**Deploying azure sample app (azure-voting-app-redis) into aks**

Install Docker to run app locally in containers

Download the sample app from git by cloning to local repo

**Steps to create a ACR and push the local docker image to ACR**

1. Create a resource group

**az group create --name agsContainersRG --location eastus2**

1. create your own private container registry using Azure Container Registry (ACR)

**az acr create --resource-group agsContainersRG --name agsacr --sku basic**

1. Login to the ACR

**az acr login --name agsacr**

1. Verify local docker images. And to use this images with ACR we need to tag the image with the login server address of our ACR. This tag is used to routing when pushing container images to ACR. So list the details of login server of ACR by using the below command

**az acr list**

1. Now tag the local image with login server address of ACR

**docker tag azure-vote-front agsacr.azurecr.io/azure-vote-front:v1**

1. Push the image to ACR

**docker push agsacr.azurecr.io/azure-vote-front:v1**

1. To see the lsit of repository in our ACR

**az acr repository list --name agsacr**

1. To see the tags in repository

**az acr repository show-tags --name agsacr --repository azure-vote-front --output table**

**Steps to create a Azure Kubernetes Cluster (AKS)**

1. First create the service principal

**az ad sp create-for-rbac --skip-assignment**

"appId": "84fc8153-1aaf-4d8a-bf28-9d2d9a6e69f9",

"displayName": "azure-cli-2019-11-12-11-38-53",

"name": "http://azure-cli-2019-11-12-11-38-53",

"password": "381db9d0-b27d-465e-ba64-1b2f09f6749f",

"tenant": "5fed7ab7-f33b-44f7-93b3-c719b17af003"

1. Configure ACR for authentication

first get the id of acr

**az acr show --resource-group agsContainersRG --name agsacr**

id: /subscriptions/0ed79c70-acdd-4c97-9aba-7a784070ee5e/resourceGroups/agsContainersRG/providers/Microsoft.ContainerRegistry/registries/agsacr",

then assign the application id to acr id

**az role assignment create --assignee 84fc8153-1aaf-4d8a-bf28-9d2d9a6e69f9 --scope /subscriptions/0ed79c70-acdd-4c97-9aba-7a784070ee5e/resourceGroups/agsContainersRG/providers/Microsoft.ContainerRegistry/registries/agsacr --role Reader**

1. Now create aks

**az aks create --resource-group** agsRG **--name** agsaks **--node-count 1 --service-principal** 84fc8153-1aaf-4d8a-bf28-9d2d9a6e69f9 **--client-secret** 381db9d0-b27d-465e-ba64-1b2f09f6749f **--generate-ssh-keys**

1. To connect to AKS from local kubectl is needed

**az aks install-cli**

1. Please add the kubectl path to search path so the kubectl.exe can be found

**$env:path += 'C:\Users\AGSPL\.azure-kubectl'**

1. Now get credentials and merge our AKS as current context in kubectl path

**az aks get-credentials --resource-group agsRG --name agsaks**

1. Run the following to see the nodes running

**kubectl get nodes**

**Steps to run our app in ACR on AKS**

1. First edit the azure-vote-all-in-one-redis.yaml file with the acr login server name in image section

image: agsacr.azurecr.io/azure-vote-front:v1

1. Apply the changes to aks

**kubectl apply -f .\azure-vote-all-in-one-redis.yaml**

1. Run the application on aks

**kubectl get service azure-vote-front –watch**

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

azure-vote-front LoadBalancer 10.0.98.104 40.81.72.177 80:30464/TCP 43s

1. To check the pods running

**kubectl get pods**

1. To scale the pods

**kubectl scale --replicas=5 deployment/azure-vote-front**

**"appId": "2ecd3be4-195f-4537-add7-0e0fb28abe28",**

**"displayName": "kiran-docker-app",**

**"name": "http://kiran-docker-app",**

**"password": "62699422-1f51-412e-b92a-e9c8a7bc2306",**

**"tenant": "7437a5b1-4b4d-4ad1-abfa-51760942d560"**