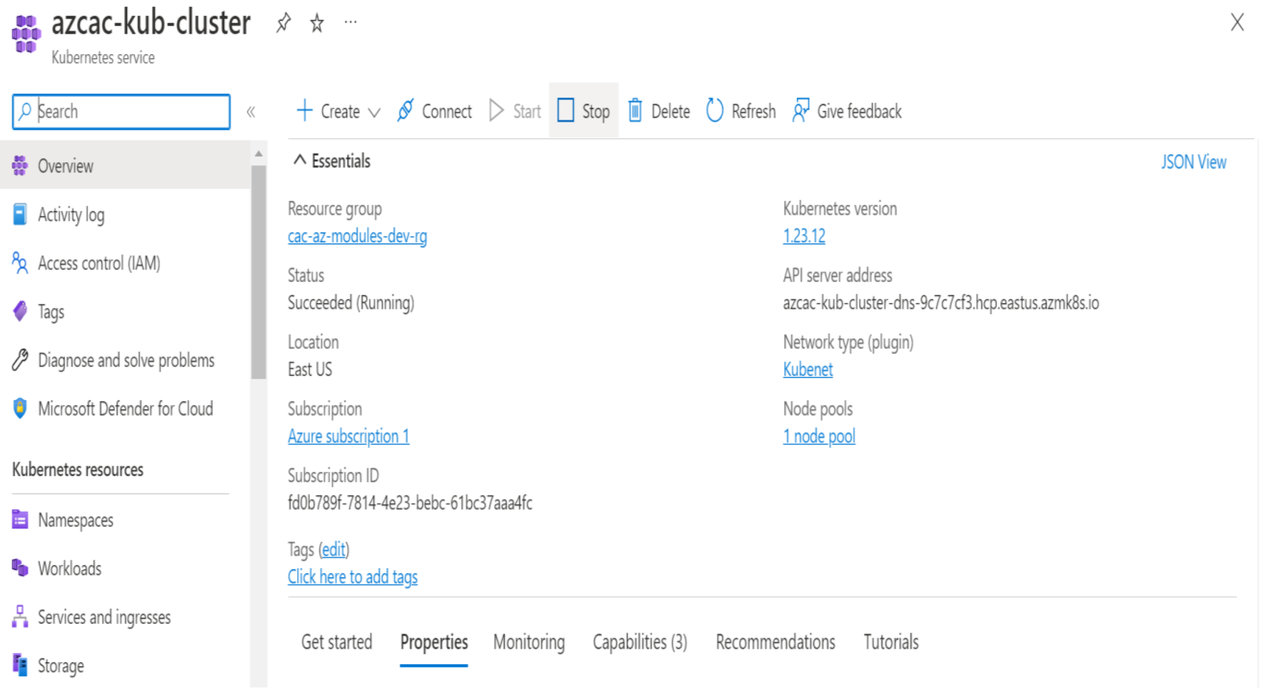
Azure Kubernetes Service Integrated with Azure Container Registry (AKS with ACR)

1. Firstly, push an existing/ built container image into the Azure Container Registry (ACR) by logging into the ACR using the login-credentials of ACR using the command:

   ➢ docker push <acr_name>.azurecr.io/<container_image_name>

2. Azure Portal Overview - After Creation and Deployment of **Azure Kubernetes Service** through Terraform:



Resource group
cac-az-modules-dev-rg

Status
Succeeded (Running)

Location
East US

Subscription
Azure subscription 1

Subscription ID
fd0b789f-7814-4e23-bebc-61bc37aaa4fc

Tags (edit)
Click here to add tags

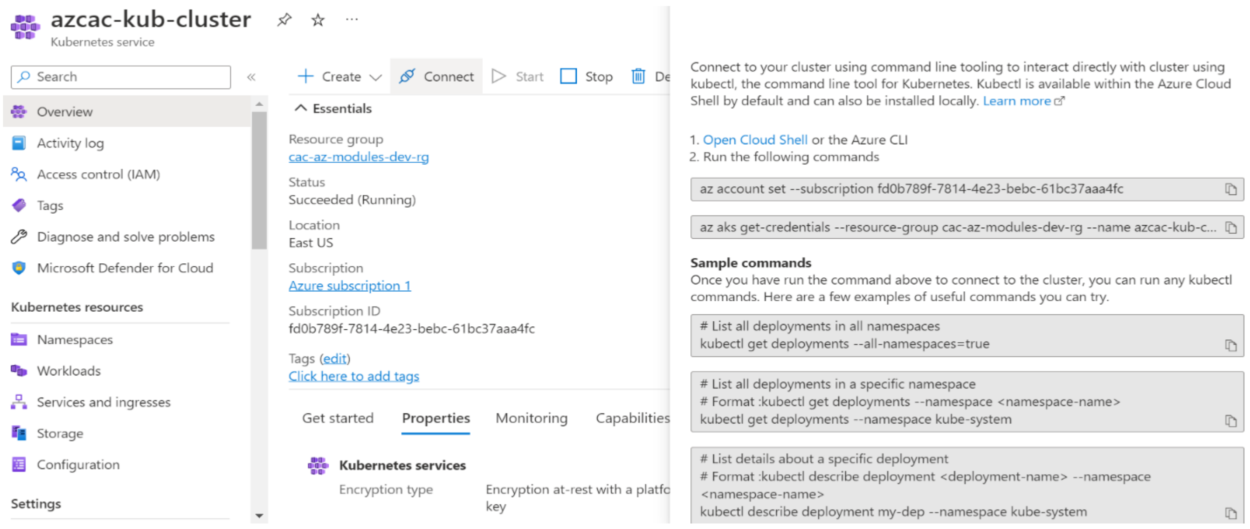
Kubernetes version
1.23.12

API server address
azcac-kub-cluster-dns-9c7c7cf3.hcp.eastus.azmk8s.io

Network type (plugin)
Kubenet

Node pools
1 node pool

3. In the Overview of AKS - Click on Connect and follow the initial two commands to login to subscription and to Merge the AKS



Resource group
cac-az-modules-dev-rg
Status
Succeeded (Running)
Location
East US
Subscription
Azure subscription 1
Subscription ID
fd0b789f-7814-4e23-bebc-61bc37aaa4fc
Tags (edit)
Click here to add tags

Encryption type — Encryption at-rest with a platfo key

Connect to your cluster using command line tooling to interact directly with cluster using kubectl, the command line tool for Kubernetes. Kubectl is available within the Azure Cloud Shell by default and can also be installed locally. Learn more

1. Open Cloud Shell or the Azure CLI
2. Run the following commands

```
az account set --subscription fd0b789f-7814-4e23-bebc-61bc37aaa4fc
```

```
az aks get-credentials --resource-group cac-az-modules-dev-rg --name azcac-kub-c...
```

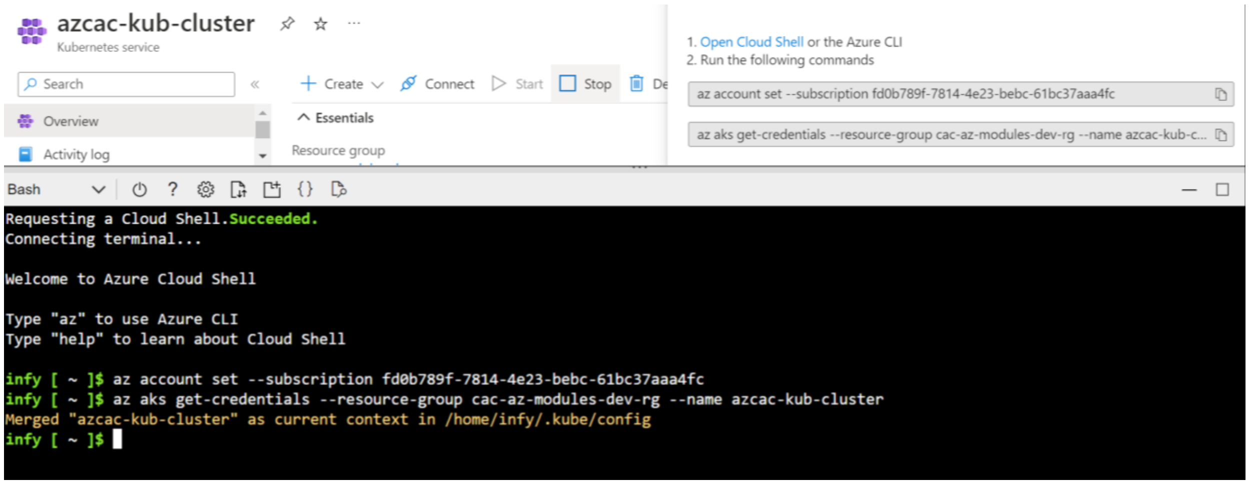
Sample commands
Once you have run the command above to connect to the cluster, you can run any kubectl commands. Here are a few examples of useful commands you can try.

```
# List all deployments in all namespaces
kubectl get deployments --all-namespaces=true
```

```
# List all deployments in a specific namespace
# Format :kubectl get deployments --namespace <namespace-name>
kubectl get deployments --namespace kube-system
```

```
# List details about a specific deployment
# Format :kubectl describe deployment <deployment-name> --namespace
<namespace-name>
kubectl describe deployment my-dep --namespace kube-system
```

4. Open the Azure CLI, move to Bash and apply the commands, after applying the two commands, the AKS gets Merged



1. Open Cloud Shell or the Azure CLI
2. Run the following commands

```
az account set --subscription fd0b789f-7814-4e23-bebc-61bc37aaa4fc
```

```
az aks get-credentials --resource-group cac-az-modules-dev-rg --name azcac-kub-c...
```

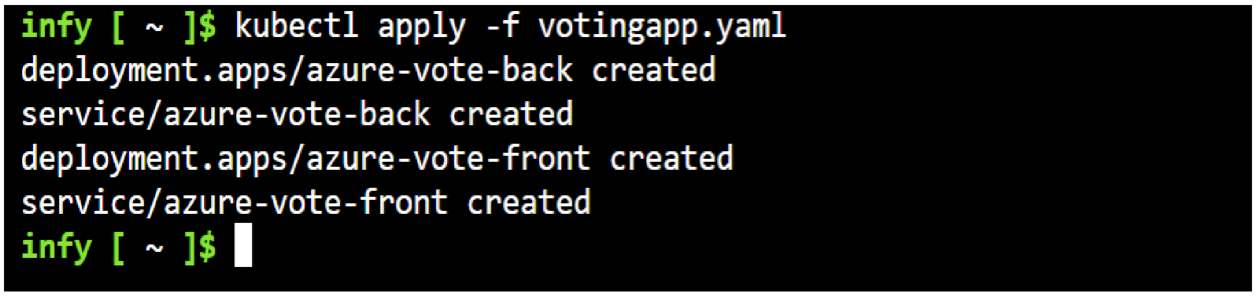
```
Requesting a Cloud Shell.Succeeded.
Connecting terminal...

Welcome to Azure Cloud Shell

Type "az" to use Azure CLI
Type "help" to learn about Cloud Shell

infy [ ~ ]$ az account set --subscription fd0b789f-7814-4e23-bebc-61bc37aaa4fc
infy [ ~ ]$ az aks get-credentials --resource-group cac-az-modules-dev-rg --name azcac-kub-cluster
Merged "azcac-kub-cluster" as current context in /home/infy/.kube/config
infy [ ~ ]$
```

5. After specifying the containers and Image Details in the YAML file. Upload the updated YAML file.

Deploy the application using the command:

➢ kubectl apply -f <filename>.yaml

6. App is deployed successfully when you get created notifications.

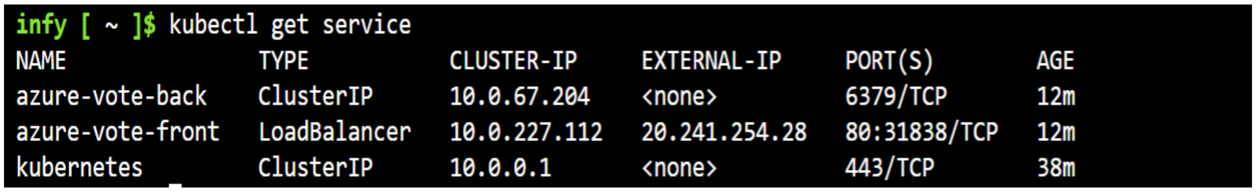


```
infy [ ~ ]$ kubectl apply -f votingapp.yaml
deployment.apps/azure-vote-back created
service/azure-vote-back created
deployment.apps/azure-vote-front created
service/azure-vote-front created
infy [ ~ ]$
```

7. To test out the App, we need Public IP Address.

To check our deployment has created an external IP, run the command:

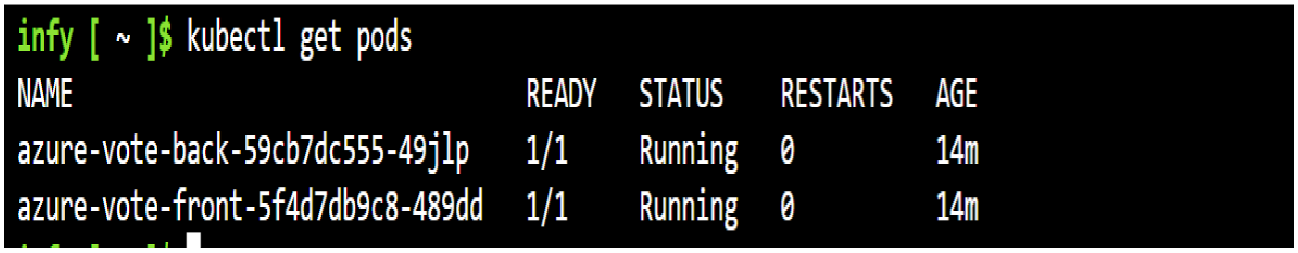
➢ kubectl get service



```
infy [ ~ ]$ kubectl get service
NAME               TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)        AGE
azure-vote-back    ClusterIP      10.0.67.204    <none>          6379/TCP       12m
azure-vote-front   LoadBalancer   10.0.227.112   20.241.254.28   80:31838/TCP   12m
kubernetes         ClusterIP      10.0.0.1       <none>          443/TCP        38m
```

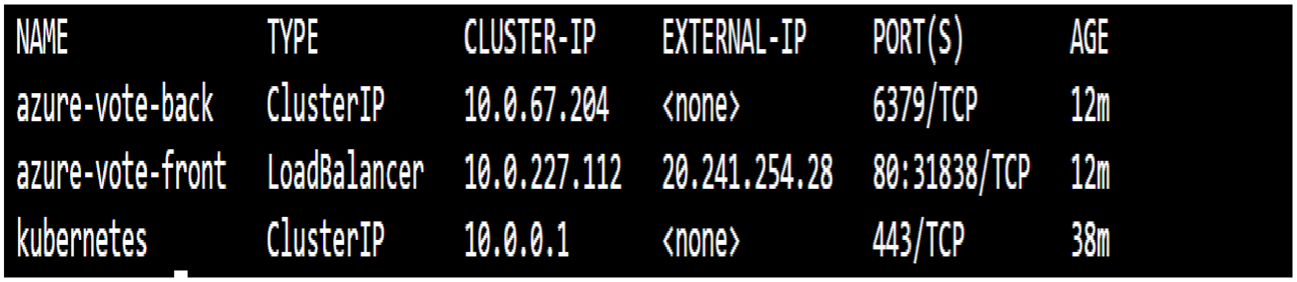
8. Run the below command to check the pods status is in running state

➢ kubectl get pods



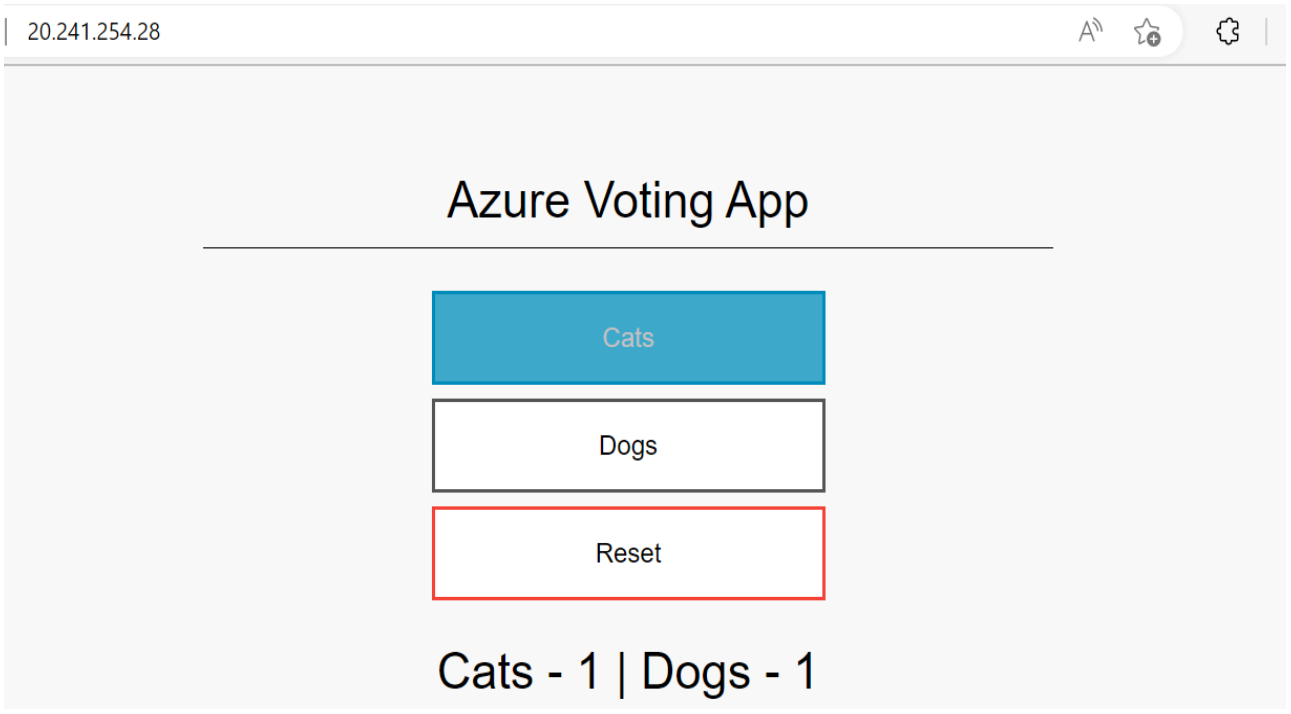
```
infy [ ~ ]$ kubectl get pods
NAME                               READY   STATUS    RESTARTS   AGE
azure-vote-back-59cb7dc555-49jlp   1/1     Running   0          14m
azure-vote-front-5f4d7db9c8-489dd  1/1     Running   0          14m
```

9. Copy the External Ip and paste it in the Web Browser and the application is up-end and running.



```
NAME              TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)        AGE
azure-vote-back   ClusterIP      10.0.67.204    <none>          6379/TCP       12m
azure-vote-front  LoadBalancer   10.0.227.112   20.241.254.28   80:31838/TCP   12m
kubernetes        ClusterIP      10.0.0.1       <none>          443/TCP        38m
```

10. After Deploying "Voting App" using Azure Kubernetes Cluster Service:


20.241.254.28
Azure Voting App
Cats
Dogs
Reset
Cats - 1 | Dogs - 1