**Deployments and Orchestration**

**Prerequisites:**

1. Create a free tier account for AWS
2. The kubectl command line tool
3. AWS CLI (https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html)
4. Install docker
5. Install kubectl
6. Install eksctl
7. Install notepad ++ , visualstudio code for creating or editing docker,terraform or yaml files
8. Install Terraform
9. Connect terraform with aws using IAM access key

**Deployment process**

* create cluster , node group , vpc , ec2 etc using terraform
* #connecting to EKS

aws eks update-kubeconfig --region us-west-1 --name devloot-eks-B8vIw1nf

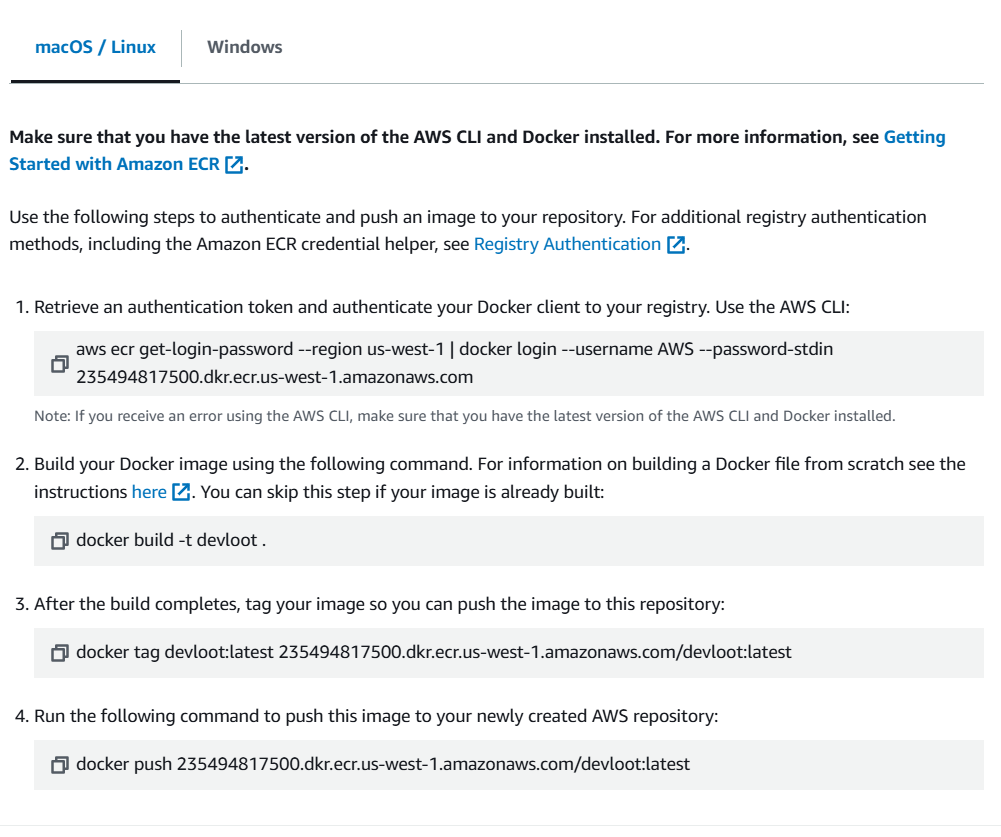
* Create IAM role for EKS Cluster
* Create Dedicated VPC for EKS Cluster

**OR**

* Create AWS Cloud Formation stack  
  3 Create EKS Cluster using Terraform
* Create Access key for the user from AWS IAM Consol
* Install and setup IAM Authenticator and Kubectl utility
* Create IAM role for EKS worker nodes
* *Create worker nodes*
* Create ECR repository

*After the infrastructure configuration*

**Deployment Steps**

* ***Push this image to ECR***
* 
* 1.Retrieve an authentication token and authenticate your Docker client to your registry. Use the AWS CLI:

aws ecr get-login-password --region us-west-1 | docker login --username AWS --password-stdin 235494817500.dkr.ecr.us-west-1.amazonaws.com

* 2. Build your Docker image using the following command. For information on building a Docker file from scratch see the instructions [here](https://docs.aws.amazon.com/AmazonECS/latest/developerguide/docker-basics.html). You can skip this step if your image is already built

docker build -t devloot .

* 3. After the build completes, tag your image so you can push the image to this repository:

docker tag devloot:latest 235494817500.dkr.ecr.us-west-1.amazonaws.com/devloot:latest

* 4.Run the following command to push this image to your newly created AWS repository:

docker push 235494817500.dkr.ecr.us-west-1.amazonaws.com/devloot:latest

**Deploy the image to EKS**

* Installing the AWS Load Balancer Controller add-on.
* Create IAM OIDC provider
* Deploy the AWS Load Balancer Controller to an Amazon EKS cluster
* Create an IAM policy using the policy downloaded

aws iam create-policy --policy-name AWSLoadBalancerControllerIAMPolicy --policy-document file://iam\_policy.json

* Create an IAM role. Create a Kubernetes service account named aws-load-balancer-controller in the kube-system namespace for the AWS Load Balancer Controller and annotate the Kubernetes service account with the name of the IAM role.
* Retrieve your cluster's OIDC provider ID and store it in a variable.

*aws eks describe-cluster --name eks-pms-dev --query "cluster.identity.oidc.issuer" --output text*

* **Create the IAM role**

*aws iam create-role --role-name AmazonEKSLoadBalancerControllerRole --assume-role-policy-document file://"load-balancer-role-trust-policy.json"*

* **Attach the required Amazon EKS managed IAM policy to the IAM role**

*aws iam attach-role-policy --policy-arn*

*arn:aws:iam::* 235494817500*:policy/AWSLoadBalancerControllerIAMPolicy --role-name AmazonEKSLoadBalancerControllerRole*

* **Create the Kubernetes service account on your cluster.**

*cat >aws-load-balancer-controller-service-account.yaml <<EOF*

*apiVersion: v1*

*kind: ServiceAccount*

*metadata:*

*labels:*

*app.kubernetes.io/component: controller*

*app.kubernetes.io/name: aws-load-balancer-controller*

*name: aws-load-balancer-controller*

*namespace: kube-system*

*annotations:*

*eks.amazonaws.com/role-arn: arn:aws:iam::* 235494817500*:role/AmazonEKSLoadBalancerControllerRole*

***kubectl apply -f aws-load-balancer-controller-service-account.yaml***

* ***Deploy the Helm chart***

***helm repo add eks https://aws.github.io/eks-charts***

*helm upgrade -i aws-load-balancer-controller eks/aws-load-balancer-controller --set clusterName=*[*devloot-eks-B8vIw1nf*](https://us-west-1.console.aws.amazon.com/eks/home?region=us-west-1#/clusters/devloot-eks-B8vIw1nf)*--set serviceAccount.create=false --set region=us-east-1 --set vpcId=*vpc-036cca00236783c25 *--set serviceAccount.name=aws-load-balancer-controller -n kube-system*

*\*/*

* *kubectl -n kube-system rollout status deployment aws-load-balancer-controller*

*kubectl get deployment -n kube-system aws-load-balancer-controller*

* ***Creating Namespace***
* ***#Create a yaml file with following content and apply using kubectl***

*apiVersion: v1*

*kind: Namespace*

*metadata:*

*name: poc-dev-ns*

*#Apply the yaml using following command*

*kubectl apply -f 1.* *poc\_namespace.yaml*

* ***Creating Deployment in EKS***

***#Create a yaml file with following content***

*apiVersion: apps/v1*

*kind: Deployment*

*metadata:*

*name: poc-dev-deployment*

*namespace: poc-dev-ns*

*spec:*

*selector:*

*matchLabels:*

*app.kubernetes.io/name: poc-dev*

*replicas: 2*

*template:*

*metadata:*

*labels:*

*app.kubernetes.io/name: poc-dev*

*spec:*

*containers:*

*- image: 235494817500.dkr.ecr.us-west-1.amazonaws.com/devloot:latest*

*imagePullPolicy: Always*

*name: poc-dev*

*ports:*

*- containerPort: 3000#Apply the yaml using following command*

***kubectl apply -f poc\_deployment.yaml***

* ***Creating Services in EKS***

***#Create a yaml file with following content***

*apiVersion: v1*

*kind: Service*

*metadata:*

*name: poc-dev-service*

*namespace: poc-dev-ns*

*spec:*

*ports:*

*- port: 80*

*targetPort: 3000*

*protocol: TCP*

*type: NodePort*

*selector:*

*app.kubernetes.io/name: poc-dev*

* ***#Apply the yaml using following command***

*kubectl apply -f poc\_service.yaml*

* ***Creating Ingress in EKS***

***#Create a yaml file with following content***

*apiVersion: networking.k8s.io/v1*

*kind: Ingress*

*metadata:*

*name: poc-dev-ingress*

*namespace: poc-dev-ns*

*annotations:*

*kubernetes.io/ingress.class: alb*

*alb.ingress.kubernetes.io/target-type: instance*

*alb.ingress.kubernetes.io/scheme: internet-facing*

*alb.ingress.kubernetes.io/listen-ports: '[{"HTTP":80} , {"HTTPS": 443}]'*

*alb.ingress.kubernetes.io/certificate-arn: arn:aws:acm:us-east-1:235494817500:certificate/4dea5f67-e30c-4d95-9fcc-dca71d402ac9*

*alb.ingress.kubernetes.io/actions.ssl-redirect: '{"Type": "redirect", "RedirectConfig": { "Protocol": "HTTPS", "Port": "443", "StatusCode": "HTTP\_301"}}'*

*spec:*

*rules:*

*- host: poc-dev.app*

*http:*

*paths:*

*- path: /*

*pathType: Prefix*

*backend:*

*service:*

*name: poc-dev-service*

*port:*

*number: 80*

*tls:*

*- hosts:*

*- poc-dev.app*

* ***#Apply the yaml using following command***

*kubectl apply -f poc\_ingress.yaml*