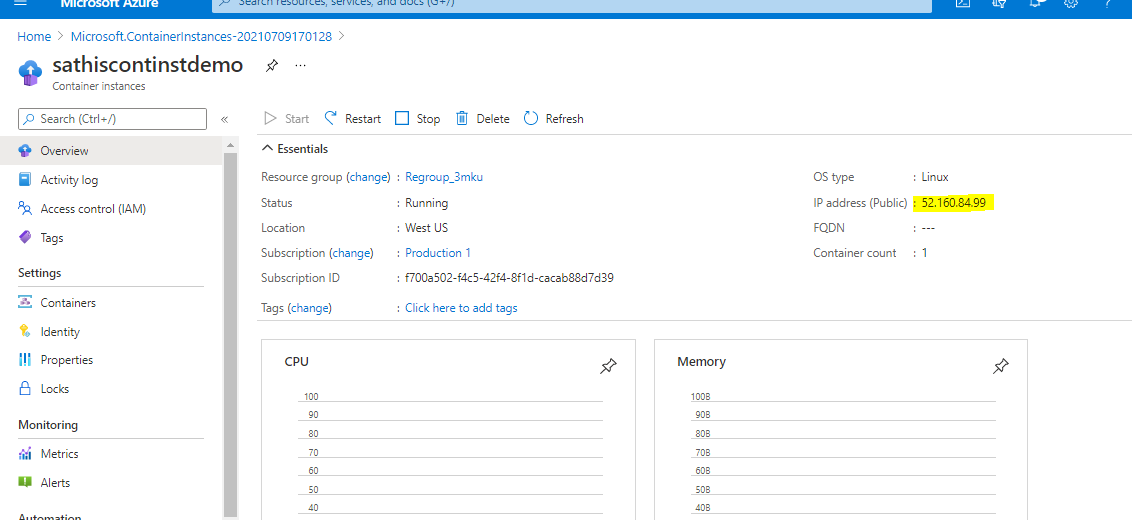
Opening the files, editing and re deploying

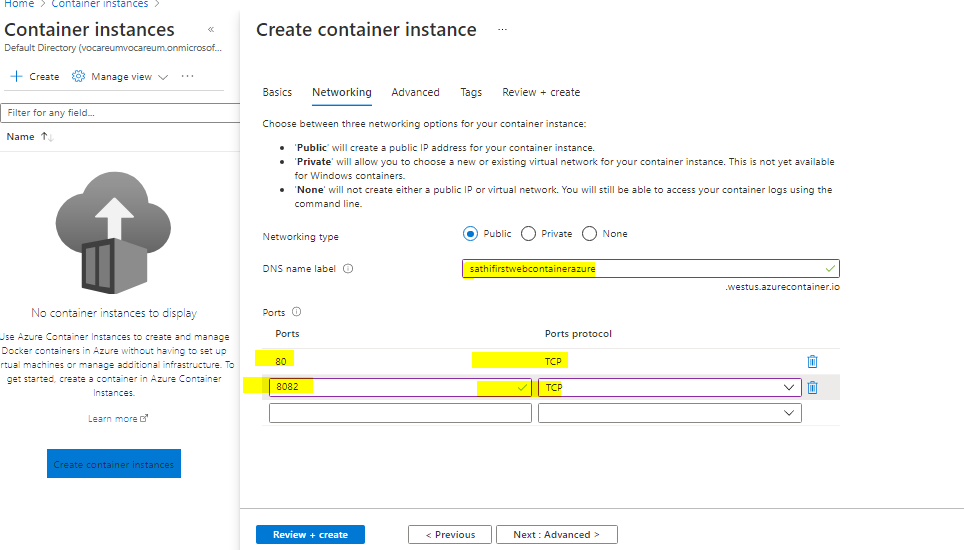
<https://docs.microsoft.com/en-us/azure/aks/tutorial-kubernetes-app-update?tabs=azure-cli>

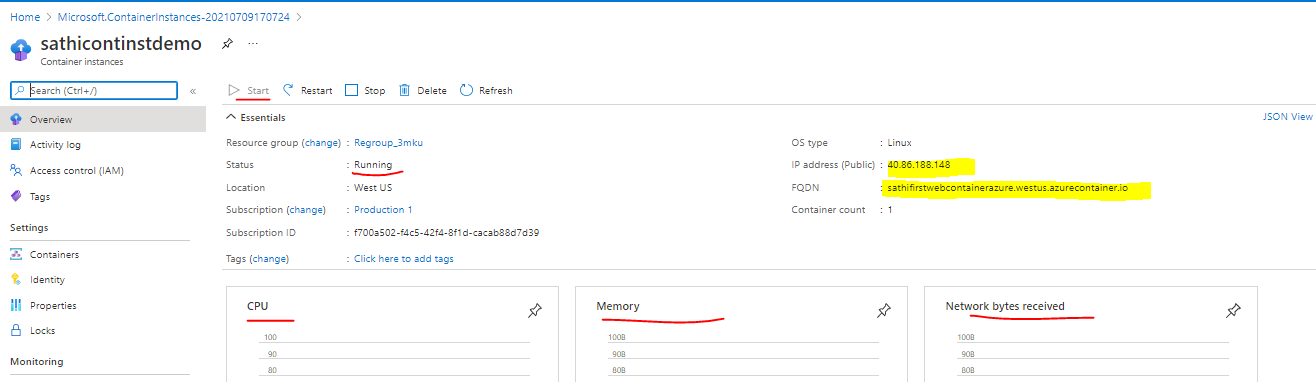
# Additional Information

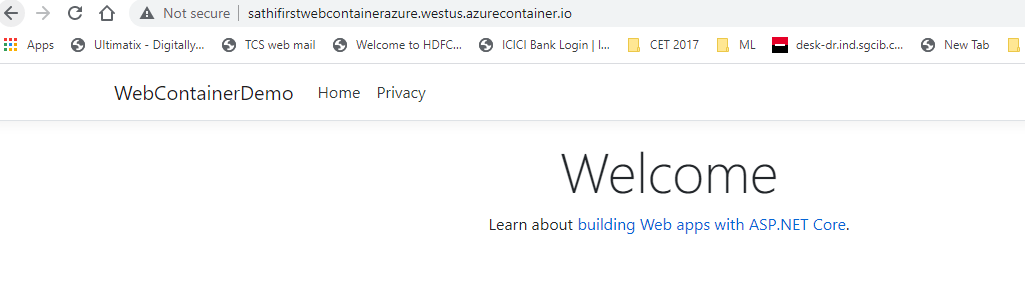
## Hosting image in container instance

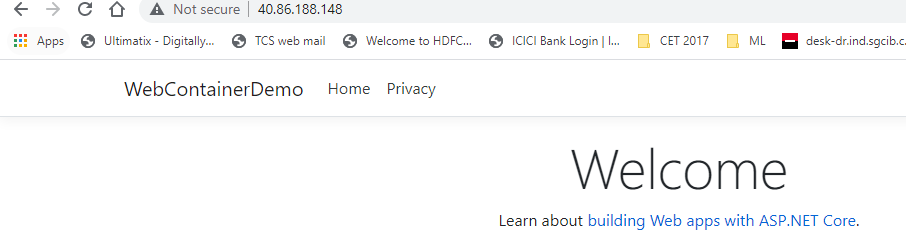












### Using Visual Studio:

