10 MICRO SERVICES PROJECT

(GitHub - usubbu/microservices-project)

STEP-1: LAUNCH T2.LARGE INSTANCE WITH ADMIN PERMISSIONS

STEP-2: Install AWS CLI, kubectl, and eksctl

Install AWS CLI LATEST VERSION

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
          unzip awscliv2.zip
          sudo ./aws/install
```

Install KUBECTL:

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.34.1/2025-09-19/bin/linux/amd64/kubectl
chmod +x ./kubectl
sudo mv ./kubectl /usr/local/bin
```

Install EKSCTL:

```
curl --silent --location "https://github.com/weaveworks/eksctl/releases/latest/download/eksctl_$(uname -s)_amd64.tar.gz" | tar xz -C /tmp
sudo mv /tmp/eksctl /usr/local/bin
eksctl version
```

(or)

STEP-3: Create EKS Cluster

create cluster:

```
eksctl create cluster --name=EKS-1 --region=ap-south-1 --zones=ap-south-1a,ap-south-1b --without-nodegroup
```

Attach IAM Role:

```
eksctl utils associate-iam-oidc-provider --region ap-south-1 --cluster EKS-1 --approve
```

create NodeGroup:

```
eksctl create nodegroup --cluster=EKS-1 --region=ap-south-1 --name=node2 --node-type=t3.medium --nodes=3 --nodes-min=2 --nodes-max=4 --node-volume-size=20 --ssh-access --ssh-public-key=mustafakey-pair --managed --asg-access --external-dns-access --full-ecr-access --appmesh-access --alb-ingress-access
```

UPDATE CLUSTER:

```
aws eks update-kubeconfig --name EKS-1 --region ap-south-1
```

STEP-4: Install Jenkins & Docker

- sudo dnf update -y
- sudo dnf install java-21-amazon-corretto -y
- sudo wget -O /etc/yum.repos.d/jenkins.repo \

- https://pkg.jenkins.io/redhat-stable/jenkins.repo
- sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io-2026.key
- sudo dnf install jenkins -y
- sudo systemctl enable jenkins
- sudo systemctl start jenkins
- yum install docker -y && systemctl start docker
- chmod 777 /var/run/docker.sock

STEP-5: Install Plugins: Install the following Jenkins plugins:

- Docker Pipeline
- Kubernetes
- Kubernetes CLI

STEP-6: Now add the dockerhub Credentials

STEP-:7 Create name space & Service Account

Namespace: kubectl create ns webapps

ServiceAccount:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: jenkins
  namespace: webapps
```

Create Role:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: app-role
  namespace: webapps
rules:
  - apiGroups:
        - ""
        - apps
        - autoscaling
        - batch
        - extensions
        - policy
        - rbac.authorization.k8s.io
    resources:
```

```
    - pods
    - componentstatuses
    - configmaps
    - daemonsets
    - deployments
    - events
    - endpoints
    - horizontalpodautoscalers
    - ingress
    - jobs
    - limitranges
    - namespaces
    - nodes
    - pods
    - persistentvolumes
    - persistentvolumeclaims
    - resourcequotas
    - replicasets
    - replicationcontrollers
    - serviceaccounts
    - services
  verbs: ["get", "list", "watch", "create", "update", "patch", "delete"]
```

Bind the role to service account:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: app-rolebinding
  namespace: webapps
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: app-role
subjects:
- namespace: webapps
  kind: ServiceAccount
  name: jenkins
```

Generate token using service account in the namespace:

```
apiVersion: v1
kind: Secret
type: kubernetes.io/service-account-token
metadata:
  name: mysecretname
  annotations:
    kubernetes.io/service-account.name: jenkins
```

Now it will generate a token, Copy this token and create the credential in jenkins named k8-token.

Go to credentials >> select secret text >> copy paste it and id as **k8s-token**

STEP-7: Set Up Multibranch Pipeline

Add this Jenkins file on your github repo

```
pipeline {

    agent any



    stages {

        stage('Deploy To Kubernetes') {

            steps {

                withKubeCredentials(kubectlCredentials: [[caCertificate: '', clusterName: 'EKS-1', contextName: '', credentialsId: 'k8-token', namespace: 'webapps', serverUrl: 'add-your-eks-cluster-url']]) {

                    sh "kubectl apply -f deployment-service.yml"



                }

            }

        }



        stage('verify Deployment') {
            steps {
                withKubeCredentials(kubectlCredentials: [[caCertificate: '', clusterName: 'EKS-1',
contextName: '', credentialsId: 'k8-token', namespace: 'add-your-eks-cluster-url']]) {
                    sh "kubectl get svc -n webapps"
                }
            }
        }
    }
}
```

command to delete cluster : eksctl delete cluster --name EKS-1 --region ap-south-1