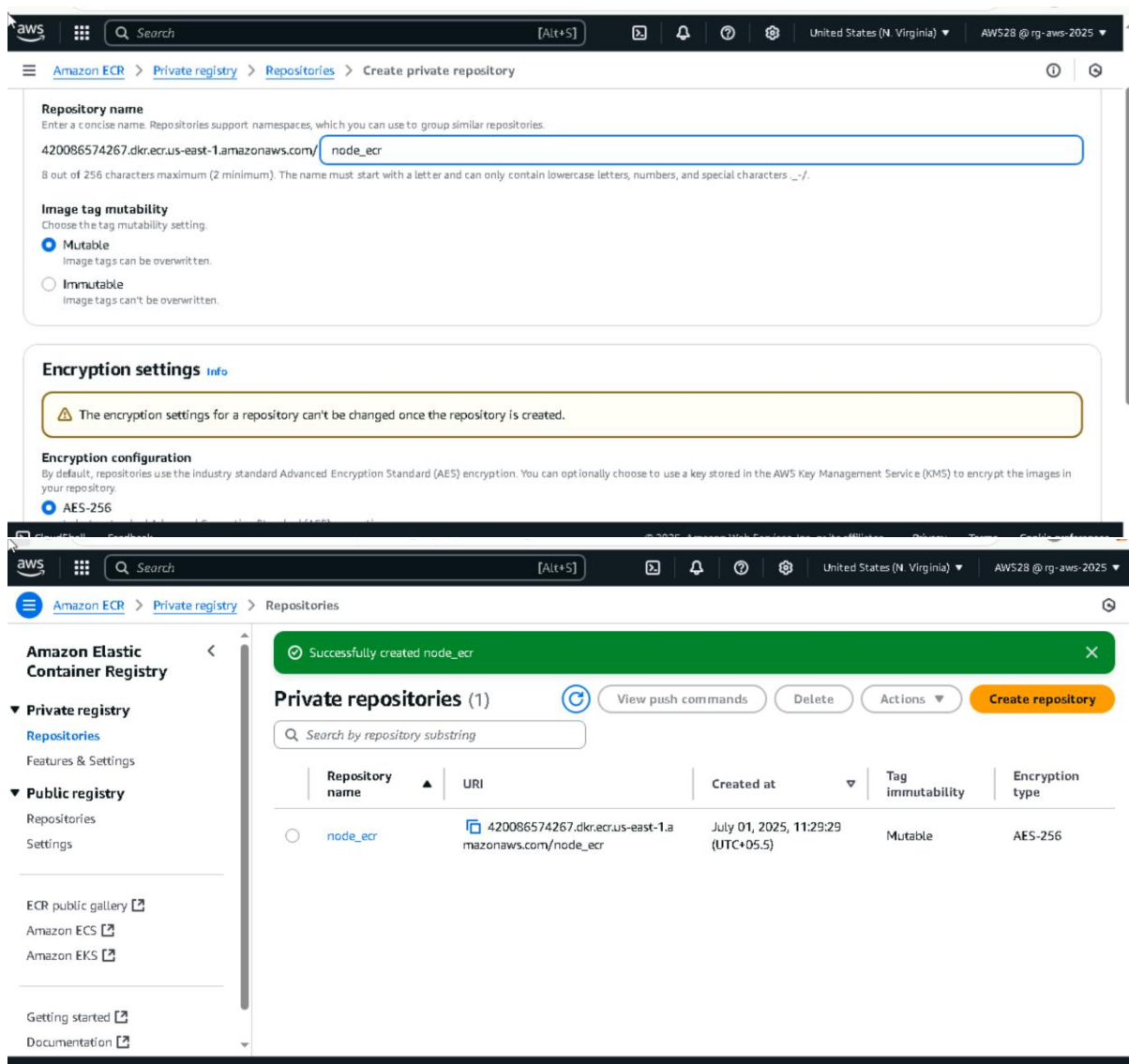


# 1. Establish an end to end ci/cd pipeline in aws to build a docker image of a simple web application, push it to amazon container registry(ecr) and then deploy it to an amazon EKS cluster.

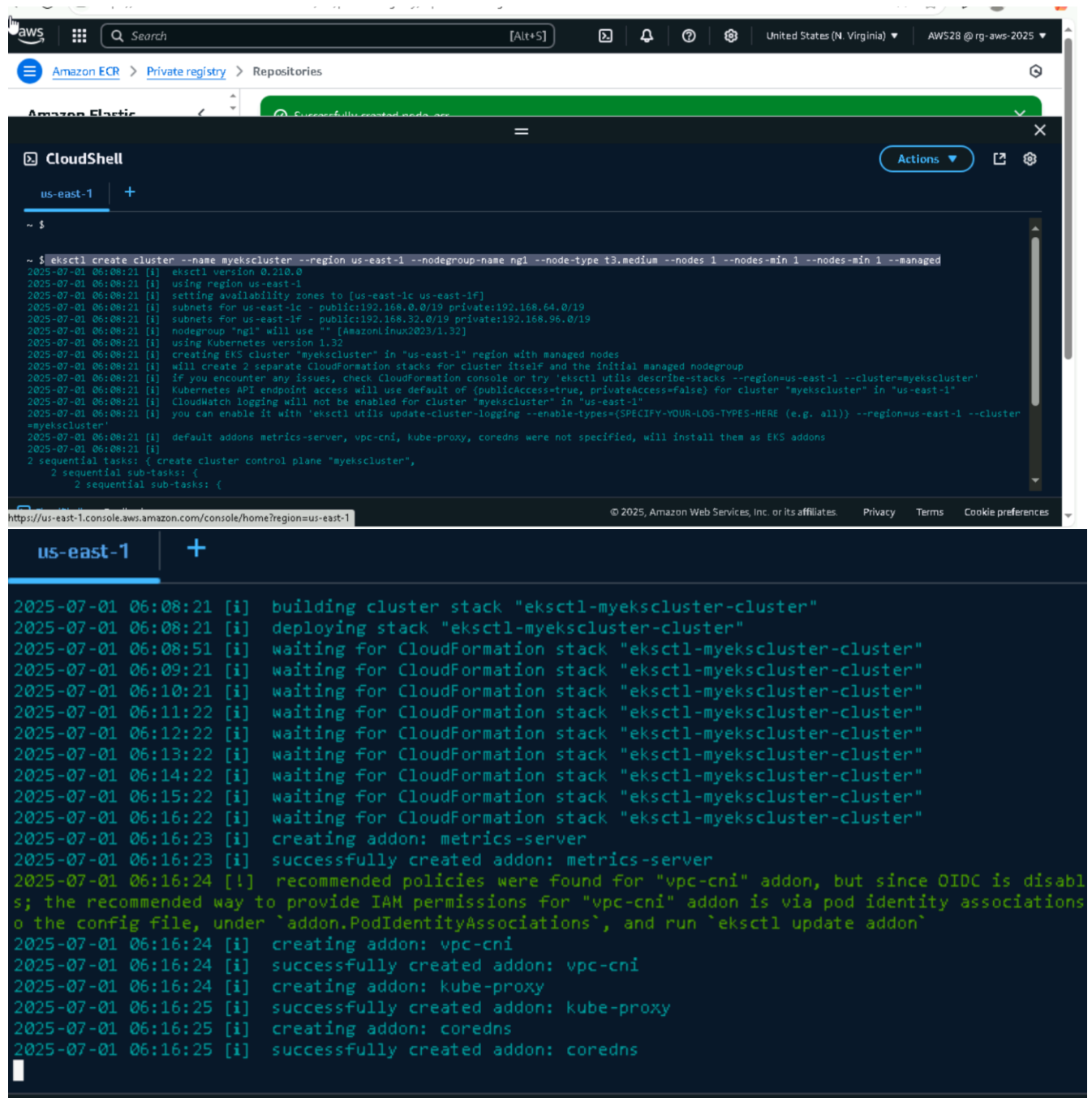
## Provision aws resources

Amazon ECR: created a ecr repository



Amazon EKS cluster: created a eks cluster using cli

Command :



```
aws
[Alt+S]
United States (N. Virginia)
AWS2B @ rg-aws-2025

Amazon ECR > Private registry > Repositories

CloudShell
us-east-1 +

~ $
~ $ eksctl create cluster --name myekscluster --region us-east-1 --nodegroup-name ng1 --node-type t3.medium --nodes 1 --nodes-min 1 --nodes-max 1 --managed
2025-07-01 06:08:21 [i] eksctl version 0.210.0
2025-07-01 06:08:21 [i] using region us-east-1
2025-07-01 06:08:21 [i] setting availability zones to [us-east-1c us-east-1f]
2025-07-01 06:08:21 [i] subnets for us-east-1c - public:192.168.0.0/19 private:192.168.64.0/19
2025-07-01 06:08:21 [i] subnets for us-east-1f - public:192.168.32.0/19 private:192.168.96.0/19
2025-07-01 06:08:21 [i] nodegroup "ng1" will use "" [AmazonLinux2023/1.32]
2025-07-01 06:08:21 [i] using Kubernetes version 1.32
2025-07-01 06:08:21 [i] creating EKS cluster "myekscluster" in "us-east-1" region with managed nodes
2025-07-01 06:08:21 [i] will create 2 separate CloudFormation stacks for cluster itself and the initial managed nodegroup
2025-07-01 06:08:21 [i] if you encounter any issues, check CloudFormation console or try 'eksctl utils describe-stacks --region=us-east-1 --cluster=myekscluster'
2025-07-01 06:08:21 [i] Kubernetes API endpoint access will use default of (publicAccess=true, privateAccess=false) for cluster "myekscluster" in "us-east-1"
2025-07-01 06:08:21 [i] CloudWatch logging will not be enabled for cluster "myekscluster" in "us-east-1"
2025-07-01 06:08:21 [i] you can enable it with 'eksctl utils update-cluster-logging --enable-types=(SPECIFY-YOUR-LOG-TYPES-HERE (e.g. all)) --region=us-east-1 --cluster=myekscluster'
2025-07-01 06:08:21 [i] default addons metrics-server, vpc-cni, kube-proxy, coredns were not specified, will install them as EKS addons
2025-07-01 06:08:21 [i]
2 sequential tasks: { create cluster control plane "myekscluster",
2 sequential sub-tasks: {
2 sequential sub-tasks: {
https://us-east-1.console.aws.amazon.com/console/home?region=us-east-1
© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

us-east-1 +

2025-07-01 06:08:21 [i] building cluster stack "eksctl-myekscluster-cluster"
2025-07-01 06:08:21 [i] deploying stack "eksctl-myekscluster-cluster"
2025-07-01 06:08:51 [i] waiting for CloudFormation stack "eksctl-myekscluster-cluster"
2025-07-01 06:09:21 [i] waiting for CloudFormation stack "eksctl-myekscluster-cluster"
2025-07-01 06:10:21 [i] waiting for CloudFormation stack "eksctl-myekscluster-cluster"
2025-07-01 06:11:22 [i] waiting for CloudFormation stack "eksctl-myekscluster-cluster"
2025-07-01 06:12:22 [i] waiting for CloudFormation stack "eksctl-myekscluster-cluster"
2025-07-01 06:13:22 [i] waiting for CloudFormation stack "eksctl-myekscluster-cluster"
2025-07-01 06:14:22 [i] waiting for CloudFormation stack "eksctl-myekscluster-cluster"
2025-07-01 06:15:22 [i] waiting for CloudFormation stack "eksctl-myekscluster-cluster"
2025-07-01 06:16:22 [i] waiting for CloudFormation stack "eksctl-myekscluster-cluster"
2025-07-01 06:16:23 [i] creating addon: metrics-server
2025-07-01 06:16:23 [i] successfully created addon: metrics-server
2025-07-01 06:16:24 [!] recommended policies were found for "vpc-cni" addon, but since OIDC is disabled; the recommended way to provide IAM permissions for "vpc-cni" addon is via pod identity associations to the config file, under `addon.PodIdentityAssociations`, and run `eksctl update addon`
2025-07-01 06:16:24 [i] creating addon: vpc-cni
2025-07-01 06:16:24 [i] successfully created addon: vpc-cni
2025-07-01 06:16:24 [i] creating addon: kube-proxy
2025-07-01 06:16:25 [i] successfully created addon: kube-proxy
2025-07-01 06:16:25 [i] creating addon: coredns
2025-07-01 06:16:25 [i] successfully created addon: coredns
```

## Cluster is created

The screenshot shows the AWS Management Console for the Amazon Elastic Kubernetes Service (EKS). The left sidebar contains the navigation menu with options like Dashboard, Clusters, Settings, Amazon EKS Anywhere, and Related services. The main content area displays the 'Clusters (1)' page. A table lists the cluster 'myeksccluster' with a status of 'Active', Kubernetes version 1.32, and a support period until March 21, 2026. The 'Create cluster' button is prominently displayed in the top right corner.

Cluster name	Status	Kubernetes version	Support period	Upgrade policy
myeksccluster	Active	1.32	Standard support until March 21, 2026	Extended

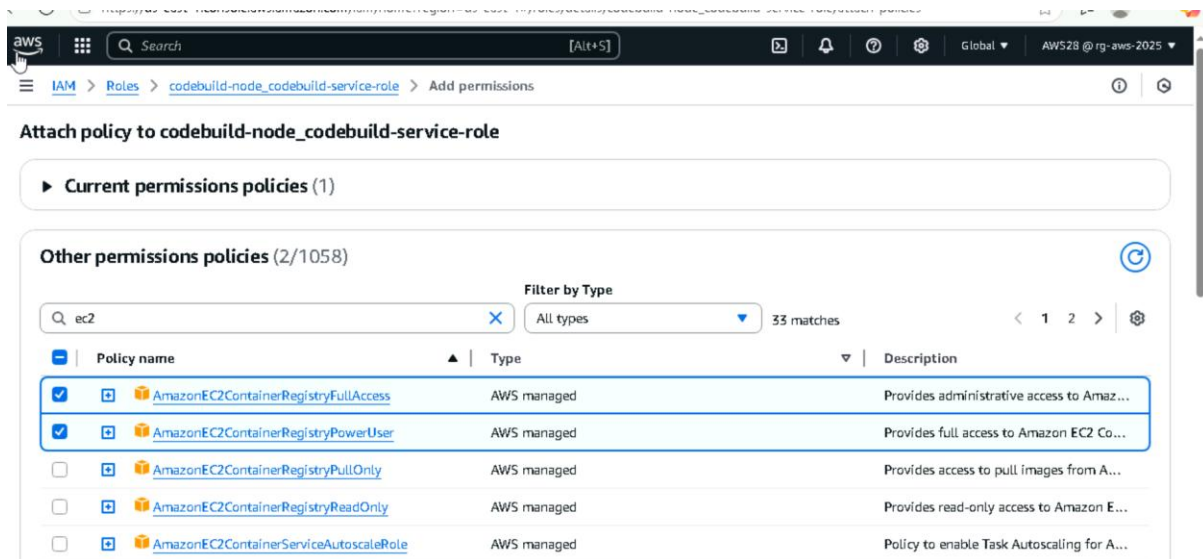
## Node group also successfully created

The screenshot shows the AWS Management Console for the Amazon Elastic Kubernetes Service (EKS), specifically the 'Node groups (1)' page for the cluster 'myeksccluster'. The left sidebar is the same as the previous screenshot. The main content area shows a message 'No Nodes' and a description of node groups. Below this, a table lists the node group 'ng1' with a desired size of 1, AMI release version 1.32.3-20250620, and a status of 'Creating'.

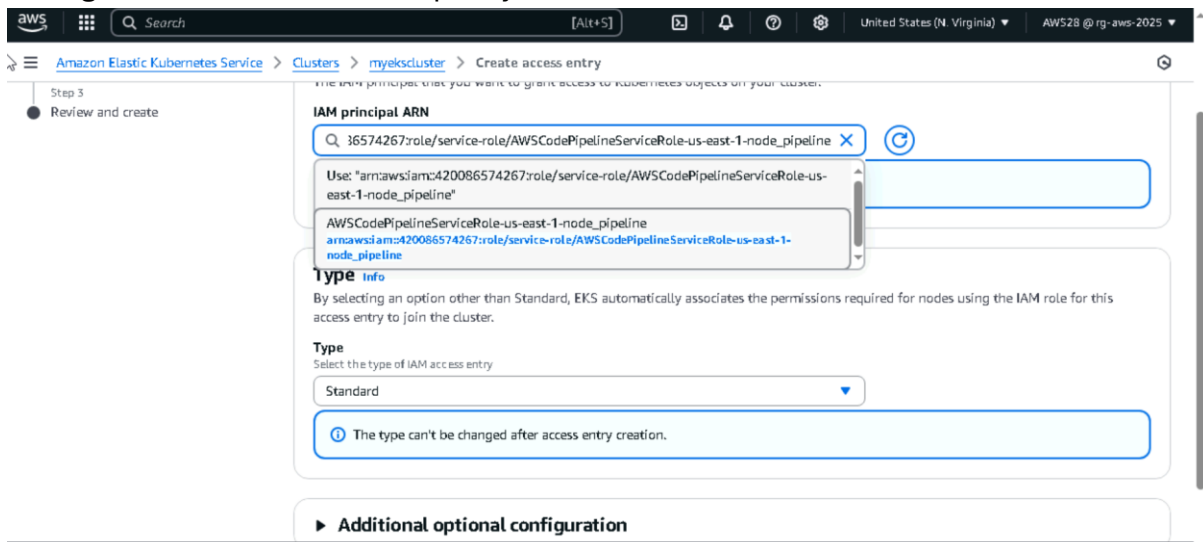
Group name	Desired size	AMI release version	Launch template	Status
ng1	1	1.32.3-20250620	eksctl-myeksccluster-nodegroup-ng1 (1)	Creating

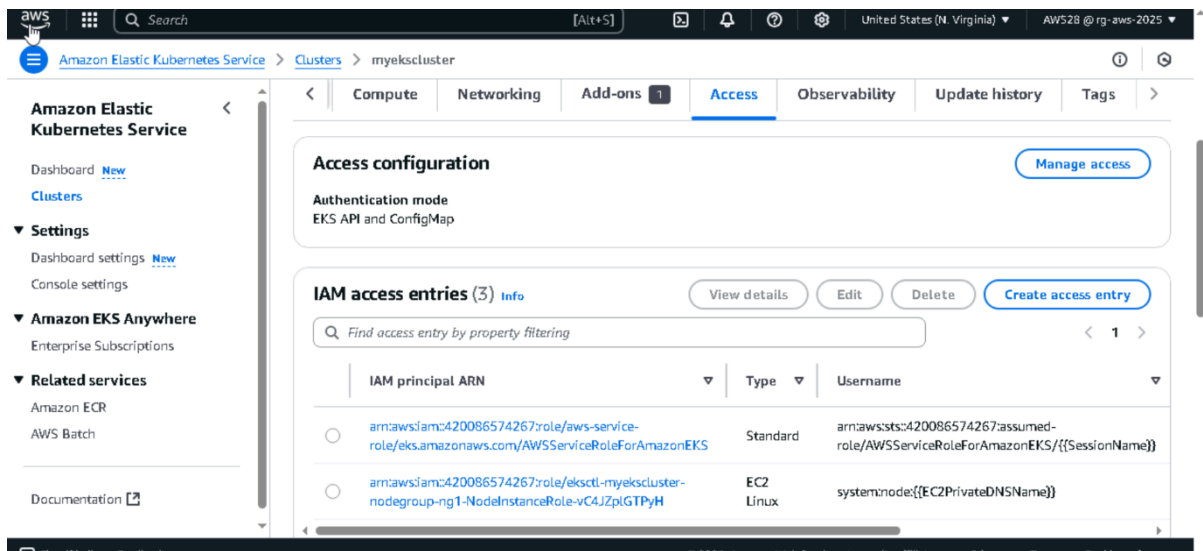
## ECR-EKS integration

For this code build need a permission to ecr that I have give me in IAM code build role for I've given permissions



**EKS integration:** I have given a permission to acces code pipeline to eks cluster..that I've in eks cluster -> acces. Given acces ..codepipeline service role arn ad into thi eks cluste and I given a eks cluster admin policy





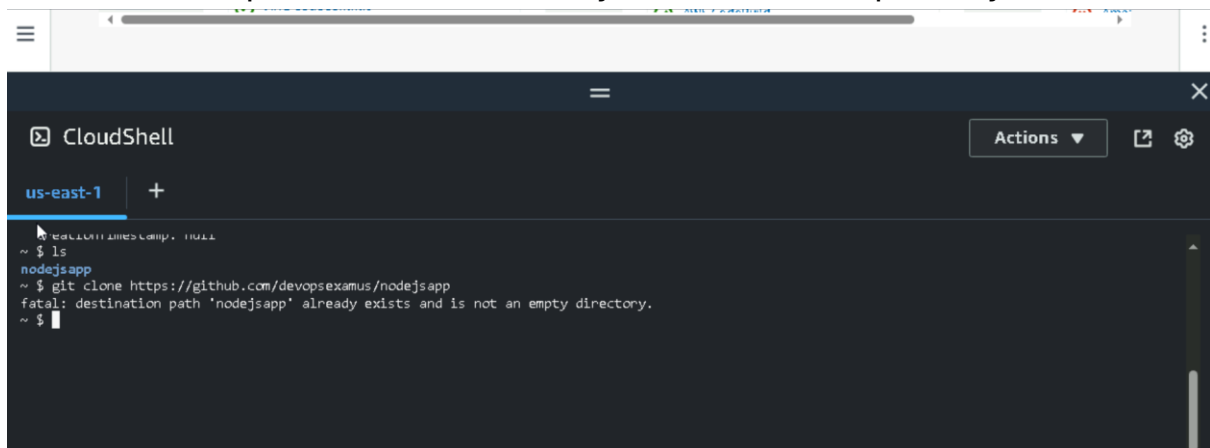
## Set up a aws cicd environment

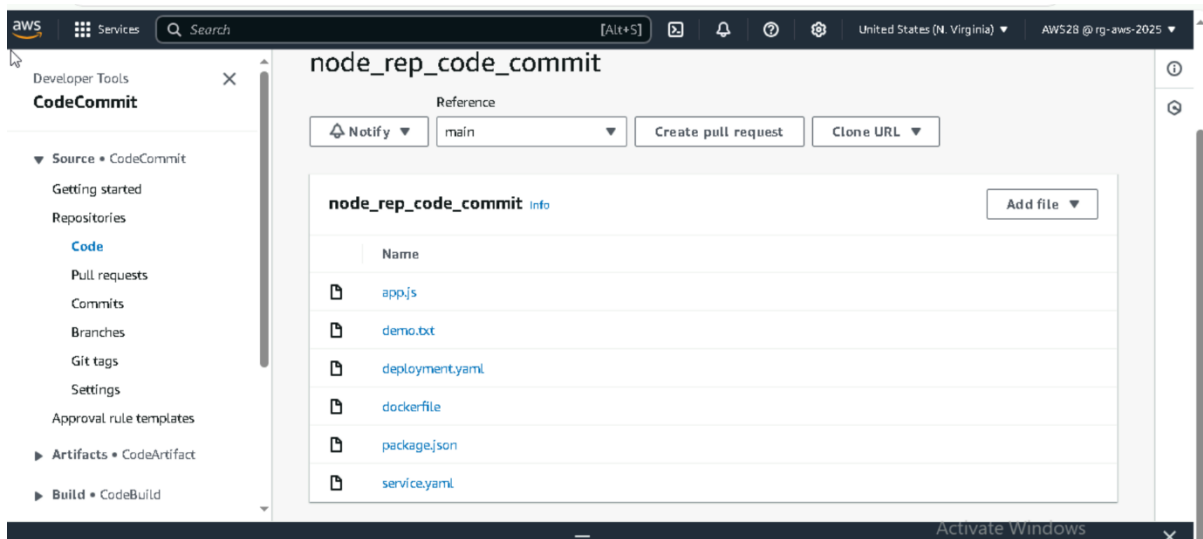
Clone repository:

-created a amazon codecommit (we don't have acces to github)

I have cloned git repository and push into **amazon code commit**

And pushed all files into my code commit repository

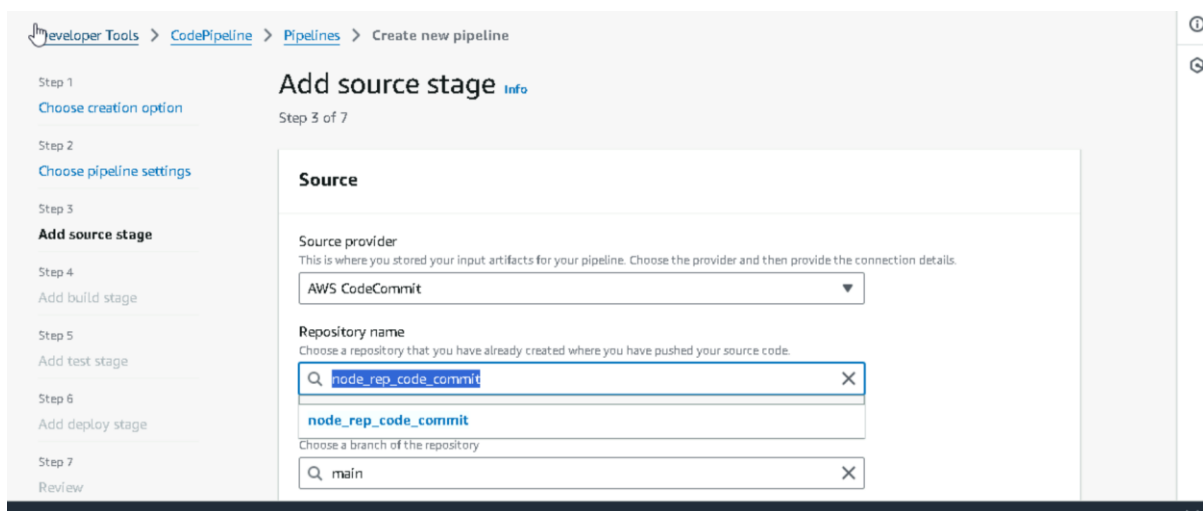




## Service connection:

Connection establishing for pipeline

I have connect my code build repository to pipeline for source artifacts



2. I have added a build stage which I created code build before

The screenshot shows the 'Edit action' configuration page in the AWS CodePipeline console. The URL is [https://us-east-1.console.aws.amazon.com/codesuite/codepipeline/pipelines/node\\_pipeline/edit?region=us-east-1](https://us-east-1.console.aws.amazon.com/codesuite/codepipeline/pipelines/node_pipeline/edit?region=us-east-1). The form includes the following fields:

- Action name:** A text input containing 'Build'.
- Action provider:** A dropdown menu set to 'AWS CodeBuild'.
- Region:** A dropdown menu set to 'United States (N. Virginia)'.
- Input artifacts:** A dropdown menu with a 'SourceArtifact' tag below it, indicating it is defined by the source.
- Project name:** A field that is currently empty.

An 'Activate Windows' watermark is visible in the bottom right corner of the console window.

And finally deploy stage ..application deploy into the eks cluster for configured

The screenshot shows the 'Edit action' configuration page in the AWS CodePipeline console for a 'deploy' action. The URL is [https://us-east-1.console.aws.amazon.com/codesuite/codepipeline/pipelines/node\\_pipeline/edit?region=us-east-1](https://us-east-1.console.aws.amazon.com/codesuite/codepipeline/pipelines/node_pipeline/edit?region=us-east-1). The form includes the following fields:

- Action name:** A text input containing 'deploy'.
- Action provider:** A dropdown menu set to 'Amazon EKS'.
- Region:** A dropdown menu set to 'United States (N. Virginia)'.
- Input artifacts:** A dropdown menu with a 'SourceArtifact' tag below it, indicating it is defined by the source.
- Deploy configuration type:** A section with a search bar containing 'myekscuster' and a list of results below it.

An 'Activate Windows' watermark is visible in the bottom right corner of the console window.

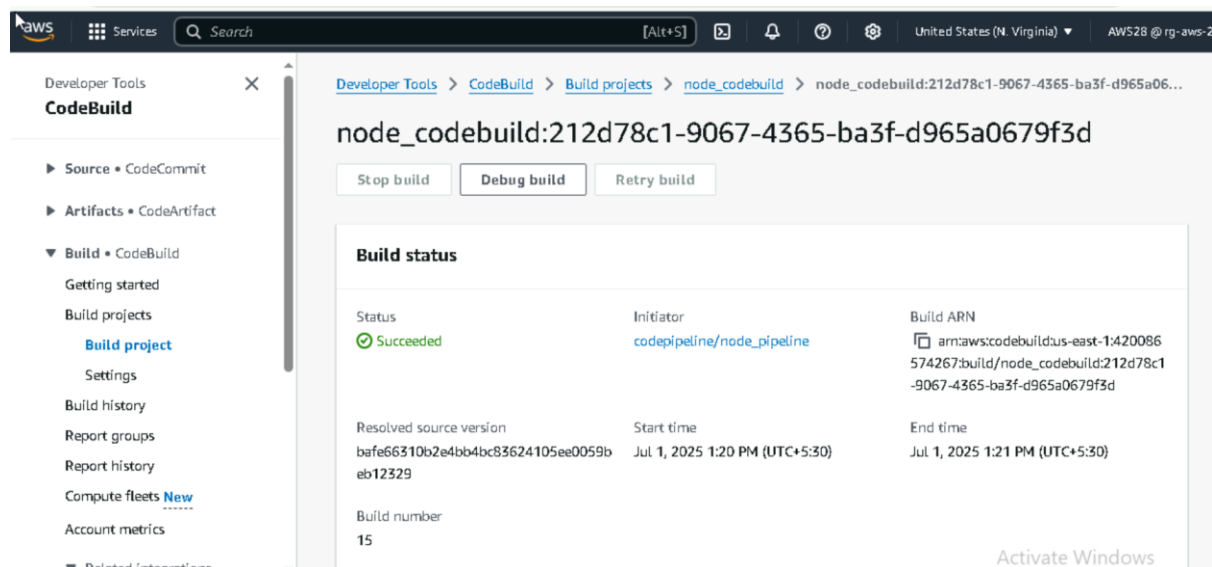
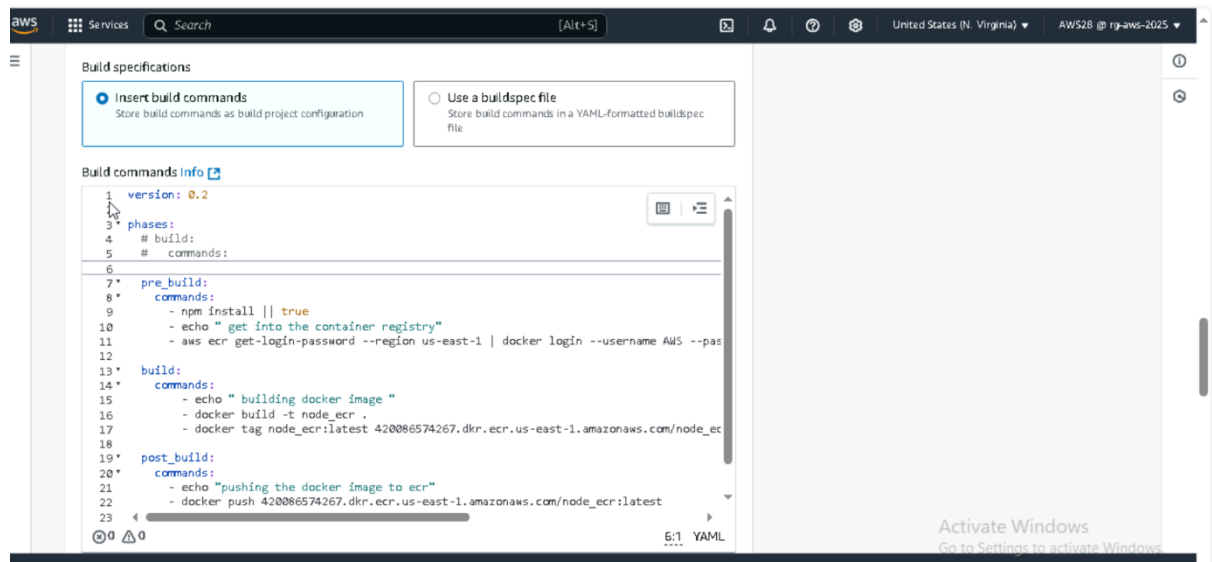
Now pipeline configuration is done..i have given permissions also

## Configure the ci cd pipeline using aws code pipeline

1.i have given a connection codebuild for source

1. created a code build

In that I have wrote buildspect yaml file for all steps





Then all phases are successful login, build, pull all phases

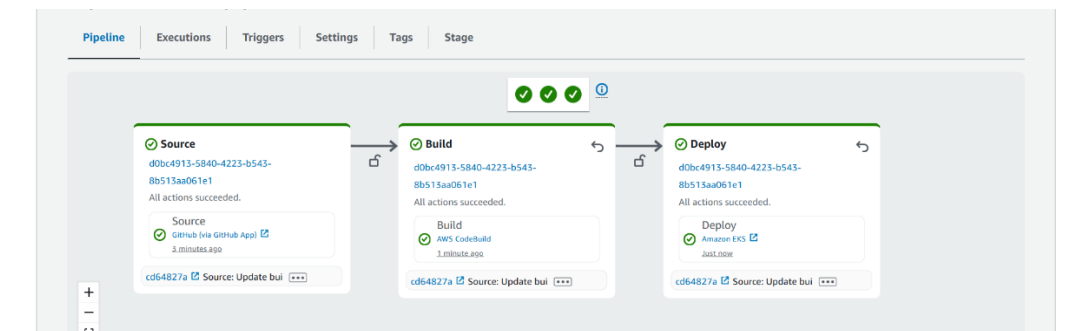
Then in code pipeline I have configured an EKS cluster to deploy stage

The screenshot shows the AWS CodePipeline console interface for configuring a new action. The URL is [https://us-east-1.console.aws.amazon.com/codesuite/codepipeline/pipelines/node\\_pipeline/edit?region=us-east-1](https://us-east-1.console.aws.amazon.com/codesuite/codepipeline/pipelines/node_pipeline/edit?region=us-east-1). The configuration fields are as follows:

- Action name:** deploy (No more than 100 characters)
- Action provider:** Amazon EKS
- Region:** United States (N. Virginia)
- Input artifacts:** Choose an input artifact for this action. [Learn more](#) (No more than 100 characters)
- SourceArtifact:** Defined by: Source
- myekscluster:** (Search results for myekscluster)
- Deploy configuration type:** Please select deploy configuration type.

At the bottom right, there is an "Activate Windows" watermark and buttons for "Cancel" and "Done".

Now all configuration is done



Stage1. Build is success

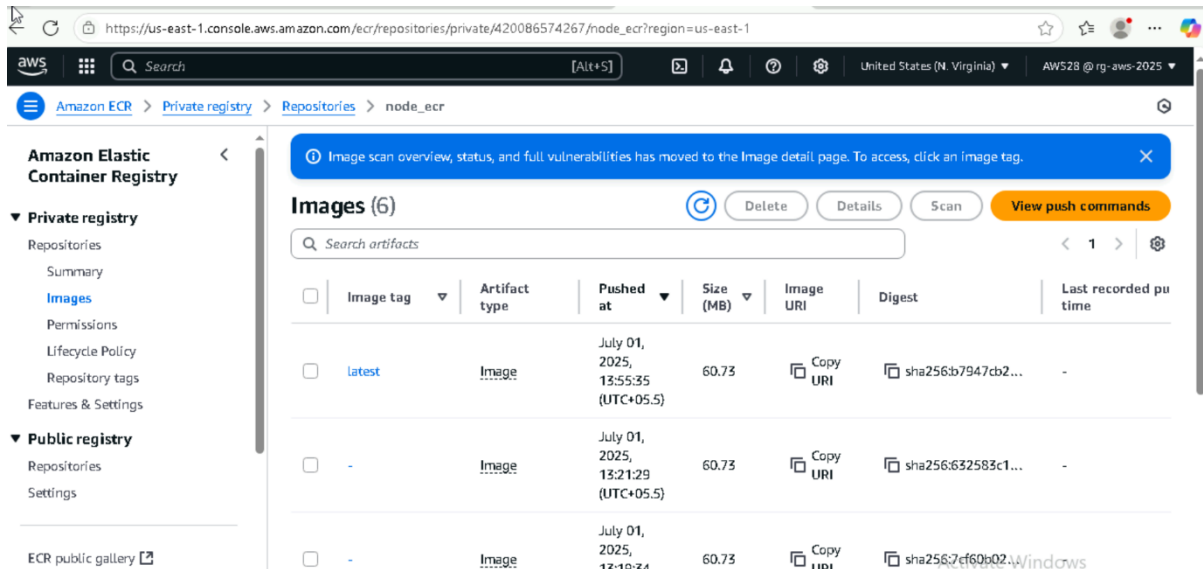
The screenshot shows the AWS CodePipeline console interface with the "Build" stage selected. The build is named "cd64827a" and is in a "Succeeded" state. The build status is "Succeeded". The build details are as follows:

Status	Initiator	Build ARN
Succeeded	codepipeline/node_pipeline	arn:aws:codebuild:us-east-1:420086574267:build/node_codebuild:212d78c1-9067-4365-ba3f-d965a0679f3d

Resolved source version: bafe66310b2e4bb4bc83624105ee0059beb12329  
Start time: Jul 1, 2025 1:20 PM (UTC+5:30)  
End time: Jul 1, 2025 1:21 PM (UTC+5:30)  
Build number: 15

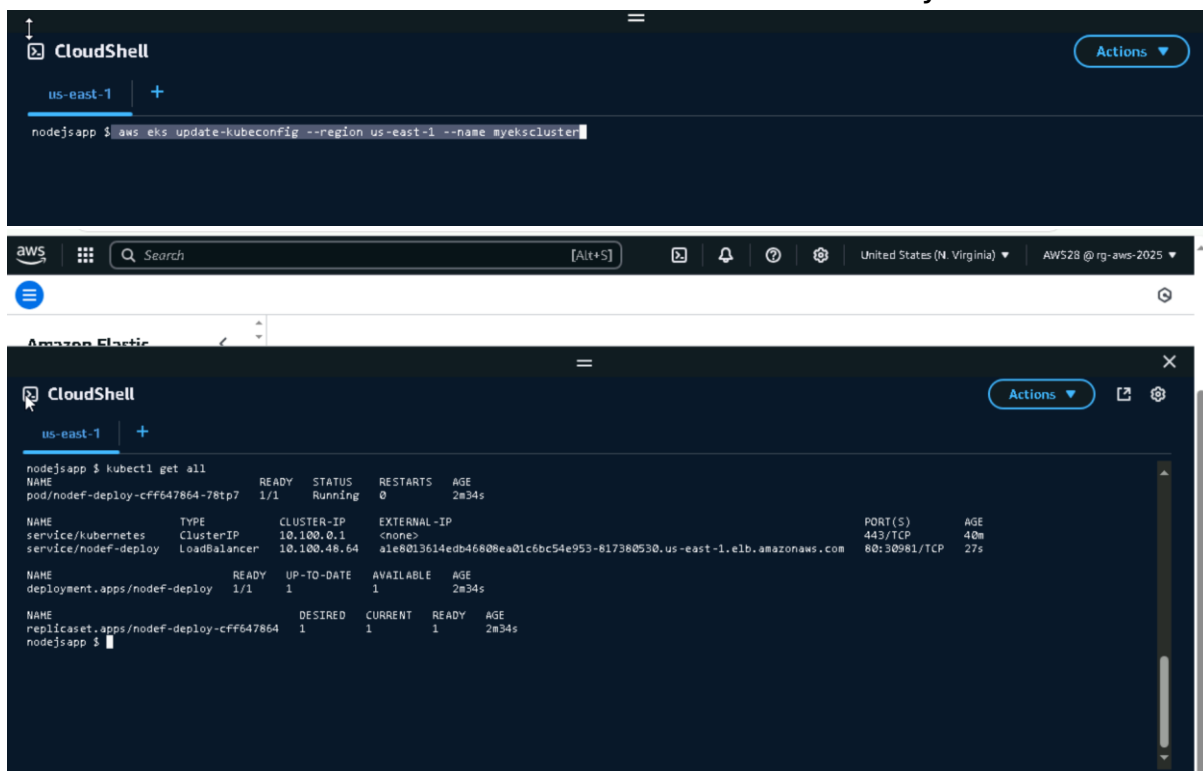
At the bottom right, there is an "Activate Windows" watermark.

Stage2. Successfully pushed the image into ecr is done



Deploy an image to eks cluster is done :

I have accessed aws cluster into cli and successfully created

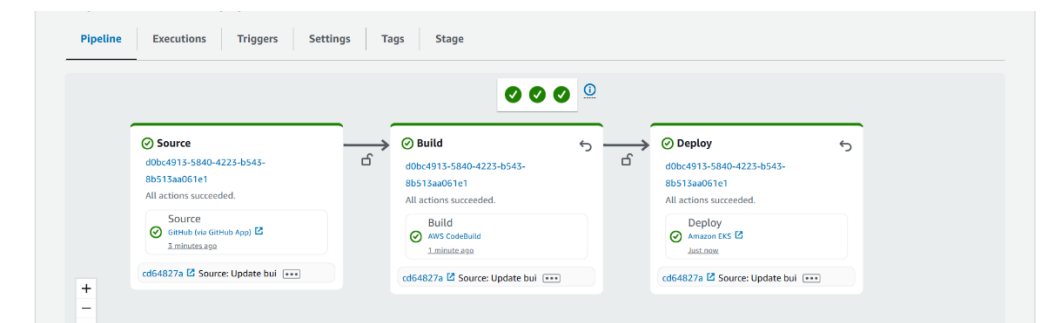


And access using load balancer service load balancer balancer :

Application is deployed successfully



Run and verify pipeline:



and evidences is logs: build push and deloy

```
118 #10 naming to docker.io/library/node_ecr done
119 #10 DONE 0.2s
120
121 [Container] 2025/07/01 07:07:24.354638 Running command docker tag node_ecr:latest 420086574267.dkr.ecr.us-east-1.amazonaws.com/node_ecr:latest
122
123 [Container] 2025/07/01 07:07:24.376823 Phase complete: BUILD State: SUCCEEDED
124 [Container] 2025/07/01 07:07:24.376836 Phase context status code: Message:
125 [Container] 2025/07/01 07:07:24.409185 Entering phase POST_BUILD
126 [Container] 2025/07/01 07:07:24.410172 Running command echo "pushing the docker image to ecr"
127 pushing the docker image to ecr
128
129 [Container] 2025/07/01 07:07:24.417757 Running command docker push 420086574267.dkr.ecr.us-east-1.amazonaws.com/node_ecr:latest
130 The push refers to repository [420086574267.dkr.ecr.us-east-1.amazonaws.com/node_ecr]
131 42ed8c457b0c: Preparing
132 1552a7af537e: Preparing
133 11024a120952: Preparing
134 c348369cdb9b: Preparing
135 45e5afb2890e: Preparing
136 a53004eec6ed: Preparing
137 30e7861e382f: Preparing
138 fd2758d7a50e: Preparing
139 a53004eec6ed: Waiting
140 fd2758d7a50e: Waiting
141 30e7861e382f: Waiting
142 11024a120952: Pushed
143 c348369cdb9b: Pushed
144 45e5afb2890e: Pushed
145 42ed8c457b0c: Pushed
21 [Container] 2025/07/01 07:06:50.687937 Phase complete: INSTALL State: SUCCEEDED
22 [Container] 2025/07/01 07:06:50.687953 Phase context status code: Message:
23 [Container] 2025/07/01 07:06:50.725064 Entering phase PRE_BUILD
24 [Container] 2025/07/01 07:06:50.763273 Running command npm install || true
25
26 added 69 packages, and audited 70 packages in 11s
27
28 14 packages are looking for funding
29   run `npm fund` for details
30
31 found 0 vulnerabilities
32
33 [Container] 2025/07/01 07:07:07.036610 Running command echo " get into the container registry"
34 get into the container registry
35
36 [Container] 2025/07/01 07:07:07.044473 Running command aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 420086574267.dkr.ecr.us-east-1.amazonaws.com
37 WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
38 Configure a credential helper to remove this warning. See
39 https://docs.docker.com/engine/reference/commandline/login/#credential-stores
40
41 Login Succeeded
42
43 [Container] 2025/07/01 07:07:17.394877 Phase complete: PRE_BUILD State: SUCCEEDED
44 [Container] 2025/07/01 07:07:17.394894 Phase context status code: Message:
45 [Container] 2025/07/01 07:07:17.429750 Entering phase BUILD
46 [Container] 2025/07/01 07:07:17.430724 Running command echo " building docker image "
47 building docker image
48
49 [Container] 2025/07/01 07:07:17.438555 Running command docker build -t node_ecr .
50 #0 building with "default" instance using docker driver
```

```

138 fd2758d7a50e: Preparing
139 a53004eec6ed: Waiting
140 fd2758d7a50e: Waiting
141 30e7861e382f: Waiting
142 11024a120952: Pushed
143 c348369cdb9b: Pushed
144 45e5afb2890e: Pushed
145 42ed8c457b0c: Pushed
146 1552a7af537e: Pushed
147 a53004eec6ed: Pushed
148 fd2758d7a50e: Pushed
149 30e7861e382f: Pushed
150 latest: digest: sha256:5545a853adfdfece636c9ae710782504035da6ed43bf9e3ff04c6220f27bfe93 size: 1994
151
152 [Container] 2025/07/01 07:07:32.836329 Phase complete: POST_BUILD State: SUCCEEDED
153 [Container] 2025/07/01 07:07:32.836346 Phase context status code: Message:
154 [Container] 2025/07/01 07:07:32.881567 Set report auto-discover timeout to 5 seconds
155 [Container] 2025/07/01 07:07:32.881608 Expanding base directory path: .
156 [Container] 2025/07/01 07:07:32.884870 Assembling file list
157 [Container] 2025/07/01 07:07:32.884883 Expanding .
158 [Container] 2025/07/01 07:07:32.888037 Expanding file paths for base directory .
159 [Container] 2025/07/01 07:07:32.888050 Assembling file list
160 [Container] 2025/07/01 07:07:32.888053 Expanding **/*
161 [Container] 2025/07/01 07:07:32.894411 No matching auto-discover report paths found
162 [Container] 2025/07/01 07:07:32.894449 Report auto-discover file discovery took 0.012882 seconds
163 [Container] 2025/07/01 07:07:32.894471 Phase complete: UPLOAD_ARTIFACTS State: SUCCEEDED
164 [Container] 2025/07/01 07:07:32.894477 Phase context status code: Message:

```

## Deployment.yaml,service.yaml files

×

node\_rep\_code\_commit / deployment.yaml [Info](#)

Edit

```

1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   creationTimestamp: null
5   labels:
6     app: nodef-deploy
7   name: nodef-deploy
8 spec:
9   replicas: 1
10  selector:
11    matchLabels:
12      app: nodef-deploy
13  strategy: {}
14  template:
15    metadata:
16      creationTimestamp: null
17      labels:
18        app: nodef-deploy
19    spec:
20      containers:
21        - image: vinayz7/nodefinal:v11
22          name: nodefinal
23          resources: {}
24  status: {}

```

Activate Windows

node\_rep\_code\_commit

Reference

Notify

main

Create pull request

Clone URL

node\_rep\_code\_commit / service.yaml

Edit

```
1 apiVersion: v1
2 kind: Service
3 metadata:
4   creationTimestamp: null
5   labels:
6     app: nodef-deploy
7   name: nodef-deploy
8 spec:
9   ports:
10    - port: 80
11      protocol: TCP
12      targetPort: 3000
13   selector:
14     app: nodef-deploy
15   type: LoadBalancer
16 status:
17   loadBalancer: {}
```

Activate Windows

t

leCommit

ad

ists

: templates

odeArtifact

Build

deDeploy