winget install Amazon.AWSCLI

#check version

aws --version

A screenshot of a computer

AI-generated content may be incorrect.



aws sts get-caller-identity

A screenshot of a computer

AI-generated content may be incorrect.



Add permission to IAM

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Create container repo

aws ecr create-repository --repository-name bankingrepo

A computer screen with white text

AI-generated content may be incorrect.

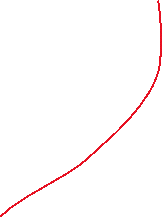
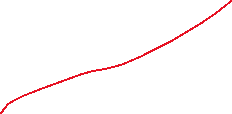


#get docker login credential

Get Screen shot from

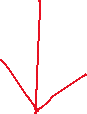
A screenshot of a computer

AI-generated content may be incorrect.



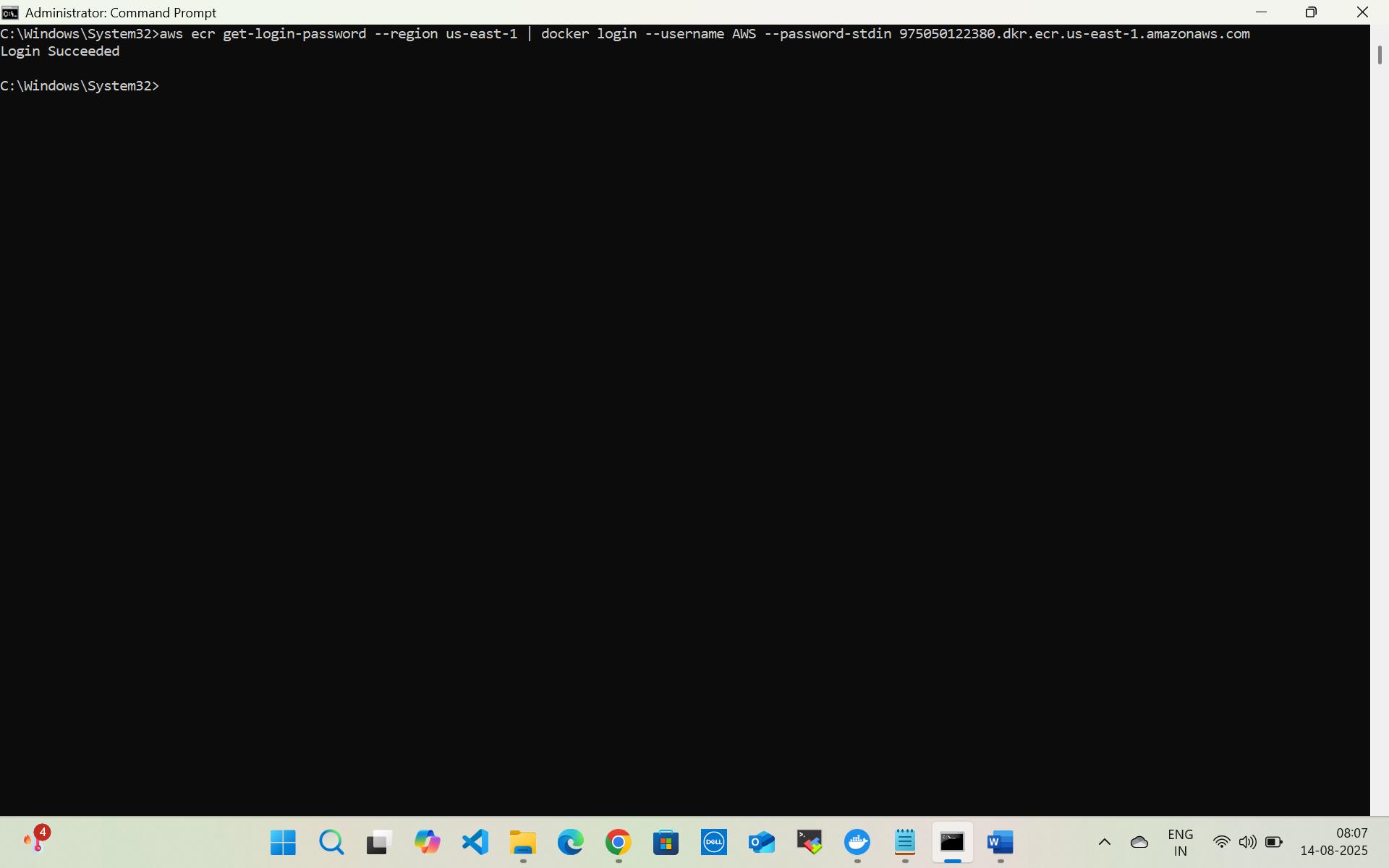
A screenshot of a computer

AI-generated content may be incorrect.



aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 975050122888.dkr.ecr.us-east-1.amazonaws.com





A screenshot of a computer

AI-generated content may be incorrect.

docker tag customerapi:latest 975050122380.dkr.ecr.us-east-1.amazonaws.com/bankingrepo:customerapiv1



docker push 975050122380.dkr.ecr.us-east-1.amazonaws.com/bankingrepo:customerapiv1



A screenshot of a computer

AI-generated content may be incorrect.



A screenshot of a computer

AI-generated content may be incorrect.



Create an EKS cluster (with eksctl)

A screenshot of a computer

AI-generated content may be incorrect.

EKS cluster Policy

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "AmazonEKSClusterPolicy",

"Effect": "Allow",

"Action": [

"autoscaling:DescribeAutoScalingGroups",

"autoscaling:UpdateAutoScalingGroup",

"ec2:AttachVolume",

"ec2:AuthorizeSecurityGroupIngress",

"ec2:CreateRoute",

"ec2:CreateSecurityGroup",

"ec2:CreateTags",

"ec2:CreateVolume",

"ec2:DeleteRoute",

"ec2:DeleteSecurityGroup",

"ec2:DeleteVolume",

"ec2:DescribeInstances",

"ec2:DescribeRouteTables",

"ec2:DescribeSecurityGroups",

"ec2:DescribeSubnets",

"ec2:DescribeVolumes",

"ec2:DescribeVolumesModifications",

"ec2:DescribeVpcs",

"ec2:DescribeDhcpOptions",

"ec2:DescribeNetworkInterfaces",

"ec2:DescribeAvailabilityZones",

"ec2:DetachVolume",

"ec2:ModifyInstanceAttribute",

"ec2:ModifyVolume",

"ec2:RevokeSecurityGroupIngress",

"ec2:DescribeAccountAttributes",

"ec2:DescribeAddresses",

"ec2:DescribeInternetGateways",

"ec2:DescribeInstanceTopology",

"elasticloadbalancing:AddTags",

"elasticloadbalancing:ApplySecurityGroupsToLoadBalancer",

"elasticloadbalancing:AttachLoadBalancerToSubnets",

"elasticloadbalancing:ConfigureHealthCheck",

"elasticloadbalancing:CreateListener",

"elasticloadbalancing:CreateLoadBalancer",

"elasticloadbalancing:CreateLoadBalancerListeners",

"elasticloadbalancing:CreateLoadBalancerPolicy",

"elasticloadbalancing:CreateTargetGroup",

"elasticloadbalancing:DeleteListener",

"elasticloadbalancing:DeleteLoadBalancer",

"elasticloadbalancing:DeleteLoadBalancerListeners",

"elasticloadbalancing:DeleteTargetGroup",

"elasticloadbalancing:DeregisterInstancesFromLoadBalancer",

"elasticloadbalancing:DeregisterTargets",

"elasticloadbalancing:DescribeListeners",

"elasticloadbalancing:DescribeLoadBalancerAttributes",

"elasticloadbalancing:DescribeLoadBalancerPolicies",

"elasticloadbalancing:DescribeLoadBalancers",

"elasticloadbalancing:DescribeTargetGroupAttributes",

"elasticloadbalancing:DescribeTargetGroups",

"elasticloadbalancing:DescribeTargetHealth",

"elasticloadbalancing:DetachLoadBalancerFromSubnets",

"elasticloadbalancing:ModifyListener",

"elasticloadbalancing:ModifyLoadBalancerAttributes",

"elasticloadbalancing:ModifyTargetGroup",

"elasticloadbalancing:ModifyTargetGroupAttributes",

"elasticloadbalancing:RegisterInstancesWithLoadBalancer",

"elasticloadbalancing:RegisterTargets",

"elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer",

"elasticloadbalancing:SetLoadBalancerPoliciesOfListener",

"kms:DescribeKey"

],

"Resource": "\*"

},

{

"Sid": "AmazonEKSClusterPolicySLRCreate",

"Effect": "Allow",

"Action": "iam:CreateServiceLinkedRole",

"Resource": "\*",

"Condition": {

"StringEquals": {

"iam:AWSServiceName": "elasticloadbalancing.amazonaws.com"

}

}

},

{

"Sid": "AmazonEKSClusterPolicyENIDelete",

"Effect": "Allow",

"Action": "ec2:DeleteNetworkInterface",

"Resource": "\*",

"Condition": {

"StringEquals": {

"ec2:ResourceTag/eks:eni:owner": "amazon-vpc-cni"

}

}

}

]

}

A screenshot of a computer

AI-generated content may be incorrect.

Secret Manager read write policy

{

"Version": "2012-10-17",

"Statement": [

{

"Sid": "BasePermissions",

"Effect": "Allow",

"Action": [

"secretsmanager:\*",

"cloudformation:CreateChangeSet",

"cloudformation:DescribeChangeSet",

"cloudformation:DescribeStackResource",

"cloudformation:DescribeStacks",

"cloudformation:ExecuteChangeSet",

"docdb-elastic:GetCluster",

"docdb-elastic:ListClusters",

"ec2:DescribeSecurityGroups",

"ec2:DescribeSubnets",

"ec2:DescribeVpcs",

"kms:DescribeKey",

"kms:ListAliases",

"kms:ListKeys",

"lambda:ListFunctions",

"rds:DescribeDBClusters",

"rds:DescribeDBInstances",

"redshift:DescribeClusters",

"redshift-serverless:ListWorkgroups",

"redshift-serverless:GetNamespace",

"tag:GetResources"

],

"Resource": "\*"

},

{

"Sid": "LambdaPermissions",

"Effect": "Allow",

"Action": [

"lambda:AddPermission",

"lambda:CreateFunction",

"lambda:GetFunction",

"lambda:InvokeFunction",

"lambda:UpdateFunctionConfiguration"

],

"Resource": "arn:aws:lambda:\*:\*:function:SecretsManager\*"

},

{

"Sid": "SARPermissions",

"Effect": "Allow",

"Action": [

"serverlessrepo:CreateCloudFormationChangeSet",

"serverlessrepo:GetApplication"

],

"Resource": "arn:aws:serverlessrepo:\*:\*:applications/SecretsManager\*"

},

{

"Sid": "S3Permissions",

"Effect": "Allow",

"Action": [

"s3:GetObject"

],

"Resource": [

"arn:aws:s3:::awsserverlessrepo-changesets\*",

"arn:aws:s3:::secrets-manager-rotation-apps-\*/\*"

]

}

]

}

A screenshot of a computer

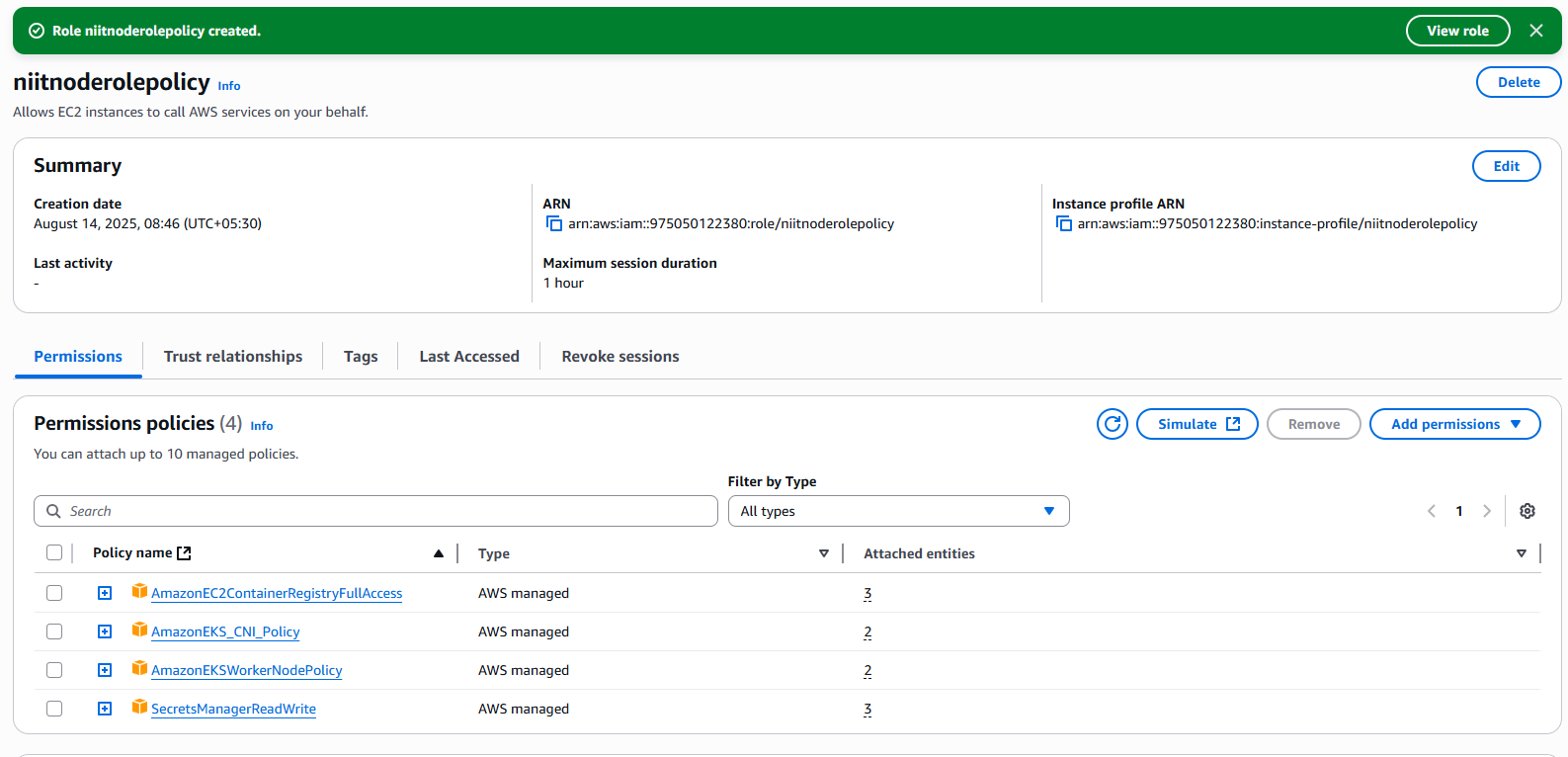
AI-generated content may be incorrect.

Noderole Policy (Ec2 usecase)

A screenshot of a computer

AI-generated content may be incorrect.

Create role



Niit cluster admin role (EKS usecase)

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Niit cluster admin role

Add

**Cluster role missing recommended managed policies**

The cluster role must have the following managed policies or equivalent permissions to use EKS Auto Mode:

* AmazonEKSBlockStoragePolicy
* AmazonEKSComputePolicy
* AmazonEKSLoadBalancingPolicy
* AmazonEKSNetworkingPolicy

A screenshot of a computer

AI-generated content may be incorrect.

Edit Trust policy of cluster admin

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Or

#eks tool

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

#create kube cluster using us-east-2

eksctl create cluster --name my-cluster --region us-east-2 --nodegroup-name standard-workers --node-type t3.medium --nodes 2 --nodes-min 1 --nodes-max 3 –managed

Add EKS permission to eksadmin

A screenshot of a computer

AI-generated content may be incorrect.

#configure aws kube in local

Use the AWS CLI update-kubeconfig command to create or update your kubeconfig for your cluster.

aws eks update-kubeconfig --name bankingcluster

A computer screen with a black screen

AI-generated content may be incorrect.

aws eks update-kubeconfig --name bankingcluster --region us-east-1

A computer screen with a black screen

AI-generated content may be incorrect.

#check status

aws eks describe-cluster --name bankingcluster—region us-east-1 --query "cluster.status" --output text

update config file

aws eks update-kubeconfig --name bankingcluster --region us-east-1 --kubeconfig "C:\Users\param\.kube\config"

kubectl config get-contexts

A screenshot of a computer

AI-generated content may be incorrect.

In powershell

Get-Command aws | Select-Object Source

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Add access policy to eks admin

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A computer screen shot of a black screen

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Create load balancer controller

Create EKS Tool

Open powershell

Invoke-WebRequest -Uri "https://github.com/eksctl-io/eksctl/releases/latest/download/eksctl\_Windows\_amd64.zip" -OutFile "eksctl.zip"

A screenshot of a computer

AI-generated content may be incorrect.

Expand-Archive -Path "eksctl.zip" -DestinationPath "C:\eksctl"

A computer screen shot of a black screen

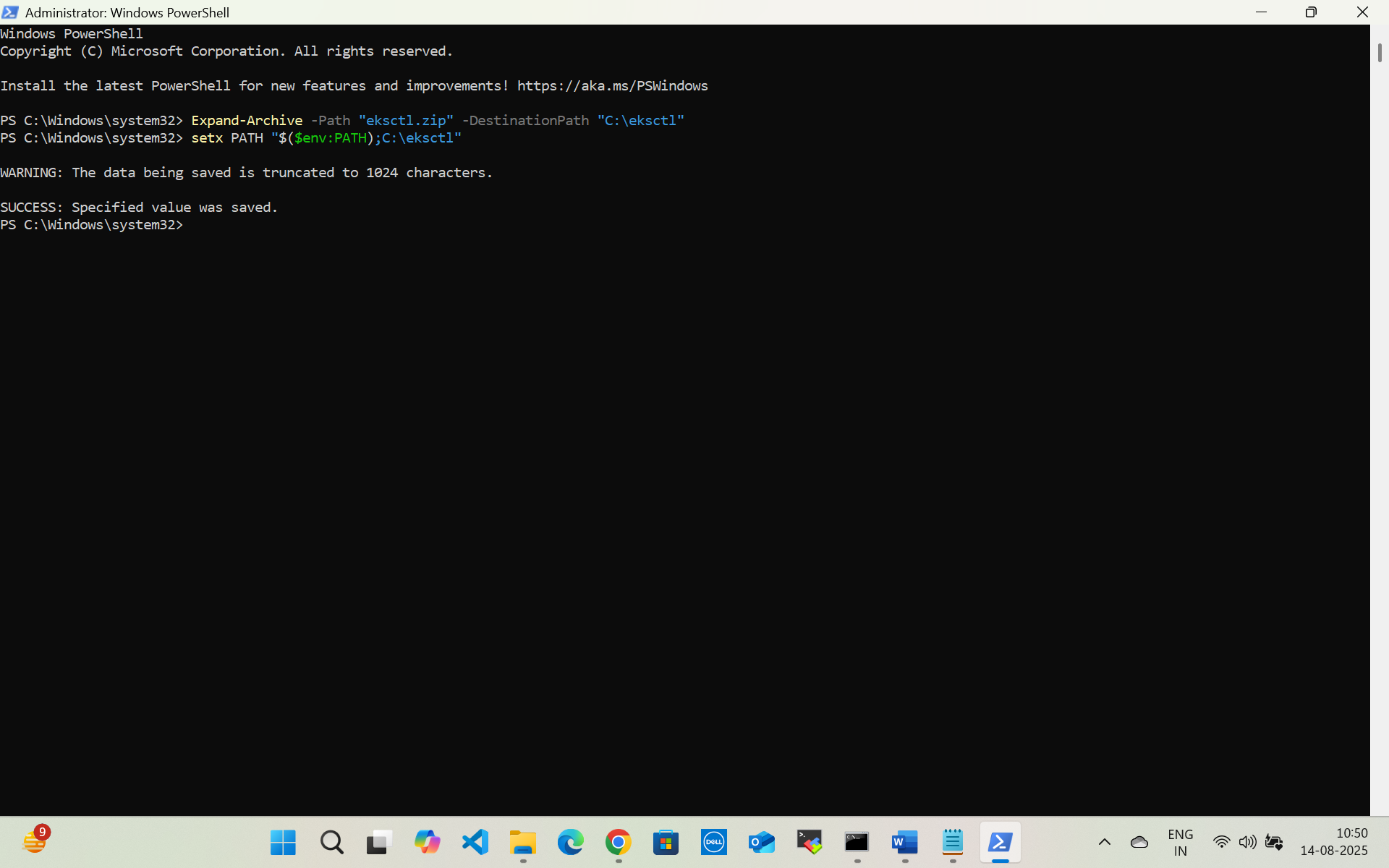
AI-generated content may be incorrect.

**Add eksctl to your PATH**

In PowerShell (run as Administrator):

setx PATH "$($env:PATH);C:\eksctl"

Close and reopen PowerShell so the PATH refreshes.



Set env variable for eksctl.exe

A screenshot of a computer

AI-generated content may be incorrect.

Go to user eksadmin add policy json

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": [

"iam:GetOpenIDConnectProvider",

"iam:CreateOpenIDConnectProvider",

"iam:DeleteOpenIDConnectProvider",

"iam:UpdateOpenIDConnectProviderThumbprint",

"iam:TagOpenIDConnectProvider",

"iam:CreatePolicy",

"iam:GetPolicy",

"iam:AttachRolePolicy",

"iam:CreateRole",

"iam:PassRole",

"iam:TagRole",

“iam:GetRole”

],

"Resource": "\*"

}

]

}

A screenshot of a computer

AI-generated content may be incorrect.

eksctl utils associate-iam-oidc-provider --cluster bankingcluster --region us-east-1 –approve

A screenshot of a computer

AI-generated content may be incorrect.

Create IAM policy

curl -o iam-policy.json <https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/main/docs/install/iam_policy.json>

aws iam create-policy --policy-name AWSLoadBalancerControllerIAMPolicy --policy-document <file://iam-policy.json>

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.



A screenshot of a computer

AI-generated content may be incorrect.

{

"Version": "2012-10-17",

"Statement": [

{

"Effect": "Allow",

"Action": [

"iam:CreateServiceLinkedRole"

],

"Resource": "\*",

"Condition": {

"StringEquals": {

"iam:AWSServiceName": "elasticloadbalancing.amazonaws.com"

}

}

},

{

"Effect": "Allow",

"Action": [

"ec2:DescribeAccountAttributes",

"ec2:DescribeAddresses",

"ec2:DescribeAvailabilityZones",

"ec2:DescribeInternetGateways",

"ec2:DescribeVpcs",

"ec2:DescribeVpcPeeringConnections",

"ec2:DescribeSubnets",

"ec2:DescribeSecurityGroups",

"ec2:DescribeInstances",

"ec2:DescribeNetworkInterfaces",

"ec2:DescribeTags",

"ec2:GetCoipPoolUsage",

"ec2:DescribeCoipPools",

"ec2:GetSecurityGroupsForVpc",

"ec2:DescribeIpamPools",

"ec2:DescribeRouteTables",

"elasticloadbalancing:DescribeLoadBalancers",

"elasticloadbalancing:DescribeLoadBalancerAttributes",

"elasticloadbalancing:DescribeListeners",

"elasticloadbalancing:DescribeListenerCertificates",

"elasticloadbalancing:DescribeSSLPolicies",

"elasticloadbalancing:DescribeRules",

"elasticloadbalancing:DescribeTargetGroups",

"elasticloadbalancing:DescribeTargetGroupAttributes",

"elasticloadbalancing:DescribeTargetHealth",

"elasticloadbalancing:DescribeTags",

"elasticloadbalancing:DescribeTrustStores",

"elasticloadbalancing:DescribeListenerAttributes",

"elasticloadbalancing:DescribeCapacityReservation"

],

"Resource": "\*"

},

{

"Effect": "Allow",

"Action": [

"cognito-idp:DescribeUserPoolClient",

"acm:ListCertificates",

"acm:DescribeCertificate",

"iam:ListServerCertificates",

"iam:GetServerCertificate",

"waf-regional:GetWebACL",

"waf-regional:GetWebACLForResource",

"waf-regional:AssociateWebACL",

"waf-regional:DisassociateWebACL",

"wafv2:GetWebACL",

"wafv2:GetWebACLForResource",

"wafv2:AssociateWebACL",

"wafv2:DisassociateWebACL",

"shield:GetSubscriptionState",

"shield:DescribeProtection",

"shield:CreateProtection",

"shield:DeleteProtection"

],

"Resource": "\*"

},

{

"Effect": "Allow",

"Action": [

"ec2:AuthorizeSecurityGroupIngress",

"ec2:RevokeSecurityGroupIngress"

],

"Resource": "\*"

},

{

"Effect": "Allow",

"Action": [

"ec2:CreateSecurityGroup"

],

"Resource": "\*"

},

{

"Effect": "Allow",

"Action": [

"ec2:CreateTags"

],

"Resource": "arn:aws:ec2:\*:\*:security-group/\*",

"Condition": {

"StringEquals": {

"ec2:CreateAction": "CreateSecurityGroup"

},

"Null": {

"aws:RequestTag/elbv2.k8s.aws/cluster": "false"

}

}

},

{

"Effect": "Allow",

"Action": [

"ec2:CreateTags",

"ec2:DeleteTags"

],

"Resource": "arn:aws:ec2:\*:\*:security-group/\*",

"Condition": {

"Null": {

"aws:RequestTag/elbv2.k8s.aws/cluster": "true",

"aws:ResourceTag/elbv2.k8s.aws/cluster": "false"

}

}

},

{

"Effect": "Allow",

"Action": [

"ec2:AuthorizeSecurityGroupIngress",

"ec2:RevokeSecurityGroupIngress",

"ec2:DeleteSecurityGroup"

],

"Resource": "\*",

"Condition": {

"Null": {

"aws:ResourceTag/elbv2.k8s.aws/cluster": "false"

}

}

},

{

"Effect": "Allow",

"Action": [

"elasticloadbalancing:CreateLoadBalancer",

"elasticloadbalancing:CreateTargetGroup"

],

"Resource": "\*",

"Condition": {

"Null": {

"aws:RequestTag/elbv2.k8s.aws/cluster": "false"

}

}

},

{

"Effect": "Allow",

"Action": [

"elasticloadbalancing:CreateListener",

"elasticloadbalancing:DeleteListener",

"elasticloadbalancing:CreateRule",

"elasticloadbalancing:DeleteRule"

],

"Resource": "\*"

},

{

"Effect": "Allow",

"Action": [

"elasticloadbalancing:AddTags",

"elasticloadbalancing:RemoveTags"

],

"Resource": [

"arn:aws:elasticloadbalancing:\*:\*:targetgroup/\*/\*",

"arn:aws:elasticloadbalancing:\*:\*:loadbalancer/net/\*/\*",

"arn:aws:elasticloadbalancing:\*:\*:loadbalancer/app/\*/\*"

],

"Condition": {

"Null": {

"aws:RequestTag/elbv2.k8s.aws/cluster": "true",

"aws:ResourceTag/elbv2.k8s.aws/cluster": "false"

}

}

},

{

"Effect": "Allow",

"Action": [

"elasticloadbalancing:AddTags",

"elasticloadbalancing:RemoveTags"

],

"Resource": [

"arn:aws:elasticloadbalancing:\*:\*:listener/net/\*/\*/\*",

"arn:aws:elasticloadbalancing:\*:\*:listener/app/\*/\*/\*",

"arn:aws:elasticloadbalancing:\*:\*:listener-rule/net/\*/\*/\*",

"arn:aws:elasticloadbalancing:\*:\*:listener-rule/app/\*/\*/\*"

]

},

{

"Effect": "Allow",

"Action": [

"elasticloadbalancing:ModifyLoadBalancerAttributes",

"elasticloadbalancing:SetIpAddressType",

"elasticloadbalancing:SetSecurityGroups",

"elasticloadbalancing:SetSubnets",

"elasticloadbalancing:DeleteLoadBalancer",

"elasticloadbalancing:ModifyTargetGroup",

"elasticloadbalancing:ModifyTargetGroupAttributes",

"elasticloadbalancing:DeleteTargetGroup",

"elasticloadbalancing:ModifyListenerAttributes",

"elasticloadbalancing:ModifyCapacityReservation",

"elasticloadbalancing:ModifyIpPools"

],

"Resource": "\*",

"Condition": {

"Null": {

"aws:ResourceTag/elbv2.k8s.aws/cluster": "false"

}

}

},

{

"Effect": "Allow",

"Action": [

"elasticloadbalancing:AddTags"

],

"Resource": [

"arn:aws:elasticloadbalancing:\*:\*:targetgroup/\*/\*",

"arn:aws:elasticloadbalancing:\*:\*:loadbalancer/net/\*/\*",

"arn:aws:elasticloadbalancing:\*:\*:loadbalancer/app/\*/\*"

],

"Condition": {

"StringEquals": {

"elasticloadbalancing:CreateAction": [

"CreateTargetGroup",

"CreateLoadBalancer"

]

},

"Null": {

"aws:RequestTag/elbv2.k8s.aws/cluster": "false"

}

}

},

{

"Effect": "Allow",

"Action": [

"elasticloadbalancing:RegisterTargets",

"elasticloadbalancing:DeregisterTargets"

],

"Resource": "arn:aws:elasticloadbalancing:\*:\*:targetgroup/\*/\*"

},

{

"Effect": "Allow",

"Action": [

"elasticloadbalancing:SetWebAcl",

"elasticloadbalancing:ModifyListener",

"elasticloadbalancing:AddListenerCertificates",

"elasticloadbalancing:RemoveListenerCertificates",

"elasticloadbalancing:ModifyRule",

"elasticloadbalancing:SetRulePriorities"

],

"Resource": "\*"

}

]

}

A screenshot of a computer

AI-generated content may be incorrect.

Create Load Balancer

eksctl create iamserviceaccount --cluster bankingcluster --region us-east-1 --namespace kube-system --name aws-load-balancer-controller --role-name AmazonEKSLoadBalancerControllerRole --attach-policy-arn arn:aws:iam::975050122380:policy/AWSLoadBalancerControllerIAMPolicy --override-existing-serviceaccounts --approve

A screenshot of a computer

AI-generated content may be incorrect.

Confirm

kubectl -n kube-system get sa aws-load-balancer-controller -o yaml

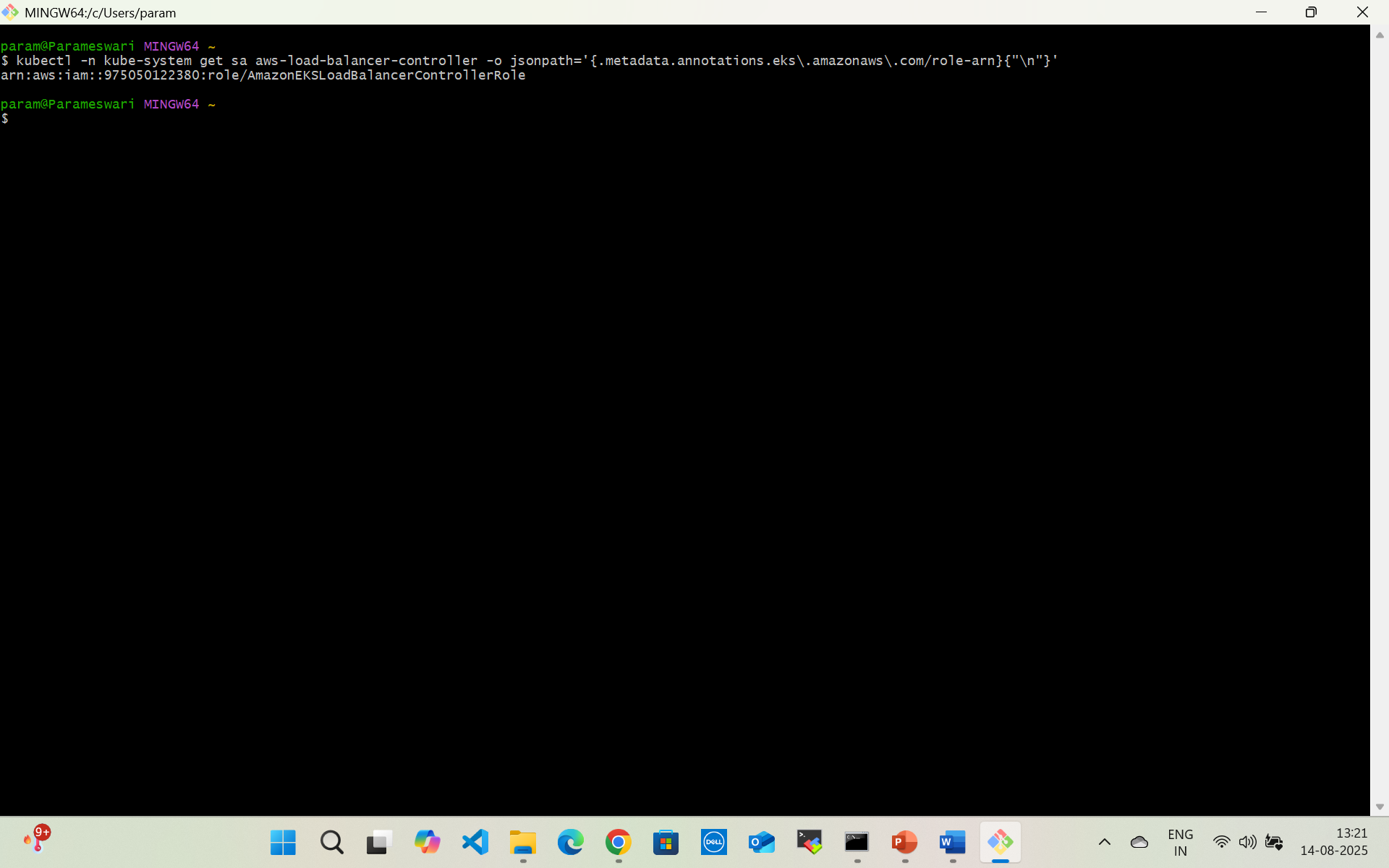
A screenshot of a computer

AI-generated content may be incorrect.

kubectl -n kube-system get sa aws-load-balancer-controller

kubectl -n kube-system get sa aws-load-balancer-controller -o jsonpath='{.metadata.annotations.eks\.amazonaws\.com/role-arn}{"\n"}'

open bash script



**C) Install the controller (Helm)**

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

helm repo update

helm upgrade -i aws-load-balancer-controller eks/aws-load-balancer-controller -n kube-system --set clusterName=bankingcluster --set region=us-east-1 --set serviceAccount.create=false --set serviceAccount.name=aws-load-balancer-controller

Verify it’s up:

kubectl -n kube-system get deploy aws-load-balancer-controller

kubectl -n kube-system get pods -l app.kubernetes.io/name=aws-load-balancer-controller -o wide

kubectl -n kube-system logs deploy/aws-load-balancer-controller | tail -n 100

**D) Create the IngressClass (fixes your earlier error)**

cat <<'YAML' | kubectl apply -f -

apiVersion: networking.k8s.io/v1

kind: IngressClass

metadata:

name: alb

spec:

controller: ingress.k8s.aws/alb

YAML

kubectl get ingressclass

Goto docker/eks

>kubectl apply -f ingress-customer.yaml

kubectl get ingress customerservice -w

kubectl get endpoints customerservice

A screenshot of a computer

AI-generated content may be incorrect.

**Do these 5 commands (safe to copy-paste)**

# 1) Tell Kubernetes this Ingress should be handled by the ALB controller

kubectl patch ingress customerservice --type='json' \

-p='[{"op":"add","path":"/spec/ingressClassName","value":"alb"}]'

# (alternative to step 1 if you prefer annotations)

kubectl annotate ingress customerservice kubernetes.io/ingress.class=alb --overwrite

# 2) Make it public (or use "internal" for VPC-only)

kubectl annotate ingress customerservice alb.ingress.kubernetes.io/scheme=internet-facing --overwrite

# 3) Choose the target type:

# - If your Service is ClusterIP -> use ip

# - If your Service is NodePort -> use instance

kubectl annotate ingress customerservice alb.ingress.kubernetes.io/target-type=ip --overwrite

# If your Service is NodePort, instead run:

# kubectl annotate ingress customerservice alb.ingress.kubernetes.io/target-type=instance --overwrite

# 4) (Recommended) set a correct healthcheck path for your app

kubectl annotate ingress customerservice alb.ingress.kubernetes.io/healthcheck-path=/actuator/health --overwrite

# 5) Watch it reconcile

kubectl get ing customerservice -w

If things are wired correctly, ADDRESS will populate with an **ALB DNS name** within 1–3 minutes.

**If ADDRESS is still empty, check these quickly**

**A) Ingress events (tells you the exact blocker)**

kubectl describe ingress customerservice | sed -n '/Events/,$p'

**B) Controller health & logs**

kubectl -n kube-system get deploy aws-load-balancer-controller

kubectl -n kube-system logs deploy/aws-load-balancer-controller | egrep -i 'error|denied|forbid|subnet|iam|quota'

**C) Service & endpoints exist**

kubectl get svc customerservice -o wide

kubectl get endpoints customerservice

* Using target-type: ip? Service should be **ClusterIP**.
* Using target-type: instance? Service can be **NodePort**.

**D) Subnet tags (most common cause)**

* On all EKS subnets: kubernetes.io/cluster/<cluster-name>=shared (or owned)
* Public ALB: kubernetes.io/role/elb=1 on public subnets
* Internal ALB: kubernetes.io/role/internal-elb=1 on private subnets

**Minimal known-good Ingress (HTTP, public, IP targets)**

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

name: customerservice

annotations:

kubernetes.io/ingress.class: alb

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

alb.ingress.kubernetes.io/target-type: ip

alb.ingress.kubernetes.io/healthcheck-path: /actuator/health # change if needed

spec:

rules:

- http:

paths:

- path: /

pathType: Prefix

backend:

service:

name: customerservice

port:

number: 7074

Apply it:

# If you use target-type: ip, ensure Service is ClusterIP

kubectl patch svc customerservice -p '{"spec":{"type":"ClusterIP"}}'

kubectl apply -f ingress-customerservice.yaml

kubectl get ing customerservice -w