How to Set Up AWS Load Balancer Controller in EKS Cluster

Step 1: Create a eks cluster in dedicated VPC

Step 2: Create IAM Role(ex: EksClusterRole) in AWS

Entity Type->Aws Service, Use Case->EKS->EKS Cluster
Role Name->EksClusterRole

Step 3: Create cluster using created VPC and IAM(ex: EksClusterRole) role cluster endpoint access: public and private

Step 4: Create Ec2 Instance(k8s_client_machine) in different vpc(ex: default VPC)

- **Install kubectl on EC2 instance**
- **Install AWS Cli on EC2 instance**
- **Configure Aws Cli with Credentials**

Note: we can use root user accesskey and secret key access

#### Note: If ec2(where install kubectl) and cluster are not in same vpc so need to update remote machine(k8s_client_machine) by this command:

update kubeconfig file in remote machine from cluster using bellow

```
aws eks update-kubeconfig --name <cluster-name> --region <region_name>
```

Step 5: Create Iam role(ex: EksWorkerNode) for Eks workder nodes(usecase EC2) with bellow polices

1. **AmazonEKSWorkerNodePolicy**
2. AmazonEKS_CNI_Policy
3. AmazonEC2ContainerRegistryReadOnly

Step 6: Create worker Node Group

Go to cluster->Compute->Node Group
->Select the Role we have created for workerNodes
->Use t2.large
->Min 2 and Max 2

Step 7: Once Node Group added then check node in k8s_client_machine

```
$ kubectl get nodes
$ kubectl get pods --all-namespaces
```

# Create POD and expose the POD using NodePort Service/LoadBalancer

### Note: Enable node port in secutriy Group to access than in out browser

```
$ kubectl get pods
$ kubectl get deployment
$ kubectl get svc
$ kubectl get pods -o wide
```

### Try to access from browser by loadbalancer DNS

<u>Manifest.yml</u>

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: portfolio-deployment
  labels:
    app: nginx
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - name: nginx
        image: nginx:latest
        ports:
        - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: nginx-service
spec:
  type: LoadBalancer
  selector:
    app: nginx
  ports:
    - protocol: TCP
      port: 80
      targetPort: 8080
```