CLOUD TRAIN
ACCELERATE YOUR GROWTH

ASSIGNMENT - MODULE 9

DEVOPS ON AWS

AWS Workshop

Contact us

TO ACCELERATE YOUR CAREER GROWTH

For questions and more details:

please call @ +91 98712 72900, or

visit https://www.thecloudtrain.com/, or

email at support@thecloudtrain.com, or

WhatsApp us @ +91 98712 72900

Exercise 1: AWS ECS

a) Create a AWS Fargate cluster using Network Only mode using hands-on doc.
b) Create an ECS task definition to deploy a containerized app using docker image: **vistasunil/devopsdemo:latest**
c) Run this ECS task on the AWS Fargate cluster that was create in step a.
d) Use the public IP of deployed ECS cluster to access your app on a web browser using URL: ***http:<public_IP>/devopsIQ .***

Exercise 2: AWS EKS

a) Install all the prerequisites to setup and connect to your EKS cluster using hands-on doc.
b) Create AWS Networking setup required for EKS cluster using CloudFormation template from URL : https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml
c) Setup all IAM roles using JSON policy documents provided in hands-on document.
d) Once everything is ready, create your EKS cluster to deploy a Kubernetes application.
e) Connect and test your cluster using kubectl command line.
f) Create a AWS Fargate profile to attach to this EKS cluster following all the IAM role and policy requirements (Refer hands-on doc). This will be required to deploy your pods on Fargate environment.
g) Deploy below yaml manifests using kubectl apply on the configured EKS cluster:

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
  namespace: default
  labels:
    app: my-app
spec:
  selector:
    app: my-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deployment
  namespace: default
  labels:
    app: my-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: beta.kubernetes.io/arch
                operator: In
                values:
                - amd64
                - arm64
      containers:
      - name: nginx
        image: public.ecr.aws/z9d2n7e1/nginx:1.19.5
        ports:
        - containerPort: 80
```

h) Verify your application is deployed successfully and all 3 pods are up and running in the provided namespace.

Exercise 3: AWS CI/CD: CodeCommit, CodeDeploy and CodePipeline

a) Replicate and practice the same CI/CD setup scenario provided in Hands-on document.

NOTE: DELETE ALL THE RESOURCES CREATED TO AVOID UNNECESSARY COSTS IN YOUR AWS ACCOUNT