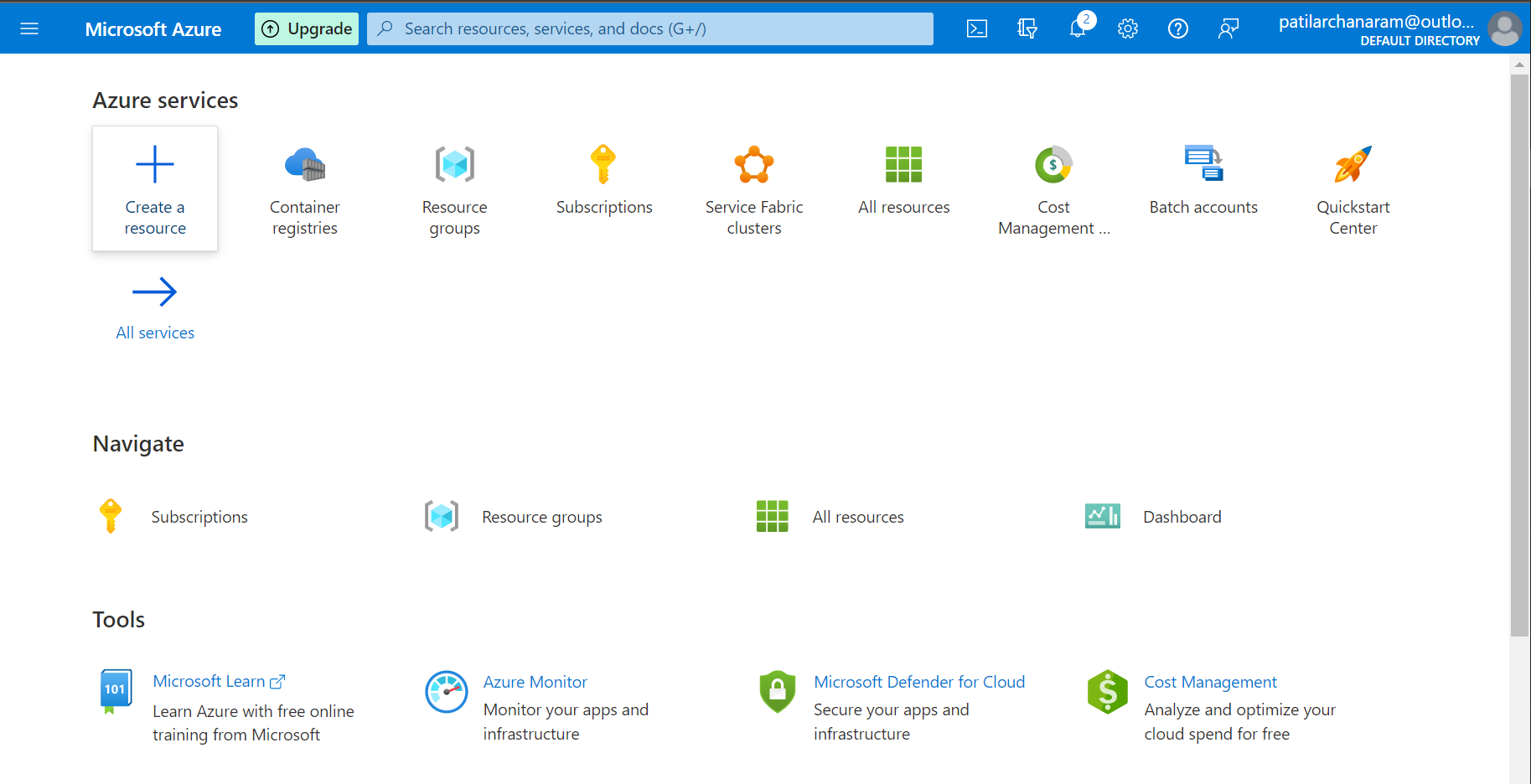
**PRACTICAL NO - 03**

**A) Create an Azure Kubernetes Service Cluster**

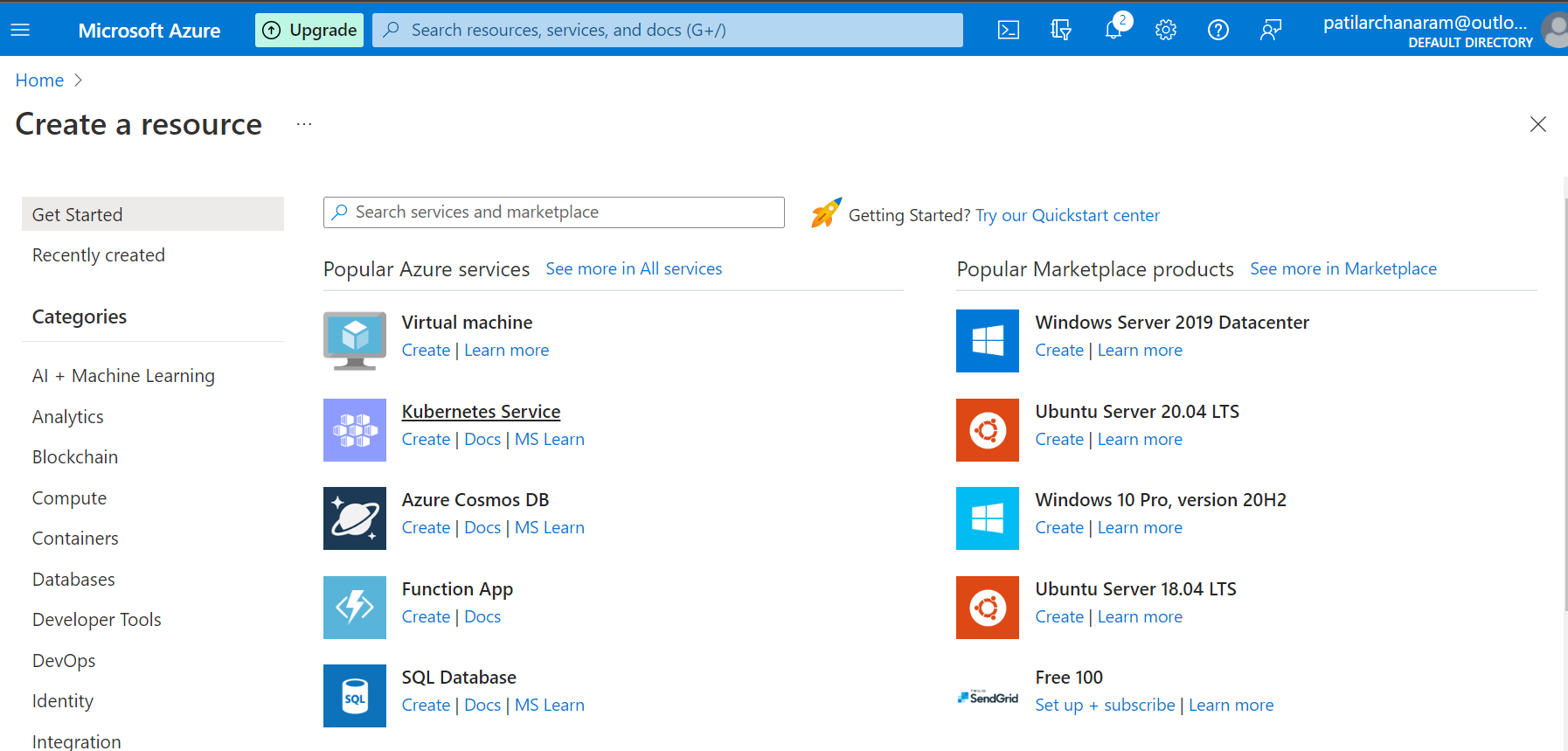
### Step I - Sign in to the Azure portal

### Step II - Create AKS Cluster

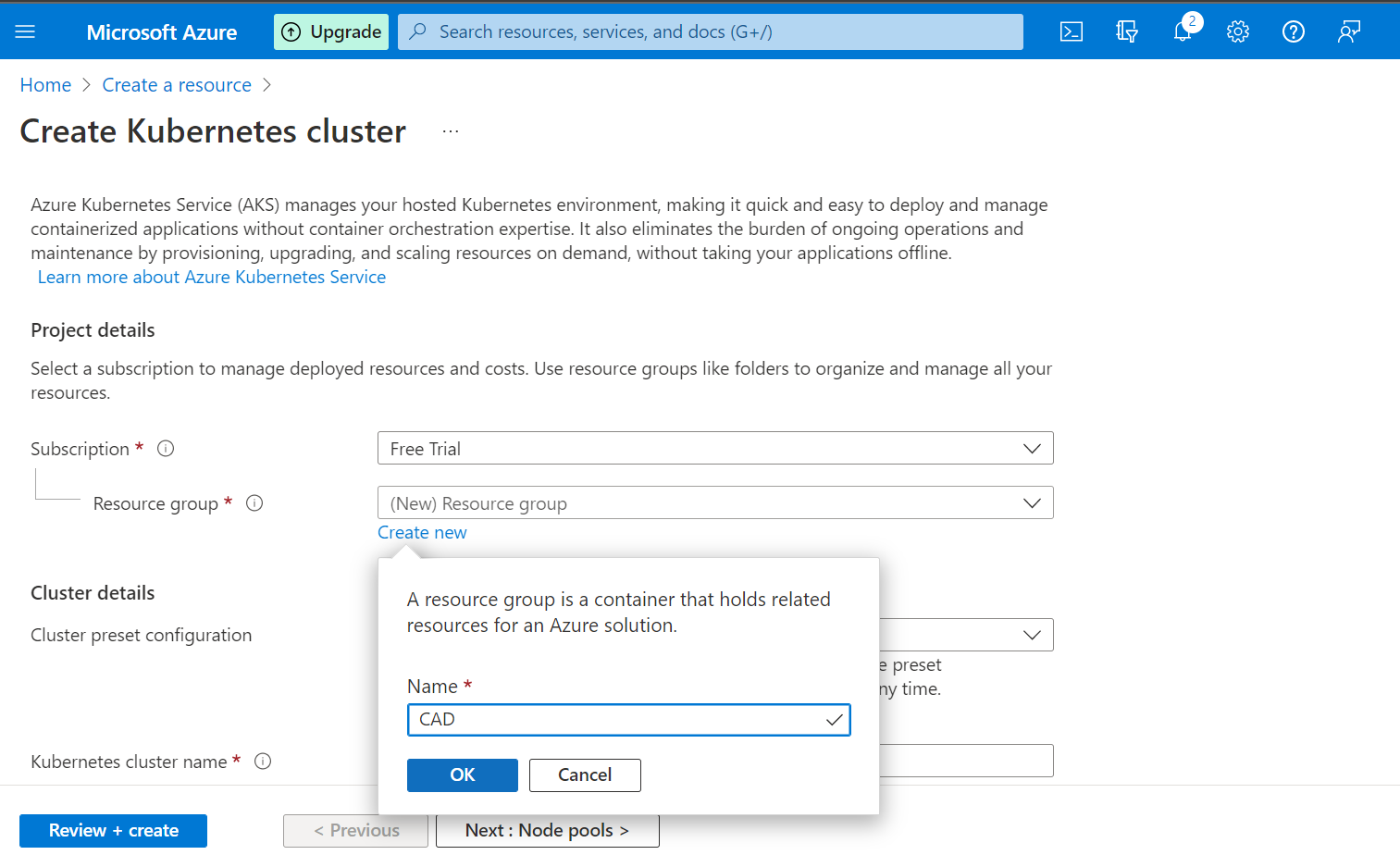
* Select ***+ Create a resource*** from the Microsoft Azure Home page.



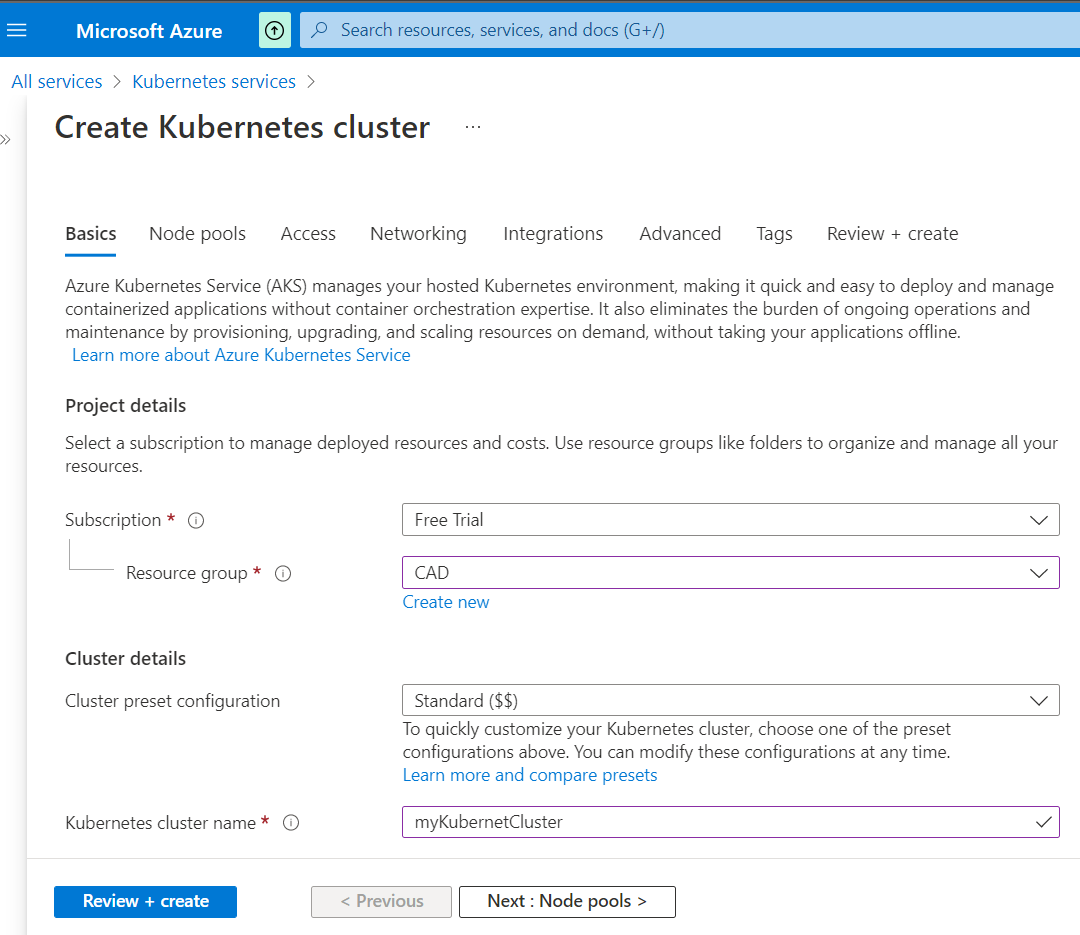
* Select Kubernetes Service



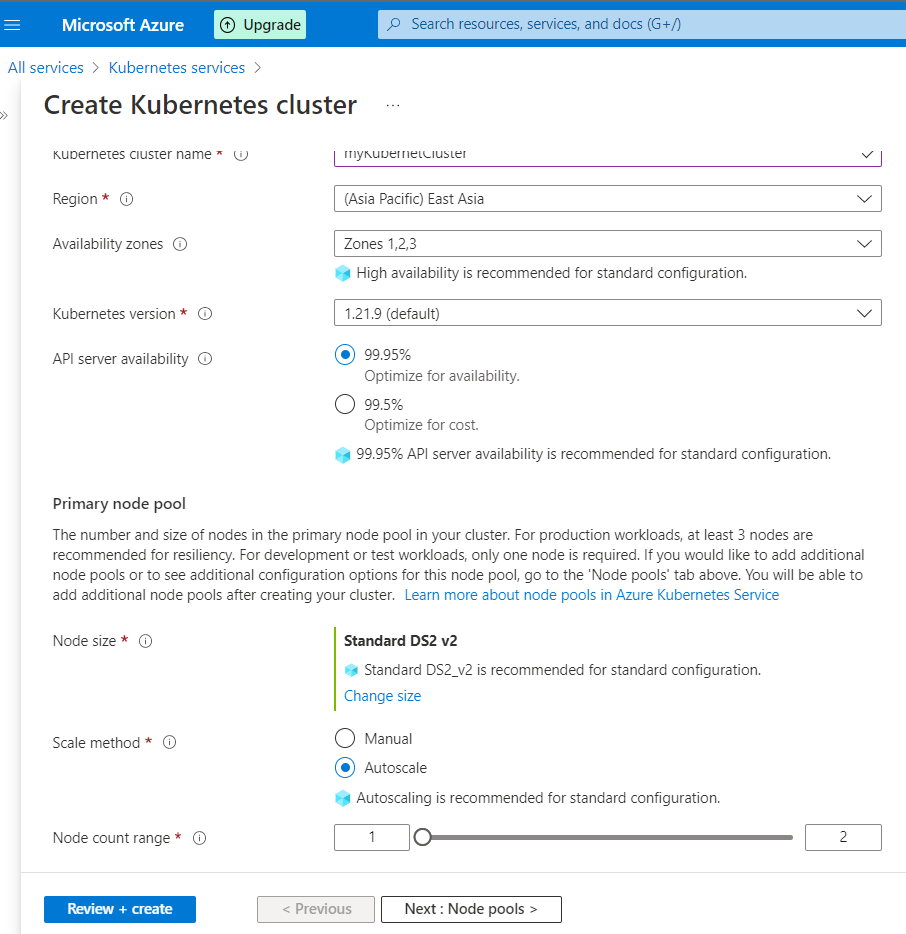
* Enter the subscription. Select resource group. If a resource group is not created then click Create new ➤ Name it ➤ Ok.



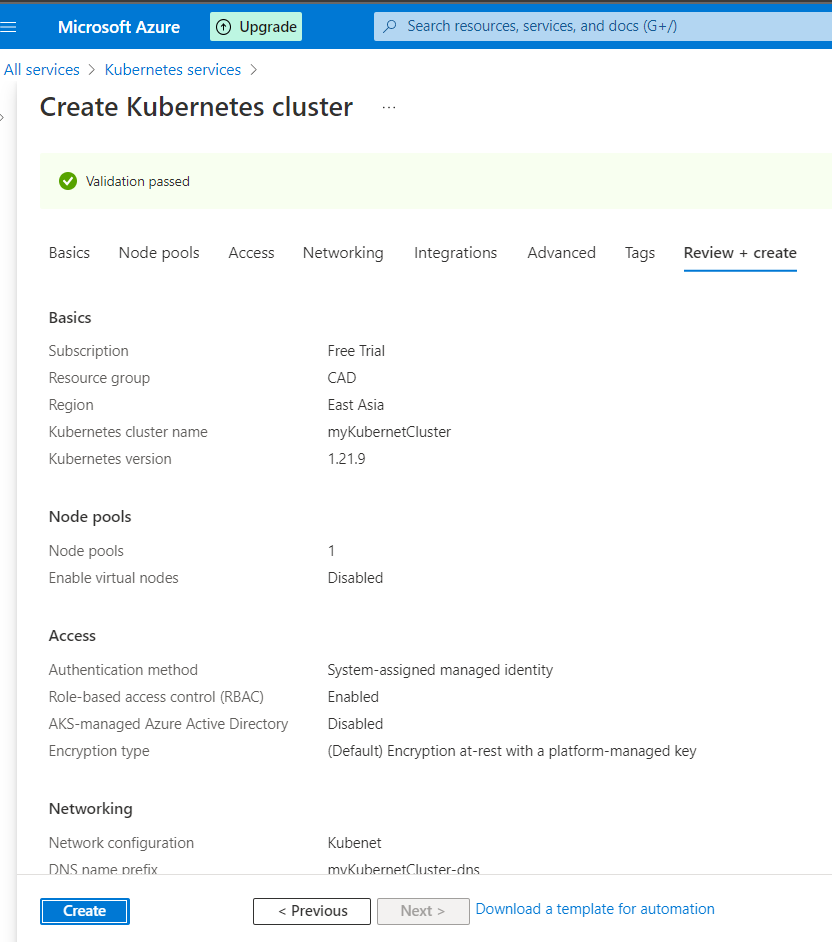
* Select Cluster preset configuration.
* Enter cluster name.



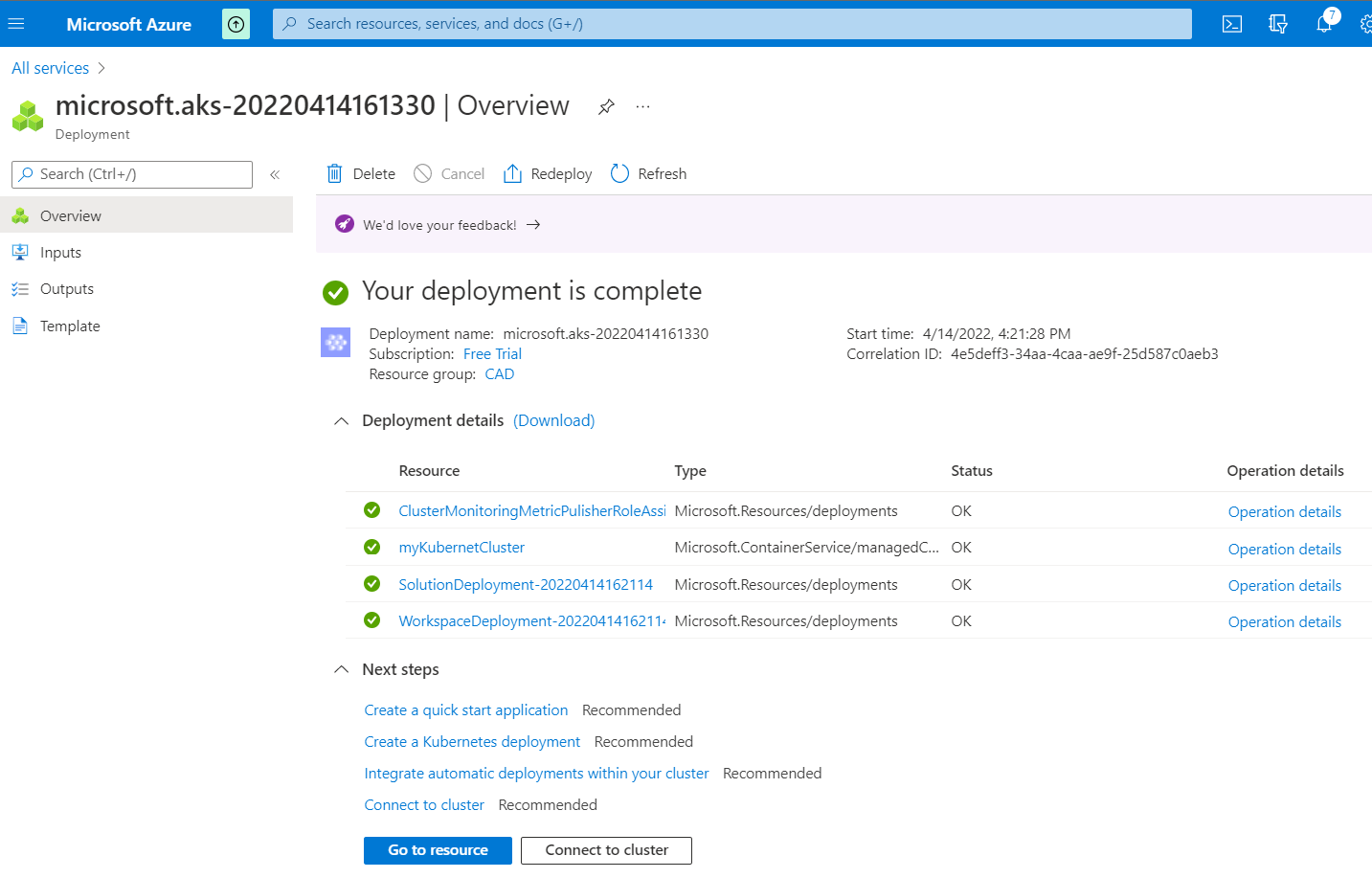
* Select region, Kubernetes version, Node count range, and scale method, as shown in the Figure below ➤ Click on Review + create



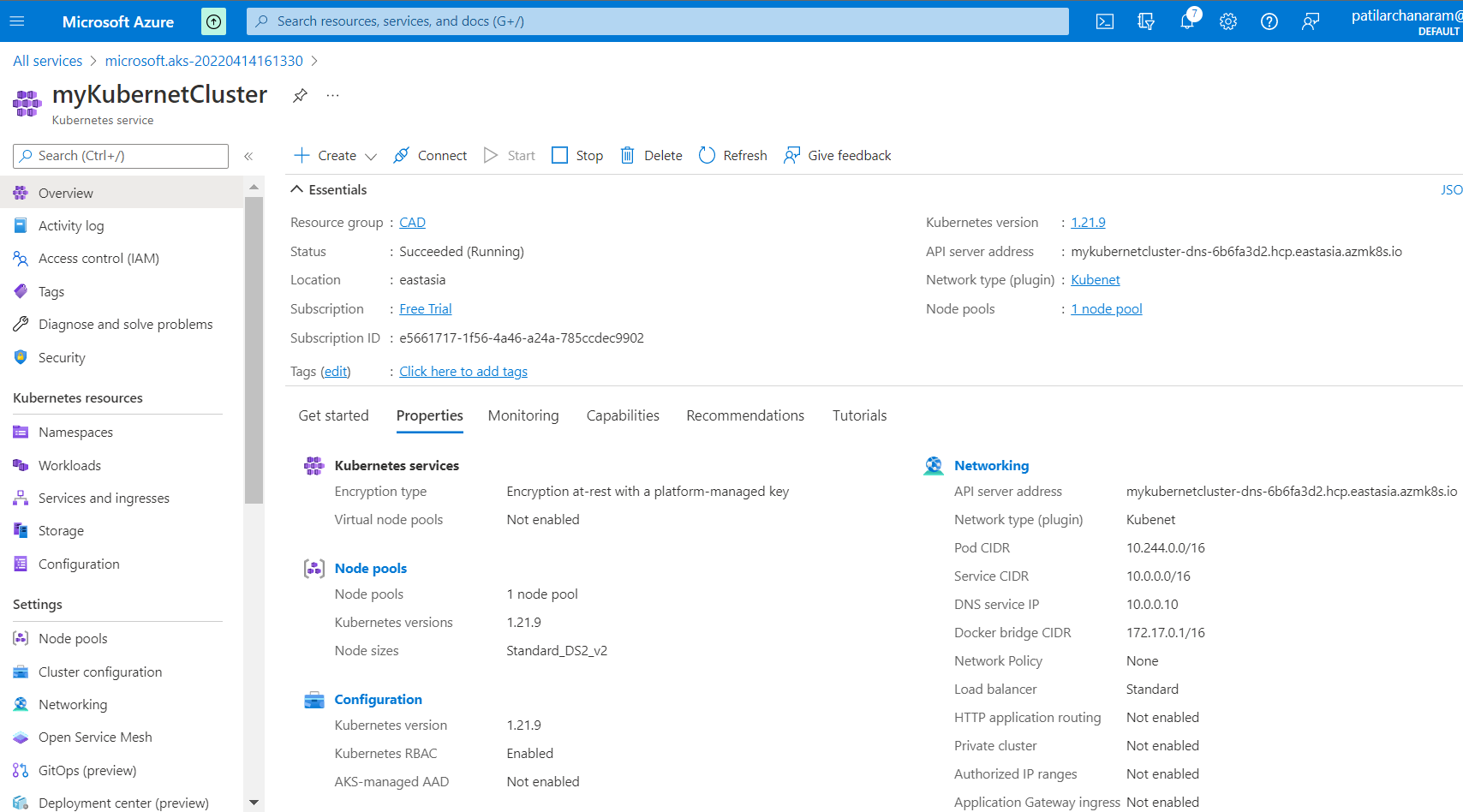
* If will get a message as ***Validation Passed*** as shown below ➤ click on Create.



* Deployment will get started. Wait for a few minutes. Once deployment is completed ➤ click on ‘***Go to resource’***.

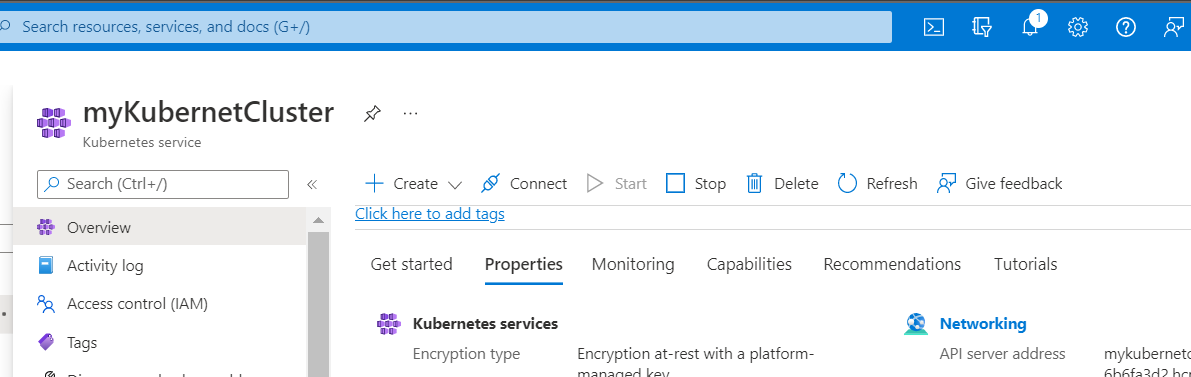


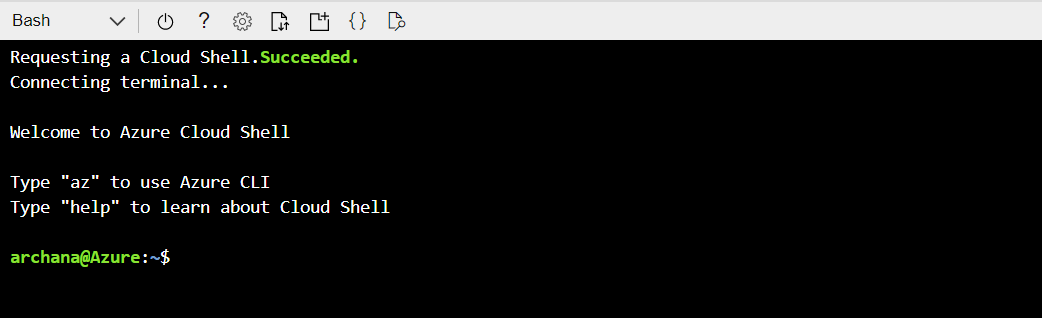
* You will get navigated to the newly created Azure Kubernetes Service Cluster.



### Step III - Connect to the cluster

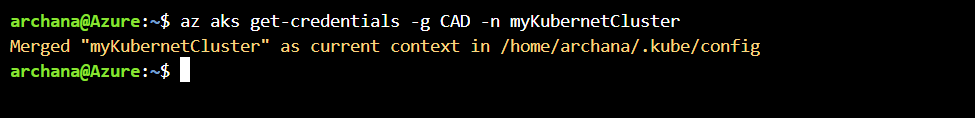
* To manage a Kubernetes cluster, use the Kubernetes command-line client, kubectl. The kubectl is already installed if you use Azure Cloud Shell.
* Open ***Cloud Shell*** using the >\_ button on the top of the Azure portal. ➤ ***Bash***





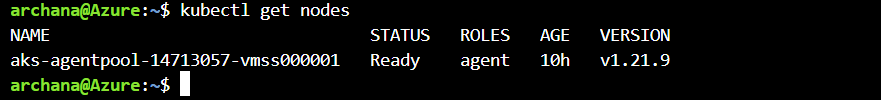
* Configure ***kubectl*** to connect to your Kubernetes cluster using the ***az aks get-credentials*** command. The following command downloads credentials and configures the Kubernetes CLI to use them.

***az aks get-credentials -g [Resource Group] -n [Name of Azure Kubernetes Service]***

******

* Verify the connection to your cluster using ***kubectl ge***t to return a list of the cluster nodes.

***kubectl get nodes***

******

**B) Deploy Application on AKS**

### Step I – Create *myfile.yaml* file

* Copy the following code and save the file as ***myfile.yaml*** and save it to Desktop. We will use this file to deploy an app to the AKS cluster using Azure CLI.
* There are two deployments and two services.

apiVersion: apps/v1

kind: Deployment

metadata:

name: azure-vote-back

spec:

replicas: 1

selector:

matchLabels:

app: azure-vote-back

template:

metadata:

labels:

app: azure-vote-back

spec:

nodeSelector:

"kubernetes.io/os": linux

containers:

- name: azure-vote-back

image: mcr.microsoft.com/oss/bitnami/redis:6.0.8

env:

- name: ALLOW\_EMPTY\_PASSWORD

value: "yes"

resources:

requests:

cpu: 100m

memory: 128Mi

limits:

cpu: 250m

memory: 256Mi

ports:

- containerPort: 6379

name: redis

---

apiVersion: v1

kind: Service

metadata:

name: azure-vote-back

spec:

ports:

- port: 6379

selector:

app: azure-vote-back

---

apiVersion: apps/v1

kind: Deployment

metadata:

name: azure-vote-front

spec:

replicas: 1

selector:

matchLabels:

app: azure-vote-front

template:

metadata:

labels:

app: azure-vote-front

spec:

nodeSelector:

"kubernetes.io/os": linux

containers:

- name: azure-vote-front

image: mcr.microsoft.com/azuredocs/azure-vote-front:v1

resources:

requests:

cpu: 100m

memory: 128Mi

limits:

cpu: 250m

memory: 256Mi

ports:

- containerPort: 80

env:

- name: REDIS

value: "azure-vote-back"

---

apiVersion: v1

kind: Service

metadata:

name: azure-vote-front

spec:

type: LoadBalancer

ports:

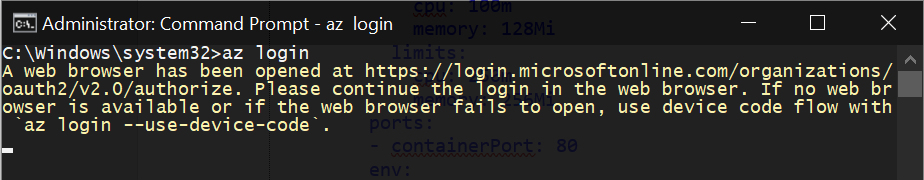
- port: 80

selector:

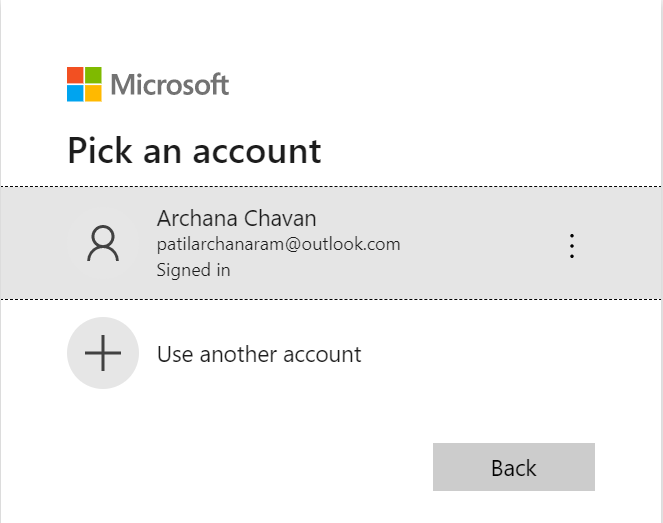
app: azure-vote-front

### Step II – Deploy app using Azure CLI

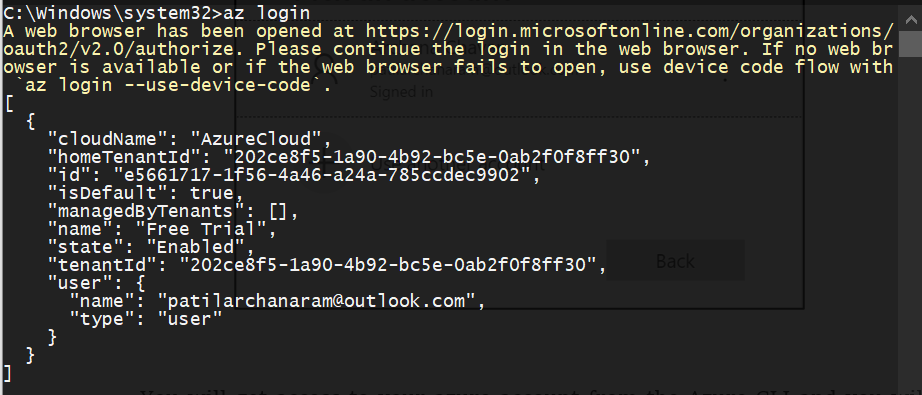
* Now open a command prompt as an administrator. Login to your azure account by using ***az login*** command as shown in the figure below.



* It will redirect you to another page. Select your Microsoft account here.

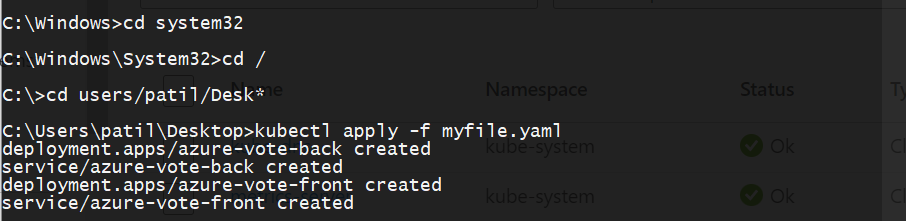


* You will get access to your azure account from the Azure CLI and you will get the following screen.



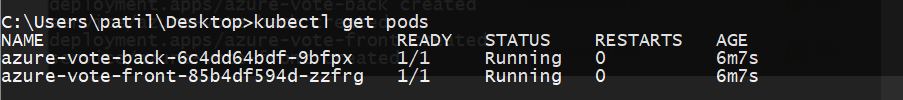
* Now navigate to the desktop where the myfile.yaml file is saved.
* Now run the following command to deploy the app on AKS cluster.

***kubectl apply -f myfile.yaml***

****

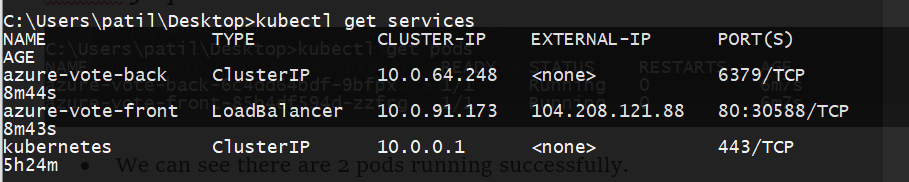
* Here we can see that there are 2 deployments and 2 services created successfully.
* Now run command

***kubectl get pods***

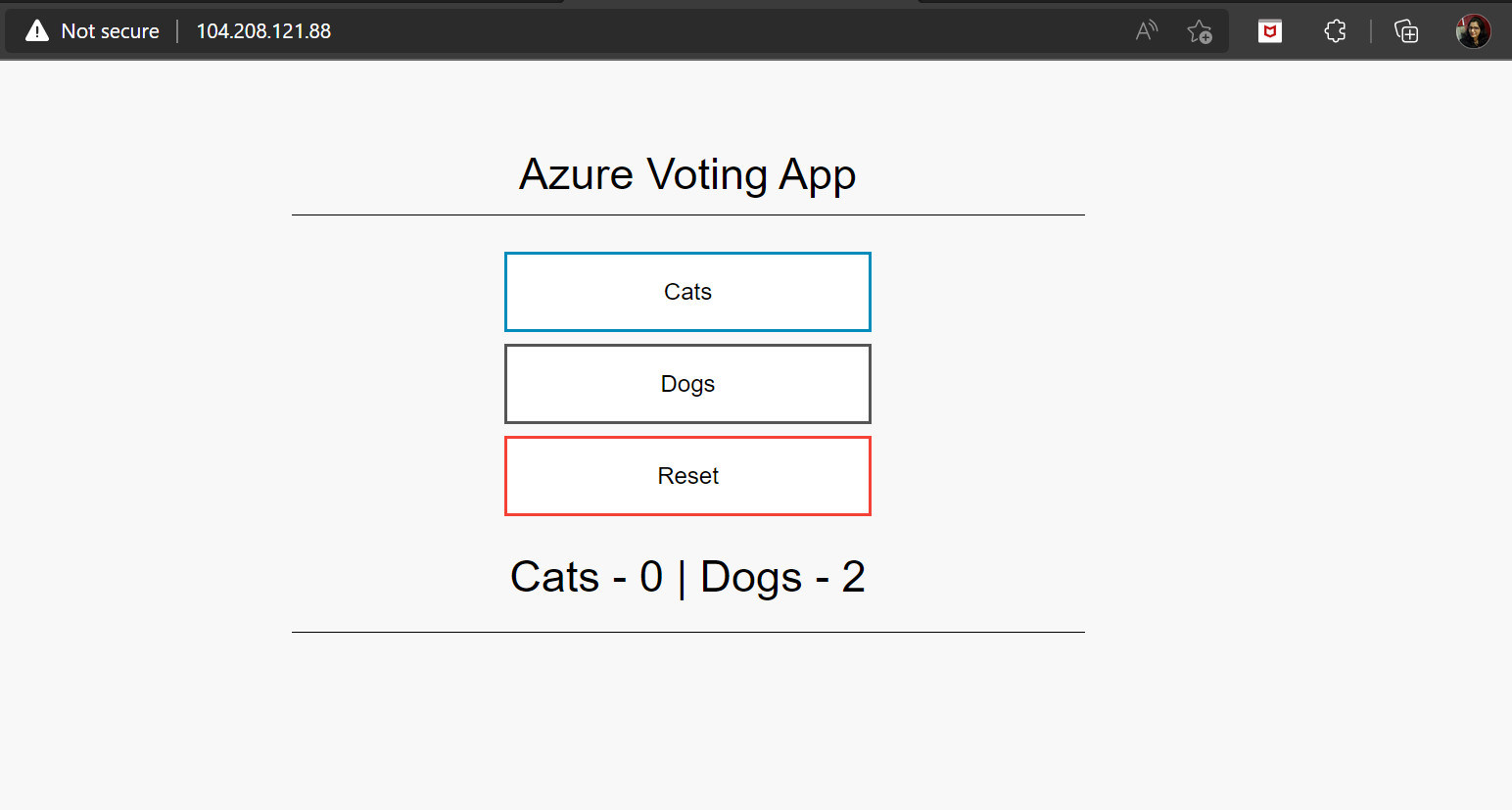


* We can see there are 2 pods running successfully.
* Now to see services run command

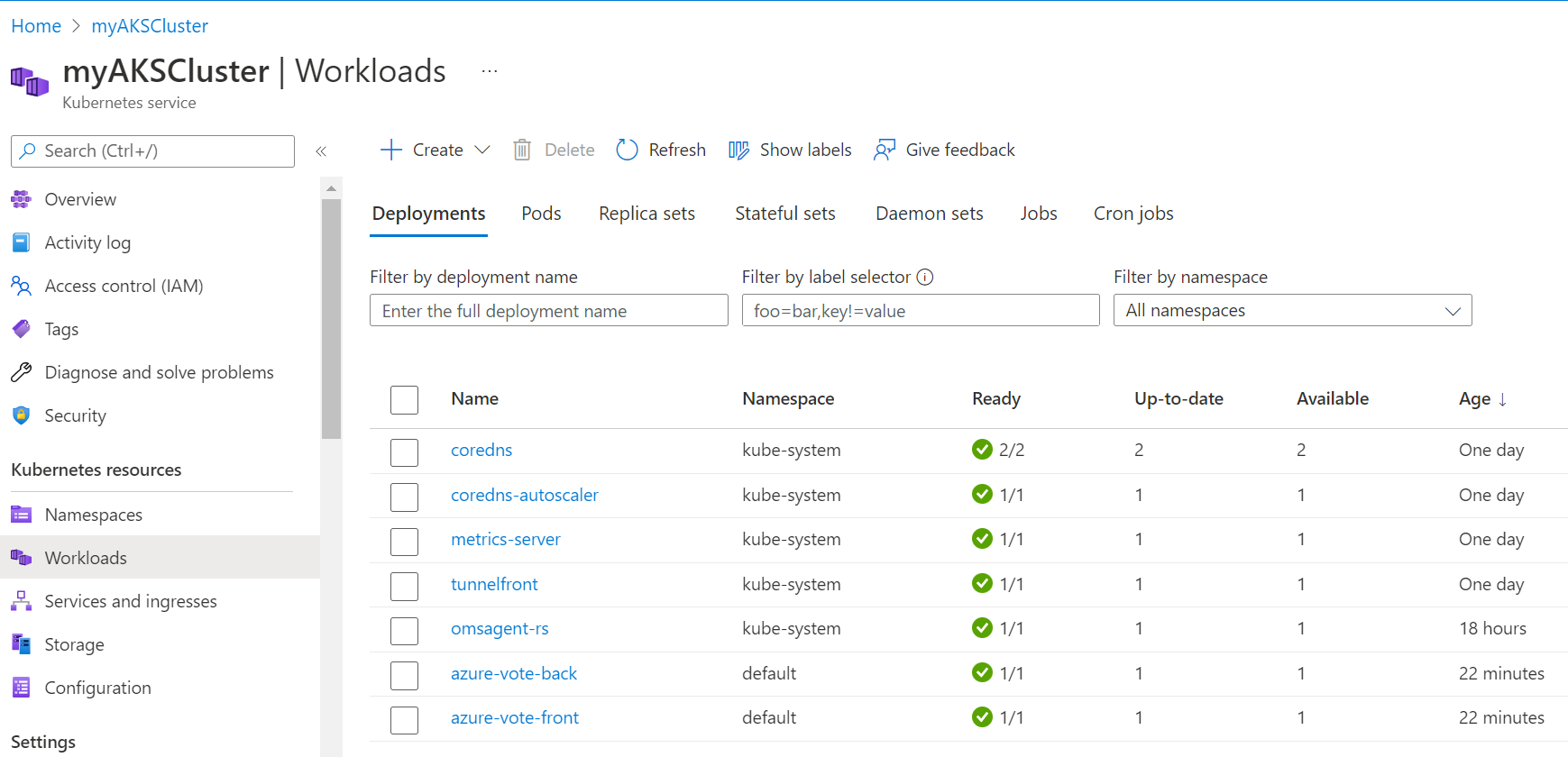
***kubectl get services***

******

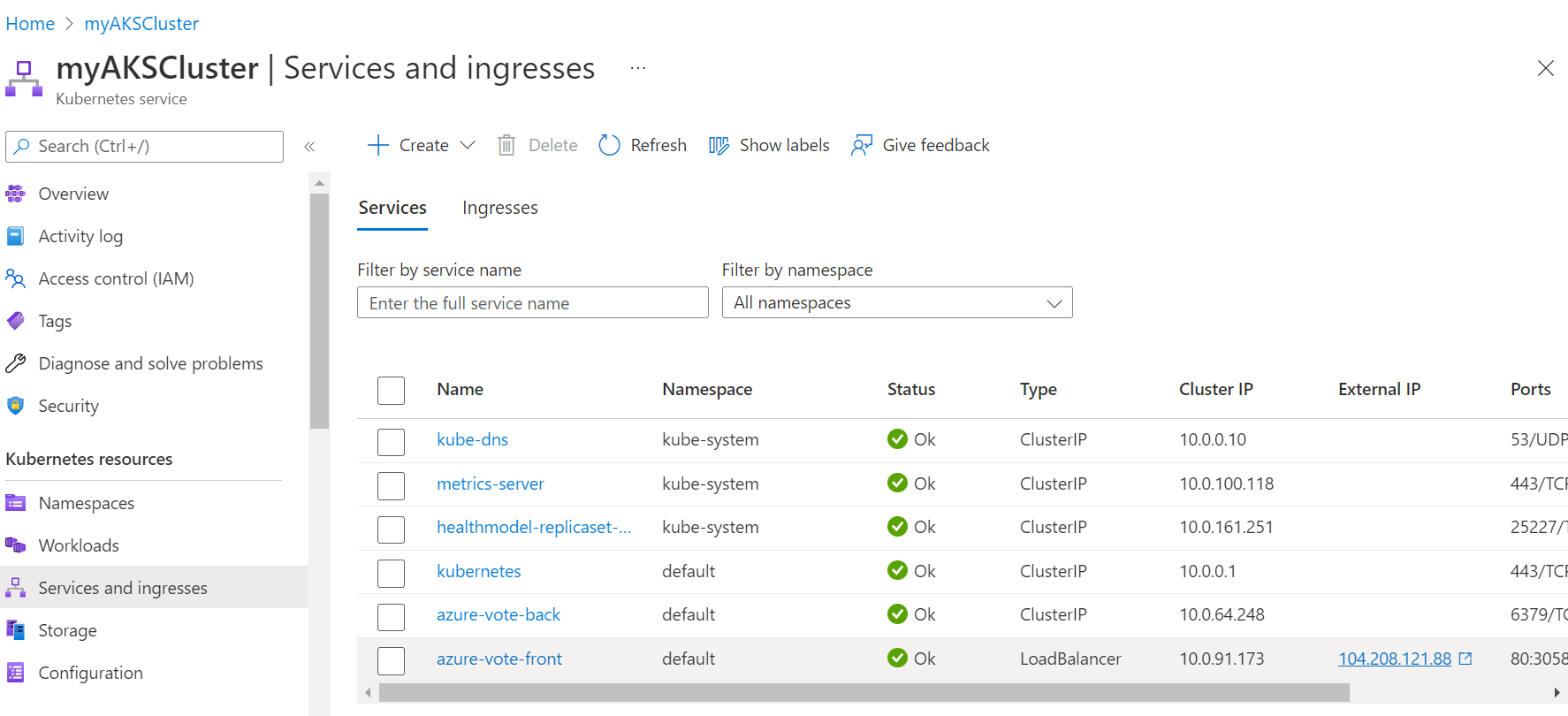
* We can see the azure-vote-back and azure-vote-front services with their corresponding Type, Custer-IP and Ports. The azure-vote-front service has External-IP that we use in the browser to run the app.
* Now copy this External-IP and paste it in the browser to run app.



* Now login to your azure portal account. > Go to your kubernet service cluster > from the left pane click on ***workloads***.
* Here in the Deployment tab you can see your current deployment



* Now click on Services and ingresses from the left pane. We can see the azure-vote-back and azure-vote-front services.



* You can see the External IP of azure-vote-front service. Just click on that ip address your app will run the browser.

