1. Create A Docker Image

2. Create An Azure Container Registry

3. Push The Docker Image To ACR

4. Create A Kubernetes Cluster

5. Create Kubernetes Deployment Manifests

6. Deploy Your Application

Step1

**Creating an Azure Cosmos DB**

1. First, we need to select an existing <>Subscription.
2. Then we need to either select an existing **Resource Group** or create a new one.
3. After that, we need to choose an **Account Name**.
4. As we discussed in the previous section, Azure Cosmos DB provides various APIs and we need to choose one. In this example, we are going to use **Core (SQL)**
5. Then we need to choose a **Location**.
6. Azure Cosmos DB provides two capacity modes. For working with traffic requiring predictable performance, we can choose **Provisioned Throughput**. **Serverless** is a good choice while working with unpredictable traffic.
7. While creating a Cosmos DB, we have an option to apply **Free Tier Discount**. Choosing this option will give us 400 Request Units per second and 5 GB of storage in one account for free.
8. There is an option to choose **Production** or **Non-Production** Account Type. This will just change the UI experience offered in the portal and Cosmos Explorer and will not impact the service behavior.
9. Choosing **Geo-Redundancy** will enable global distribution of databases by pairing certain Regions together. Let’s disable it in this example.
10. Finally, the **Multi-region Writes** option improves the throughput of our databases and containers. We can disable this option for now.

AKS

1. Az login
2. Az acr login –-name rpsregistry
3. docker tag yourImageName yourLoginServer/yourRegistryName:yourTag
4. docker push yourLoginServer/yourRegistryName:yourTag
5. docker tag aksservice:dev rpsregistry.azurecr.io/aksservicedemo
6. docker push rpsregistry.azurecr.io/aksservicedemo
7. az acr repository list --name rpsregistry --output table

Text

Description automatically generated

A screenshot of a computer

Description automatically generated with medium confidence

1. az aks install-cli

A screenshot of a computer

Description automatically generated with medium confidence

1. az aks get-credentials –g rpsgroup –n rpscluster

Text

Description automatically generated

1. **kubectl create secret docker-registry SECRET\_NAME --docker-server=REGISTRY\_NAME.azurecr.io --docker-username=USERNAME --docker-password=PASSWORD --docker-email=ANY\_VALID\_EMAIL**

A screenshot of a computer

Description automatically generated

kubectl create secret docker-registry globalaks-key --docker-server rpsregistry.azurecr.io --docker-username= rpsregistry --docker-password= = qJEfEYAIZlQbhepyj+b+6wLNL0/bYa1s [--docker-email=parameswari.bala@rpsconsulting.in](mailto:--docker-email=parameswari.bala@rpsconsulting.in)

Text

Description automatically generated

11 kubectl apply -f <deploymentfile.yml>

A screenshot of a computer

Description automatically generated with medium confidence

12. az aks browse --resource-group rpsgroup –name rpscluster